Package: broom (via r-universe)

October 21, 2024

Type Package

Title Convert Statistical Objects into Tidy Tibbles

Version 1.0.7.9000

Description Summarizes key information about statistical objects in tidy tibbles. This makes it easy to report results, create plots and consistently work with large numbers of models at once. Broom provides three verbs that each provide different types of information about a model. tidy() summarizes information about model components such as coefficients of a regression. glance() reports information about an entire model, such as goodness of fit measures like AIC and BIC. augment() adds information about individual observations to a dataset, such as fitted values or influence measures.

License MIT + file LICENSE

URL https://broom.tidymodels.org/, https://github.com/tidymodels/broom

BugReports https://github.com/tidymodels/broom/issues

Depends R (>= 3.5)

Imports backports, cli, dplyr (>= 1.0.0), generics (>= 0.0.2), glue, lifecycle, purrr, rlang (>= 1.1.0), stringr, tibble (>= 3.0.0), tidyr (>= 1.0.0)

Suggests AER, AUC, bbmle, betareg (>= 3.2-1), biglm, binGroup, boot, btergm (>= 1.10.6), car (>= 3.1-2), carData, caret, cluster, cmprsk, coda, covr, drc, e1071, emmeans, epiR, ergm (>= 3.10.4), fixest (>= 0.9.0), gam (>= 1.15), gee, geepack, ggplot2, glmnet, glmnetUtils, gmm, Hmisc, irlba, interp, joineRML, Kendall, knitr, ks, Lahman, lavaan (>= 0.6.18), leaps, lfe, lm.beta, lme4, lmodel2, lmtest (>= 0.9.38), lsmeans, maps, margins, MASS, mclust, mediation, metafor, mfx, mgcv, mlogit, modeldata, modeltests (>= 0.1.6), muhaz, multcomp, network, nnet, orcutt (>= 2.2), ordinal, plm, poLCA, psych, quantreg, rmarkdown, robust, robustbase, rsample, sandwich, spdep (>= 1.1), spatialreg, speedglm, spelling, survey, survival (>= 3.6-4), systemfit, testthat (>= 3.0.0), tseries, vars, zoo

VignetteBuilder knitr

Encoding UTF-8 Roxygen list(markdown = TRUE) RoxygenNote 7.3.2 Language en-US Collate 'aaa-documentation-helper.R' 'null-and-default.R' 'aer.R' 'auc.R' 'base.R' 'bbmle.R' 'betareg.R' 'biglm.R' 'bingroup.R' 'boot.R' 'broom-package.R' 'broom.R' 'btergm.R' 'car.R' 'caret.R' 'cluster.R' 'cmprsk.R' 'data-frame.R' 'deprecated-0-7-0.R' 'drc.R' 'emmeans.R' 'epiR.R' 'ergm.R' 'fixest.R' 'gam.R' 'geepack.R' 'glmnet-ev-glmnet.R' 'glmnet-glmnet.R' 'glmnet-glmnet.R' 'mport-standalone-obj-type.R' 'import-standalone-obj-type.R' 'inst-optim.R' 'layavan.R' 'lays.R' 'lfc.R' 'list-iriba.R' 'list-optim.R' 'list-svd.R' 'list-xyz.R' 'list.R' 'lm-beta.R' 'lmodel2.R' 'lmtest.R' 'maps.R' 'margins.R' 'mass-fitdistr.R' 'mass-relpin.R' 'mass-polr.R' 'mass-ridgelm.R' 'stats-lm.R' 'mass-rlm.R' 'mclust.R' 'mediation.R' 'metafor.R' 'mfx.R' 'mgcv.R' 'mlogit.R' 'muhaz.R' 'multcomp.R' 'nnet.R' 'nobs.R' 'orout.R' 'ordinal-clmm.R' 'plm.R' 'ploca.R' 'psych.R' 'stats-nls.R' 'quantreg-rap.R' 'quantreg-rap.R' 'quantreg-rap.R' 'quantreg-rap.R' 'quantreg-rap.R' 'robust-glmrob.R' 'robust-lmrob.R' 'robust-glmrob.R' 'robust-lmrob.R' 'robust-glmrob.R' 'stats-anova.R' 'stats-alm.R' 'stats-beedplm.R' 'stats-anova.R' 'stats-alm.R' 'stats-breomps.R' 'stats-factanal.R' 'stats-alm.R' 'stats-breomps.R' 'stats-smooth.spline.R' 'stats-smlm.R' 'stats-breomps.R' 'stats-smooth.spline.R' 'stats-smlm.R' 'stats-breomps.R' 'stats-smooth.spline.R' 'stats-smlm.R' 'stats-breomps.R' 'stats-smooth.spline.R' 'stats-smlm.R' 'stats-breomps.R' 'survival-oxph.R' 'survival-pycars.R' 'survival-surveq.R' 'survival-survexp.R' 'survival-survef.R' 'survival-survexp.R' 'survival-survef.R' 'survival-survexp.R' 'survival-survef.R' 'survival-survexp.R' 'survival-survit.R' 'survival-coh.R' 'survival-survexp.R' 'survival-survef.R' 'survival-bedafa e480deb74c8ea172e1d5e69630971c4b59bb43dd Contents augment.betareg.	Config/Needs/website tidyverse/tidytemplate
RoxygenNote 7.3.2 Language en-US Collate 'aaa-documentation-helper.R' 'null-and-default.R' 'aer.R' 'auc.R' 'base.R' 'bbmle.R' 'betareg.R' 'biglm.R' 'bingroup.R' 'boot.R' 'broom-package.R' 'broom R' 'btergm.R' 'car.R' 'caret.R' 'cluster.R' 'cmprsk.R' 'data-frame.R' 'deprecated-0-7-0.R' 'drc.R' 'emmeans.R' 'epiR.R' 'ergm.R' 'fixest.R' 'gam.R' 'geepack.R' 'glmnet-cv-glmnet.R' glmnet-glmnet.R' glmnet-pye.R' 'import-standalone-obj-type.R' 'import-standalone-obj-type.R' 'import-standalone-types-check.R' 'joinerml.R' 'kendall.R' 'ks.R' 'lavaan.R' 'leaps.R' 'lfe.R' 'list-irlba.R' 'list-optim.R' 'list-svd.R' 'list-xyz.R' 'list.R' 'lm-beta.R' 'lmodel.R' 'lmsts.vd.R' 'list-xyz.R' 'list.R' 'lm-beta.R' 'mass-negbin.R' 'mass-polr.R' 'mass-ridgelm.R' 'stats-lm.R' 'mass-rim.R' 'mass-negbin.R' 'mass-ridgelm.R' 'stats-lm.R' 'mgcv.R' 'mlogit.R' 'muhaz.R' 'multcomp.R' 'nnet.R' 'nobs.R' 'orcutt.R' 'ordinal-clm.R' 'ordinal-clm.R' 'plm.R' 'polea.R' 'psych.R' 'stats-nls.R' 'quantreg-rq.R' 'quantreg-rq.R' 'quantreg-rq.R' 'quantreg-rq.R' 'quantreg-rq.R' 'robust-glmrob.R' 'robust-lmrob.R' 'speedglm-speedglm.R' 'speedglm-speedlm.R' 'stats-anova.R' 'stats-arima.R' 'stats-decompose.R' 'stats-factanal.R' 'stats-arima.R' 'stats-decompose.R' 'stats-factanal.R' 'stats-arima.R' 'stats-decompose.R' 'stats-sens.R' 'stats-loes.R' 'stats-mlm.R' 'stats-prcomp.R' 'stats-smooth.spline.R' 'stats-mlm.R' 'stats-prcomp.R' 'stats-smooth.spline.R' 'stats-mlm.R' 'stats-prcomp.R' 'stats-smooth.spline.R' 'stats-summary-lm.R' 'stats-time-series.R' 'survey.R' 'survival-areg.R' 'survival-areg.R' 'survival-coxph.R' 'survival-coxph.R' 'survival-survep.R' 'survival-survep.	Encoding UTF-8
Language en-US Collate 'aaa-documentation-helper.R' 'null-and-default.R' 'aer.R' 'auc.R' 'base.R' 'bbmle.R' 'betareg.R' 'biglm.R' 'bingroup.R' 'boot.R' 'broom-package.R' 'broom.R' 'btergm.R' 'car.R' 'caret.R' 'cluster.R' 'cmprsk.R' 'data-frame.R' 'deprecated-0-7-0.R' 'drc.R' 'emmeans.R' 'epiR.R' 'ergm.R' 'fixest.R' 'gam.R' 'gepack.R' 'glmnet-ev-glmnet.R' 'glmnet-glmet.R' 'gmm.R' 'hmisc.R' 'import-standalone-obj-type.R' 'import-standalone-obj-type.R' 'import-standalone-types-check.R' 'joinerml.R' 'kendall.R' 'ks.R' 'lavaan.R' 'leaps.R' 'lfe.R' 'list-iriba.R' 'list-optim.R' 'list-svd.R' 'list-xyz.R' 'list.R' 'lm-beta.R' 'model2.R' 'lmtest.R' 'maps.R' 'margsn.R' 'mass-riddistr.R' 'mass-negbin.R' mass-polr.R' 'mass-ridgelm.R' 'stats-lm.R' 'mroust.R' 'mediation.R' 'metafor.R' 'mfx.R' 'mgcv.R' mlogit.R' 'muhaz.R' 'multcomp.R' 'nnet.R' 'nobs.R' 'orcutt.R' 'ordinal-clm.R' 'ordinal-clm.R' 'pim.R' 'polca.R' 'psych.R' 'stats-sl.R' 'quantreg-rqs.R' 'quantreg-rqs.R' 'robust-glmrob.R' robust-lmrob.R' 'psych.R' 'stats-glm.R' 'speedglm-speedlm.R' 'stats-arima.R' 'stats-arima.R' 'stats-decompose.R' 'stats-factanal.R' 'stats-arima.R' 'stats-bres.R' 'stats-smooth.spline.R' 'stats-summary-lm.R' 'stats-smooth.spline.R' 'stats-summary-lm.R' 'stats-smooth.spline.R' 'stats-summary-lm.R' 'stats-smooth.spline.R' 'stats-summary-lm.R' 'stats-smooth.spline.R' 'stats-survival-survit.R' 'survival-survidif.R' 'survival-survexp.R' 'survival-survit.R' 'su	Roxygen list(markdown = TRUE)
Collate 'aaa-documentation-helper.R' 'null-and-default.R' 'aer.R' 'auc.R' 'base.R' 'bbmle.R' 'betareg.R' 'biglm.R' 'bingroup.R' boot.R' 'broom-package.R' 'broom.R' btergm.R' 'car.R' caret.R' 'cluster.R' 'emprs.R. 'data-frame.R' 'deprecated-0-7-0.R' 'drc.R' 'emmeans.R' 'epiR.R' 'ergm.R' 'fixest.R' 'gam.R' 'gepack.R' 'glmnet-ev-glmnet.R' 'glmnet-glmnet.R' 'gemm.R' 'hisis.c.R' 'import-standalone-obj-type.R' 'import-standalone-types-check.R' 'joinerml.R' 'kendall.R' 'ks.R' 'lavaan.R' 'leaps.R' 'lfe.R' 'list-iriba.R' 'list-optim.R' 'list-svd.R' 'list-syz.R' 'list.R' 'lm-beta.R' 'lmodel2.R' 'lmtest.R' 'maps.R' 'margin.R' 'mass-ridgelm.R' 'stats-lm.R' 'muss-negbin.R' mass-plc.R' 'mass-ridgelm.R' 'stats-lm.R' 'muss-rim.R' 'mult.st.R' 'mdiation.R' 'metafor.R' 'mfx.R' 'mgcv.R' 'mlogit.R' 'muhaz.R' 'multcomp.R' 'nnet.R' 'nobs.R' 'orcutt.R' 'ordinal-clm.R' 'ordinal-clm.R' 'plm.R' 'polca.R' 'psych.R' 'stats-ls.R' 'quantreg-rq.R' 'quantreg-rqs.R' 'quantreg-rqs.R' 'quantreg-rgs.R' 'robust-glmrob.R' 'robust-lmrob.R' 'speedglm-speedglm.R' 'speedglm-speedglm.R' 'stats-arima.R' 'stats-arima.R' 'stats-decompose.R' 'stats-factanal.R' 'stats-arima.R' 'stats-btes.R' 'stats-kmeans.R' 'stats-loess.R' 'stats-mlm.R' 'stats-btes.R' 'stats-kmeans.R' 'stats-loess.R' 'stats-mlm.R' 'stats-time-series.R' 'survival-survit.R' 'survival-sur	RoxygenNote 7.3.2
'auc.R' 'base.R' 'betne.R' 'betareg.R' 'biglm.R' 'bingroup.R' boot.R' 'broom-package.R' 'broom.R' 'btergm.R' 'car.R' 'caret.R' 'cluster.R' 'cmprsk.R' 'data-frame.R' 'deprecated-0-7-0.R' 'drc.R' 'emmeans.R' 'epiR.R' 'ergm.R' 'fixest.R' 'gam.R' 'geopack.R' 'glmnet-ev-glmnet.R' 'glmnet-glmnet.B' 'gmm.R' 'hmisc.R' 'import-standalone-obj-type.R' 'import-standalone-types-check.R' 'joinerml.R' 'kendall.R' 'ks.R' 'lavaan.R' 'leaps.R' 'lfc.R' 'list-irlba.R' 'list-optim.R' 'list-svd.R' 'list-xyz.R' 'list.R' 'lm-beta.R' 'lmodel2.R' 'lmtest.R' 'maps.R' 'mass-ridgelm.R' 'stats-lm.R' 'mass-negbin.R' 'mass-polr.R' 'mass-ridgelm.R' 'stats-lm.R' 'mass-rim.R' 'mclust.R' 'mediation.R' 'metafor.R' 'mfx.R' 'mgcv.R' 'mlogit.R' 'mulacom.R' 'plm.R' 'polca.R' 'psych.R' 'stats-nls.R' 'quantreg-rq.R' 'quantreg-rq.R' 'quantreg-rq.R' 'quantreg-rq.R' 'quantreg-rq.R' 'spoustbase-glmrob.R' 'robust-lmrob.R' 'spedglm-speedglm.R' 'spedglm-speedlm.R' 'stats-anova.R' 'stats-arima.R' 'stats-decompose.R' 'stats-factanal.R' 'stats-anova.R' 'stats-mlm.R' 'stats-btest.R' 'stats-summary-lm.R' 'stats-sumooth.spline.R' 'stats-summary-lm.R' 'stats-time-series.R' 'survival-pyears.R' 'survival-ca.R' 'survival-coxph.R' 'survival-pyears.R' 'survival-surveg.R' 'systemfit.R' 'survival-pyears.R' 'survival-surveg.R' 'systemfit.R' 'series.R' 'utilities.R' 'vars.R' 'zoo.R' 'zzz.R' Config/testthat/edition 3 Repository https://razvanazamfirei.r-universe.dev RemoteCrl https://github.com/tidymodels/broom RemoteSha e480deb74c8ea172e1d5e69630971c4b59bb43dd Contents augment.betamfx	Language en-US
RemoteUrl https://github.com/tidymodels/broom RemoteRef HEAD RemoteSha e480deb74c8ea172e1d5e69630971c4b59bb43dd Contents augment.betamfx	'auc.R' 'base.R' 'bbmle.R' 'betareg.R' 'biglm.R' 'bingroup.R' 'boot.R' 'broom-package.R' 'broom.R' 'btergm.R' 'car.R' 'caret.R' 'cluster.R' 'cmprsk.R' 'data-frame.R' 'deprecated-0-7-0.R' 'drc.R' 'emmeans.R' 'epiR.R' 'ergm.R' 'fixest.R' 'gam.R' 'geepack.R' 'glmnet-cv-glmnet.R' 'glmnet-glmnet.R' 'gmm.R' 'hmisc.R' 'import-standalone-obj-type.R' 'import-standalone-types-check.R' 'joinerml.R' 'kendall.R' 'ks.R' 'lavaan.R' 'leaps.R' 'lfe.R' 'list-irlba.R' 'list-optim.R' 'list-svd.R' 'list-xyz.R' 'list.R' 'lm-beta.R' 'lmodel2.R' 'lmtest.R' 'maps.R' 'margins.R' 'mass-fidistr.R' 'mass-negbin.R' 'mass-polr.R' 'mass-ridgelm.R' 'stats-lm.R' 'mass-rlm.R' 'mclust.R' 'mediation.R' 'metafor.R' 'mfx.R' 'mgcv.R' 'mlogit.R' 'muhaz.R' 'multcomp.R' 'nnet.R' 'nobs.R' 'orcutt.R' 'ordinal-clm.R' 'ordinal-clmm.R' 'plm.R' 'polca.R' 'psych.R' 'stats-nls.R' 'quantreg-nlrq.R' 'quantreg-rq.R' 'quantreg-rqs.R' 'robust-glmrob.R' 'robust-lmrob.R' 'robustbase-glmrob.R' 'robust-lmrob.R' 'speedglm-speedglm.R' 'speedglm-speedlm.R' 'stats-anova.R' 'stats-arima.R' 'stats-decompose.R' 'stats-factanal.R' 'stats-alm.R' 'stats-decompose.R' 'stats-factanal.R' 'stats-alm.R' 'stats-hest.R' 'stats-kmeans.R' 'stats-loess.R' 'stats-mlm.R' 'stats-time-series.R' 'survey.R' 'survival-aareg.R' 'survival-cch.R' 'survival-coxph.R' 'survival-aareg.R' 'survival-survifit.R' 'survival-survep.R' 'survival-survifit.R' 'survival-surveg.R' 'systemfit.R' 'tseries.R' 'utilities.R' 'vars.R' 'zoo.R' 'zzz.R'
RemoteUrl https://github.com/tidymodels/broom RemoteRef HEAD RemoteSha e480deb74c8ea172e1d5e69630971c4b59bb43dd Contents augment.betamfx	
RemoteRef HEAD RemoteSha e480deb74c8ea172e1d5e69630971c4b59bb43dd Contents augment.betamfx	• •
Contents augment.betamfx	•
augment.betamfx	RemoteSha e480deb74c8ea172e1d5e69630971c4b59bb43dd
	augment.betamfx

augment.clm
augment.coxph
augment.decomposed.ts
augment.drc
augment.factanal
augment.felm
augment.fixest
augment.gam
augment.glm
augment.glmRob
augment.glmrob
augment.htest
augment.ivreg
augment.kmeans
augment.lm
augment.lmRob
augment.lmrob
augment.loess
augment.Mclust
augment.mfx
e
c ,
e i
e
\mathcal{E}
augment.plm
augment.poLCA
augment.polr
augment.prcomp
augment.rlm
augment.rma
augment.rq
augment.rqs
augment.sarlm
augment.smooth.spline
augment.speedlm
augment.stl
augment.survreg
augment_columns
bootstrap
confint_tidy
data.frame_tidiers
durbinWatsonTest_tidiers
finish_glance
fix_data_frame
glance.aareg
glance.anova
glance.aov

glance.Arima								
glance.betamfx .								
glance.betareg	 	. 107						
glance.biglm								
glance.binDesign	 	. 110						
glance.cch	 	. 111						
glance.clm	 	. 113						
glance.clmm	 	. 115						
glance.coeftest .	 	. 117						
glance.coxph	 	. 119						
glance.crr	 	. 121						
glance.cv.glmnet	 	. 123						
glance.drc	 	. 125						
glance.ergm								
glance.factanal .								
glance.felm								
glance.fitdistr								
glance.fixest								
glance.Gam								
glance.gam								
glance.garch								
glance.geeglm								
glance.glm								
glance.glm glance.glmnet								
glance.glmRob .								
glance.gmm								
glance.ivreg								
glance.kmeans								
glance.lavaan								
U								
glance.lm glance.lmodel2 .								
-								
glance.lmRob								
glance.lmrob								
glance.margins .								
glance.Mclust								
glance.mfx								
glance.mjoint								
glance.mlogit								
glance.muhaz								
glance.multinom								
glance.negbin								
glance.nlrq								
glance.nls								
glance.orcutt								
glance.pam								
glance.plm	 	. 180						
glance.poLCA								
glance.polr	 	. 184						

glance.pyears	
glance.ridgelm	188
glance.rlm	189
glance.rma	191
glance.rq	193
glance.sarlm	195
glance.smooth.spline	197
glance.speedglm	198
glance.speedlm	200
glance.summary.lm	
glance.survdiff	
glance.survexp	206
glance.survfit	
glance.survreg	
glance.svyglm	
glance.svyolr	
glance.varest	
glance_optim	
leveneTest_tidiers	
list_tidiers	
null_tidiers	
sp_tidiers	
summary_tidiers	
tidy.aareg	
tidy.acf	
tidy.anova	
tidy.aov	
tidy.aovlist	
tidy.Arima	
tidy.betamfx	
tidy.betareg	
tidy.biglm	
tidy.binDesign	
tidy.binWidth	
tidy.boot	
tidy.btergm	
tidy.cch	
tidy.cld	
tidy.clm	
tidy.clmm	
tidy.coeftest	
tidy.confint.glht	
tidy.confusionMatrix	
tidy.coxph	
tidy.crr	
tidy.cv.glmnet	
tidy.density	
tidy.dist	260

4: 4-, 4								261
tidy.drc tidy.emmGrid								
•								
tidy.epi.2by2								
tidy.ergm								
tidy.factanal .								
tidy.felm								
tidy.fitdistr								
tidy.fixest								
tidy.ftable								
tidy.Gam								
tidy.gam								
tidy.garch	 	 . 279						
tidy.geeglm .	 	 . 280						
tidy.glht	 	 . 282						
tidy.glm	 	 . 283						
tidy.glmnet .	 	 . 284						
tidy.glmRob .	 	 . 286						
tidy.glmrob .	 	 . 287						
tidy.gmm	 	 . 289						
tidy.htest	 	 . 291						
idy.ivreg	 	 . 293						
tidy.kappa								
tidy.kde								
tidy.Kendall .								
tidy.kmeans .								
tidy.lavaan								
idy.lm								
idy.lm.beta .								
dy.lmodel2.								
tidy.lmRob .								
tidy.lmrob								
tidy.lsmobj .								
tidy.nanova .								
tidy.map								
tidy.margins .								
tidy.Mclust .								
•								
tidy.mediate .								
-								. 322
tidy.mjoint								
•								. 326
tidy.mlm								
tidy.mlogit								
tidy.muhaz								
tidy. multinom	 	 . 332						
tidy.negbin	 	 . 334						
tidy.nlrq	 	 . 335						
tidy.nls	 	 . 336						
tidy.numeric .	 	 . 338						

tidy.orcutt	9
tidy.pairwise.htest	0
tidy.pam	
tidy.plm	3
tidy.poLCA	5
tidy.polr	
tidy.power.htest	9
tidy.prcomp	
tidy.pyears	
tidy.rcorr	
tidy.ref.grid	
tidy.regsubsets	
tidy.ridgelm	
tidy.rlm	
tidy.rma	
tidy.roc	
tidy.rq	
tidy.rqs	
tidy.sarlm	
tidy.spec	
tidy.speedglm	
tidy.speedlm	
tidy.summary.glht	
tidy.summary.lm	
tidy.summary_emm	
tidy.survdiff	
tidy.survexp	
tidy.survfit	
tidy.survreg	
tidy.svyglm	
tidy.svyolr	
tidy.systemfit	
tidy.table	
tidy.ts	
tidy.TukeyHSD	
tidy.varest	
tidy.zoo	
tidy_irlba	
tidy_optim	
tidy_svd	
tidy_xyz	2
40	4

Index

8 augment.betamfx

augment.betamfx

Augment data with information from a(n) betamfx object

Description

Augment accepts a model object and a dataset and adds information about each observation in the dataset. Most commonly, this includes predicted values in the .fitted column, residuals in the .resid column, and standard errors for the fitted values in a .se.fit column. New columns always begin with a . prefix to avoid overwriting columns in the original dataset.

Users may pass data to augment via either the data argument or the newdata argument. If the user passes data to the data argument, it **must** be exactly the data that was used to fit the model object. Pass datasets to newdata to augment data that was not used during model fitting. This still requires that at least all predictor variable columns used to fit the model are present. If the original outcome variable used to fit the model is not included in newdata, then no .resid column will be included in the output.

Augment will often behave differently depending on whether data or newdata is given. This is because there is often information associated with training observations (such as influences or related) measures that is not meaningfully defined for new observations.

For convenience, many augment methods provide default data arguments, so that augment(fit) will return the augmented training data. In these cases, augment tries to reconstruct the original data based on the model object with varying degrees of success.

The augmented dataset is always returned as a tibble::tibble with the **same number of rows** as the passed dataset. This means that the passed data must be coercible to a tibble. If a predictor enters the model as part of a matrix of covariates, such as when the model formula uses splines::ns(), stats::poly(), or survival::Surv(), it is represented as a matrix column.

We are in the process of defining behaviors for models fit with various na.action arguments, but make no guarantees about behavior when data is missing at this time.

Usage

Arguments

x A betamfx object.

augment.betamfx 9

data A base::data.frame or tibble::tibble() containing the original data that was

used to produce the object x. Defaults to stats::model.frame(x) so that augment(my_fit) returns the augmented original data. **Do not** pass new data to the data argument. Augment will report information such as influence and cooks distance for data passed to the data argument. These measures are only

defined for the original training data.

newdata A base::data.frame() or tibble::tibble() containing all the original pre-

dictors used to create x. Defaults to NULL, indicating that nothing has been passed to newdata. If newdata is specified, the data argument will be ignored.

type.predict Character indicating type of prediction to use. Passed to the type argument of

betareg::predict.betareg(). Defaults to "response".

type.residuals Character indicating type of residuals to use. Passed to the type argument of

betareg::residuals.betareg(). Defaults to "sweighted2.

Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in . . ., where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

Details

This augment method wraps augment.betareg() for mfx::betamfx() objects.

Value

```
A tibble::tibble() with columns:
```

. cooksd Cooks distance.

.fitted Fitted or predicted value.

resid The difference between observed and fitted values.

See Also

```
augment.betareg(), mfx::betamfx()
Other mfx tidiers: augment.mfx(), glance.betamfx(), glance.mfx(), tidy.betamfx(), tidy.mfx()
```

Examples

```
library(mfx)
# Simulate some data
set.seed(12345)
n <- 1000</pre>
```

10 augment.betareg

```
x <- rnorm(n)

# Beta outcome
y <- rbeta(n, shape1 = plogis(1 + 0.5 * x), shape2 = (abs(0.2 * x)))
# Use Smithson and Verkuilen correction
y <- (y * (n - 1) + 0.5) / n

d <- data.frame(y, x)
mod_betamfx <- betamfx(y ~ x | x, data = d)

tidy(mod_betamfx, conf.int = TRUE)

# Compare with the naive model coefficients of the equivalent betareg call (not run)
# tidy(betamfx(y ~ x | x, data = d), conf.int = TRUE)

augment(mod_betamfx)
glance(mod_betamfx)</pre>
```

augment.betareg

Augment data with information from a(n) betareg object

Description

Augment accepts a model object and a dataset and adds information about each observation in the dataset. Most commonly, this includes predicted values in the .fitted column, residuals in the .resid column, and standard errors for the fitted values in a .se.fit column. New columns always begin with a . prefix to avoid overwriting columns in the original dataset.

Users may pass data to augment via either the data argument or the newdata argument. If the user passes data to the data argument, it **must** be exactly the data that was used to fit the model object. Pass datasets to newdata to augment data that was not used during model fitting. This still requires that at least all predictor variable columns used to fit the model are present. If the original outcome variable used to fit the model is not included in newdata, then no .resid column will be included in the output.

Augment will often behave differently depending on whether data or newdata is given. This is because there is often information associated with training observations (such as influences or related) measures that is not meaningfully defined for new observations.

For convenience, many augment methods provide default data arguments, so that augment(fit) will return the augmented training data. In these cases, augment tries to reconstruct the original data based on the model object with varying degrees of success.

The augmented dataset is always returned as a tibble::tibble with the **same number of rows** as the passed dataset. This means that the passed data must be coercible to a tibble. If a predictor enters the model as part of a matrix of covariates, such as when the model formula uses splines::ns(), stats::poly(), or survival::Surv(), it is represented as a matrix column.

We are in the process of defining behaviors for models fit with various na.action arguments, but make no guarantees about behavior when data is missing at this time.

augment.betareg 11

Usage

```
## S3 method for class 'betareg'
augment(
    x,
    data = model.frame(x),
    newdata = NULL,
    type.predict,
    type.residuals,
    ...
)
```

Arguments

A betareg object produced by a call to betareg::betareg().

data A base::data.frame or tibble::tibble() containing the original data that was

used to produce the object x. Defaults to stats::model.frame(x) so that $augment(my_fit)$ returns the augmented original data. **Do not** pass new data to the data argument. Augment will report information such as influence and cooks distance for data passed to the data argument. These measures are only

defined for the original training data.

newdata A base::data.frame() or tibble::tibble() containing all the original predictors used to create x. Defaults to NULL, indicating that nothing has been

passed to newdata. If newdata is specified, the data argument will be ignored.

type.predict Character indicating type of prediction to use. Passed to the type argument of the stats::predict() generic. Allowed arguments vary with model class, so

be sure to read the predict.my_class documentation.

type.residuals Character indicating type of residuals to use. Passed to the type argument of stats::residuals() generic. Allowed arguments vary with model class, so

be sure to read the residuals.my_class documentation.

Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in . . ., where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

Details

. . .

For additional details on Cook's distance, see stats::cooks.distance().

Value

```
A tibble::tibble() with columns:
```

12 augment.clm

. cooksd Cooks distance.

.fitted Fitted or predicted value.

.resid The difference between observed and fitted values.

See Also

```
augment(), betareg()
```

Examples

```
# load libraries for models and data
library(betareg)

# load dats
data("GasolineYield", package = "betareg")

# fit model
mod <- betareg(yield ~ batch + temp, data = GasolineYield)

mod

# summarize model fit with tidiers
tidy(mod)
tidy(mod, conf.int = TRUE)
tidy(mod, conf.int = TRUE, conf.level = .99)
augment(mod)
glance(mod)</pre>
```

augment.clm

Augment data with information from a(n) clm object

Description

Augment accepts a model object and a dataset and adds information about each observation in the dataset. Most commonly, this includes predicted values in the .fitted column, residuals in the .resid column, and standard errors for the fitted values in a .se.fit column. New columns always begin with a . prefix to avoid overwriting columns in the original dataset.

Users may pass data to augment via either the data argument or the newdata argument. If the user passes data to the data argument, it **must** be exactly the data that was used to fit the model object. Pass datasets to newdata to augment data that was not used during model fitting. This still requires that at least all predictor variable columns used to fit the model are present. If the original outcome variable used to fit the model is not included in newdata, then no .resid column will be included in the output.

13 augment.clm

Augment will often behave differently depending on whether data or newdata is given. This is because there is often information associated with training observations (such as influences or related) measures that is not meaningfully defined for new observations.

For convenience, many augment methods provide default data arguments, so that augment(fit) will return the augmented training data. In these cases, augment tries to reconstruct the original data based on the model object with varying degrees of success.

The augmented dataset is always returned as a tibble::tibble with the same number of rows as the passed dataset. This means that the passed data must be coercible to a tibble. If a predictor enters the model as part of a matrix of covariates, such as when the model formula uses splines::ns(), stats::poly(), or survival::Surv(), it is represented as a matrix column.

We are in the process of defining behaviors for models fit with various na.action arguments, but make no guarantees about behavior when data is missing at this time.

Usage

```
## S3 method for class 'clm'
augment(
  Х,
  data = model.frame(x),
  newdata = NULL,
  type.predict = c("prob", "class"),
)
```

Arguments

A clm object returned from ordinal::clm(). X

A base::data.frame or tibble::tibble() containing the original data that was data

> used to produce the object x. Defaults to stats::model.frame(x) so that augment(my_fit) returns the augmented original data. **Do not** pass new data to the data argument. Augment will report information such as influence and cooks distance for data passed to the data argument. These measures are only

defined for the original training data.

newdata A base::data.frame() or tibble::tibble() containing all the original predictors used to create x. Defaults to NULL, indicating that nothing has been

passed to newdata. If newdata is specified, the data argument will be ignored.

Which type of prediction to compute, either "prob" or "class", passed to ordinal::predict.clm(). Defaults to "prob".

> Additional arguments. Not used. Needed to match generic signature only. Cautionary note: Misspelled arguments will be absorbed in . . . , where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

type.predict

14 augment.coxph

See Also

```
tidy, ordinal::clm(), ordinal::predict.clm()
Other ordinal tidiers: augment.polr(), glance.clm(), glance.clmm(), glance.polr(), glance.svyolr(),
tidy.clm(), tidy.clmm(), tidy.polr(), tidy.svyolr()
```

Examples

```
# load libraries for models and data
library(ordinal)

# fit model
fit <- clm(rating ~ temp * contact, data = wine)

# summarize model fit with tidiers
tidy(fit)
tidy(fit, conf.int = TRUE, conf.level = 0.9)
tidy(fit, conf.int = TRUE, conf.type = "Wald", exponentiate = TRUE)

glance(fit)
augment(fit, type.predict = "prob")
augment(fit, type.predict = "class")

# ...and again with another model specification
fit2 <- clm(rating ~ temp, nominal = ~contact, data = wine)

tidy(fit2)
glance(fit2)</pre>
```

augment.coxph

Augment data with information from a(n) coxph object

Description

Augment accepts a model object and a dataset and adds information about each observation in the dataset. Most commonly, this includes predicted values in the .fitted column, residuals in the .resid column, and standard errors for the fitted values in a .se.fit column. New columns always begin with a . prefix to avoid overwriting columns in the original dataset.

Users may pass data to augment via either the data argument or the newdata argument. If the user passes data to the data argument, it **must** be exactly the data that was used to fit the model object. Pass datasets to newdata to augment data that was not used during model fitting. This still requires that at least all predictor variable columns used to fit the model are present. If the original outcome variable used to fit the model is not included in newdata, then no .resid column will be included in the output.

Augment will often behave differently depending on whether data or newdata is given. This is because there is often information associated with training observations (such as influences or related) measures that is not meaningfully defined for new observations.

augment.coxph 15

For convenience, many augment methods provide default data arguments, so that augment(fit) will return the augmented training data. In these cases, augment tries to reconstruct the original data based on the model object with varying degrees of success.

The augmented dataset is always returned as a tibble::tibble with the **same number of rows** as the passed dataset. This means that the passed data must be coercible to a tibble. If a predictor enters the model as part of a matrix of covariates, such as when the model formula uses splines::ns(), stats::poly(), or survival::Surv(), it is represented as a matrix column.

We are in the process of defining behaviors for models fit with various na.action arguments, but make no guarantees about behavior when data is missing at this time.

Usage

```
## S3 method for class 'coxph'
augment(
    x,
    data = model.frame(x),
    newdata = NULL,
    type.predict = "lp",
    type.residuals = "martingale",
    ...
)
```

Arguments

x A coxph object returned from survival::coxph().

data A base::data.frame or tibble::tibble() containing the original data that was

used to produce the object x. Defaults to stats::model.frame(x) so that augment(my_fit) returns the augmented original data. **Do not** pass new data to the data argument. Augment will report information such as influence and cooks distance for data passed to the data argument. These measures are only

defined for the original training data.

newdata A base::data.frame() or tibble::tibble() containing all the original pre-

dictors used to create x. Defaults to NULL, indicating that nothing has been passed to newdata. If newdata is specified, the data argument will be ignored.

type.predict Character indicating type of prediction to use. Passed to the type argument of the stats::predict() generic. Allowed arguments vary with model class, so

be sure to read the predict.my_class documentation.

type.residuals Character indicating type of residuals to use. Passed to the type argument of stats::residuals() generic. Allowed arguments vary with model class, so

be sure to read the residuals.my_class documentation.

Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in . . . , where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

• tidy() methods will warn when supplied an exponentiate argument if it will be ignored.

16 augment.coxph

• augment() methods will warn when supplied a newdata argument if it will be ignored.

Details

When the modeling was performed with na.action = "na.omit" (as is the typical default), rows with NA in the initial data are omitted entirely from the augmented data frame. When the modeling was performed with na.action = "na.exclude", one should provide the original data as a second argument, at which point the augmented data will contain those rows (typically with NAs in place of the new columns). If the original data is not provided to augment() and na.action = "na.exclude", a warning is raised and the incomplete rows are dropped.

Value

```
A tibble::tibble() with columns:

.fitted Fitted or predicted value.

.resid The difference between observed and fitted values.

.se.fit Standard errors of fitted values.
```

See Also

```
stats::na.action
augment(), survival::coxph()
Other coxph tidiers: glance.coxph(), tidy.coxph()
Other survival tidiers: augment.survreg(), glance.aareg(), glance.cch(), glance.coxph(),
glance.pyears(), glance.survdiff(), glance.survexp(), glance.survfit(), glance.survreg(),
tidy.aareg(), tidy.cch(), tidy.coxph(), tidy.pyears(), tidy.survdiff(), tidy.survexp(),
tidy.survfit(), tidy.survreg()
```

Examples

```
# load libraries for models and data
library(survival)

# fit model
cfit <- coxph(Surv(time, status) ~ age + sex, lung)

# summarize model fit with tidiers
tidy(cfit)
tidy(cfit, exponentiate = TRUE)

lp <- augment(cfit, lung)
risks <- augment(cfit, lung, type.predict = "risk")
expected <- augment(cfit, lung, type.predict = "expected")
glance(cfit)

# also works on clogit models</pre>
```

```
resp <- levels(logan$occupation)</pre>
n <- nrow(logan)</pre>
indx <- rep(1:n, length(resp))</pre>
logan2 <- data.frame(</pre>
  logan[indx, ],
  id = indx,
  tocc = factor(rep(resp, each = n))
logan2$case <- (logan2$occupation == logan2$tocc)</pre>
cl <- clogit(case ~ tocc + tocc:education + strata(id), logan2)</pre>
tidy(cl)
glance(cl)
library(ggplot2)
ggplot(lp, aes(age, .fitted, color = sex)) +
  geom_point()
ggplot(risks, aes(age, .fitted, color = sex)) +
  geom_point()
ggplot(expected, aes(time, .fitted, color = sex)) +
  geom_point()
```

augment.decomposed.ts Augment data with information from a(n) decomposed.ts object

Description

Augment accepts a model object and a dataset and adds information about each observation in the dataset. Most commonly, this includes predicted values in the .fitted column, residuals in the .resid column, and standard errors for the fitted values in a .se.fit column. New columns always begin with a . prefix to avoid overwriting columns in the original dataset.

Users may pass data to augment via either the data argument or the newdata argument. If the user passes data to the data argument, it **must** be exactly the data that was used to fit the model object. Pass datasets to newdata to augment data that was not used during model fitting. This still requires that at least all predictor variable columns used to fit the model are present. If the original outcome variable used to fit the model is not included in newdata, then no .resid column will be included in the output.

Augment will often behave differently depending on whether data or newdata is given. This is because there is often information associated with training observations (such as influences or related) measures that is not meaningfully defined for new observations.

For convenience, many augment methods provide default data arguments, so that augment(fit) will return the augmented training data. In these cases, augment tries to reconstruct the original data based on the model object with varying degrees of success.

The augmented dataset is always returned as a tibble::tibble with the **same number of rows** as the passed dataset. This means that the passed data must be coercible to a tibble. If a predictor enters the model as part of a matrix of covariates, such as when the model formula uses splines::ns(), stats::poly(), or survival::Surv(), it is represented as a matrix column.

We are in the process of defining behaviors for models fit with various na.action arguments, but make no guarantees about behavior when data is missing at this time.

Usage

```
## S3 method for class 'decomposed.ts'
augment(x, ...)
```

Arguments

. . .

x A decomposed.ts object returned from stats::decompose().

Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in . . . , where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

Value

A tibble::tibble with one row for each observation in the original times series:

. seasonal The seasonal component of the decomposition.. trend The trend component of the decomposition.

remainder The remainder, or "random" component of the decomposition.

. weight The final robust weights (stl only).

. seasadj The seasonally adjusted (or "deseasonalised") series.

See Also

```
augment(), stats::decompose()
Other decompose tidiers: augment.stl()
```

Examples

```
# time series of temperatures in Nottingham, 1920-1939:
nottem
# perform seasonal decomposition on the data with both decompose
# and stl:
```

augment.drc 19

```
d1 <- decompose(nottem)</pre>
d2 <- stl(nottem, s.window = "periodic", robust = TRUE)</pre>
# compare the original series to its decompositions.
cbind(
  tidy(nottem), augment(d1),
  augment(d2)
)
# visually compare seasonal decompositions in tidy data frames.
library(tibble)
library(dplyr)
library(tidyr)
library(ggplot2)
decomps <- tibble(</pre>
  # turn the ts objects into data frames.
  series = list(as.data.frame(nottem), as.data.frame(nottem)),
  # add the models in, one for each row.
  decomp = c("decompose", "stl"),
  model = list(d1, d2)
) %>%
  rowwise() %>%
  # pull out the fitted data using broom::augment.
  mutate(augment = list(broom::augment(model))) %>%
  ungroup() %>%
  # unnest the data frames into a tidy arrangement of
  # the series next to its seasonal decomposition, grouped
  # by the method (stl or decompose).
  group_by(decomp) %>%
  unnest(c(series, augment)) %>%
  mutate(index = 1:n()) %>%
  ungroup() %>%
  select(decomp, index, x, adjusted = .seasadj)
ggplot(decomps) +
  geom\_line(aes(x = index, y = x), colour = "black") +
  geom_line(aes(
   x = index, y = adjusted, colour = decomp,
   group = decomp
  ))
```

20 augment.drc

Description

Augment accepts a model object and a dataset and adds information about each observation in the dataset. Most commonly, this includes predicted values in the .fitted column, residuals in the .resid column, and standard errors for the fitted values in a .se.fit column. New columns always begin with a . prefix to avoid overwriting columns in the original dataset.

Users may pass data to augment via either the data argument or the newdata argument. If the user passes data to the data argument, it **must** be exactly the data that was used to fit the model object. Pass datasets to newdata to augment data that was not used during model fitting. This still requires that at least all predictor variable columns used to fit the model are present. If the original outcome variable used to fit the model is not included in newdata, then no .resid column will be included in the output.

Augment will often behave differently depending on whether data or newdata is given. This is because there is often information associated with training observations (such as influences or related) measures that is not meaningfully defined for new observations.

For convenience, many augment methods provide default data arguments, so that augment(fit) will return the augmented training data. In these cases, augment tries to reconstruct the original data based on the model object with varying degrees of success.

The augmented dataset is always returned as a tibble::tibble with the **same number of rows** as the passed dataset. This means that the passed data must be coercible to a tibble. If a predictor enters the model as part of a matrix of covariates, such as when the model formula uses splines::ns(), stats::poly(), or survival::Surv(), it is represented as a matrix column.

We are in the process of defining behaviors for models fit with various na.action arguments, but make no guarantees about behavior when data is missing at this time.

Usage

```
## S3 method for class 'drc'
augment(
    x,
    data = NULL,
    newdata = NULL,
    se_fit = FALSE,
    conf.int = FALSE,
    conf.level = 0.95,
    ...
)
```

Arguments

A drc object produced by a call to drc::drm().

data

A base::data.frame or tibble::tibble() containing the original data that was used to produce the object x. Defaults to stats::model.frame(x) so that augment(my_fit) returns the augmented original data. **Do not** pass new data to the data argument. Augment will report information such as influence and cooks distance for data passed to the data argument. These measures are only defined for the original training data.

augment.drc 21

newdata	A base::data.frame() or tibble::tibble() containing all the original predictors used to create x. Defaults to NULL, indicating that nothing has been passed to newdata. If newdata is specified, the data argument will be ignored.
se_fit	Logical indicating whether or not a .se.fit column should be added to the augmented output. For some models, this calculation can be somewhat time-consuming. Defaults to FALSE.
conf.int	Logical indicating whether or not to include a confidence interval in the tidied output. Defaults to FALSE.
conf.level	The confidence level to use for the confidence interval if conf.int = TRUE. Must be strictly greater than 0 and less than 1. Defaults to 0.95, which corresponds to a 95 percent confidence interval.
	Additional arguments. Not used. Needed to match generic signature only. Cautionary note: Misspelled arguments will be absorbed in , where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:
	• tidy() methods will warn when supplied an exponentiate argument if it will be ignored.

• augment() methods will warn when supplied a newdata argument if it will

Value

A tibble::tibble() with columns:

```
. cooksd
. fitted
. Fitted or predicted value.
. lower
. Lower bound on interval for fitted values.
. resid
. se.fit
. Standard errors of fitted values.
. upper
. Upper bound on interval for fitted values.
```

be ignored.

See Also

```
augment(), drc::drm()
Other drc tidiers: glance.drc(), tidy.drc()
```

Examples

```
# load libraries for models and data
library(drc)

# fit model
mod <- drm(dead / total ~ conc, type,
  weights = total, data = selenium, fct = LL.2(), type = "binomial"
)</pre>
```

22 augment.factanal

```
# summarize model fit with tidiers
tidy(mod)
tidy(mod, conf.int = TRUE)
glance(mod)
augment(mod, selenium)
```

augment.factanal

Augment data with information from a(n) factanal object

Description

Augment accepts a model object and a dataset and adds information about each observation in the dataset. Most commonly, this includes predicted values in the .fitted column, residuals in the .resid column, and standard errors for the fitted values in a .se.fit column. New columns always begin with a . prefix to avoid overwriting columns in the original dataset.

Users may pass data to augment via either the data argument or the newdata argument. If the user passes data to the data argument, it **must** be exactly the data that was used to fit the model object. Pass datasets to newdata to augment data that was not used during model fitting. This still requires that at least all predictor variable columns used to fit the model are present. If the original outcome variable used to fit the model is not included in newdata, then no .resid column will be included in the output.

Augment will often behave differently depending on whether data or newdata is given. This is because there is often information associated with training observations (such as influences or related) measures that is not meaningfully defined for new observations.

For convenience, many augment methods provide default data arguments, so that augment(fit) will return the augmented training data. In these cases, augment tries to reconstruct the original data based on the model object with varying degrees of success.

The augmented dataset is always returned as a tibble::tibble with the **same number of rows** as the passed dataset. This means that the passed data must be coercible to a tibble. If a predictor enters the model as part of a matrix of covariates, such as when the model formula uses splines::ns(), stats::poly(), or survival::Surv(), it is represented as a matrix column.

We are in the process of defining behaviors for models fit with various na. action arguments, but make no guarantees about behavior when data is missing at this time.

Usage

```
## S3 method for class 'factanal'
augment(x, data, ...)
```

augment.felm 23

Arguments

Х

A factanal object created by stats::factanal().

data

A base::data.frame or tibble::tibble() containing the original data that was used to produce the object x. Defaults to stats::model.frame(x) so that augment(my_fit) returns the augmented original data. **Do not** pass new data to the data argument. Augment will report information such as influence and cooks distance for data passed to the data argument. These measures are only defined for the original training data.

Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in . . . , where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

Value

When data is not supplied augment.factanal returns one row for each observation, with a factor score column added for each factor X, (.fsX). This is because $\mathtt{stats::factanal}()$, unlike other stats methods like $\mathtt{stats::lm}()$, does not retain the original data.

When data is supplied, augment.factanal returns one row for each observation, with a factor score column added for each factor X, (.fsX).

See Also

```
augment(), stats::factanal()
Other factanal tidiers: glance.factanal(), tidy.factanal()
```

augment.felm

Augment data with information from a(n) felm object

Description

Augment accepts a model object and a dataset and adds information about each observation in the dataset. Most commonly, this includes predicted values in the .fitted column, residuals in the .resid column, and standard errors for the fitted values in a .se.fit column. New columns always begin with a . prefix to avoid overwriting columns in the original dataset.

Users may pass data to augment via either the data argument or the newdata argument. If the user passes data to the data argument, it **must** be exactly the data that was used to fit the model object. Pass datasets to newdata to augment data that was not used during model fitting. This still requires that at least all predictor variable columns used to fit the model are present. If the original outcome

24 augment.felm

variable used to fit the model is not included in newdata, then no .resid column will be included in the output.

Augment will often behave differently depending on whether data or newdata is given. This is because there is often information associated with training observations (such as influences or related) measures that is not meaningfully defined for new observations.

For convenience, many augment methods provide default data arguments, so that augment(fit) will return the augmented training data. In these cases, augment tries to reconstruct the original data based on the model object with varying degrees of success.

The augmented dataset is always returned as a tibble::tibble with the **same number of rows** as the passed dataset. This means that the passed data must be coercible to a tibble. If a predictor enters the model as part of a matrix of covariates, such as when the model formula uses splines::ns(), stats::poly(), or survival::Surv(), it is represented as a matrix column.

We are in the process of defining behaviors for models fit with various na.action arguments, but make no guarantees about behavior when data is missing at this time.

Usage

```
## S3 method for class 'felm'
augment(x, data = model.frame(x), ...)
```

Arguments

Χ

A felm object returned from lfe::felm().

data

A base::data.frame or tibble::tibble() containing the original data that was used to produce the object x. Defaults to stats::model.frame(x) so that augment(my_fit) returns the augmented original data. **Do not** pass new data to the data argument. Augment will report information such as influence and cooks distance for data passed to the data argument. These measures are only defined for the original training data.

. . .

Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in . . . , where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

Value

A tibble::tibble() with columns:

.fitted Fitted or predicted value.

. resid The difference between observed and fitted values.

augment.fixest 25

See Also

```
augment(), lfe::felm()
Other felm tidiers: tidy.felm()
```

Examples

```
# load libraries for models and data
library(lfe)
# use built-in `airquality` dataset
head(airquality)
# no FEs; same as lm()
est0 <- felm(Ozone ~ Temp + Wind + Solar.R, airquality)</pre>
# summarize model fit with tidiers
tidy(est0)
augment(est0)
# add month fixed effects
est1 <- felm(Ozone ~ Temp + Wind + Solar.R | Month, airquality)</pre>
# summarize model fit with tidiers
tidy(est1)
tidy(est1, fe = TRUE)
augment(est1)
glance(est1)
# the "se.type" argument can be used to switch out different standard errors
# types on the fly. In turn, this can be useful exploring the effect of
# different error structures on model inference.
tidy(est1, se.type = "iid")
tidy(est1, se.type = "robust")
# add clustered SEs (also by month)
est2 <- felm(Ozone ~ Temp + Wind + Solar.R | Month | 0 | Month, airquality)</pre>
# summarize model fit with tidiers
tidy(est2, conf.int = TRUE)
tidy(est2, conf.int = TRUE, se.type = "cluster")
tidy(est2, conf.int = TRUE, se.type = "robust")
tidy(est2, conf.int = TRUE, se.type = "iid")
```

26 augment.fixest

Description

Augment accepts a model object and a dataset and adds information about each observation in the dataset. Most commonly, this includes predicted values in the .fitted column, residuals in the .resid column, and standard errors for the fitted values in a .se.fit column. New columns always begin with a . prefix to avoid overwriting columns in the original dataset.

Users may pass data to augment via either the data argument or the newdata argument. If the user passes data to the data argument, it **must** be exactly the data that was used to fit the model object. Pass datasets to newdata to augment data that was not used during model fitting. This still requires that at least all predictor variable columns used to fit the model are present. If the original outcome variable used to fit the model is not included in newdata, then no .resid column will be included in the output.

Augment will often behave differently depending on whether data or newdata is given. This is because there is often information associated with training observations (such as influences or related) measures that is not meaningfully defined for new observations.

For convenience, many augment methods provide default data arguments, so that augment(fit) will return the augmented training data. In these cases, augment tries to reconstruct the original data based on the model object with varying degrees of success.

The augmented dataset is always returned as a tibble::tibble with the **same number of rows** as the passed dataset. This means that the passed data must be coercible to a tibble. If a predictor enters the model as part of a matrix of covariates, such as when the model formula uses splines::ns(), stats::poly(), or survival::Surv(), it is represented as a matrix column.

We are in the process of defining behaviors for models fit with various na.action arguments, but make no guarantees about behavior when data is missing at this time.

Usage

```
## S3 method for class 'fixest'
augment(
    x,
    data = NULL,
    newdata = NULL,
    type.predict = c("link", "response"),
    type.residuals = c("response", "deviance", "pearson", "working"),
    ...
)
```

Arguments

x A fixest object returned from any of the fixest estimators

data

A base::data.frame or tibble::tibble() containing the original data that was used to produce the object x. Defaults to stats::model.frame(x) so that augment(my_fit) returns the augmented original data. **Do not** pass new data to the data argument. Augment will report information such as influence and cooks distance for data passed to the data argument. These measures are only defined for the original training data.

augment.fixest 27

```
A base::data.frame() or tibble::tibble() containing all the original predictors used to create x. Defaults to NULL, indicating that nothing has been passed to newdata. If newdata is specified, the data argument will be ignored.

type.predict Passed to predict.fixest type argument. Defaults to "link" (like predict.glm).

type.residuals Passed to predict.fixest type argument. Defaults to "response" (like residuals.lm, but unlike residuals.glm).

Additional arguments passed to summary and confint. Important arguments are se and cluster. Other arguments are dof, exact_dof, forceCovariance, and keepBounded. See summary.fixest.
```

Value

A tibble::tibble() with columns:

.fitted Fitted or predicted value.

resid The difference between observed and fitted values.

Note

Important note: fixest models do not include a copy of the input data, so you must provide it manually.

```
augment.fixest only works for fixest::feols(), fixest::feglm(), and fixest::femlm() mod-
els. It does not work with results from fixest::fenegbin(), fixest::feNmlm(), or fixest::fepois().
```

See Also

```
augment(), fixest::feglm(), fixest::femlm(), fixest::feols()
Other fixest tidiers: tidy.fixest()
```

Examples

```
# load libraries for models and data
library(fixest)

gravity <-
    feols(
        log(Euros) ~ log(dist_km) | Origin + Destination + Product + Year, trade
    )

tidy(gravity)
glance(gravity)
augment(gravity, trade)

# to get robust or clustered SEs, users can either:
# 1) specify the arguments directly in the `tidy()` call

tidy(gravity, conf.int = TRUE, cluster = c("Product", "Year"))</pre>
```

28 augment.gam

```
tidy(gravity, conf.int = TRUE, se = "threeway")
# 2) or, feed tidy() a summary.fixest object that has already accepted
# these arguments
gravity_summ <- summary(gravity, cluster = c("Product", "Year"))
tidy(gravity_summ, conf.int = TRUE)
# approach (1) is preferred.</pre>
```

augment.gam

Augment data with information from a(n) gam object

Description

Augment accepts a model object and a dataset and adds information about each observation in the dataset. Most commonly, this includes predicted values in the .fitted column, residuals in the .resid column, and standard errors for the fitted values in a .se.fit column. New columns always begin with a . prefix to avoid overwriting columns in the original dataset.

Users may pass data to augment via either the data argument or the newdata argument. If the user passes data to the data argument, it **must** be exactly the data that was used to fit the model object. Pass datasets to newdata to augment data that was not used during model fitting. This still requires that at least all predictor variable columns used to fit the model are present. If the original outcome variable used to fit the model is not included in newdata, then no .resid column will be included in the output.

Augment will often behave differently depending on whether data or newdata is given. This is because there is often information associated with training observations (such as influences or related) measures that is not meaningfully defined for new observations.

For convenience, many augment methods provide default data arguments, so that augment(fit) will return the augmented training data. In these cases, augment tries to reconstruct the original data based on the model object with varying degrees of success.

The augmented dataset is always returned as a tibble::tibble with the **same number of rows** as the passed dataset. This means that the passed data must be coercible to a tibble. If a predictor enters the model as part of a matrix of covariates, such as when the model formula uses splines::ns(), stats::poly(), or survival::Surv(), it is represented as a matrix column.

We are in the process of defining behaviors for models fit with various na.action arguments, but make no guarantees about behavior when data is missing at this time.

Usage

```
## S3 method for class 'gam'
augment(
    x,
```

augment.gam 29

```
data = model.frame(x),
newdata = NULL,
type.predict,
type.residuals,
...
)
```

Arguments

A gam object returned from a call to mgcv::gam().

data A base::data.frame or tibble::tibble() containing the original data that was

used to produce the object x. Defaults to stats::model.frame(x) so that augment(my_fit) returns the augmented original data. **Do not** pass new data to the data argument. Augment will report information such as influence and cooks distance for data passed to the data argument. These measures are only

defined for the original training data.

newdata A base::data.frame() or tibble::tibble() containing all the original pre-

dictors used to create x. Defaults to NULL, indicating that nothing has been passed to newdata. If newdata is specified, the data argument will be ignored.

type.predict Character indicating type of prediction to use. Passed to the type argument of

the stats::predict() generic. Allowed arguments vary with model class, so

be sure to read the predict.my_class documentation.

type.residuals Character indicating type of residuals to use. Passed to the type argument of stats::residuals() generic. Allowed arguments vary with model class, so

be sure to read the $residuals.my_class$ documentation.

Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in . . ., where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

Details

For additional details on Cook's distance, see stats::cooks.distance().

Value

A tibble::tibble() with columns:

. cooksd Cooks distance.

.fitted Fitted or predicted value..hat Diagonal of the hat matrix.

resid The difference between observed and fitted values.

30 augment.glm

```
. se.fit Standard errors of fitted values.. sigma Estimated residual standard deviation when corresponding observation is dropped from model.
```

See Also

```
augment(), mgcv::gam()
```

Examples

```
# load libraries for models and data
library(mgcv)

# fit model
g <- gam(mpg ~ s(hp) + am + qsec, data = mtcars)

# summarize model fit with tidiers
tidy(g)
tidy(g, parametric = TRUE)
glance(g)
augment(g)</pre>
```

augment.glm

Augment data with information from a(n) glm object

Description

Augment accepts a model object and a dataset and adds information about each observation in the dataset. Most commonly, this includes predicted values in the .fitted column, residuals in the .resid column, and standard errors for the fitted values in a .se.fit column. New columns always begin with a . prefix to avoid overwriting columns in the original dataset.

Users may pass data to augment via either the data argument or the newdata argument. If the user passes data to the data argument, it **must** be exactly the data that was used to fit the model object. Pass datasets to newdata to augment data that was not used during model fitting. This still requires that at least all predictor variable columns used to fit the model are present. If the original outcome variable used to fit the model is not included in newdata, then no .resid column will be included in the output.

Augment will often behave differently depending on whether data or newdata is given. This is because there is often information associated with training observations (such as influences or related) measures that is not meaningfully defined for new observations.

For convenience, many augment methods provide default data arguments, so that augment(fit) will return the augmented training data. In these cases, augment tries to reconstruct the original data based on the model object with varying degrees of success.

The augmented dataset is always returned as a tibble::tibble with the **same number of rows** as the passed dataset. This means that the passed data must be coercible to a tibble. If a predictor enters

augment.glm 31

the model as part of a matrix of covariates, such as when the model formula uses splines::ns(), stats::poly(), or survival::Surv(), it is represented as a matrix column.

We are in the process of defining behaviors for models fit with various na.action arguments, but make no guarantees about behavior when data is missing at this time.

Usage

```
## S3 method for class 'glm'
augment(
    x,
    data = model.frame(x),
    newdata = NULL,
    type.predict = c("link", "response", "terms"),
    type.residuals = c("deviance", "pearson"),
    se_fit = FALSE,
    ...
)
```

Arguments

x A glm object returned from stats::glm().

data A base::data.frame or tibble::tibble() containing the original data that was

used to produce the object x. Defaults to stats::model.frame(x) so that augment(my_fit) returns the augmented original data. **Do not** pass new data to the data argument. Augment will report information such as influence and cooks distance for data passed to the data argument. These measures are only

defined for the original training data.

newdata A base::data.frame() or tibble::tibble() containing all the original pre-

dictors used to create x. Defaults to NULL, indicating that nothing has been passed to newdata. If newdata is specified, the data argument will be ignored.

type.predict Passed to stats::predict.glm() type argument. Defaults to "link".

using conf.level = 0.95. Two exceptions here are:

type.residuals Passed to stats::residuals.glm() and to stats::rstandard.glm() type

arguments. Defaults to "deviance".

se_fit Logical indicating whether or not a .se.fit column should be added to the augmented output. For some models, this calculation can be somewhat time-

consuming. Defaults to FALSE.

Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in . . . , where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

32 augment.glmRob

Details

If the weights for any of the observations in the model are 0, then columns ".infl" and ".hat" in the result will be 0 for those observations.

A .resid column is not calculated when data is specified via the newdata argument.

Value

A tibble::tibble() with columns:

.cooksd
.fitted
.fitted or predicted value.
.hat
.hat
.piagonal of the hat matrix.
.resid
.se.fit
.standard errors of fitted values.
.sigma
.sigma
Estimated residual standard deviation when corresponding observation is dropped from model.
.std.resid
.std.resid
Standardised residuals.

See Also

```
stats::glm()
```

```
Other Im tidiers: augment.lm(), glance.glm(), glance.lm(), glance.summary.lm(), glance.svyglm(), tidy.glm(), tidy.lm(), tidy.lm(), tidy.lm(), tidy.summary.lm()
```

augment.glmRob

Augment data with information from a(n) glmRob object

Description

Augment accepts a model object and a dataset and adds information about each observation in the dataset. Most commonly, this includes predicted values in the .fitted column, residuals in the .resid column, and standard errors for the fitted values in a .se.fit column. New columns always begin with a . prefix to avoid overwriting columns in the original dataset.

Users may pass data to augment via either the data argument or the newdata argument. If the user passes data to the data argument, it **must** be exactly the data that was used to fit the model object. Pass datasets to newdata to augment data that was not used during model fitting. This still requires that at least all predictor variable columns used to fit the model are present. If the original outcome variable used to fit the model is not included in newdata, then no .resid column will be included in the output.

Augment will often behave differently depending on whether data or newdata is given. This is because there is often information associated with training observations (such as influences or related) measures that is not meaningfully defined for new observations.

augment.glmrob 33

For convenience, many augment methods provide default data arguments, so that augment(fit) will return the augmented training data. In these cases, augment tries to reconstruct the original data based on the model object with varying degrees of success.

The augmented dataset is always returned as a tibble::tibble with the **same number of rows** as the passed dataset. This means that the passed data must be coercible to a tibble. If a predictor enters the model as part of a matrix of covariates, such as when the model formula uses splines::ns(), stats::poly(), or survival::Surv(), it is represented as a matrix column.

We are in the process of defining behaviors for models fit with various na.action arguments, but make no guarantees about behavior when data is missing at this time.

Usage

```
## S3 method for class 'glmRob' augment(x, \ldots)
```

Arguments

x Unused. . . . Unused.

augment.glmrob

Augment data with information from a(n) glmrob object

Description

Augment accepts a model object and a dataset and adds information about each observation in the dataset. Most commonly, this includes predicted values in the .fitted column, residuals in the .resid column, and standard errors for the fitted values in a .se.fit column. New columns always begin with a . prefix to avoid overwriting columns in the original dataset.

Users may pass data to augment via either the data argument or the newdata argument. If the user passes data to the data argument, it **must** be exactly the data that was used to fit the model object. Pass datasets to newdata to augment data that was not used during model fitting. This still requires that at least all predictor variable columns used to fit the model are present. If the original outcome variable used to fit the model is not included in newdata, then no .resid column will be included in the output.

Augment will often behave differently depending on whether data or newdata is given. This is because there is often information associated with training observations (such as influences or related) measures that is not meaningfully defined for new observations.

For convenience, many augment methods provide default data arguments, so that augment(fit) will return the augmented training data. In these cases, augment tries to reconstruct the original data based on the model object with varying degrees of success.

The augmented dataset is always returned as a tibble::tibble with the **same number of rows** as the passed dataset. This means that the passed data must be coercible to a tibble. If a predictor enters the model as part of a matrix of covariates, such as when the model formula uses splines::ns(), stats::poly(), or survival::Surv(), it is represented as a matrix column.

We are in the process of defining behaviors for models fit with various na.action arguments, but make no guarantees about behavior when data is missing at this time.

34 augment.glmrob

Usage

```
## S3 method for class 'glmrob'
augment(
  х,
  data = model.frame(x),
  newdata = NULL,
  type.predict = c("link", "response"),
  type.residuals = c("deviance", "pearson"),
  se_fit = FALSE,
)
```

Arguments

A glmrob object returned from robustbase::glmrob(). Х

data A base::data.frame or tibble::tibble() containing the original data that was

> used to produce the object x. Defaults to stats::model.frame(x) so that augment(my_fit) returns the augmented original data. Do not pass new data to the data argument. Augment will report information such as influence and cooks distance for data passed to the data argument. These measures are only

defined for the original training data.

newdata A base::data.frame() or tibble::tibble() containing all the original pre-

dictors used to create x. Defaults to NULL, indicating that nothing has been passed to newdata. If newdata is specified, the data argument will be ignored.

Character indicating type of prediction to use. Passed to the type argument of type.predict the stats::predict() generic. Allowed arguments vary with model class, so

be sure to read the predict.my_class documentation.

type.residuals Character indicating type of residuals to use. Passed to the type argument of stats::residuals() generic. Allowed arguments vary with model class, so

be sure to read the residuals.my_class documentation.

se_fit Logical indicating whether or not a .se.fit column should be added to the

augmented output. For some models, this calculation can be somewhat timeconsuming. Defaults to FALSE.

Additional arguments. Not used. Needed to match generic signature only. Cautionary note: Misspelled arguments will be absorbed in . . . , where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed

using conf.level = 0.95. Two exceptions here are:

• tidy() methods will warn when supplied an exponentiate argument if it will be ignored.

• augment() methods will warn when supplied a newdata argument if it will be ignored.

Details

For tidiers for robust models from the MASS package see tidy.rlm().

augment.htest 35

Value

```
A tibble::tibble() with columns:

.fitted Fitted or predicted value.

.resid The difference between observed and fitted values.
```

See Also

```
robustbase::glmrob()
Other robustbase tidiers: augment.lmrob(), glance.lmrob(), tidy.glmrob(), tidy.lmrob()
```

Examples

```
if (requireNamespace("robustbase", quietly = TRUE)) {
 # load libraries for models and data
 library(robustbase)
 data(coleman)
 set.seed(0)
 m <- lmrob(Y ~ ., data = coleman)</pre>
 tidy(m)
 augment(m)
 glance(m)
 data(carrots)
 Rfit <- glmrob(cbind(success, total - success) ~ logdose + block,
    family = binomial, data = carrots, method = "Mqle",
    control = glmrobMqle.control(tcc = 1.2)
 tidy(Rfit)
 augment(Rfit)
}
```

augment.htest

Augment data with information from a(n) htest object

Description

Augment accepts a model object and a dataset and adds information about each observation in the dataset. Most commonly, this includes predicted values in the .fitted column, residuals in the .resid column, and standard errors for the fitted values in a .se.fit column. New columns always begin with a . prefix to avoid overwriting columns in the original dataset.

Users may pass data to augment via either the data argument or the newdata argument. If the user passes data to the data argument, it **must** be exactly the data that was used to fit the model object. Pass datasets to newdata to augment data that was not used during model fitting. This still requires

36 augment.htest

that at least all predictor variable columns used to fit the model are present. If the original outcome variable used to fit the model is not included in newdata, then no .resid column will be included in the output.

Augment will often behave differently depending on whether data or newdata is given. This is because there is often information associated with training observations (such as influences or related) measures that is not meaningfully defined for new observations.

For convenience, many augment methods provide default data arguments, so that augment(fit) will return the augmented training data. In these cases, augment tries to reconstruct the original data based on the model object with varying degrees of success.

The augmented dataset is always returned as a tibble::tibble with the **same number of rows** as the passed dataset. This means that the passed data must be coercible to a tibble. If a predictor enters the model as part of a matrix of covariates, such as when the model formula uses splines::ns(), stats::poly(), or survival::Surv(), it is represented as a matrix column.

We are in the process of defining behaviors for models fit with various na.action arguments, but make no guarantees about behavior when data is missing at this time.

Usage

```
## S3 method for class 'htest'
augment(x, ...)
```

Arguments

Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in . . . , where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

Details

See stats::chisq.test() for more details on how residuals are computed.

Value

A tibble::tibble() with exactly one row and columns:

.observed Observed count.
.prop Proportion of the total.
.row.prop Row proportion (2 dimensions table only).
.col.prop Column proportion (2 dimensions table only).

augment.ivreg 37

```
. expected Expected count under the null hypothesis.. resid Pearson residuals.. std.resid Standardized residual.
```

See Also

```
augment(), stats::chisq.test()
Other htest tidiers: tidy.htest(), tidy.pairwise.htest(), tidy.power.htest()
```

Examples

```
tt <- t.test(rnorm(10))
tidy(tt)

# the glance output will be the same for each of the below tests
glance(tt)

tt <- t.test(mpg ~ am, data = mtcars)
tidy(tt)

wt <- wilcox.test(mpg ~ am, data = mtcars, conf.int = TRUE, exact = FALSE)
tidy(wt)

ct <- cor.test(mtcars$wt, mtcars$mpg)
tidy(ct)
chit <- chisq.test(xtabs(Freq ~ Sex + Class, data = as.data.frame(Titanic)))
tidy(chit)
augment(chit)</pre>
```

augment.ivreg

Augment data with information from a(n) ivreg object

Description

Augment accepts a model object and a dataset and adds information about each observation in the dataset. Most commonly, this includes predicted values in the .fitted column, residuals in the .resid column, and standard errors for the fitted values in a .se.fit column. New columns always begin with a . prefix to avoid overwriting columns in the original dataset.

Users may pass data to augment via either the data argument or the newdata argument. If the user passes data to the data argument, it **must** be exactly the data that was used to fit the model object. Pass datasets to newdata to augment data that was not used during model fitting. This still requires

38 augment.ivreg

that at least all predictor variable columns used to fit the model are present. If the original outcome variable used to fit the model is not included in newdata, then no .resid column will be included in the output.

Augment will often behave differently depending on whether data or newdata is given. This is because there is often information associated with training observations (such as influences or related) measures that is not meaningfully defined for new observations.

For convenience, many augment methods provide default data arguments, so that augment(fit) will return the augmented training data. In these cases, augment tries to reconstruct the original data based on the model object with varying degrees of success.

The augmented dataset is always returned as a tibble::tibble with the **same number of rows** as the passed dataset. This means that the passed data must be coercible to a tibble. If a predictor enters the model as part of a matrix of covariates, such as when the model formula uses splines::ns(), stats::poly(), or survival::Surv(), it is represented as a matrix column.

We are in the process of defining behaviors for models fit with various na.action arguments, but make no guarantees about behavior when data is missing at this time.

Usage

```
## S3 method for class 'ivreg'
augment(x, data = model.frame(x), newdata = NULL, ...)
```

Arguments

Χ

An ivreg object created by a call to AER::ivreg().

data

A base::data.frame or tibble::tibble() containing the original data that was used to produce the object x. Defaults to stats::model.frame(x) so that augment(my_fit) returns the augmented original data. **Do not** pass new data to the data argument. Augment will report information such as influence and cooks distance for data passed to the data argument. These measures are only defined for the original training data.

newdata

A base::data.frame() or tibble::tibble() containing all the original predictors used to create x. Defaults to NULL, indicating that nothing has been passed to newdata. If newdata is specified, the data argument will be ignored.

. . .

Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in . . . , where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

Details

This tidier currently only supports ivreg-classed objects outputted by the AER package. The ivreg package also outputs objects of class ivreg, and will be supported in a later release.

augment.kmeans 39

Value

```
A tibble::tibble() with columns:

.fitted Fitted or predicted value.

.resid The difference between observed and fitted values.
```

See Also

```
augment(), AER::ivreg()
Other ivreg tidiers: glance.ivreg(), tidy.ivreg()
```

Examples

```
# load libraries for models and data
library(AER)
# load data
data("CigarettesSW", package = "AER")
# fit model
ivr <- ivreg(</pre>
 log(packs) ~ income | population,
 data = CigarettesSW,
 subset = year == "1995"
)
# summarize model fit with tidiers
tidy(ivr)
tidy(ivr, conf.int = TRUE)
tidy(ivr, conf.int = TRUE, instruments = TRUE)
augment(ivr)
augment(ivr, data = CigarettesSW)
augment(ivr, newdata = CigarettesSW)
glance(ivr)
```

 ${\it augment.kmeans}$

Augment data with information from a(n) kmeans object

Description

Augment accepts a model object and a dataset and adds information about each observation in the dataset. Most commonly, this includes predicted values in the .fitted column, residuals in the .resid column, and standard errors for the fitted values in a .se.fit column. New columns always begin with a . prefix to avoid overwriting columns in the original dataset.

40 augment.kmeans

Users may pass data to augment via either the data argument or the newdata argument. If the user passes data to the data argument, it **must** be exactly the data that was used to fit the model object. Pass datasets to newdata to augment data that was not used during model fitting. This still requires that at least all predictor variable columns used to fit the model are present. If the original outcome variable used to fit the model is not included in newdata, then no .resid column will be included in the output.

Augment will often behave differently depending on whether data or newdata is given. This is because there is often information associated with training observations (such as influences or related) measures that is not meaningfully defined for new observations.

For convenience, many augment methods provide default data arguments, so that augment(fit) will return the augmented training data. In these cases, augment tries to reconstruct the original data based on the model object with varying degrees of success.

The augmented dataset is always returned as a tibble::tibble with the **same number of rows** as the passed dataset. This means that the passed data must be coercible to a tibble. If a predictor enters the model as part of a matrix of covariates, such as when the model formula uses splines::ns(), stats::poly(), or survival::Surv(), it is represented as a matrix column.

We are in the process of defining behaviors for models fit with various na.action arguments, but make no guarantees about behavior when data is missing at this time.

Usage

```
## S3 method for class 'kmeans'
augment(x, data, ...)
```

Arguments

Χ

A kmeans object created by stats::kmeans().

data

A base::data.frame or tibble::tibble() containing the original data that was used to produce the object x. Defaults to stats::model.frame(x) so that augment(my_fit) returns the augmented original data. **Do not** pass new data to the data argument. Augment will report information such as influence and cooks distance for data passed to the data argument. These measures are only defined for the original training data.

Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in . . . , where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

Value

A tibble::tibble() with columns:
.cluster Cluster assignment.

See Also

```
augment(), stats::kmeans()
Other kmeans tidiers: glance.kmeans(), tidy.kmeans()
```

Examples

```
library(cluster)
library(modeldata)
library(dplyr)

data(hpc_data)

x <- hpc_data[, 2:5]

fit <- pam(x, k = 4)

tidy(fit)
glance(fit)
augment(fit, x)</pre>
```

augment.lm

Augment data with information from a(n) lm object

Description

Augment accepts a model object and a dataset and adds information about each observation in the dataset. Most commonly, this includes predicted values in the .fitted column, residuals in the .resid column, and standard errors for the fitted values in a .se.fit column. New columns always begin with a . prefix to avoid overwriting columns in the original dataset.

Users may pass data to augment via either the data argument or the newdata argument. If the user passes data to the data argument, it **must** be exactly the data that was used to fit the model object. Pass datasets to newdata to augment data that was not used during model fitting. This still requires that at least all predictor variable columns used to fit the model are present. If the original outcome variable used to fit the model is not included in newdata, then no .resid column will be included in the output.

Augment will often behave differently depending on whether data or newdata is given. This is because there is often information associated with training observations (such as influences or related) measures that is not meaningfully defined for new observations.

For convenience, many augment methods provide default data arguments, so that augment(fit) will return the augmented training data. In these cases, augment tries to reconstruct the original data based on the model object with varying degrees of success.

The augmented dataset is always returned as a tibble::tibble with the **same number of rows** as the passed dataset. This means that the passed data must be coercible to a tibble. If a predictor enters

the model as part of a matrix of covariates, such as when the model formula uses splines::ns(), stats::poly(), or survival::Surv(), it is represented as a matrix column.

We are in the process of defining behaviors for models fit with various na.action arguments, but make no guarantees about behavior when data is missing at this time.

Usage

```
## S3 method for class 'lm'
augment(
  х,
  data = model.frame(x),
  newdata = NULL,
  se_fit = FALSE,
  interval = c("none", "confidence", "prediction"),
  conf.level = 0.95,
)
```

Arguments

An 1m object created by stats::lm(). Х

data A base::data.frame or tibble::tibble() containing the original data that was used to produce the object x. Defaults to stats::model.frame(x) so that augment(my_fit) returns the augmented original data. Do not pass new data to the data argument. Augment will report information such as influence and

defined for the original training data.

A base::data.frame() or tibble::tibble() containing all the original prenewdata

dictors used to create x. Defaults to NULL, indicating that nothing has been passed to newdata. If newdata is specified, the data argument will be ignored.

cooks distance for data passed to the data argument. These measures are only

se_fit Logical indicating whether or not a .se.fit column should be added to the augmented output. For some models, this calculation can be somewhat time-

consuming. Defaults to FALSE.

interval Character indicating the type of confidence interval columns to be added to the

augmented output. Passed on to predict() and defaults to "none".

The confidence level to use for the interval created if interval is "confidence" or "prediction". Must be strictly greater than 0 and less than 1. Defaults to 0.95, which corresponds to a 95 percent confidence/prediction interval.

Additional arguments. Not used. Needed to match generic signature only. Cautionary note: Misspelled arguments will be absorbed in ..., where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

conf.level

Details

When the modeling was performed with na.action = "na.omit" (as is the typical default), rows with NA in the initial data are omitted entirely from the augmented data frame. When the modeling was performed with na.action = "na.exclude", one should provide the original data as a second argument, at which point the augmented data will contain those rows (typically with NAs in place of the new columns). If the original data is not provided to augment() and na.action = "na.exclude", a warning is raised and the incomplete rows are dropped.

Some unusual lm objects, such as rlm from MASS, may omit .cooksd and .std.resid. gam from mgcv omits .sigma.

When newdata is supplied, only returns .fitted, .resid and .se.fit columns.

Value

A tibble::tibble() with columns:

. cooksd Cooks distance.

.fitted Fitted or predicted value..hat Diagonal of the hat matrix.

.lower Lower bound on interval for fitted values.

.resid The difference between observed and fitted values.

.se.fit Standard errors of fitted values.

. sigma Estimated residual standard deviation when corresponding observation is dropped

from model.

.std.resid Standardised residuals.

. upper Upper bound on interval for fitted values.

See Also

```
stats::na.action
augment(), stats::predict.lm()
Other lm tidiers: augment.glm(), glance.glm(), glance.lm(), glance.summary.lm(), glance.svyglm(),
tidy.glm(), tidy.lm(), tidy.lm.beta(), tidy.mlm(), tidy.summary.lm()
```

```
library(ggplot2)
library(dplyr)

mod <- lm(mpg ~ wt + qsec, data = mtcars)

tidy(mod)
glance(mod)

# coefficient plot
d <- tidy(mod, conf.int = TRUE)</pre>
```

```
ggplot(d, aes(estimate, term, xmin = conf.low, xmax = conf.high, height = 0)) +
 geom_point() +
 geom_vline(xintercept = 0, lty = 4) +
 geom_errorbarh()
# aside: There are tidy() and glance() methods for lm.summary objects too.
# this can be useful when you want to conserve memory by converting large lm
# objects into their leaner summary.lm equivalents.
s <- summary(mod)</pre>
tidy(s, conf.int = TRUE)
glance(s)
augment(mod)
augment(mod, mtcars, interval = "confidence")
# predict on new data
newdata <- mtcars %>%
 head(6) %>%
 mutate(wt = wt + 1)
augment(mod, newdata = newdata)
# ggplot2 example where we also construct 95% prediction interval
# simpler bivariate model since we're plotting in 2D
mod2 <- lm(mpg ~ wt, data = mtcars)</pre>
au <- augment(mod2, newdata = newdata, interval = "prediction")</pre>
ggplot(au, aes(wt, mpg)) +
 geom_point() +
 geom\_line(aes(y = .fitted)) +
 geom_ribbon(aes(ymin = .lower, ymax = .upper), col = NA, alpha = 0.3)
# predict on new data without outcome variable. Output does not include .resid
newdata <- newdata %>%
 select(-mpg)
augment(mod, newdata = newdata)
au <- augment(mod, data = mtcars)</pre>
ggplot(au, aes(.hat, .std.resid)) +
 geom_vline(size = 2, colour = "white", xintercept = 0) +
 geom_hline(size = 2, colour = "white", yintercept = 0) +
 geom_point() +
 geom_smooth(se = FALSE)
plot(mod, which = 6)
ggplot(au, aes(.hat, .cooksd)) +
 geom_vline(xintercept = 0, colour = NA) +
 geom_abline(slope = seq(0, 3, by = 0.5), colour = "white") +
```

augment.ImRob 45

```
geom_smooth(se = FALSE) +
geom_point()

# column-wise models
a <- matrix(rnorm(20), nrow = 10)
b <- a + rnorm(length(a))
result <- lm(b ~ a)

tidy(result)</pre>
```

augment.lmRob

Augment data with information from a(n) lmRob object

Description

Augment accepts a model object and a dataset and adds information about each observation in the dataset. Most commonly, this includes predicted values in the .fitted column, residuals in the .resid column, and standard errors for the fitted values in a .se.fit column. New columns always begin with a . prefix to avoid overwriting columns in the original dataset.

Users may pass data to augment via either the data argument or the newdata argument. If the user passes data to the data argument, it **must** be exactly the data that was used to fit the model object. Pass datasets to newdata to augment data that was not used during model fitting. This still requires that at least all predictor variable columns used to fit the model are present. If the original outcome variable used to fit the model is not included in newdata, then no .resid column will be included in the output.

Augment will often behave differently depending on whether data or newdata is given. This is because there is often information associated with training observations (such as influences or related) measures that is not meaningfully defined for new observations.

For convenience, many augment methods provide default data arguments, so that augment(fit) will return the augmented training data. In these cases, augment tries to reconstruct the original data based on the model object with varying degrees of success.

The augmented dataset is always returned as a tibble::tibble with the **same number of rows** as the passed dataset. This means that the passed data must be coercible to a tibble. If a predictor enters the model as part of a matrix of covariates, such as when the model formula uses splines::ns(), stats::poly(), or survival::Surv(), it is represented as a matrix column.

We are in the process of defining behaviors for models fit with various na.action arguments, but make no guarantees about behavior when data is missing at this time.

Usage

```
## S3 method for class 'lmRob'
augment(x, data = model.frame(x), newdata = NULL, ...)
```

46 augment.lmRob

Arguments

A lmRob object returned from robust::lmRob(). Χ

data A base::data.frame or tibble::tibble() containing the original data that was

> used to produce the object x. Defaults to stats::model.frame(x) so that augment(my_fit) returns the augmented original data. Do not pass new data to the data argument. Augment will report information such as influence and cooks distance for data passed to the data argument. These measures are only

defined for the original training data.

newdata A base::data.frame() or tibble::tibble() containing all the original predictors used to create x. Defaults to NULL, indicating that nothing has been

passed to newdata. If newdata is specified, the data argument will be ignored.

Additional arguments. Not used. Needed to match generic signature only. Cautionary note: Misspelled arguments will be absorbed in ..., where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

Details

For tidiers for robust models from the MASS package see tidy.rlm().

See Also

```
robust::lmRob()
Other robust tidiers: glance.glmRob(), glance.lmRob(), tidy.glmRob(), tidy.lmRob()
```

```
# load modeling library
library(robust)
# fit model
m \leftarrow lmRob(mpg \sim wt, data = mtcars)
# summarize model fit with tidiers
tidy(m)
augment(m)
glance(m)
```

47 augment.lmrob

augment.lmrob

Augment data with information from a(n) lmrob object

Description

Augment accepts a model object and a dataset and adds information about each observation in the dataset. Most commonly, this includes predicted values in the .fitted column, residuals in the .resid column, and standard errors for the fitted values in a .se.fit column. New columns always begin with a . prefix to avoid overwriting columns in the original dataset.

Users may pass data to augment via either the data argument or the newdata argument. If the user passes data to the data argument, it must be exactly the data that was used to fit the model object. Pass datasets to newdata to augment data that was not used during model fitting. This still requires that at least all predictor variable columns used to fit the model are present. If the original outcome variable used to fit the model is not included in newdata, then no .resid column will be included in the output.

Augment will often behave differently depending on whether data or newdata is given. This is because there is often information associated with training observations (such as influences or related) measures that is not meaningfully defined for new observations.

For convenience, many augment methods provide default data arguments, so that augment(fit) will return the augmented training data. In these cases, augment tries to reconstruct the original data based on the model object with varying degrees of success.

The augmented dataset is always returned as a tibble::tibble with the same number of rows as the passed dataset. This means that the passed data must be coercible to a tibble. If a predictor enters the model as part of a matrix of covariates, such as when the model formula uses splines::ns(), stats::poly(), or survival::Surv(), it is represented as a matrix column.

We are in the process of defining behaviors for models fit with various na.action arguments, but make no guarantees about behavior when data is missing at this time.

Usage

```
## S3 method for class 'lmrob'
augment(x, data = model.frame(x), newdata = NULL, se_fit = FALSE, ...)
```

Arguments

A 1mrob object returned from robustbase::1mrob().

data A base::data.frame or tibble::tibble() containing the original data that was

used to produce the object x. Defaults to stats::model.frame(x) so that augment(my_fit) returns the augmented original data. **Do not** pass new data to the data argument. Augment will report information such as influence and cooks distance for data passed to the data argument. These measures are only

defined for the original training data.

newdata A base::data.frame() or tibble::tibble() containing all the original predictors used to create x. Defaults to NULL, indicating that nothing has been

passed to newdata. If newdata is specified, the data argument will be ignored.

48 augment.Imrob

se_fit

Logical indicating whether or not a .se.fit column should be added to the augmented output. For some models, this calculation can be somewhat time-consuming. Defaults to FALSE.

. . .

Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in . . . , where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

Details

For tidiers for robust models from the MASS package see tidy.rlm().

Value

```
A tibble::tibble() with columns:
```

.fitted Fitted or predicted value.

.resid The difference between observed and fitted values.

See Also

```
robustbase::lmrob()
```

Other robustbase tidiers: augment.glmrob(), glance.lmrob(), tidy.glmrob(), tidy.lmrob()

```
if (requireNamespace("robustbase", quietly = TRUE)) {
  # load libraries for models and data
  library(robustbase)

  data(coleman)
  set.seed(0)

  m <- lmrob(Y ~ ., data = coleman)
  tidy(m)
  augment(m)
  glance(m)

  data(carrots)

Rfit <- glmrob(cbind(success, total - success) ~ logdose + block,
  family = binomial, data = carrots, method = "Mqle",
  control = glmrobMqle.control(tcc = 1.2)
)</pre>
```

augment.loess 49

```
tidy(Rfit)
augment(Rfit)
}
```

augment.loess

Tidy a(n) loess object

Description

Tidy summarizes information about the components of a model. A model component might be a single term in a regression, a single hypothesis, a cluster, or a class. Exactly what tidy considers to be a model component varies across models but is usually self-evident. If a model has several distinct types of components, you will need to specify which components to return.

Usage

```
## S3 method for class 'loess'
augment(x, data = model.frame(x), newdata = NULL, se_fit = FALSE, ...)
```

Arguments

Χ

A loess objects returned by stats::loess().

data

A base::data.frame or tibble::tibble() containing the original data that was used to produce the object x. Defaults to stats::model.frame(x) so that augment(my_fit) returns the augmented original data. **Do not** pass new data to the data argument. Augment will report information such as influence and cooks distance for data passed to the data argument. These measures are only defined for the original training data.

newdata

A base::data.frame() or tibble::tibble() containing all the original predictors used to create x. Defaults to NULL, indicating that nothing has been passed to newdata. If newdata is specified, the data argument will be ignored.

se_fit

Logical indicating whether or not a .se.fit column should be added to the augmented output. For some models, this calculation can be somewhat time-consuming. Defaults to FALSE.

. . .

Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in . . . , where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

50 augment.loess

Details

When the modeling was performed with na.action = "na.omit" (as is the typical default), rows with NA in the initial data are omitted entirely from the augmented data frame. When the modeling was performed with na.action = "na.exclude", one should provide the original data as a second argument, at which point the augmented data will contain those rows (typically with NAs in place of the new columns). If the original data is not provided to augment() and na.action = "na.exclude", a warning is raised and the incomplete rows are dropped.

Note that loess objects by default will not predict on data outside of a bounding hypercube defined by the training data unless the original loess object was fit with control = loess.control(surface = \"direct\")). See stats::predict.loess() for details.

Value

A tibble::tibble() with columns:

.fitted Fitted or predicted value.

resid The difference between observed and fitted values.

.se.fit Standard errors of fitted values.

See Also

```
stats::na.action
augment(), stats::loess(), stats::predict.loess()
```

```
lo <- loess(
  mpg ~ hp + wt,
  mtcars,
  control = loess.control(surface = "direct")
)
augment(lo)
# with all columns of original data
augment(lo, mtcars)
# with a new dataset
augment(lo, newdata = head(mtcars))</pre>
```

augment.Mclust 51

augment.Mclust

Augment data with information from a(n) Mclust object

Description

Augment accepts a model object and a dataset and adds information about each observation in the dataset. Most commonly, this includes predicted values in the .fitted column, residuals in the .resid column, and standard errors for the fitted values in a .se.fit column. New columns always begin with a . prefix to avoid overwriting columns in the original dataset.

Users may pass data to augment via either the data argument or the newdata argument. If the user passes data to the data argument, it must be exactly the data that was used to fit the model object. Pass datasets to newdata to augment data that was not used during model fitting. This still requires that at least all predictor variable columns used to fit the model are present. If the original outcome variable used to fit the model is not included in newdata, then no .resid column will be included in the output.

Augment will often behave differently depending on whether data or newdata is given. This is because there is often information associated with training observations (such as influences or related) measures that is not meaningfully defined for new observations.

For convenience, many augment methods provide default data arguments, so that augment(fit) will return the augmented training data. In these cases, augment tries to reconstruct the original data based on the model object with varying degrees of success.

The augmented dataset is always returned as a tibble::tibble with the same number of rows as the passed dataset. This means that the passed data must be coercible to a tibble. If a predictor enters the model as part of a matrix of covariates, such as when the model formula uses splines::ns(), stats::poly(), or survival::Surv(), it is represented as a matrix column.

We are in the process of defining behaviors for models fit with various na.action arguments, but make no guarantees about behavior when data is missing at this time.

Usage

```
## S3 method for class 'Mclust'
augment(x, data = NULL, ...)
```

Arguments

An Mclust object return from mclust::Mclust().

data A base::data.frame or tibble::tibble() containing the original data that was used to produce the object x. Defaults to stats::model.frame(x) so that augment(my_fit) returns the augmented original data. Do not pass new data to the data argument. Augment will report information such as influence and

cooks distance for data passed to the data argument. These measures are only defined for the original training data.

Additional arguments. Not used. Needed to match generic signature only. Cautionary note: Misspelled arguments will be absorbed in . . . , where they will be ignored. If the misspelled argument has a default value, the default value will be 52 augment.Mclust

used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

Value

A tibble::tibble() with columns:

.class Predicted class.

class probability.

See Also

```
augment(), mclust::Mclust()
Other mclust tidiers: tidy.Mclust()
```

```
# load library for models and data
library(mclust)
# load data manipulation libraries
library(dplyr)
library(tibble)
library(purrr)
library(tidyr)
set.seed(27)
centers <- tibble(</pre>
  cluster = factor(1:3),
  # number points in each cluster
  num_points = c(100, 150, 50),
  # x1 coordinate of cluster center
  x1 = c(5, 0, -3),
  # x2 coordinate of cluster center
  x2 = c(-1, 1, -2)
)
points <- centers %>%
  mutate(
   x1 = map2(num_points, x1, rnorm),
   x2 = map2(num_points, x2, rnorm)
  ) %>%
  select(-num_points, -cluster) %>%
  unnest(c(x1, x2))
```

augment.mfx 53

```
# fit model
m <- Mclust(points)

# summarize model fit with tidiers
tidy(m)
augment(m, points)
glance(m)</pre>
```

augment.mfx

Augment data with information from a(n) mfx object

Description

Augment accepts a model object and a dataset and adds information about each observation in the dataset. Most commonly, this includes predicted values in the .fitted column, residuals in the .resid column, and standard errors for the fitted values in a .se.fit column. New columns always begin with a . prefix to avoid overwriting columns in the original dataset.

Users may pass data to augment via either the data argument or the newdata argument. If the user passes data to the data argument, it **must** be exactly the data that was used to fit the model object. Pass datasets to newdata to augment data that was not used during model fitting. This still requires that at least all predictor variable columns used to fit the model are present. If the original outcome variable used to fit the model is not included in newdata, then no .resid column will be included in the output.

Augment will often behave differently depending on whether data or newdata is given. This is because there is often information associated with training observations (such as influences or related) measures that is not meaningfully defined for new observations.

For convenience, many augment methods provide default data arguments, so that augment(fit) will return the augmented training data. In these cases, augment tries to reconstruct the original data based on the model object with varying degrees of success.

The augmented dataset is always returned as a tibble::tibble with the **same number of rows** as the passed dataset. This means that the passed data must be coercible to a tibble. If a predictor enters the model as part of a matrix of covariates, such as when the model formula uses splines::ns(), stats::poly(), or survival::Surv(), it is represented as a matrix column.

We are in the process of defining behaviors for models fit with various na.action arguments, but make no guarantees about behavior when data is missing at this time.

Usage

```
## S3 method for class 'mfx'
augment(
    x,
    data = model.frame(x$fit),
    newdata = NULL,
    type.predict = c("link", "response", "terms"),
```

54 augment.mfx

```
type.residuals = c("deviance", "pearson"),
  se_fit = FALSE,
)
## S3 method for class 'logitmfx'
augment(
 data = model.frame(x$fit),
 newdata = NULL,
  type.predict = c("link", "response", "terms"),
  type.residuals = c("deviance", "pearson"),
  se_fit = FALSE,
)
## S3 method for class 'negbinmfx'
augment(
 Х,
 data = model.frame(x$fit),
 newdata = NULL,
 type.predict = c("link", "response", "terms"),
  type.residuals = c("deviance", "pearson"),
 se_fit = FALSE,
)
## S3 method for class 'poissonmfx'
augment(
 Х,
 data = model.frame(x$fit),
 newdata = NULL,
  type.predict = c("link", "response", "terms"),
  type.residuals = c("deviance", "pearson"),
  se_fit = FALSE,
)
## S3 method for class 'probitmfx'
augment(
 Х,
 data = model.frame(x$fit),
 newdata = NULL,
  type.predict = c("link", "response", "terms"),
  type.residuals = c("deviance", "pearson"),
  se_fit = FALSE,
  . . .
)
```

augment.mfx 55

Arguments

x A logitmfx, negbinmfx, poissonmfx, or probitmfx object. (Note that betamfx objects receive their own set of tidiers.)

data A base::data.frame or tibble::tibble() containing the original data that was

A base::data.frame or tibble::tibble() containing the original data that was used to produce the object x. Defaults to stats::model.frame(x) so that augment(my_fit) returns the augmented original data. **Do not** pass new data to the data argument. Augment will report information such as influence and cooks distance for data passed to the data argument. These measures are only

defined for the original training data.

newdata A base::data.frame() or tibble::tibble() containing all the original pre-

dictors used to create x. Defaults to NULL, indicating that nothing has been passed to newdata. If newdata is specified, the data argument will be ignored.

type.predict Passed to stats::predict.glm() type argument. Defaults to "link".

type.residuals Passed to stats::residuals.glm() and to stats::rstandard.glm() type

arguments. Defaults to "deviance".

se_fit Logical indicating whether or not a .se.fit column should be added to the

augmented output. For some models, this calculation can be somewhat time-

consuming. Defaults to FALSE.

.. Additional arguments. Not used. Needed to match generic signature only. Cautionary note: Misspelled arguments will be absorbed in . . . , where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed

using conf.level = 0.95. Two exceptions here are:

• tidy() methods will warn when supplied an exponentiate argument if it

will be ignored.

• augment() methods will warn when supplied a newdata argument if it will

be ignored.

Details

This generic augment method wraps augment.glm() for applicable objects from the mfx package.

Value

A tibble::tibble() with columns:

. cooksd Cooks distance.

.fitted Fitted or predicted value..hat Diagonal of the hat matrix.

resid The difference between observed and fitted values.

.se.fit Standard errors of fitted values.

. sigma Estimated residual standard deviation when corresponding observation is dropped

from model.

.std.resid Standardised residuals.

56 augment.mjoint

See Also

```
augment.glm(), mfx::logitmfx(), mfx::negbinmfx(), mfx::poissonmfx(), mfx::probitmfx()
Other mfx tidiers: augment.betamfx(), glance.betamfx(), glance.mfx(), tidy.betamfx(),
tidy.mfx()
```

Examples

```
# load libraries for models and data
library(mfx)
# get the marginal effects from a logit regression
mod_logmfx <- logitmfx(am ~ cyl + hp + wt, atmean = TRUE, data = mtcars)</pre>
tidy(mod_logmfx, conf.int = TRUE)
# compare with the naive model coefficients of the same logit call
 glm(am ~ cyl + hp + wt, family = binomial, data = mtcars),
 conf.int = TRUE
)
augment(mod_logmfx)
glance(mod_logmfx)
# another example, this time using probit regression
mod_probmfx <- probitmfx(am ~ cyl + hp + wt, atmean = TRUE, data = mtcars)</pre>
tidy(mod_probmfx, conf.int = TRUE)
augment(mod_probmfx)
glance(mod_probmfx)
```

augment.mjoint

Augment data with information from a(n) mjoint object

Description

Augment accepts a model object and a dataset and adds information about each observation in the dataset. Most commonly, this includes predicted values in the .fitted column, residuals in the .resid column, and standard errors for the fitted values in a .se.fit column. New columns always begin with a . prefix to avoid overwriting columns in the original dataset.

Users may pass data to augment via either the data argument or the newdata argument. If the user passes data to the data argument, it **must** be exactly the data that was used to fit the model object. Pass datasets to newdata to augment data that was not used during model fitting. This still requires that at least all predictor variable columns used to fit the model are present. If the original outcome variable used to fit the model is not included in newdata, then no .resid column will be included in the output.

augment.mjoint 57

Augment will often behave differently depending on whether data or newdata is given. This is because there is often information associated with training observations (such as influences or related) measures that is not meaningfully defined for new observations.

For convenience, many augment methods provide default data arguments, so that augment(fit) will return the augmented training data. In these cases, augment tries to reconstruct the original data based on the model object with varying degrees of success.

The augmented dataset is always returned as a tibble::tibble with the **same number of rows** as the passed dataset. This means that the passed data must be coercible to a tibble. If a predictor enters the model as part of a matrix of covariates, such as when the model formula uses splines::ns(), stats::poly(), or survival::Surv(), it is represented as a matrix column.

We are in the process of defining behaviors for models fit with various na.action arguments, but make no guarantees about behavior when data is missing at this time.

Usage

```
## S3 method for class 'mjoint'
augment(x, data = x$data, ...)
```

Arguments

x An mjoint object returned from joineRML::mjoint().

data

A base::data.frame or tibble::tibble() containing the original data that was used to produce the object x. Defaults to stats::model.frame(x) so that augment(my_fit) returns the augmented original data. **Do not** pass new data to the data argument. Augment will report information such as influence and cooks distance for data passed to the data argument. These measures are only defined for the original training data.

. . .

Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in . . . , where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

Details

See joineRML::fitted.mjoint() and joineRML::residuals.mjoint() for more information on the difference between population-level and individual-level fitted values and residuals.

If fitting a joint model with a single longitudinal process, make sure you are using a named list to define the formula for the fixed and random effects of the longitudinal submodel.

58 augment.mjoint

Value

A tibble::tibble() with one row for each original observation with addition columns:

```
.fitted_j_0 population-level fitted values for the j-th longitudinal process .fitted_j_1 individuals-level fitted values for the j-th longitudinal process .resid_j_0 population-level residuals for the j-th longitudinal process .resid_j_1 individual-level residuals for the j-th longitudinal process
```

```
# broom only skips running these examples because the example models take a
# while to generate—they should run just fine, though!
## Not run:
# load libraries for models and data
library(joineRML)
# fit a joint model with bivariate longitudinal outcomes
data(heart.valve)
hvd <- heart.valve[!is.na(heart.valve$log.grad) &</pre>
 !is.na(heart.valve$log.lvmi) &
 heart.valve$num <= 50, ]
fit <- mjoint(</pre>
 formLongFixed = list(
    "grad" = log.grad \sim time + sex + hs,
   "lvmi" = log.lvmi ~ time + sex
 formLongRandom = list(
    "grad" = ~1 \mid num,
    "lvmi" = ~ time | num
 ),
 formSurv = Surv(fuyrs, status) ~ age,
 data = hvd,
 inits = list("gamma" = c(0.11, 1.51, 0.80)),
 timeVar = "time"
# extract the survival fixed effects
tidy(fit)
# extract the longitudinal fixed effects
tidy(fit, component = "longitudinal")
# extract the survival fixed effects with confidence intervals
tidy(fit, ci = TRUE)
# extract the survival fixed effects with confidence intervals based
```

augment.mlogit 59

```
# on bootstrapped standard errors
bSE <- bootsE(fit, nboot = 5, safe.boot = TRUE)
tidy(fit, boot_se = bSE, ci = TRUE)

# augment original data with fitted longitudinal values and residuals
hvd2 <- augment(fit)

# extract model statistics
glance(fit)

## End(Not run)</pre>
```

augment.mlogit

Augment data with information from a(n) mlogit object

Description

Augment accepts a model object and a dataset and adds information about each observation in the dataset. Most commonly, this includes predicted values in the .fitted column, residuals in the .resid column, and standard errors for the fitted values in a .se.fit column. New columns always begin with a . prefix to avoid overwriting columns in the original dataset.

Users may pass data to augment via either the data argument or the newdata argument. If the user passes data to the data argument, it **must** be exactly the data that was used to fit the model object. Pass datasets to newdata to augment data that was not used during model fitting. This still requires that at least all predictor variable columns used to fit the model are present. If the original outcome variable used to fit the model is not included in newdata, then no .resid column will be included in the output.

Augment will often behave differently depending on whether data or newdata is given. This is because there is often information associated with training observations (such as influences or related) measures that is not meaningfully defined for new observations.

For convenience, many augment methods provide default data arguments, so that augment(fit) will return the augmented training data. In these cases, augment tries to reconstruct the original data based on the model object with varying degrees of success.

The augmented dataset is always returned as a tibble::tibble with the **same number of rows** as the passed dataset. This means that the passed data must be coercible to a tibble. If a predictor enters the model as part of a matrix of covariates, such as when the model formula uses splines::ns(), stats::poly(), or survival::Surv(), it is represented as a matrix column.

We are in the process of defining behaviors for models fit with various na.action arguments, but make no guarantees about behavior when data is missing at this time.

Usage

```
## S3 method for class 'mlogit'
augment(x, data = x$model, ...)
```

60 augment.mlogit

Arguments

. . .

x an object returned from mlogit::mlogit().

data Not currently used

Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in . . . , where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

Details

At the moment this only works on the estimation dataset. Need to set it up to predict on another dataset.

Value

```
A tibble::tibble() with columns:
```

.fitted Fitted or predicted value..probability Class probability of modal class.

resid The difference between observed and fitted values.

See Also

```
augment()
Other mlogit tidiers: glance.mlogit(), tidy.mlogit()
```

```
# load libraries for models and data
library(mlogit)

data("Fishing", package = "mlogit")
Fish <- dfidx(Fishing, varying = 2:9, shape = "wide", choice = "mode")

# fit model
m <- mlogit(mode ~ price + catch | income, data = Fish)

# summarize model fit with tidiers
tidy(m)
augment(m)
glance(m)</pre>
```

augment.nlrq 61

augment.nlrq

Tidy a(n) nlrq object

Description

Tidy summarizes information about the components of a model. A model component might be a single term in a regression, a single hypothesis, a cluster, or a class. Exactly what tidy considers to be a model component varies across models but is usually self-evident. If a model has several distinct types of components, you will need to specify which components to return.

Usage

```
## S3 method for class 'nlrq'
augment(x, data = NULL, newdata = NULL, ...)
```

Arguments

Χ

A nlrq object returned from quantreg::nlrq().

data

A base::data.frame or tibble::tibble() containing the original data that was used to produce the object x. Defaults to stats::model.frame(x) so that augment(my_fit) returns the augmented original data. **Do not** pass new data to the data argument. Augment will report information such as influence and cooks distance for data passed to the data argument. These measures are only defined for the original training data.

newdata

A base::data.frame() or tibble::tibble() containing all the original predictors used to create x. Defaults to NULL, indicating that nothing has been passed to newdata. If newdata is specified, the data argument will be ignored.

. . .

Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in . . ., where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

See Also

```
augment(), quantreg::nlrq()
Other quantreg tidiers: augment.rq(), augment.rqs(), glance.nlrq(), glance.rq(), tidy.nlrq(),
tidy.rq(), tidy.rqs()
```

62 augment.nls

Examples

```
# fit model
n <- nls(mpg ~ k * e^wt, data = mtcars, start = list(k = 1, e = 2))
# summarize model fit with tidiers + visualization
tidy(n)
augment(n)
glance(n)
library(ggplot2)
ggplot(augment(n), aes(wt, mpg)) +
geom_point() +
geom_line(aes(y = .fitted))
newdata <- head(mtcars)
newdata$wt <- newdata$wt + 1
augment(n, newdata = newdata)</pre>
```

augment.nls

Augment data with information from a(n) nls object

Description

Augment accepts a model object and a dataset and adds information about each observation in the dataset. Most commonly, this includes predicted values in the .fitted column, residuals in the .resid column, and standard errors for the fitted values in a .se.fit column. New columns always begin with a . prefix to avoid overwriting columns in the original dataset.

Users may pass data to augment via either the data argument or the newdata argument. If the user passes data to the data argument, it **must** be exactly the data that was used to fit the model object. Pass datasets to newdata to augment data that was not used during model fitting. This still requires that at least all predictor variable columns used to fit the model are present. If the original outcome variable used to fit the model is not included in newdata, then no .resid column will be included in the output.

Augment will often behave differently depending on whether data or newdata is given. This is because there is often information associated with training observations (such as influences or related) measures that is not meaningfully defined for new observations.

For convenience, many augment methods provide default data arguments, so that augment(fit) will return the augmented training data. In these cases, augment tries to reconstruct the original data based on the model object with varying degrees of success.

The augmented dataset is always returned as a tibble::tibble with the **same number of rows** as the passed dataset. This means that the passed data must be coercible to a tibble. If a predictor enters the model as part of a matrix of covariates, such as when the model formula uses splines::ns(), stats::poly(), or survival::Surv(), it is represented as a matrix column.

augment.nls 63

We are in the process of defining behaviors for models fit with various na.action arguments, but make no guarantees about behavior when data is missing at this time.

Usage

```
## S3 method for class 'nls'
augment(x, data = NULL, newdata = NULL, ...)
```

Arguments

x An nls object returned from stats::nls().

data A base::data.frame or tibble::tibble() containing the original data that was

used to produce the object x. Defaults to stats::model.frame(x) so that augment(my_fit) returns the augmented original data. **Do not** pass new data to the data argument. Augment will report information such as influence and cooks distance for data passed to the data argument. These measures are only

defined for the original training data.

newdata A base::data.frame() or tibble::tibble() containing all the original pre-

dictors used to create x. Defaults to NULL, indicating that nothing has been passed to newdata. If newdata is specified, the data argument will be ignored.

... Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in ..., where they will be ignored. If the misspelled argument has a default value, the default value will be

used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

Details

augment.nls does not currently support confidence intervals due to a lack of support in stats::predict.nls().

Value

```
A tibble::tibble() with columns:
```

.fitted Fitted or predicted value.

resid The difference between observed and fitted values.

See Also

```
tidy, stats::nls(), stats::predict.nls()
Other nls tidiers: glance.nls(), tidy.nls()
```

64 augment.pam

Examples

```
# fit model
n <- nls(mpg ~ k * e^wt, data = mtcars, start = list(k = 1, e = 2))
# summarize model fit with tidiers + visualization
tidy(n)
augment(n)
glance(n)
library(ggplot2)
ggplot(augment(n), aes(wt, mpg)) +
    geom_point() +
    geom_line(aes(y = .fitted))

newdata <- head(mtcars)
newdata$wt <- newdata$wt + 1
augment(n, newdata = newdata)</pre>
```

augment.pam

Augment data with information from a(n) pam object

Description

Augment accepts a model object and a dataset and adds information about each observation in the dataset. Most commonly, this includes predicted values in the .fitted column, residuals in the .resid column, and standard errors for the fitted values in a .se.fit column. New columns always begin with a . prefix to avoid overwriting columns in the original dataset.

Users may pass data to augment via either the data argument or the newdata argument. If the user passes data to the data argument, it **must** be exactly the data that was used to fit the model object. Pass datasets to newdata to augment data that was not used during model fitting. This still requires that at least all predictor variable columns used to fit the model are present. If the original outcome variable used to fit the model is not included in newdata, then no .resid column will be included in the output.

Augment will often behave differently depending on whether data or newdata is given. This is because there is often information associated with training observations (such as influences or related) measures that is not meaningfully defined for new observations.

For convenience, many augment methods provide default data arguments, so that augment(fit) will return the augmented training data. In these cases, augment tries to reconstruct the original data based on the model object with varying degrees of success.

The augmented dataset is always returned as a tibble::tibble with the **same number of rows** as the passed dataset. This means that the passed data must be coercible to a tibble. If a predictor enters the model as part of a matrix of covariates, such as when the model formula uses splines::ns(), stats::poly(), or survival::Surv(), it is represented as a matrix column.

augment.pam 65

We are in the process of defining behaviors for models fit with various na.action arguments, but make no guarantees about behavior when data is missing at this time.

Usage

```
## S3 method for class 'pam'
augment(x, data = NULL, ...)
```

Arguments

x An pam object returned from cluster::pam()

data A base::data.frame or tibble::tibble() containing the original data that was

used to produce the object x. Defaults to stats::model.frame(x) so that $augment(my_fit)$ returns the augmented original data. **Do not** pass new data to the data argument. Augment will report information such as influence and cooks distance for data passed to the data argument. These measures are only

defined for the original training data.

Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in . . . , where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

Value

```
A tibble::tibble() with columns:
```

. cluster Cluster assignment.
. fitted Fitted or predicted value.

resid The difference between observed and fitted values.

See Also

```
augment(), cluster::pam()
Other pam tidiers: glance.pam(), tidy.pam()
```

```
# load libraries for models and data
library(dplyr)
library(ggplot2)
library(cluster)
library(modeldata)
```

```
data(hpc_data)

x <- hpc_data[, 2:5]
p <- pam(x, k = 4)

# summarize model fit with tidiers + visualization
tidy(p)
glance(p)
augment(p, x)

augment(p, x)

ggplot(aes(compounds, input_fields)) +
geom_point(aes(color = .cluster)) +
geom_text(aes(label = cluster), data = tidy(p), size = 10)</pre>
```

augment.plm

Augment data with information from a(n) plm object

Description

Augment accepts a model object and a dataset and adds information about each observation in the dataset. Most commonly, this includes predicted values in the .fitted column, residuals in the .resid column, and standard errors for the fitted values in a .se.fit column. New columns always begin with a . prefix to avoid overwriting columns in the original dataset.

Users may pass data to augment via either the data argument or the newdata argument. If the user passes data to the data argument, it **must** be exactly the data that was used to fit the model object. Pass datasets to newdata to augment data that was not used during model fitting. This still requires that at least all predictor variable columns used to fit the model are present. If the original outcome variable used to fit the model is not included in newdata, then no .resid column will be included in the output.

Augment will often behave differently depending on whether data or newdata is given. This is because there is often information associated with training observations (such as influences or related) measures that is not meaningfully defined for new observations.

For convenience, many augment methods provide default data arguments, so that augment(fit) will return the augmented training data. In these cases, augment tries to reconstruct the original data based on the model object with varying degrees of success.

The augmented dataset is always returned as a tibble::tibble with the **same number of rows** as the passed dataset. This means that the passed data must be coercible to a tibble. If a predictor enters the model as part of a matrix of covariates, such as when the model formula uses splines::ns(), stats::poly(), or survival::Surv(), it is represented as a matrix column.

We are in the process of defining behaviors for models fit with various na.action arguments, but make no guarantees about behavior when data is missing at this time.

Usage

```
## S3 method for class 'plm'
augment(x, data = model.frame(x), ...)
```

Arguments

x A plm objected returned by plm::plm().

data

A base::data.frame or tibble::tibble() containing the original data that was used to produce the object x. Defaults to stats::model.frame(x) so that augment(my_fit) returns the augmented original data. **Do not** pass new data to the data argument. Augment will report information such as influence and cooks distance for data passed to the data argument. These measures are only defined for the original training data.

. . .

Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in . . . , where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

Value

```
A tibble::tibble() with columns:
```

.fitted Fitted or predicted value.

.resid The difference between observed and fitted values.

See Also

```
augment(), plm::plm()
Other plm tidiers: glance.plm(), tidy.plm()
```

```
# load libraries for models and data
library(plm)

# load data
data("Produc", package = "plm")

# fit model
zz <- plm(log(gsp) ~ log(pcap) + log(pc) + log(emp) + unemp,
data = Produc, index = c("state", "year")
)

# summarize model fit with tidiers
summary(zz)

tidy(zz)
tidy(zz, conf.int = TRUE)</pre>
```

68 augment.poLCA

```
tidy(zz, conf.int = TRUE, conf.level = 0.9)
augment(zz)
glance(zz)
```

augment.poLCA

Augment data with information from a(n) poLCA object

Description

Augment accepts a model object and a dataset and adds information about each observation in the dataset. Most commonly, this includes predicted values in the .fitted column, residuals in the .resid column, and standard errors for the fitted values in a .se.fit column. New columns always begin with a . prefix to avoid overwriting columns in the original dataset.

Users may pass data to augment via either the data argument or the newdata argument. If the user passes data to the data argument, it **must** be exactly the data that was used to fit the model object. Pass datasets to newdata to augment data that was not used during model fitting. This still requires that at least all predictor variable columns used to fit the model are present. If the original outcome variable used to fit the model is not included in newdata, then no .resid column will be included in the output.

Augment will often behave differently depending on whether data or newdata is given. This is because there is often information associated with training observations (such as influences or related) measures that is not meaningfully defined for new observations.

For convenience, many augment methods provide default data arguments, so that augment(fit) will return the augmented training data. In these cases, augment tries to reconstruct the original data based on the model object with varying degrees of success.

The augmented dataset is always returned as a tibble::tibble with the **same number of rows** as the passed dataset. This means that the passed data must be coercible to a tibble. If a predictor enters the model as part of a matrix of covariates, such as when the model formula uses splines::ns(), stats::poly(), or survival::Surv(), it is represented as a matrix column.

We are in the process of defining behaviors for models fit with various na.action arguments, but make no guarantees about behavior when data is missing at this time.

Usage

```
## S3 method for class 'poLCA'
augment(x, data = NULL, ...)
```

Arguments

x A poLCA object returned from poLCA::poLCA().

data A base::data.frame or tibble::tibble() cont

A base::data.frame or tibble::tibble() containing the original data that was used to produce the object x. Defaults to stats::model.frame(x) so that augment(my_fit) returns the augmented original data. **Do not** pass new data to the data argument. Augment will report information such as influence and

augment.poLCA 69

cooks distance for data passed to the data argument. These measures are only defined for the original training data.

Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in . . . , where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

Details

If the data argument is given, those columns are included in the output (only rows for which predictions could be made). Otherwise, the y element of the poLCA object, which contains the manifest variables used to fit the model, are used, along with any covariates, if present, in x.

Note that while the probability of all the classes (not just the predicted modal class) can be found in the posterior element, these are not included in the augmented output.

Value

```
A tibble::tibble() with columns:

.class Predicted class.

.probability Class probability of modal class.
```

See Also

```
augment(), poLCA::poLCA()
Other poLCA tidiers: glance.poLCA(), tidy.poLCA()
```

```
# load libraries for models and data
library(poLCA)
library(dplyr)

# generate data
data(values)

f <- cbind(A, B, C, D) ~ 1

# fit model
M1 <- poLCA(f, values, nclass = 2, verbose = FALSE)</pre>
M1
```

70 augment.polr

```
# summarize model fit with tidiers + visualization
tidy(M1)
augment(M1)
glance(M1)
library(ggplot2)
ggplot(tidy(M1), aes(factor(class), estimate, fill = factor(outcome))) +
 geom_bar(stat = "identity", width = 1) +
 facet_wrap(~variable)
# three-class model with a single covariate.
data(election)
f2a <- cbind(
 MORALG, CARESG, KNOWG, LEADG, DISHONG, INTELG,
 MORALB, CARESB, KNOWB, LEADB, DISHONB, INTELB
) ~ PARTY
nes2a <- poLCA(f2a, election, nclass = 3, nrep = 5, verbose = FALSE)</pre>
td <- tidy(nes2a)
td
ggplot(td, aes(outcome, estimate, color = factor(class), group = class)) +
 geom_line() +
 facet_wrap(~variable, nrow = 2) +
 theme(axis.text.x = element_text(angle = 90, hjust = 1))
au <- augment(nes2a)</pre>
au
count(au, .class)
# if the original data is provided, it leads to NAs in new columns
# for rows that weren't predicted
au2 <- augment(nes2a, data = election)</pre>
au2
dim(au2)
```

augment.polr

Augment data with information from a(n) polr object

Description

Augment accepts a model object and a dataset and adds information about each observation in the dataset. Most commonly, this includes predicted values in the .fitted column, residuals in

augment.polr 71

the .resid column, and standard errors for the fitted values in a .se.fit column. New columns always begin with a . prefix to avoid overwriting columns in the original dataset.

Users may pass data to augment via either the data argument or the newdata argument. If the user passes data to the data argument, it **must** be exactly the data that was used to fit the model object. Pass datasets to newdata to augment data that was not used during model fitting. This still requires that at least all predictor variable columns used to fit the model are present. If the original outcome variable used to fit the model is not included in newdata, then no .resid column will be included in the output.

Augment will often behave differently depending on whether data or newdata is given. This is because there is often information associated with training observations (such as influences or related) measures that is not meaningfully defined for new observations.

For convenience, many augment methods provide default data arguments, so that augment(fit) will return the augmented training data. In these cases, augment tries to reconstruct the original data based on the model object with varying degrees of success.

The augmented dataset is always returned as a tibble::tibble with the **same number of rows** as the passed dataset. This means that the passed data must be coercible to a tibble. If a predictor enters the model as part of a matrix of covariates, such as when the model formula uses splines::ns(), stats::poly(), or survival::Surv(), it is represented as a matrix column.

We are in the process of defining behaviors for models fit with various na.action arguments, but make no guarantees about behavior when data is missing at this time.

Usage

```
## S3 method for class 'polr'
augment(
    x,
    data = model.frame(x),
    newdata = NULL,
    type.predict = c("class"),
    ...
)
```

Arguments

x A	polr	object return	ed from	MASS: :	polr().

data A base::data.frame or tibble::tibble() containing the original data that was

used to produce the object x. Defaults to stats::model.frame(x) so that augment(my_fit) returns the augmented original data. **Do not** pass new data to the data argument. Augment will report information such as influence and cooks distance for data passed to the data argument. These measures are only

defined for the original training data.

newdata A base::data.frame() or tibble::tibble() containing all the original pre-

dictors used to create x. Defaults to NULL, indicating that nothing has been passed to newdata. If newdata is specified, the data argument will be ignored.

 $type.predict \qquad Which type of prediction to compute, passed to \texttt{MASS:::predict.polr()}. \ Only$

supports "class" at the moment.

72 augment.prcomp

Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in . . . , where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

See Also

. . .

```
tidy(), MASS::polr()
Other ordinal tidiers: augment.clm(), glance.clm(), glance.clmm(), glance.polr(), glance.svyolr(),
tidy.clm(), tidy.clmm(), tidy.polr(), tidy.svyolr()
```

Examples

```
# load libraries for models and data
library(MASS)

# fit model
fit <- polr(Sat ~ Infl + Type + Cont, weights = Freq, data = housing)

# summarize model fit with tidiers
tidy(fit, exponentiate = TRUE, conf.int = TRUE)

glance(fit)
augment(fit, type.predict = "class")

fit2 <- polr(factor(gear) ~ am + mpg + qsec, data = mtcars)

tidy(fit, p.values = TRUE)</pre>
```

augment.prcomp

Augment data with information from a(n) prcomp object

Description

Augment accepts a model object and a dataset and adds information about each observation in the dataset. Most commonly, this includes predicted values in the .fitted column, residuals in the .resid column, and standard errors for the fitted values in a .se.fit column. New columns always begin with a . prefix to avoid overwriting columns in the original dataset.

Users may pass data to augment via either the data argument or the newdata argument. If the user passes data to the data argument, it **must** be exactly the data that was used to fit the model object.

augment.prcomp 73

Pass datasets to newdata to augment data that was not used during model fitting. This still requires that at least all predictor variable columns used to fit the model are present. If the original outcome variable used to fit the model is not included in newdata, then no .resid column will be included in the output.

Augment will often behave differently depending on whether data or newdata is given. This is because there is often information associated with training observations (such as influences or related) measures that is not meaningfully defined for new observations.

For convenience, many augment methods provide default data arguments, so that augment(fit) will return the augmented training data. In these cases, augment tries to reconstruct the original data based on the model object with varying degrees of success.

The augmented dataset is always returned as a tibble::tibble with the **same number of rows** as the passed dataset. This means that the passed data must be coercible to a tibble. If a predictor enters the model as part of a matrix of covariates, such as when the model formula uses splines::ns(), stats::poly(), or survival::Surv(), it is represented as a matrix column.

We are in the process of defining behaviors for models fit with various na.action arguments, but make no guarantees about behavior when data is missing at this time.

Usage

```
## S3 method for class 'prcomp'
augment(x, data = NULL, newdata, ...)
```

Arguments

A prcomp object returned by stats::prcomp().

data

A base::data.frame or tibble::tibble() containing the original data that was used to produce the object x. Defaults to stats::model.frame(x) so that augment(my_fit) returns the augmented original data. **Do not** pass new data to the data argument. Augment will report information such as influence and cooks distance for data passed to the data argument. These measures are only defined for the original training data.

newdata

A base::data.frame() or tibble::tibble() containing all the original predictors used to create x. Defaults to NULL, indicating that nothing has been passed to newdata. If newdata is specified, the data argument will be ignored.

Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in . . . , where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

74 augment.rlm

Value

A tibble::tibble containing the original data along with additional columns containing each observation's projection into PCA space.

See Also

```
stats::prcomp(), svd_tidiers
Other svd tidiers: tidy.prcomp(), tidy_irlba(), tidy_svd()
```

augment.rlm

Augment data with information from a(n) rlm object

Description

Augment accepts a model object and a dataset and adds information about each observation in the dataset. Most commonly, this includes predicted values in the .fitted column, residuals in the .resid column, and standard errors for the fitted values in a .se.fit column. New columns always begin with a . prefix to avoid overwriting columns in the original dataset.

Users may pass data to augment via either the data argument or the newdata argument. If the user passes data to the data argument, it **must** be exactly the data that was used to fit the model object. Pass datasets to newdata to augment data that was not used during model fitting. This still requires that at least all predictor variable columns used to fit the model are present. If the original outcome variable used to fit the model is not included in newdata, then no .resid column will be included in the output.

Augment will often behave differently depending on whether data or newdata is given. This is because there is often information associated with training observations (such as influences or related) measures that is not meaningfully defined for new observations.

For convenience, many augment methods provide default data arguments, so that augment(fit) will return the augmented training data. In these cases, augment tries to reconstruct the original data based on the model object with varying degrees of success.

The augmented dataset is always returned as a tibble::tibble with the **same number of rows** as the passed dataset. This means that the passed data must be coercible to a tibble. If a predictor enters the model as part of a matrix of covariates, such as when the model formula uses splines::ns(), stats::poly(), or survival::Surv(), it is represented as a matrix column.

We are in the process of defining behaviors for models fit with various na.action arguments, but make no guarantees about behavior when data is missing at this time.

Usage

```
## S3 method for class 'rlm'
augment(x, data = model.frame(x), newdata = NULL, se_fit = FALSE, ...)
```

augment.rlm 75

Arguments

x An rlm object returned by MASS::rlm().

data A base::data.frame or tibble::tibble() containing the original data that was

used to produce the object x. Defaults to stats::model.frame(x) so that augment(my_fit) returns the augmented original data. **Do not** pass new data to the data argument. Augment will report information such as influence and cooks distance for data passed to the data argument. These measures are only

defined for the original training data.

newdata A base::data.frame() or tibble::tibble() containing all the original pre-

dictors used to create x. Defaults to NULL, indicating that nothing has been passed to newdata. If newdata is specified, the data argument will be ignored.

se_fit Logical indicating whether or not a .se.fit column should be added to the

augmented output. For some models, this calculation can be somewhat time-

consuming. Defaults to FALSE.

. Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in . . ., where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed

using conf.level = 0.95. Two exceptions here are:

• tidy() methods will warn when supplied an exponentiate argument if it will be ignored.

 augment() methods will warn when supplied a newdata argument if it will be ignored.

Value

A tibble::tibble() with columns:

.fitted Fitted or predicted value..hat Diagonal of the hat matrix.

. resid The difference between observed and fitted values.

.se.fit Standard errors of fitted values.

. sigma Estimated residual standard deviation when corresponding observation is dropped

from model.

See Also

```
MASS::rlm()
```

Other rlm tidiers: glance.rlm(), tidy.rlm()

```
# load libraries for models and data
library(MASS)
```

76 augment.rma

```
# fit model
r <- rlm(stack.loss ~ ., stackloss)
# summarize model fit with tidiers
tidy(r)
augment(r)
glance(r)</pre>
```

augment.rma

Augment data with information from a(n) rma object

Description

Augment accepts a model object and a dataset and adds information about each observation in the dataset. Most commonly, this includes predicted values in the .fitted column, residuals in the .resid column, and standard errors for the fitted values in a .se.fit column. New columns always begin with a . prefix to avoid overwriting columns in the original dataset.

Users may pass data to augment via either the data argument or the newdata argument. If the user passes data to the data argument, it **must** be exactly the data that was used to fit the model object. Pass datasets to newdata to augment data that was not used during model fitting. This still requires that at least all predictor variable columns used to fit the model are present. If the original outcome variable used to fit the model is not included in newdata, then no .resid column will be included in the output.

Augment will often behave differently depending on whether data or newdata is given. This is because there is often information associated with training observations (such as influences or related) measures that is not meaningfully defined for new observations.

For convenience, many augment methods provide default data arguments, so that augment(fit) will return the augmented training data. In these cases, augment tries to reconstruct the original data based on the model object with varying degrees of success.

The augmented dataset is always returned as a tibble::tibble with the **same number of rows** as the passed dataset. This means that the passed data must be coercible to a tibble. If a predictor enters the model as part of a matrix of covariates, such as when the model formula uses splines::ns(), stats::poly(), or survival::Surv(), it is represented as a matrix column.

We are in the process of defining behaviors for models fit with various na.action arguments, but make no guarantees about behavior when data is missing at this time.

Usage

```
## S3 method for class 'rma'
augment(x, interval = c("prediction", "confidence"), ...)
```

augment.rma 77

Arguments

x An rma object such as those created by metafor::rma(), metafor::rma.uni(), metafor::rma.glmm(), metafor::rma.mh(), metafor::rma.mv(), or metafor::rma.peto().

interval For rma.mv models, should prediction intervals ("prediction", default) or con-

fidence intervals ("confidence") intervals be returned? For rma.uni models, prediction intervals are always returned. For rma.mh and rma.peto models,

confidence intervals are always returned.

Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in . . ., where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it
 will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

Value

A tibble::tibble() with columns:

.fitted Fitted or predicted value.

. lower Lower bound on interval for fitted values.

. moderator In meta-analysis, the moderators used to calculate the predicted values.

.moderator.level

In meta-analysis, the level of the moderators used to calculate the predicted

values

resid The difference between observed and fitted values.

.se.fit Standard errors of fitted values.

. upper Upper bound on interval for fitted values.. observed The observed values for the individual studies

```
# load modeling library
library(metafor)

# generate data and fit
df <-
    escalc(
    measure = "RR",
    ai = tpos,
    bi = tneg,
    ci = cpos,
    di = cneg,
    data = dat.bcg</pre>
```

78 augment.rq

```
meta_analysis <- rma(yi, vi, data = df, method = "EB")
# summarize model fit with tidiers
augment(meta_analysis)</pre>
```

augment.rq

Augment data with information from a(n) rq object

Description

Augment accepts a model object and a dataset and adds information about each observation in the dataset. Most commonly, this includes predicted values in the .fitted column, residuals in the .resid column, and standard errors for the fitted values in a .se.fit column. New columns always begin with a . prefix to avoid overwriting columns in the original dataset.

Users may pass data to augment via either the data argument or the newdata argument. If the user passes data to the data argument, it **must** be exactly the data that was used to fit the model object. Pass datasets to newdata to augment data that was not used during model fitting. This still requires that at least all predictor variable columns used to fit the model are present. If the original outcome variable used to fit the model is not included in newdata, then no .resid column will be included in the output.

Augment will often behave differently depending on whether data or newdata is given. This is because there is often information associated with training observations (such as influences or related) measures that is not meaningfully defined for new observations.

For convenience, many augment methods provide default data arguments, so that augment(fit) will return the augmented training data. In these cases, augment tries to reconstruct the original data based on the model object with varying degrees of success.

The augmented dataset is always returned as a tibble::tibble with the **same number of rows** as the passed dataset. This means that the passed data must be coercible to a tibble. If a predictor enters the model as part of a matrix of covariates, such as when the model formula uses splines::ns(), stats::poly(), or survival::Surv(), it is represented as a matrix column.

We are in the process of defining behaviors for models fit with various na.action arguments, but make no guarantees about behavior when data is missing at this time.

Usage

```
## S3 method for class 'rq'
augment(x, data = model.frame(x), newdata = NULL, ...)
```

Arguments

Х

An rq object returned from quantreg::rq().

augment.rq 79

data

A base::data.frame or tibble::tibble() containing the original data that was used to produce the object x. Defaults to stats::model.frame(x) so that augment(my_fit) returns the augmented original data. **Do not** pass new data to the data argument. Augment will report information such as influence and cooks distance for data passed to the data argument. These measures are only defined for the original training data.

newdata

A base::data.frame() or tibble::tibble() containing all the original predictors used to create x. Defaults to NULL, indicating that nothing has been passed to newdata. If newdata is specified, the data argument will be ignored.

. . .

Arguments passed on to quantreg::predict.rq

object object of class rq or rqs or rq.process produced by rq

interval type of interval desired: default is 'none', when set to 'confidence' the function returns a matrix predictions with point predictions for each of the 'newdata' points as well as lower and upper confidence limits.

level converage probability for the 'confidence' intervals.

type For predict.rq, the method for 'confidence' intervals, if desired. If 'percentile' then one of the bootstrap methods is used to generate percentile intervals for each prediction, if 'direct' then a version of the Portnoy and Zhou (1998) method is used, and otherwise an estimated covariance matrix for the parameter estimates is used. Further arguments to determine the choice of bootstrap method or covariance matrix estimate can be passed via the ... argument. For predict.rqs and predict.rq.process when stepfun = TRUE, type is "Qhat", "Fhat" or "fhat" depending on whether the user would like to have estimates of the conditional quantile, distribution or density functions respectively. As noted below the two former estimates can be monotonized with the function rearrange. When the "fhat" option is invoked, a list of conditional density functions is returned based on Silverman's adaptive kernel method as implemented in akj and approxfun.

na.action function determining what should be done with missing values in 'newdata'. The default is to predict 'NA'.

Details

Depending on the arguments passed on to predict.rq via ..., a confidence interval is also calculated on the fitted values resulting in columns .lower and .upper. Does not provide confidence intervals when data is specified via the newdata argument.

Value

A tibble::tibble() with columns:

.fitted Fitted or predicted value.

. resid The difference between observed and fitted values.

. tau Quantile.

80 augment.rqs

See Also

```
augment, quantreg::rq(), quantreg::predict.rq()
Other quantreg tidiers: augment.nlrq(), augment.rqs(), glance.nlrq(), glance.rq(), tidy.nlrq(),
tidy.rq(), tidy.rqs()
```

Examples

```
# load modeling library and data
library(quantreg)
data(stackloss)
# median (11) regression fit for the stackloss data.
mod1 <- rq(stack.loss ~ stack.x, .5)</pre>
# weighted sample median
mod2 \leftarrow rq(rnorm(50) \sim 1, weights = runif(50))
# summarize model fit with tidiers
tidy(mod1)
glance(mod1)
augment(mod1)
tidy(mod2)
glance(mod2)
augment(mod2)
# varying tau to generate an rqs object
mod3 \leftarrow rq(stack.loss \sim stack.x, tau = c(.25, .5))
tidy(mod3)
augment(mod3)
# glance cannot handle rqs objects like `mod3`--use a purrr
# `map`-based workflow instead
```

augment.rqs

Augment data with information from a(n) rqs object

Description

Augment accepts a model object and a dataset and adds information about each observation in the dataset. Most commonly, this includes predicted values in the .fitted column, residuals in the .resid column, and standard errors for the fitted values in a .se.fit column. New columns always begin with a . prefix to avoid overwriting columns in the original dataset.

Users may pass data to augment via either the data argument or the newdata argument. If the user passes data to the data argument, it **must** be exactly the data that was used to fit the model object.

81 augment.rqs

Pass datasets to newdata to augment data that was not used during model fitting. This still requires that at least all predictor variable columns used to fit the model are present. If the original outcome variable used to fit the model is not included in newdata, then no .resid column will be included in the output.

Augment will often behave differently depending on whether data or newdata is given. This is because there is often information associated with training observations (such as influences or related) measures that is not meaningfully defined for new observations.

For convenience, many augment methods provide default data arguments, so that augment(fit) will return the augmented training data. In these cases, augment tries to reconstruct the original data based on the model object with varying degrees of success.

The augmented dataset is always returned as a tibble::tibble with the same number of rows as the passed dataset. This means that the passed data must be coercible to a tibble. If a predictor enters the model as part of a matrix of covariates, such as when the model formula uses splines::ns(), stats::poly(), or survival::Surv(), it is represented as a matrix column.

We are in the process of defining behaviors for models fit with various na.action arguments, but make no guarantees about behavior when data is missing at this time.

Usage

```
## S3 method for class 'rgs'
augment(x, data = model.frame(x), newdata, ...)
```

Arguments

Х An rgs object returned from quantreg::rg().

data A base::data.frame or tibble::tibble() containing the original data that was used to produce the object x. Defaults to stats::model.frame(x) so that

augment(my_fit) returns the augmented original data. **Do not** pass new data to the data argument. Augment will report information such as influence and cooks distance for data passed to the data argument. These measures are only

defined for the original training data.

A base::data.frame() or tibble::tibble() containing all the original prenewdata dictors used to create x. Defaults to NULL, indicating that nothing has been

passed to newdata. If newdata is specified, the data argument will be ignored.

Arguments passed on to quantreg::predict.rq

object object of class rq or rqs or rq.process produced by rq

interval type of interval desired: default is 'none', when set to 'confidence' the function returns a matrix predictions with point predictions for each of the 'newdata' points as well as lower and upper confidence limits.

level converage probability for the 'confidence' intervals.

type For predict.rg, the method for 'confidence' intervals, if desired. If 'percentile' then one of the bootstrap methods is used to generate percentile intervals for each prediction, if 'direct' then a version of the Portnoy and Zhou (1998) method is used, and otherwise an estimated covariance matrix for the parameter estimates is used. Further arguments to determine the choice of bootstrap method or covariance matrix estimate can be passed

82 augment.rqs

via the ... argument. For predict.rqs and predict.rq.process when stepfun = TRUE, type is "Qhat", "Fhat" or "fhat" depending on whether the user would like to have estimates of the conditional quantile, distribution or density functions respectively. As noted below the two former estimates can be monotonized with the function rearrange. When the "fhat" option is invoked, a list of conditional density functions is returned based on Silverman's adaptive kernel method as implemented in akj and approxfun.

na.action function determining what should be done with missing values in 'newdata'. The default is to predict 'NA'.

Details

Depending on the arguments passed on to predict.rq via ..., a confidence interval is also calculated on the fitted values resulting in columns .lower and .upper. Does not provide confidence intervals when data is specified via the newdata argument.

See Also

```
augment, quantreg::rq(), quantreg::predict.rqs()
Other quantreg tidiers: augment.nlrq(), augment.rq(), glance.nlrq(), glance.rq(), tidy.nlrq(),
tidy.rq(), tidy.rqs()
```

```
# load modeling library and data
library(quantreg)
data(stackloss)
# median (11) regression fit for the stackloss data.
mod1 <- rq(stack.loss ~ stack.x, .5)</pre>
# weighted sample median
mod2 \leftarrow rq(rnorm(50) \sim 1, weights = runif(50))
# summarize model fit with tidiers
tidy(mod1)
glance(mod1)
augment(mod1)
tidy(mod2)
glance(mod2)
augment(mod2)
# varying tau to generate an rqs object
mod3 \leftarrow rq(stack.loss \sim stack.x, tau = c(.25, .5))
tidy(mod3)
augment(mod3)
```

augment.sarlm 83

```
# glance cannot handle rqs objects like `mod3`--use a purrr
# `map`-based workflow instead
```

augment.sarlm

Augment data with information from a(n) spatialreg object

Description

Augment accepts a model object and a dataset and adds information about each observation in the dataset. Most commonly, this includes predicted values in the .fitted column, residuals in the .resid column, and standard errors for the fitted values in a .se.fit column. New columns always begin with a . prefix to avoid overwriting columns in the original dataset.

Users may pass data to augment via either the data argument or the newdata argument. If the user passes data to the data argument, it **must** be exactly the data that was used to fit the model object. Pass datasets to newdata to augment data that was not used during model fitting. This still requires that at least all predictor variable columns used to fit the model are present. If the original outcome variable used to fit the model is not included in newdata, then no .resid column will be included in the output.

Augment will often behave differently depending on whether data or newdata is given. This is because there is often information associated with training observations (such as influences or related) measures that is not meaningfully defined for new observations.

For convenience, many augment methods provide default data arguments, so that augment(fit) will return the augmented training data. In these cases, augment tries to reconstruct the original data based on the model object with varying degrees of success.

The augmented dataset is always returned as a tibble::tibble with the **same number of rows** as the passed dataset. This means that the passed data must be coercible to a tibble. If a predictor enters the model as part of a matrix of covariates, such as when the model formula uses splines::ns(), stats::poly(), or survival::Surv(), it is represented as a matrix column.

We are in the process of defining behaviors for models fit with various na.action arguments, but make no guarantees about behavior when data is missing at this time.

Usage

```
## S3 method for class 'sarlm'
augment(x, data = x$X, ...)
```

Arguments

An object returned from spatialreg::lagsarlm() or spatialreg::errorsarlm().

Ignored, but included for internal consistency. See the details below.

Additional arguments. Not used. Needed to match generic signature only. Cautionary note: Misspelled arguments will be absorbed in . . . , where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed

using conf.level = 0.95. Two exceptions here are:

84 augment.sarlm

• tidy() methods will warn when supplied an exponentiate argument if it will be ignored.

• augment() methods will warn when supplied a newdata argument if it will be ignored.

Details

The predict method for sarlm objects assumes that the response is known. See ?predict.sarlm for more discussion. As a result, since the original data can be recovered from the fit object, this method currently does not take in data or newdata arguments.

Value

```
A tibble::tibble() with columns:

.fitted Fitted or predicted value.

.resid The difference between observed and fitted values.
```

See Also

```
augment()
Other spatialreg tidiers: glance.sarlm(), tidy.sarlm()
```

```
# load libraries for models and data
library(spatialreg)
library(spdep)
# load data
data(oldcol, package = "spdep")
listw <- nb2listw(COL.nb, style = "W")</pre>
# fit model
crime_sar <-</pre>
  lagsarlm(CRIME ~ INC + HOVAL,
    data = COL.OLD,
    listw = listw,
    method = "eigen"
  )
# summarize model fit with tidiers
tidy(crime_sar)
tidy(crime_sar, conf.int = TRUE)
glance(crime_sar)
augment(crime_sar)
# fit another model
```

augment.smooth.spline

```
crime_sem <- errorsarlm(CRIME ~ INC + HOVAL, data = COL.OLD, listw)

# summarize model fit with tidiers
tidy(crime_sem)
tidy(crime_sem, conf.int = TRUE)
glance(crime_sem)
augment(crime_sem)

# fit another model
crime_sac <- sacsarlm(CRIME ~ INC + HOVAL, data = COL.OLD, listw)

# summarize model fit with tidiers
tidy(crime_sac)
tidy(crime_sac, conf.int = TRUE)
glance(crime_sac)
augment(crime_sac)</pre>
```

augment.smooth.spline $Tidy \ a(n) \ smooth.spline \ object$

Description

Tidy summarizes information about the components of a model. A model component might be a single term in a regression, a single hypothesis, a cluster, or a class. Exactly what tidy considers to be a model component varies across models but is usually self-evident. If a model has several distinct types of components, you will need to specify which components to return.

Usage

```
## S3 method for class 'smooth.spline'
augment(x, data = x$data, ...)
```

Arguments

Χ

A smooth.spline object returned from stats::smooth.spline().

data

A base::data.frame or tibble::tibble() containing the original data that was used to produce the object x. Defaults to stats::model.frame(x) so that augment(my_fit) returns the augmented original data. **Do not** pass new data to the data argument. Augment will report information such as influence and cooks distance for data passed to the data argument. These measures are only defined for the original training data.

. . .

Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in . . ., where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

86 augment.speedlm

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

Value

```
A tibble::tibble() with columns:

.fitted Fitted or predicted value.

.resid The difference between observed and fitted values.
```

See Also

```
augment(), stats::smooth.spline(), stats::predict.smooth.spline()
Other smoothing spline tidiers: glance.smooth.spline()
```

Examples

```
# fit model
spl <- smooth.spline(mtcars$wt, mtcars$mpg, df = 4)
# summarize model fit with tidiers
augment(spl, mtcars)
# calls original columns x and y
augment(spl)
library(ggplot2)
ggplot(augment(spl, mtcars), aes(wt, mpg)) +
    geom_point() +
    geom_line(aes(y = .fitted))</pre>
```

augment.speedlm

Augment data with information from a(n) speedlm object

Description

Augment accepts a model object and a dataset and adds information about each observation in the dataset. Most commonly, this includes predicted values in the .fitted column, residuals in the .resid column, and standard errors for the fitted values in a .se.fit column. New columns always begin with a . prefix to avoid overwriting columns in the original dataset.

Users may pass data to augment via either the data argument or the newdata argument. If the user passes data to the data argument, it **must** be exactly the data that was used to fit the model object. Pass datasets to newdata to augment data that was not used during model fitting. This still requires that at least all predictor variable columns used to fit the model are present. If the original outcome

augment.speedIm 87

variable used to fit the model is not included in newdata, then no .resid column will be included in the output.

Augment will often behave differently depending on whether data or newdata is given. This is because there is often information associated with training observations (such as influences or related) measures that is not meaningfully defined for new observations.

For convenience, many augment methods provide default data arguments, so that augment(fit) will return the augmented training data. In these cases, augment tries to reconstruct the original data based on the model object with varying degrees of success.

The augmented dataset is always returned as a tibble::tibble with the **same number of rows** as the passed dataset. This means that the passed data must be coercible to a tibble. If a predictor enters the model as part of a matrix of covariates, such as when the model formula uses splines::ns(), stats::poly(), or survival::Surv(), it is represented as a matrix column.

We are in the process of defining behaviors for models fit with various na. action arguments, but make no guarantees about behavior when data is missing at this time.

Usage

```
## S3 method for class 'speedlm'
augment(x, data = model.frame(x), newdata = NULL, ...)
```

Arguments

Χ....

A speedlm object returned from speedglm::speedlm().

data

A base::data.frame or tibble::tibble() containing the original data that was used to produce the object x. Defaults to stats::model.frame(x) so that augment(my_fit) returns the augmented original data. **Do not** pass new data to the data argument. Augment will report information such as influence and cooks distance for data passed to the data argument. These measures are only defined for the original training data.

newdata

A base::data.frame() or tibble::tibble() containing all the original predictors used to create x. Defaults to NULL, indicating that nothing has been passed to newdata. If newdata is specified, the data argument will be ignored.

. . .

Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in . . . , where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

Value

A tibble::tibble() with columns:

. fitted or predicted value.

. resid The difference between observed and fitted values.

88 augment.stl

See Also

```
speedglm::speedlm()
Other speedlm tidiers: glance.speedglm(), glance.speedlm(), tidy.speedglm(), tidy.speedlm()
```

Examples

```
# load modeling library
library(speedglm)

# fit model
mod <- speedlm(mpg ~ wt + qsec, data = mtcars, fitted = TRUE)

# summarize model fit with tidiers
tidy(mod)
glance(mod)
augment(mod)</pre>
```

augment.stl

Augment data with information from a(n) stl object

Description

Augment accepts a model object and a dataset and adds information about each observation in the dataset. Most commonly, this includes predicted values in the .fitted column, residuals in the .resid column, and standard errors for the fitted values in a .se.fit column. New columns always begin with a . prefix to avoid overwriting columns in the original dataset.

Users may pass data to augment via either the data argument or the newdata argument. If the user passes data to the data argument, it **must** be exactly the data that was used to fit the model object. Pass datasets to newdata to augment data that was not used during model fitting. This still requires that at least all predictor variable columns used to fit the model are present. If the original outcome variable used to fit the model is not included in newdata, then no .resid column will be included in the output.

Augment will often behave differently depending on whether data or newdata is given. This is because there is often information associated with training observations (such as influences or related) measures that is not meaningfully defined for new observations.

For convenience, many augment methods provide default data arguments, so that augment(fit) will return the augmented training data. In these cases, augment tries to reconstruct the original data based on the model object with varying degrees of success.

The augmented dataset is always returned as a tibble::tibble with the **same number of rows** as the passed dataset. This means that the passed data must be coercible to a tibble. If a predictor enters the model as part of a matrix of covariates, such as when the model formula uses splines::ns(), stats::poly(), or survival::Surv(), it is represented as a matrix column.

We are in the process of defining behaviors for models fit with various na.action arguments, but make no guarantees about behavior when data is missing at this time.

augment.survreg 89

Usage

```
## S3 method for class 'stl'
augment(x, data = NULL, weights = TRUE, ...)
```

Arguments

x An stl object returned from stats::stl().

data Ignored, included for consistency with the augment generic signature only.

weights Logical indicating whether or not to include the robust weights in the output.

using conf.level = 0.95. Two exceptions here are:

... Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in ..., where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

Value

A tibble::tibble with one row for each observation in the original times series:

. seasonal The seasonal component of the decomposition.

 $.\ trend \qquad \qquad The\ trend\ component\ of\ the\ decomposition.$

 $. \, remainder \qquad \quad The \, remainder, \, or \, "random" \, component \, of \, the \, decomposition.$

.weight The final robust weights, if requested.

. seasadj The seasonally adjusted (or "deseasonalised") series.

See Also

```
augment(), stats::stl()
```

Other decompose tidiers: augment.decomposed.ts()

90 augment.survreg

Description

Augment accepts a model object and a dataset and adds information about each observation in the dataset. Most commonly, this includes predicted values in the .fitted column, residuals in the .resid column, and standard errors for the fitted values in a .se.fit column. New columns always begin with a . prefix to avoid overwriting columns in the original dataset.

Users may pass data to augment via either the data argument or the newdata argument. If the user passes data to the data argument, it **must** be exactly the data that was used to fit the model object. Pass datasets to newdata to augment data that was not used during model fitting. This still requires that at least all predictor variable columns used to fit the model are present. If the original outcome variable used to fit the model is not included in newdata, then no .resid column will be included in the output.

Augment will often behave differently depending on whether data or newdata is given. This is because there is often information associated with training observations (such as influences or related) measures that is not meaningfully defined for new observations.

For convenience, many augment methods provide default data arguments, so that augment(fit) will return the augmented training data. In these cases, augment tries to reconstruct the original data based on the model object with varying degrees of success.

The augmented dataset is always returned as a tibble::tibble with the **same number of rows** as the passed dataset. This means that the passed data must be coercible to a tibble. If a predictor enters the model as part of a matrix of covariates, such as when the model formula uses splines::ns(), stats::poly(), or survival::Surv(), it is represented as a matrix column.

We are in the process of defining behaviors for models fit with various na.action arguments, but make no guarantees about behavior when data is missing at this time.

Usage

```
## S3 method for class 'survreg'
augment(
    x,
    data = model.frame(x),
    newdata = NULL,
    type.predict = "response",
    type.residuals = "response",
    ...
)
```

Arguments

x An survreg object returned from survival::survreg().

data

A base::data.frame or tibble::tibble() containing the original data that was used to produce the object x. Defaults to stats::model.frame(x) so that augment(my_fit) returns the augmented original data. **Do not** pass new data to the data argument. Augment will report information such as influence and cooks distance for data passed to the data argument. These measures are only defined for the original training data.

augment.survreg 91

newdata A base::data.frame() or tibble::tibble() containing all the original pre-

dictors used to create x. Defaults to NULL, indicating that nothing has been passed to newdata. If newdata is specified, the data argument will be ignored.

type.predict Character indicating type of prediction to use. Passed to the type argument of

the stats::predict() generic. Allowed arguments vary with model class, so be sure to read the predict.my_class documentation.

be sure to read the predict. my_class documentation.

type.residuals Character indicating type of residuals to use. Passed to the type argument of stats::residuals() generic. Allowed arguments vary with model class, so

be sure to read the residuals.my_class documentation.

Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in . . ., where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

Value

```
A tibble::tibble() with columns:
```

.fitted Fitted or predicted value.

resid The difference between observed and fitted values.

.se.fit Standard errors of fitted values.

See Also

```
augment(), survival::survreg()
Other survreg tidiers: glance.survreg(), tidy.survreg()
Other survival tidiers: augment.coxph(), glance.aareg(), glance.cch(), glance.coxph(),
glance.pyears(), glance.survdiff(), glance.survexp(), glance.survfit(), glance.survreg(),
tidy.aareg(), tidy.cch(), tidy.coxph(), tidy.pyears(), tidy.survdiff(), tidy.survexp(),
tidy.survfit(), tidy.survreg()
```

```
# load libraries for models and data
library(survival)

# fit model
sr <- survreg(
   Surv(futime, fustat) ~ ecog.ps + rx,
   ovarian,
   dist = "exponential"
)</pre>
```

92 augment_columns

```
# summarize model fit with tidiers + visualization
tidy(sr)
augment(sr, ovarian)
glance(sr)

# coefficient plot
td <- tidy(sr, conf.int = TRUE)

library(ggplot2)

ggplot(td, aes(estimate, term)) +
  geom_point() +
  geom_errorbarh(aes(xmin = conf.low, xmax = conf.high), height = 0) +
  geom_vline(xintercept = 0)</pre>
```

augment_columns

Add fitted values, residuals, and other common outputs to an augment call

Description

augment_columns is intended for use in the internals of augment methods only and is exported for developers extending the broom package. Please instead use augment() to appropriately make use of the functionality in augment_columns().

Usage

```
augment_columns(
    x,
    data,
    newdata = NULL,
    type,
    type.predict = type,
    type.residuals = type,
    se.fit = TRUE,
    ...
)
```

Arguments

x a model

data original data onto which columns should be added

newdata new data to predict on, optional

type Type of prediction and residuals to compute

type.predict Type of prediction to compute; by default same as type

bootstrap 93

type.residuals Type of residuals to compute; by default same as type

se.fit Value to pass to predict's se.fit, or NULL for no value. Ignored for model

types that do not accept an se.fit argument

... extra arguments (not used)

Details

Note that, in the case that a residuals() or influence() generic is not implemented for the supplied model x, the function will fail quietly.

bootstrap

Set up bootstrap replicates of a dplyr operation

Description

The bootstrap() function is deprecated and will be removed from an upcoming release of broom. For tidy resampling, please use the rsample package instead. Functionality is no longer supported for this method.

Usage

```
bootstrap(df, m, by_group = FALSE)
```

Arguments

df a data frame

m number of bootstrap replicates to perform

by_group If TRUE, then bootstrap within each group if df is a grouped tibble.

Details

This code originates from Hadley Wickham (with a few small corrections) here: https://github.com/tidyverse/dplyr/issues/269

See Also

```
Other deprecated: confint_tidy(), data.frame_tidiers, finish_glance(), fix_data_frame(), summary_tidiers, tidy.density(), tidy.dist(), tidy.ftable(), tidy.numeric()
```

94 confint_tidy

cont	fint	_tidy

(Deprecated) Calculate confidence interval as a tidy data frame

Description

This function is now deprecated and will be removed from a future release of broom.

Usage

```
confint_tidy(x, conf.level = 0.95, func = stats::confint, ...)
```

Arguments

x a model object for which confint() can be calculated

conf.level confidence level

func A function to compute a confidence interval for x. Calling func(x, level =

conf.level, ...) must return an object coercible to a tibble. This dataframe like object should have to columns corresponding the lower and upper bounds

on the confidence interval.

... extra arguments passed on to confint

Details

Return a confidence interval as a tidy data frame. This directly wraps the confint() function, but ensures it follows broom conventions: column names of conf.low and conf.high, and no row names.

```
confint_tidy
```

Value

A tibble with two columns: conf.low and conf.high.

See Also

```
Other deprecated: bootstrap(), data.frame_tidiers, finish_glance(), fix_data_frame(), summary_tidiers, tidy.density(), tidy.dist(), tidy.ftable(), tidy.numeric()
```

data.frame_tidiers 95

ects	
------	--

Description

Data frame tidiers are deprecated and will be removed from an upcoming release of broom.

Usage

```
## S3 method for class 'data.frame'
tidy(x, ..., na.rm = TRUE, trim = 0.1)
## S3 method for class 'data.frame'
augment(x, data, ...)
## S3 method for class 'data.frame'
glance(x, ...)
```

Arguments

X	A data.frame
	Additional arguments for other methods.
na.rm	a logical value indicating whether NA values should be stripped before the computation proceeds.
trim	the fraction (0 to 0.5) of observations to be trimmed from each end of x before the mean is computed. Passed to the trim argument of mean
data	data, not used

Details

These perform tidy summaries of data.frame objects. tidy produces summary statistics about each column, while glance simply reports the number of rows and columns. Note that augment.data.frame will throw an error.

Value

tidy.data.frame produces a data frame with one row per original column, containing summary statistics of each:

statistics of cacif.	
column	name of original column
n	Number of valid (non-NA) values
mean	mean
sd	standard deviation

median median

trimmed trimmed mean, with trim defaulting to .1

96 data.frame_tidiers

mad median absolute deviation (from the median)

min minimum value max maximum value

range range
skew skew
kurtosis kurtosis
se standard error

glance returns a one-row data.frame with

nrow number of rows ncol number of columns

complete.obs number of rows that have no missing values

na.fraction fraction of values across all rows and columns that are missing

Author(s)

David Robinson, Benjamin Nutter

Source

Skew and Kurtosis functions are adapted from implementations in the moments package: Lukasz Komsta and Frederick Novomestky (2015). moments: Moments, cumulants, skewness, kurtosis and related tests. R package version 0.14. https://CRAN.R-project.org/package=moments

See Also

```
Other deprecated: bootstrap(), confint_tidy(), finish_glance(), fix_data_frame(), summary_tidiers, tidy.density(), tidy.dist(), tidy.ftable(), tidy.numeric()

Other deprecated: bootstrap(), confint_tidy(), finish_glance(), fix_data_frame(), summary_tidiers, tidy.density(), tidy.dist(), tidy.ftable(), tidy.numeric()

Other deprecated: bootstrap(), confint_tidy(), finish_glance(), fix_data_frame(), summary_tidiers, tidy.density(), tidy.dist(), tidy.ftable(), tidy.numeric()
```

```
durbinWatsonTest_tidiers
```

Tidy/glance a(n) durbinWatsonTest object

Description

For models that have only a single component, the tidy() and glance() methods are identical. Please see the documentation for both of those methods.

Usage

```
## $3 method for class 'durbinWatsonTest'
tidy(x, ...)
## $3 method for class 'durbinWatsonTest'
glance(x, ...)
```

Arguments

x An object of class durbinWatsonTest created by a call to car::durbinWatsonTest().

Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in . . . , where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

Value

A tibble::tibble() with columns:

alternative Alternative hypothesis (character).

autocorrelation

Autocorrelation.

p.value The two-sided p-value associated with the observed statistic.

statistic Test statistic for Durbin-Watson test.
method Always 'Durbin-Watson Test'.

See Also

```
tidy(), glance(), car::durbinWatsonTest()
Other car tidiers: leveneTest_tidiers
```

98 finish_glance

Examples

```
# load modeling library
library(car)

# fit model
dw <- durbinWatsonTest(lm(mpg ~ wt, data = mtcars))

# summarize model fit with tidiers
tidy(dw)

# same output for all durbinWatsonTests
glance(dw)</pre>
```

finish_glance

(Deprecated) Add logLik, AIC, BIC, and other common measurements to a glance of a prediction

Description

This function is now deprecated in favor of using custom logic and the appropriate nobs() method.

Usage

```
finish_glance(ret, x)
```

Arguments

ret a one-row data frame (a partially complete glance)

x the prediction model

Value

a one-row data frame with additional columns added, such as

logLik log likelihoods

AIC Akaike Information Criterion
BIC Bayesian Information Criterion

deviance deviance

df.residual residual degrees of freedom

See Also

```
Other deprecated: bootstrap(), confint_tidy(), data.frame_tidiers, fix_data_frame(), summary_tidiers, tidy.density(), tidy.dist(), tidy.ftable(), tidy.numeric()
```

fix_data_frame 99

c ·		C
†1X	data	frame

Ensure an object is a data frame, with rownames moved into a column

Description

This function is deprecated as of broom 0.7.0 and will be removed from a future release. Please see tibble::as_tibble.

Usage

```
fix_data_frame(x, newnames = NULL, newcol = "term")
```

Arguments

x a data.frame or matrix

newnames new column names, not including the rownames

newcol the name of the new rownames column

Value

a data.frame, with rownames moved into a column and new column names assigned

See Also

```
Other deprecated: bootstrap(), confint_tidy(), data.frame_tidiers, finish_glance(), summary_tidiers, tidy.density(), tidy.dist(), tidy.ftable(), tidy.numeric()
```

glance.aareg

Glance at a(n) aareg object

Description

Glance accepts a model object and returns a tibble::tibble() with exactly one row of model summaries. The summaries are typically goodness of fit measures, p-values for hypothesis tests on residuals, or model convergence information.

Glance never returns information from the original call to the modeling function. This includes the name of the modeling function or any arguments passed to the modeling function.

Glance does not calculate summary measures. Rather, it farms out these computations to appropriate methods and gathers the results together. Sometimes a goodness of fit measure will be undefined. In these cases the measure will be reported as NA.

Glance returns the same number of columns regardless of whether the model matrix is rank-deficient or not. If so, entries in columns that no longer have a well-defined value are filled in with an NA of the appropriate type.

100 glance.aareg

Usage

```
## S3 method for class 'aareg'
glance(x, ...)
```

Arguments

x An aareg object returned from survival::aareg().

Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in . . . , where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

Value

A tibble::tibble() with exactly one row and columns:

df Degrees of freedom used by the model.

nobs Number of observations used.

p.value P-value corresponding to the test statistic.

statistic Test statistic.

See Also

```
glance(), survival::aareg()
Other aareg tidiers: tidy.aareg()
Other survival tidiers: augment.coxph(), augment.survreg(), glance.cch(), glance.coxph(),
glance.pyears(), glance.survdiff(), glance.survexp(), glance.survfit(), glance.survreg(),
tidy.aareg(), tidy.cch(), tidy.coxph(), tidy.pyears(), tidy.survdiff(), tidy.survexp(),
tidy.survfit(), tidy.survreg()
```

```
# load libraries for models and data
library(survival)

# fit model
afit <- aareg(
   Surv(time, status) ~ age + sex + ph.ecog,
   data = lung,
   dfbeta = TRUE
)</pre>
```

glance.anova 101

```
# summarize model fit with tidiers
tidy(afit)
```

glance.anova

Glance at a(n) anova object

Description

Glance accepts a model object and returns a tibble::tibble() with exactly one row of model summaries. The summaries are typically goodness of fit measures, p-values for hypothesis tests on residuals, or model convergence information.

Glance never returns information from the original call to the modeling function. This includes the name of the modeling function or any arguments passed to the modeling function.

Glance does not calculate summary measures. Rather, it farms out these computations to appropriate methods and gathers the results together. Sometimes a goodness of fit measure will be undefined. In these cases the measure will be reported as NA.

Glance returns the same number of columns regardless of whether the model matrix is rank-deficient or not. If so, entries in columns that no longer have a well-defined value are filled in with an NA of the appropriate type.

Usage

```
## S3 method for class 'anova'
glance(x, ...)
```

Arguments

An anova object, such as those created by stats::anova(), car::Anova(), car::leveneTest(), or car::linearHypothesis().

Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in . . . , where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

Value

A tibble::tibble() with exactly one row and columns:

deviance Deviance of the model.

df.residual Residual degrees of freedom.

102 glance.aov

Note

Note that the output of glance.anova() will vary depending on the initializing anova call. In some cases, it will just return an empty data frame. In other cases, glance.anova() may return columns that are also common to tidy.anova(). This is partly to preserve backwards compatibility with early versions of broom, but also because the underlying anova model yields components that could reasonably be interpreted as goodness-of-fit summaries too.

See Also

```
glance()
Other anova tidiers: glance.aov(), tidy.TukeyHSD(), tidy.anova(), tidy.aov(), tidy.aovlist(),
tidy.manova()
```

Examples

```
# fit models
a <- lm(mpg ~ wt + qsec + disp, mtcars)
b <- lm(mpg ~ wt + qsec, mtcars)

mod <- anova(a, b)

# summarize model fit with tidiers
tidy(mod)
glance(mod)

# car::linearHypothesis() example
library(car)
mod_lht <- linearHypothesis(a, "wt - disp")
tidy(mod_lht)
glance(mod_lht)</pre>
```

glance.aov

Glance at a(n) lm object

Description

Glance accepts a model object and returns a tibble::tibble() with exactly one row of model summaries. The summaries are typically goodness of fit measures, p-values for hypothesis tests on residuals, or model convergence information.

Glance never returns information from the original call to the modeling function. This includes the name of the modeling function or any arguments passed to the modeling function.

Glance does not calculate summary measures. Rather, it farms out these computations to appropriate methods and gathers the results together. Sometimes a goodness of fit measure will be undefined. In these cases the measure will be reported as NA.

glance.aov 103

Glance returns the same number of columns regardless of whether the model matrix is rank-deficient or not. If so, entries in columns that no longer have a well-defined value are filled in with an NA of the appropriate type.

Usage

```
## S3 method for class 'aov'
glance(x, ...)
```

Arguments

x An aov object, such as those created by stats::aov().

... Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in ..., where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

Value

A tibble::tibble() with exactly one row and columns:

AIC Akaike's Information Criterion for the model.

BIC Bayesian Information Criterion for the model.

deviance Deviance of the model.

logLik The log-likelihood of the model. [stats::logLik()] may be a useful reference.

nobs Number of observations used.

Note

Note that tidy.aov() now contains the numerator and denominator degrees of freedom, which were included in the output of glance.aov() in some previous versions of the package.

See Also

```
glance()
Other anova tidiers: glance.anova(), tidy.TukeyHSD(), tidy.anova(), tidy.aov(), tidy.aov(), tidy.aov(), tidy.aov(), tidy.aov()
```

```
a <- aov(mpg ~ wt + qsec + disp, mtcars)
tidy(a)</pre>
```

104 glance.Arima

. Arima Glance at a(n) Arima object

Description

Glance accepts a model object and returns a tibble::tibble() with exactly one row of model summaries. The summaries are typically goodness of fit measures, p-values for hypothesis tests on residuals, or model convergence information.

Glance never returns information from the original call to the modeling function. This includes the name of the modeling function or any arguments passed to the modeling function.

Glance does not calculate summary measures. Rather, it farms out these computations to appropriate methods and gathers the results together. Sometimes a goodness of fit measure will be undefined. In these cases the measure will be reported as NA.

Glance returns the same number of columns regardless of whether the model matrix is rank-deficient or not. If so, entries in columns that no longer have a well-defined value are filled in with an NA of the appropriate type.

Usage

```
## S3 method for class 'Arima'
glance(x, ...)
```

Arguments

An object of class Arima created by stats::arima(). х

> Additional arguments. Not used. Needed to match generic signature only. Cau**tionary note:** Misspelled arguments will be absorbed in . . . , where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

Value

sigma

A tibble::tibble() with exactly one row and columns:

AIC Akaike's Information Criterion for the model. BIC Bayesian Information Criterion for the model. logLik The log-likelihood of the model. [stats::logLik()] may be a useful reference. nobs Number of observations used. Estimated standard error of the residuals.

glance.betamfx 105

See Also

```
stats::arima()
Other Arima tidiers: tidy.Arima()
```

Examples

```
# fit model
fit <- arima(lh, order = c(1, 0, 0))
# summarize model fit with tidiers
tidy(fit)
glance(fit)</pre>
```

glance.betamfx

Glance at a(n) betamfx object

Description

Glance accepts a model object and returns a tibble::tibble() with exactly one row of model summaries. The summaries are typically goodness of fit measures, p-values for hypothesis tests on residuals, or model convergence information.

Glance never returns information from the original call to the modeling function. This includes the name of the modeling function or any arguments passed to the modeling function.

Glance does not calculate summary measures. Rather, it farms out these computations to appropriate methods and gathers the results together. Sometimes a goodness of fit measure will be undefined. In these cases the measure will be reported as NA.

Glance returns the same number of columns regardless of whether the model matrix is rank-deficient or not. If so, entries in columns that no longer have a well-defined value are filled in with an NA of the appropriate type.

Usage

```
## S3 method for class 'betamfx'
glance(x, ...)
```

Arguments

x A betamfx object.

Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in . . . , where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

• tidy() methods will warn when supplied an exponentiate argument if it will be ignored.

106 glance.betamfx

• augment() methods will warn when supplied a newdata argument if it will be ignored.

Details

This glance method wraps glance.betareg() for mfx::betamfx() objects.

Value

A tibble::tibble() with exactly one row and columns:

AIC Akaike's Information Criterion for the model.

BIC Bayesian Information Criterion for the model.

df.null Degrees of freedom used by the null model.

df.residual Residual degrees of freedom.

logLik The log-likelihood of the model. [stats::logLik()] may be a useful reference.

nobs Number of observations used.

pseudo.r.squared

Like the R squared statistic, but for situations when the R squared statistic isn't

defined.

See Also

```
glance.betareg(), mfx::betamfx()
Other mfx tidiers: augment.betamfx(), augment.mfx(), glance.mfx(), tidy.betamfx(), tidy.mfx()
```

```
library(mfx)

# Simulate some data
set.seed(12345)
n <- 1000
x <- rnorm(n)

# Beta outcome
y <- rbeta(n, shape1 = plogis(1 + 0.5 * x), shape2 = (abs(0.2 * x)))
# Use Smithson and Verkuilen correction
y <- (y * (n - 1) + 0.5) / n

d <- data.frame(y, x)
mod_betamfx <- betamfx(y ~ x | x, data = d)

tidy(mod_betamfx, conf.int = TRUE)

# Compare with the naive model coefficients of the equivalent betareg call (not run)
# tidy(betamfx(y ~ x | x, data = d), conf.int = TRUE)

augment(mod_betamfx)</pre>
```

glance.betareg 107

```
glance(mod_betamfx)
```

glance.betareg

Glance at a(n) betareg object

Description

Glance accepts a model object and returns a tibble::tibble() with exactly one row of model summaries. The summaries are typically goodness of fit measures, p-values for hypothesis tests on residuals, or model convergence information.

Glance never returns information from the original call to the modeling function. This includes the name of the modeling function or any arguments passed to the modeling function.

Glance does not calculate summary measures. Rather, it farms out these computations to appropriate methods and gathers the results together. Sometimes a goodness of fit measure will be undefined. In these cases the measure will be reported as NA.

Glance returns the same number of columns regardless of whether the model matrix is rank-deficient or not. If so, entries in columns that no longer have a well-defined value are filled in with an NA of the appropriate type.

Usage

```
## S3 method for class 'betareg'
glance(x, ...)
```

Arguments

. . .

x A betareg object produced by a call to betareg::betareg().

Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in . . . , where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

Value

A tibble::tibble() with exactly one row and columns:

AIC Akaike's Information Criterion for the model.

BIC Bayesian Information Criterion for the model.

df.null Degrees of freedom used by the null model.

108 glance.biglm

df.residual Residual degrees of freedom.

logLik The log-likelihood of the model. [stats::logLik()] may be a useful reference.

nobs Number of observations used.

pseudo.r.squared

Like the R squared statistic, but for situations when the R squared statistic isn't defined.

See Also

```
glance(), betareg::betareg()
```

Examples

```
# load libraries for models and data
library(betareg)

# load dats
data("GasolineYield", package = "betareg")

# fit model
mod <- betareg(yield ~ batch + temp, data = GasolineYield)

mod

# summarize model fit with tidiers
tidy(mod)
tidy(mod, conf.int = TRUE)
tidy(mod, conf.int = TRUE, conf.level = .99)
augment(mod)
glance(mod)</pre>
```

glance.biglm

Glance at a(n) biglm object

Description

Glance accepts a model object and returns a tibble::tibble() with exactly one row of model summaries. The summaries are typically goodness of fit measures, p-values for hypothesis tests on residuals, or model convergence information.

Glance never returns information from the original call to the modeling function. This includes the name of the modeling function or any arguments passed to the modeling function.

Glance does not calculate summary measures. Rather, it farms out these computations to appropriate methods and gathers the results together. Sometimes a goodness of fit measure will be undefined. In these cases the measure will be reported as NA.

glance.biglm 109

Glance returns the same number of columns regardless of whether the model matrix is rank-deficient or not. If so, entries in columns that no longer have a well-defined value are filled in with an NA of the appropriate type.

Usage

```
## S3 method for class 'biglm'
glance(x, ...)
```

Arguments

. . .

x A biglm object created by a call to biglm::biglm() or biglm::bigglm().

Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in . . . , where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

Value

A tibble::tibble() with exactly one row and columns:

AIC Akaike's Information Criterion for the model.

deviance Deviance of the model.

df.residual Residual degrees of freedom.
nobs Number of observations used.

r.squared R squared statistic, or the percent of variation explained by the model. Also

known as the coefficient of determination.

See Also

```
glance(), biglm::biglm(), biglm::bigglm()
Other biglm tidiers: tidy.biglm()
```

```
# load modeling library
library(biglm)

# fit model -- linear regression
bfit <- biglm(mpg ~ wt + disp, mtcars)
# summarize model fit with tidiers</pre>
```

110 glance.binDesign

```
tidy(bfit)
tidy(bfit, conf.int = TRUE)
tidy(bfit, conf.int = TRUE, conf.level = .9)
glance(bfit)

# fit model -- logistic regression
bgfit <- bigglm(am ~ mpg, mtcars, family = binomial())

# summarize model fit with tidiers
tidy(bgfit)
tidy(bgfit, exponentiate = TRUE)
tidy(bgfit, conf.int = TRUE)
tidy(bgfit, conf.int = TRUE, conf.level = .9)
tidy(bgfit, conf.int = TRUE, conf.level = .9, exponentiate = TRUE)
glance(bgfit)</pre>
```

glance.binDesign

Glance at a(n) binDesign object

Description

Glance accepts a model object and returns a tibble::tibble() with exactly one row of model summaries. The summaries are typically goodness of fit measures, p-values for hypothesis tests on residuals, or model convergence information.

Glance never returns information from the original call to the modeling function. This includes the name of the modeling function or any arguments passed to the modeling function.

Glance does not calculate summary measures. Rather, it farms out these computations to appropriate methods and gathers the results together. Sometimes a goodness of fit measure will be undefined. In these cases the measure will be reported as NA.

Glance returns the same number of columns regardless of whether the model matrix is rank-deficient or not. If so, entries in columns that no longer have a well-defined value are filled in with an NA of the appropriate type.

Usage

```
## S3 method for class 'binDesign'
glance(x, ...)
```

Arguments

x A binGroup::binDesign object.

Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in . . . , where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

glance.cch 111

• tidy() methods will warn when supplied an exponentiate argument if it will be ignored.

• augment() methods will warn when supplied a newdata argument if it will be ignored.

Value

A tibble::tibble() with exactly one row and columns:

power Power achieved by the analysis.

n Sample size used to achieve this power.

power.reached Whether the desired power was reached.

maxit Number of iterations performed.

See Also

```
glance(), binGroup::binDesign()
Other bingroup tidiers: tidy.binDesign(), tidy.binWidth()
```

```
# load libraries for models and data
library(binGroup)

des <- binDesign(
   nmax = 300, delta = 0.06,
   p.hyp = 0.1, power = .8
)

glance(des)
tidy(des)
library(ggplot2)
ggplot(tidy(des), aes(n, power)) +
   geom_line()</pre>
```

112 glance.cch

Description

Glance accepts a model object and returns a tibble::tibble() with exactly one row of model summaries. The summaries are typically goodness of fit measures, p-values for hypothesis tests on residuals, or model convergence information.

Glance never returns information from the original call to the modeling function. This includes the name of the modeling function or any arguments passed to the modeling function.

Glance does not calculate summary measures. Rather, it farms out these computations to appropriate methods and gathers the results together. Sometimes a goodness of fit measure will be undefined. In these cases the measure will be reported as NA.

Glance returns the same number of columns regardless of whether the model matrix is rank-deficient or not. If so, entries in columns that no longer have a well-defined value are filled in with an NA of the appropriate type.

Usage

```
## S3 method for class 'cch'
glance(x, ...)
```

Arguments

x An cch object returned from survival::cch().

Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in . . . , where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

Value

A tibble::tibble() with exactly one row and columns:

iter Iterations of algorithm/fitting procedure completed.

p.value P-value corresponding to the test statistic.

rscore Robust log-rank statistic

score Score.

n number of predictions

nevent number of events

glance.clm 113

See Also

```
glance(), survival::cch()
Other cch tidiers: glance.survfit(), tidy.cch()
Other survival tidiers: augment.coxph(), augment.survreg(), glance.aareg(), glance.coxph(),
glance.pyears(), glance.survdiff(), glance.survexp(), glance.survfit(), glance.survreg(),
tidy.aareg(), tidy.cch(), tidy.coxph(), tidy.pyears(), tidy.survdiff(), tidy.survexp(),
tidy.survfit(), tidy.survreg()
```

```
# load libraries for models and data
library(survival)
# examples come from cch documentation
subcoh <- nwtco$in.subcohort</pre>
selccoh <- with(nwtco, rel == 1 | subcoh == 1)</pre>
ccoh.data <- nwtco[selccoh, ]</pre>
ccoh.data$subcohort <- subcoh[selccoh]</pre>
# central-lab histology
ccoh.data$histol <- factor(ccoh.data$histol, labels = c("FH", "UH"))</pre>
# tumour stage
ccoh.data$stage <- factor(ccoh.data$stage, labels = c("I", "II", "III", "IV"))</pre>
ccoh.data$age <- ccoh.data$age / 12 # age in years</pre>
# fit model
fit.ccP <- cch(Surv(edrel, rel) ~ stage + histol + age,</pre>
 data = ccoh.data,
 subcoh = ~subcohort, id = ~seqno, cohort.size = 4028
)
# summarize model fit with tidiers + visualization
tidy(fit.ccP)
# coefficient plot
library(ggplot2)
ggplot(tidy(fit.ccP), aes(x = estimate, y = term)) +
 geom_point() +
 geom_errorbarh(aes(xmin = conf.low, xmax = conf.high), height = 0) +
 geom_vline(xintercept = 0)
```

114 glance.clm

Description

Glance accepts a model object and returns a tibble::tibble() with exactly one row of model summaries. The summaries are typically goodness of fit measures, p-values for hypothesis tests on residuals, or model convergence information.

Glance never returns information from the original call to the modeling function. This includes the name of the modeling function or any arguments passed to the modeling function.

Glance does not calculate summary measures. Rather, it farms out these computations to appropriate methods and gathers the results together. Sometimes a goodness of fit measure will be undefined. In these cases the measure will be reported as NA.

Glance returns the same number of columns regardless of whether the model matrix is rank-deficient or not. If so, entries in columns that no longer have a well-defined value are filled in with an NA of the appropriate type.

Usage

```
## S3 method for class 'clm'
glance(x, ...)
```

Arguments

. . .

x A clm object returned from ordinal::clm().

Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in . . . , where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

Value

A tibble::tibble() with exactly one row and columns:

AIC Akaike's Information Criterion for the model.

BIC Bayesian Information Criterion for the model.

df.residual Residual degrees of freedom.
edf The effective degrees of freedom.

logLik The log-likelihood of the model. [stats::logLik()] may be a useful reference.

nobs Number of observations used.

See Also

```
tidy, ordinal::clm()
Other ordinal tidiers: augment.clm(), augment.polr(), glance.clmm(), glance.polr(), glance.svyolr(),
tidy.clm(), tidy.clmm(), tidy.polr(), tidy.svyolr()
```

glance.clmm 115

Examples

```
# load libraries for models and data
library(ordinal)

# fit model
fit <- clm(rating ~ temp * contact, data = wine)

# summarize model fit with tidiers
tidy(fit)
tidy(fit, conf.int = TRUE, conf.level = 0.9)
tidy(fit, conf.int = TRUE, conf.type = "Wald", exponentiate = TRUE)
glance(fit)
augment(fit, type.predict = "prob")
augment(fit, type.predict = "class")

# ...and again with another model specification
fit2 <- clm(rating ~ temp, nominal = ~contact, data = wine)
tidy(fit2)
glance(fit2)</pre>
```

glance.clmm

Glance at a(n) clmm object

Description

Glance accepts a model object and returns a tibble::tibble() with exactly one row of model summaries. The summaries are typically goodness of fit measures, p-values for hypothesis tests on residuals, or model convergence information.

Glance never returns information from the original call to the modeling function. This includes the name of the modeling function or any arguments passed to the modeling function.

Glance does not calculate summary measures. Rather, it farms out these computations to appropriate methods and gathers the results together. Sometimes a goodness of fit measure will be undefined. In these cases the measure will be reported as NA.

Glance returns the same number of columns regardless of whether the model matrix is rank-deficient or not. If so, entries in columns that no longer have a well-defined value are filled in with an NA of the appropriate type.

Usage

```
## S3 method for class 'clmm'
glance(x, ...)
```

116 glance.clmm

Arguments

. . .

x A clmm object returned from ordinal::clmm().

Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in . . . , where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

Value

A tibble::tibble() with exactly one row and columns:

AIC Akaike's Information Criterion for the model.

BIC Bayesian Information Criterion for the model.

edf The effective degrees of freedom.

logLik The log-likelihood of the model. [stats::logLik()] may be a useful reference.

nobs Number of observations used.

See Also

```
tidy, ordinal::clmm()
Other ordinal tidiers: augment.clm(), augment.polr(), glance.clm(), glance.polr(), glance.svyolr(),
tidy.clm(), tidy.clmm(), tidy.polr(), tidy.svyolr()
```

```
# load libraries for models and data
library(ordinal)

# fit model
fit <- clmm(rating ~ temp + contact + (1 | judge), data = wine)

# summarize model fit with tidiers
tidy(fit)
tidy(fit, conf.int = TRUE, conf.level = 0.9)
tidy(fit, conf.int = TRUE, exponentiate = TRUE)

glance(fit)

# ...and again with another model specification
fit2 <- clmm(rating ~ temp + (1 | judge), nominal = ~contact, data = wine)
tidy(fit2)</pre>
```

glance.coeftest 117

```
glance(fit2)
```

glance.coeftest

Glance at a(n) coeftest object

Description

Glance accepts a model object and returns a tibble::tibble() with exactly one row of model summaries. The summaries are typically goodness of fit measures, p-values for hypothesis tests on residuals, or model convergence information.

Glance never returns information from the original call to the modeling function. This includes the name of the modeling function or any arguments passed to the modeling function.

Glance does not calculate summary measures. Rather, it farms out these computations to appropriate methods and gathers the results together. Sometimes a goodness of fit measure will be undefined. In these cases the measure will be reported as NA.

Glance returns the same number of columns regardless of whether the model matrix is rank-deficient or not. If so, entries in columns that no longer have a well-defined value are filled in with an NA of the appropriate type.

Usage

```
## S3 method for class 'coeftest'
glance(x, ...)
```

Arguments

. . .

x A coeftest object returned from lmtest::coeftest().

Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in . . ., where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

Value

A tibble::tibble() with exactly one row and columns:

adj.r.squared Adjusted R squared statistic, which is like the R squared statistic except taking degrees of freedom into account.

AIC Akaike's Information Criterion for the model.

118 glance.coeftest

BIC Bayesian Information Criterion for the model.

deviance Deviance of the model.

df Degrees of freedom used by the model.

df.residual Residual degrees of freedom.

logLik The log-likelihood of the model. [stats::logLik()] may be a useful reference.

nobs Number of observations used.

p.value P-value corresponding to the test statistic.

r.squared R squared statistic, or the percent of variation explained by the model. Also

known as the coefficient of determination.

sigma Estimated standard error of the residuals.

statistic Test statistic.

Note

Because of the way that lmtest::coeftest() retains information about the underlying model object, the returned columns for glance.coeftest() will vary depending on the arguments. Specifically, four columns are returned regardless: "Loglik", "AIC", "BIC", and "nobs". Users can obtain additional columns (e.g. "r.squared", "df") by invoking the "save = TRUE" argument as part of lmtest::coeftest(). See examples.

As an aside, goodness-of-fit measures such as R-squared are unaffected by the presence of heteroskedasticity. For further discussion see, e.g. chapter 8.1 of Wooldridge (2016).

References

Wooldridge, Jeffrey M. (2016) *Introductory econometrics: A modern approach*. (6th edition). Nelson Education.

See Also

```
glance(), lmtest::coeftest()
```

```
# load libraries for models and data
library(lmtest)

m <- lm(dist ~ speed, data = cars)

coeftest(m)
tidy(coeftest(m))
tidy(coeftest(m, conf.int = TRUE))

# a very common workflow is to combine lmtest::coeftest with alternate
# variance-covariance matrices via the sandwich package. The lmtest
# tidiers support this workflow too, enabling you to adjust the standard
# errors of your tidied models on the fly.</pre>
```

glance.coxph 119

```
library(sandwich)
# "HC3" (default) robust SEs
tidy(coeftest(m, vcov = vcovHC))

# "HC2" robust SEs
tidy(coeftest(m, vcov = vcovHC, type = "HC2"))

# N-W HAC robust SEs
tidy(coeftest(m, vcov = NeweyWest))

# the columns of the returned tibble for glance.coeftest() will vary
# depending on whether the coeftest object retains the underlying model.
# Users can control this with the "save = TRUE" argument of coeftest().
glance(coeftest(m))
glance(coeftest(m, save = TRUE))
```

glance.coxph

Glance at a(n) coxph object

Description

Glance accepts a model object and returns a tibble::tibble() with exactly one row of model summaries. The summaries are typically goodness of fit measures, p-values for hypothesis tests on residuals, or model convergence information.

Glance never returns information from the original call to the modeling function. This includes the name of the modeling function or any arguments passed to the modeling function.

Glance does not calculate summary measures. Rather, it farms out these computations to appropriate methods and gathers the results together. Sometimes a goodness of fit measure will be undefined. In these cases the measure will be reported as NA.

Glance returns the same number of columns regardless of whether the model matrix is rank-deficient or not. If so, entries in columns that no longer have a well-defined value are filled in with an NA of the appropriate type.

Usage

```
## S3 method for class 'coxph'
glance(x, ...)
```

Arguments

```
x A coxph object returned from survival::coxph().
```

... For tidy(), additional arguments passed to summary(x, ...). Otherwise ignored.

120 glance.coxph

Value

A tibble::tibble() with exactly one row and columns:

AIC Akaike's Information Criterion for the model.

BIC Bayesian Information Criterion for the model.

logLik The log-likelihood of the model. [stats::logLik()] may be a useful reference.

n The total number of observations.

nevent Number of events.

nobs Number of observations used.

See survival::coxph.object for additional column descriptions.

See Also

```
glance(), survival::coxph()
Other coxph tidiers: augment.coxph(), tidy.coxph()
Other survival tidiers: augment.coxph(), augment.survreg(), glance.aareg(), glance.cch(),
glance.pyears(), glance.survdiff(), glance.survexp(), glance.survfit(), glance.survreg(),
tidy.aareg(), tidy.cch(), tidy.coxph(), tidy.pyears(), tidy.survdiff(), tidy.survexp(),
tidy.survfit(), tidy.survreg()
```

```
# load libraries for models and data
library(survival)
# fit model
cfit <- coxph(Surv(time, status) ~ age + sex, lung)</pre>
# summarize model fit with tidiers
tidy(cfit)
tidy(cfit, exponentiate = TRUE)
lp <- augment(cfit, lung)</pre>
risks <- augment(cfit, lung, type.predict = "risk")</pre>
expected <- augment(cfit, lung, type.predict = "expected")</pre>
glance(cfit)
# also works on clogit models
resp <- levels(logan$occupation)</pre>
n <- nrow(logan)</pre>
indx <- rep(1:n, length(resp))</pre>
logan2 <- data.frame(</pre>
  logan[indx, ],
  id = indx,
  tocc = factor(rep(resp, each = n))
)
```

glance.crr 121

```
logan2$case <- (logan2$occupation == logan2$tocc)

cl <- clogit(case ~ tocc + tocc:education + strata(id), logan2)

tidy(cl)
glance(cl)

library(ggplot2)

ggplot(lp, aes(age, .fitted, color = sex)) +
    geom_point()

ggplot(risks, aes(age, .fitted, color = sex)) +
    geom_point()

ggplot(expected, aes(time, .fitted, color = sex)) +
    geom_point()</pre>
```

glance.crr

Glance at a(n) crr object

Description

Glance accepts a model object and returns a tibble::tibble() with exactly one row of model summaries. The summaries are typically goodness of fit measures, p-values for hypothesis tests on residuals, or model convergence information.

Glance never returns information from the original call to the modeling function. This includes the name of the modeling function or any arguments passed to the modeling function.

Glance does not calculate summary measures. Rather, it farms out these computations to appropriate methods and gathers the results together. Sometimes a goodness of fit measure will be undefined. In these cases the measure will be reported as NA.

Glance returns the same number of columns regardless of whether the model matrix is rank-deficient or not. If so, entries in columns that no longer have a well-defined value are filled in with an NA of the appropriate type.

Usage

```
## S3 method for class 'crr'
glance(x, ...)
```

Arguments

A crr object returned from cmprsk::crr().

122 glance.crr

Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in . . ., where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

Value

. . .

A tibble::tibble() with exactly one row and columns:

converged Logical indicating if the model fitting procedure was successful and converged.

df Degrees of freedom used by the model.

logLik The log-likelihood of the model. [stats::logLik()] may be a useful reference.

nobs Number of observations used.

statistic Test statistic.

See Also

```
glance(), cmprsk::crr()
Other cmprsk tidiers: tidy.crr()
```

```
library(cmprsk)

# time to loco-regional failure (lrf)
lrf_time <- rexp(100)
lrf_event <- sample(0:2, 100, replace = TRUE)
trt <- sample(0:1, 100, replace = TRUE)
strt <- sample(1:2, 100, replace = TRUE)

# fit model
x <- crr(lrf_time, lrf_event, cbind(trt, strt))

# summarize model fit with tidiers
tidy(x, conf.int = TRUE)
glance(x)</pre>
```

glance.cv.glmnet 123

glance.cv.glmnet

Glance at a(n) cv.glmnet object

Description

Glance accepts a model object and returns a tibble::tibble() with exactly one row of model summaries. The summaries are typically goodness of fit measures, p-values for hypothesis tests on residuals, or model convergence information.

Glance never returns information from the original call to the modeling function. This includes the name of the modeling function or any arguments passed to the modeling function.

Glance does not calculate summary measures. Rather, it farms out these computations to appropriate methods and gathers the results together. Sometimes a goodness of fit measure will be undefined. In these cases the measure will be reported as NA.

Glance returns the same number of columns regardless of whether the model matrix is rank-deficient or not. If so, entries in columns that no longer have a well-defined value are filled in with an NA of the appropriate type.

Usage

```
## S3 method for class 'cv.glmnet'
glance(x, ...)
```

Arguments

x A cv.glmnet object returned from glmnet::cv.glmnet().

Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in . . . , where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

Value

A tibble::tibble() with exactly one row and columns:

lambda.1se The value of the penalization parameter lambda that results in the sparsest model while remaining within one standard error of the minimum loss.

lambda.min The value of the penalization parameter lambda that achieved minimum loss as

estimated by cross validation.

nobs Number of observations used.

124 glance.cv.glmnet

See Also

```
glance(), glmnet::cv.glmnet()
Other glmnet tidiers: glance.glmnet(), tidy.cv.glmnet(), tidy.glmnet()
```

```
# load libraries for models and data
library(glmnet)
set.seed(27)
nobs <- 100
nvar <- 50
real <- 5
x <- matrix(rnorm(nobs * nvar), nobs, nvar)</pre>
beta <- c(rnorm(real, 0, 1), rep(0, nvar - real))
y \leftarrow c(t(beta) %*% t(x)) + rnorm(nvar, sd = 3)
cvfit1 <- cv.glmnet(x, y)</pre>
tidy(cvfit1)
glance(cvfit1)
library(ggplot2)
tidied_cv <- tidy(cvfit1)</pre>
glance_cv <- glance(cvfit1)</pre>
# plot of MSE as a function of lambda
g <- ggplot(tidied_cv, aes(lambda, estimate)) +</pre>
  geom_line() +
  scale_x_log10()
g
# plot of MSE as a function of lambda with confidence ribbon
g <- g + geom_ribbon(aes(ymin = conf.low, ymax = conf.high), alpha = .25)</pre>
# plot of MSE as a function of lambda with confidence ribbon and choices
# of minimum lambda marked
g <- g +
  geom_vline(xintercept = glance_cv$lambda.min) +
  geom_vline(xintercept = glance_cv$lambda.1se, lty = 2)
g
# plot of number of zeros for each choice of lambda
ggplot(tidied_cv, aes(lambda, nzero)) +
  geom_line() +
  scale_x_log10()
```

glance.drc 125

```
# coefficient plot with min lambda shown
tidied <- tidy(cvfit1$glmnet.fit)

ggplot(tidied, aes(lambda, estimate, group = term)) +
    scale_x_log10() +
    geom_line() +
    geom_vline(xintercept = glance_cv$lambda.min) +
    geom_vline(xintercept = glance_cv$lambda.1se, lty = 2)</pre>
```

glance.drc

Glance at a(n) drc object

Description

Glance accepts a model object and returns a tibble::tibble() with exactly one row of model summaries. The summaries are typically goodness of fit measures, p-values for hypothesis tests on residuals, or model convergence information.

Glance never returns information from the original call to the modeling function. This includes the name of the modeling function or any arguments passed to the modeling function.

Glance does not calculate summary measures. Rather, it farms out these computations to appropriate methods and gathers the results together. Sometimes a goodness of fit measure will be undefined. In these cases the measure will be reported as NA.

Glance returns the same number of columns regardless of whether the model matrix is rank-deficient or not. If so, entries in columns that no longer have a well-defined value are filled in with an NA of the appropriate type.

Usage

```
## S3 method for class 'drc'
glance(x, ...)
```

Arguments

x A dro

A drc object produced by a call to drc::drm().

. .

Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in . . . , where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

126 glance.ergm

Value

A tibble::tibble() with exactly one row and columns:

AIC Akaike's Information Criterion for the model.

BIC Bayesian Information Criterion for the model.

df.residual Residual degrees of freedom.

logLik The log-likelihood of the model. [stats::logLik()] may be a useful reference.

AICc AIC corrected for small samples

See Also

```
glance(), drc::drm()
Other drc tidiers: augment.drc(), tidy.drc()
```

Examples

```
# load libraries for models and data
library(drc)

# fit model
mod <- drm(dead / total ~ conc, type,
   weights = total, data = selenium, fct = LL.2(), type = "binomial"
)

# summarize model fit with tidiers
tidy(mod)
tidy(mod, conf.int = TRUE)
glance(mod)
augment(mod, selenium)</pre>
```

glance.ergm

Glance at a(n) ergm object

Description

Glance accepts a model object and returns a tibble::tibble() with exactly one row of model summaries. The summaries are typically goodness of fit measures, p-values for hypothesis tests on residuals, or model convergence information.

Glance never returns information from the original call to the modeling function. This includes the name of the modeling function or any arguments passed to the modeling function.

Glance does not calculate summary measures. Rather, it farms out these computations to appropriate methods and gathers the results together. Sometimes a goodness of fit measure will be undefined. In these cases the measure will be reported as NA.

glance.ergm 127

Glance returns the same number of columns regardless of whether the model matrix is rank-deficient or not. If so, entries in columns that no longer have a well-defined value are filled in with an NA of the appropriate type.

Usage

```
## S3 method for class 'ergm'
glance(x, deviance = FALSE, mcmc = FALSE, ...)
```

Arguments

x An ergm object returned from a call to ergm::ergm().

deviance Logical indicating whether or not to report null and residual deviance for the

model, as well as degrees of freedom. Defaults to FALSE.

mcmc Logical indicating whether or not to report MCMC interval, burn-in and sample

size used to estimate the model. Defaults to FALSE.

... Additional arguments to pass to ergm::summary(). Cautionary note: Mis-

specified arguments may be silently ignored.

Value

glance.ergm returns a one-row tibble with the columns

independence Whether the model assumed dyadic independence

iterations The number of MCMLE iterations performed before convergence

logLik If applicable, the log-likelihood associated with the model

AIC The Akaike Information Criterion

BIC The Bayesian Information Criterion

If deviance = TRUE, and if the model supports it, the tibble will also contain the columns

null.deviance The null deviance of the model

df.null The degrees of freedom of the null deviance

residual.deviance

The residual deviance of the model

df.residual The degrees of freedom of the residual deviance

See Also

```
glance(), ergm::ergm(), ergm::summary.ergm()
Other ergm tidiers: tidy.ergm()
```

128 glance.factanal

glance.factanal

Glance at a(n) factanal object

Description

Glance accepts a model object and returns a tibble::tibble() with exactly one row of model summaries. The summaries are typically goodness of fit measures, p-values for hypothesis tests on residuals, or model convergence information.

Glance never returns information from the original call to the modeling function. This includes the name of the modeling function or any arguments passed to the modeling function.

Glance does not calculate summary measures. Rather, it farms out these computations to appropriate methods and gathers the results together. Sometimes a goodness of fit measure will be undefined. In these cases the measure will be reported as NA.

Glance returns the same number of columns regardless of whether the model matrix is rank-deficient or not. If so, entries in columns that no longer have a well-defined value are filled in with an NA of the appropriate type.

Usage

```
## S3 method for class 'factanal' glance(x, ...)
```

Arguments

x A factanal object created by stats::factanal().

Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in . . . , where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

Value

A tibble::tibble() with exactly one row and columns:

converged Logical indicating if the model fitting procedure was successful and converged.

df Degrees of freedom used by the model.

method Which method was used.

n The total number of observations.

n. factors The number of fitted factors.

glance.factanal 129

```
nobs Number of observations used.

p.value P-value corresponding to the test statistic.

statistic Test statistic.

total.variance Total cumulative proportion of variance accounted for by all factors.
```

See Also

```
glance(), stats::factanal()
Other factanal tidiers: augment.factanal(), tidy.factanal()
```

```
set.seed(123)
# generate data
library(dplyr)
library(purrr)
m1 <- tibble(
  v1 = c(1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 3, 3, 3, 3, 3, 4, 5, 6),
  v2 = c(1, 2, 1, 1, 1, 1, 2, 1, 2, 1, 3, 4, 3, 3, 3, 4, 6, 5),
  v3 = c(3, 3, 3, 3, 3, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 5, 4, 6),
  v4 = c(3, 3, 4, 3, 3, 1, 1, 2, 1, 1, 1, 1, 2, 1, 1, 5, 6, 4),
  v5 = c(1, 1, 1, 1, 1, 3, 3, 3, 3, 1, 1, 1, 1, 1, 6, 4, 5),
  v6 = c(1, 1, 1, 2, 1, 3, 3, 3, 4, 3, 1, 1, 1, 2, 1, 6, 5, 4)
)
# new data
m2 <- map_dfr(m1, rev)</pre>
# factor analysis objects
fit1 <- factanal(m1, factors = 3, scores = "Bartlett")</pre>
fit2 <- factanal(m1, factors = 3, scores = "regression")</pre>
# tidying the object
tidy(fit1)
tidy(fit2)
# augmented dataframe
augment(fit1)
augment(fit2)
# augmented dataframe (with new data)
augment(fit1, data = m2)
augment(fit2, data = m2)
```

130 glance.felm

glance.felm

Glance at a(n) felm object

Description

Glance accepts a model object and returns a tibble::tibble() with exactly one row of model summaries. The summaries are typically goodness of fit measures, p-values for hypothesis tests on residuals, or model convergence information.

Glance never returns information from the original call to the modeling function. This includes the name of the modeling function or any arguments passed to the modeling function.

Glance does not calculate summary measures. Rather, it farms out these computations to appropriate methods and gathers the results together. Sometimes a goodness of fit measure will be undefined. In these cases the measure will be reported as NA.

Glance returns the same number of columns regardless of whether the model matrix is rank-deficient or not. If so, entries in columns that no longer have a well-defined value are filled in with an NA of the appropriate type.

Usage

```
## S3 method for class 'felm'
glance(x, ...)
```

Arguments

. . .

x A felm object returned from lfe::felm().

Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in . . . , where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

Value

A tibble::tibble() with exactly one row and columns:

adj.r.squared Adjusted R squared statistic, which is like the R squared statistic except taking

degrees of freedom into account.

df Degrees of freedom used by the model.

df.residual Residual degrees of freedom.

Number of observations used.

p. value P-value corresponding to the test statistic.

glance.fitdistr 131

r. squared R squared statistic, or the percent of variation explained by the model. Also

known as the coefficient of determination.

sigma Estimated standard error of the residuals.

statistic Test statistic.

```
# load libraries for models and data
library(lfe)
# use built-in `airquality` dataset
head(airquality)
# no FEs; same as lm()
est0 <- felm(Ozone ~ Temp + Wind + Solar.R, airquality)</pre>
# summarize model fit with tidiers
tidy(est0)
augment(est0)
# add month fixed effects
est1 <- felm(Ozone ~ Temp + Wind + Solar.R | Month, airquality)</pre>
# summarize model fit with tidiers
tidy(est1)
tidy(est1, fe = TRUE)
augment(est1)
glance(est1)
# the "se.type" argument can be used to switch out different standard errors
# types on the fly. In turn, this can be useful exploring the effect of
# different error structures on model inference.
tidy(est1, se.type = "iid")
tidy(est1, se.type = "robust")
# add clustered SEs (also by month)
est2 <- felm(Ozone ~ Temp + Wind + Solar.R | Month | 0 | Month, airquality)</pre>
# summarize model fit with tidiers
tidy(est2, conf.int = TRUE)
tidy(est2, conf.int = TRUE, se.type = "cluster")
tidy(est2, conf.int = TRUE, se.type = "robust")
tidy(est2, conf.int = TRUE, se.type = "iid")
```

132 glance.fitdistr

Description

Glance accepts a model object and returns a tibble::tibble() with exactly one row of model summaries. The summaries are typically goodness of fit measures, p-values for hypothesis tests on residuals, or model convergence information.

Glance never returns information from the original call to the modeling function. This includes the name of the modeling function or any arguments passed to the modeling function.

Glance does not calculate summary measures. Rather, it farms out these computations to appropriate methods and gathers the results together. Sometimes a goodness of fit measure will be undefined. In these cases the measure will be reported as NA.

Glance returns the same number of columns regardless of whether the model matrix is rank-deficient or not. If so, entries in columns that no longer have a well-defined value are filled in with an NA of the appropriate type.

Usage

```
## S3 method for class 'fitdistr'
glance(x, ...)
```

Arguments

x A fitdistr object returned by MASS::fitdistr().

Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in . . . , where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

Value

A tibble::tibble() with exactly one row and columns:

AIC Akaike's Information Criterion for the model.

BIC Bayesian Information Criterion for the model.

logLik The log-likelihood of the model. [stats::logLik()] may be a useful reference.

nobs Number of observations used.

See Also

```
tidy(), MASS::fitdistr()
Other fitdistr tidiers: tidy.fitdistr()
```

glance.fixest 133

Examples

```
# load libraries for models and data
library(MASS)

# generate data
set.seed(2015)
x <- rnorm(100, 5, 2)

# fit models
fit <- fitdistr(x, dnorm, list(mean = 3, sd = 1))

# summarize model fit with tidiers
tidy(fit)
glance(fit)</pre>
```

glance.fixest

Glance at a(n) fixest object

Description

Glance accepts a model object and returns a tibble::tibble() with exactly one row of model summaries. The summaries are typically goodness of fit measures, p-values for hypothesis tests on residuals, or model convergence information.

Glance never returns information from the original call to the modeling function. This includes the name of the modeling function or any arguments passed to the modeling function.

Glance does not calculate summary measures. Rather, it farms out these computations to appropriate methods and gathers the results together. Sometimes a goodness of fit measure will be undefined. In these cases the measure will be reported as NA.

Glance returns the same number of columns regardless of whether the model matrix is rank-deficient or not. If so, entries in columns that no longer have a well-defined value are filled in with an NA of the appropriate type.

Usage

```
## S3 method for class 'fixest'
glance(x, ...)
```

Arguments

x A fixest object returned from any of the fixest estimators

Additional arguments passed to summary and confint. Important arguments are se and cluster. Other arguments are dof, exact_dof, forceCovariance, and keepBounded. See summary.fixest.

134 glance.fixest

Value

A tibble::tibble() with exactly one row and columns:

adj.r.squared Adjusted R squared statistic, which is like the R squared statistic except taking

degrees of freedom into account.

AIC Akaike's Information Criterion for the model.

BIC Bayesian Information Criterion for the model.

logLik The log-likelihood of the model. [stats::logLik()] may be a useful reference.

nobs Number of observations used.

pseudo.r.squared

Like the R squared statistic, but for situations when the R squared statistic isn't

defined.

r.squared R squared statistic, or the percent of variation explained by the model. Also

known as the coefficient of determination.

sigma Estimated standard error of the residuals.

within.r.squared

R squared within fixed-effect groups.

Note

All columns listed below will be returned, but some will be NA, depending on the type of model estimated. sigma, r.squared, adj.r.squared, and within.r.squared will be NA for any model other than feols. pseudo.r.squared will be NA for feols.

```
# load libraries for models and data
library(fixest)

gravity <-
    feols(
    log(Euros) ~ log(dist_km) | Origin + Destination + Product + Year, trade
)

tidy(gravity)
glance(gravity)
augment(gravity, trade)

# to get robust or clustered SEs, users can either:

# 1) specify the arguments directly in the `tidy()` call

tidy(gravity, conf.int = TRUE, cluster = c("Product", "Year"))

tidy(gravity, conf.int = TRUE, se = "threeway")

# 2) or, feed tidy() a summary.fixest object that has already accepted
# these arguments</pre>
```

glance.Gam 135

```
gravity_summ <- summary(gravity, cluster = c("Product", "Year"))
tidy(gravity_summ, conf.int = TRUE)
# approach (1) is preferred.</pre>
```

glance.Gam

Glance at a(n) Gam object

Description

Glance accepts a model object and returns a tibble::tibble() with exactly one row of model summaries. The summaries are typically goodness of fit measures, p-values for hypothesis tests on residuals, or model convergence information.

Glance never returns information from the original call to the modeling function. This includes the name of the modeling function or any arguments passed to the modeling function.

Glance does not calculate summary measures. Rather, it farms out these computations to appropriate methods and gathers the results together. Sometimes a goodness of fit measure will be undefined. In these cases the measure will be reported as NA.

Glance returns the same number of columns regardless of whether the model matrix is rank-deficient or not. If so, entries in columns that no longer have a well-defined value are filled in with an NA of the appropriate type.

Usage

```
## S3 method for class 'Gam'
glance(x, ...)
```

Arguments

х

A Gam object returned from a call to gam::gam().

. . .

Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in . . . , where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

Details

Glance at gam objects created by calls to mgcv::gam() with glance.gam().

136 glance.gam

Value

A tibble::tibble() with exactly one row and columns:

AIC Akaike's Information Criterion for the model.

BIC Bayesian Information Criterion for the model.

deviance Deviance of the model.

df Degrees of freedom used by the model.

df.residual Residual degrees of freedom.

logLik The log-likelihood of the model. [stats::logLik()] may be a useful reference.

nobs Number of observations used.

See Also

```
glance(), gam::gam()
Other gam tidiers: tidy.Gam()
```

glance.gam

Glance at a(n) gam object

Description

Glance accepts a model object and returns a tibble::tibble() with exactly one row of model summaries. The summaries are typically goodness of fit measures, p-values for hypothesis tests on residuals, or model convergence information.

Glance never returns information from the original call to the modeling function. This includes the name of the modeling function or any arguments passed to the modeling function.

Glance does not calculate summary measures. Rather, it farms out these computations to appropriate methods and gathers the results together. Sometimes a goodness of fit measure will be undefined. In these cases the measure will be reported as NA.

Glance returns the same number of columns regardless of whether the model matrix is rank-deficient or not. If so, entries in columns that no longer have a well-defined value are filled in with an NA of the appropriate type.

Usage

```
## S3 method for class 'gam'
glance(x, ...)
```

glance.gam 137

Arguments

. . .

x A gam object returned from a call to mgcv::gam().

Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in . . ., where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

Value

A tibble::tibble() with exactly one row and columns:

adj.r.squared Adjusted R squared statistic, which is like the R squared statistic except taking

degrees of freedom into account.

AIC Akaike's Information Criterion for the model.

BIC Bayesian Information Criterion for the model.

deviance Deviance of the model.

df Degrees of freedom used by the model.

df.residual Residual degrees of freedom.

logLik The log-likelihood of the model. [stats::logLik()] may be a useful reference.

nobs Number of observations used.

npar Number of parameters in the model.

See Also

```
glance(), mgcv::gam()
Other mgcv tidiers: tidy.gam()
```

```
# load libraries for models and data
library(mgcv)

# fit model
g <- gam(mpg ~ s(hp) + am + qsec, data = mtcars)

# summarize model fit with tidiers
tidy(g)
tidy(g, parametric = TRUE)
glance(g)
augment(g)</pre>
```

138 glance.garch

glance.garch	Tidy a(n) garch object

Description

Tidy summarizes information about the components of a model. A model component might be a single term in a regression, a single hypothesis, a cluster, or a class. Exactly what tidy considers to be a model component varies across models but is usually self-evident. If a model has several distinct types of components, you will need to specify which components to return.

Usage

```
## S3 method for class 'garch'
glance(x, test = c("box-ljung-test", "jarque-bera-test"), ...)
```

Arguments

A garch object returned by tseries::garch(). Х

test Character specification of which hypothesis test to use. The garch function

reports 2 hypothesis tests: Jarque-Bera to residuals and Box-Ljung to squared

residuals.

Additional arguments. Not used. Needed to match generic signature only. Cau-

tionary note: Misspelled arguments will be absorbed in . . . , where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed

using conf.level = 0.95. Two exceptions here are:

• tidy() methods will warn when supplied an exponentiate argument if it

will be ignored.

• augment() methods will warn when supplied a newdata argument if it will be ignored.

Value

A tibble::tibble() with exactly one row and columns:

AIC Akaike's Information Criterion for the model. BIC Bayesian Information Criterion for the model.

The log-likelihood of the model. [stats::logLik()] may be a useful reference. logLik

Which method was used. method

Number of observations used. nobs

p.value P-value corresponding to the test statistic.

Test statistic. statistic

Parameter field in the htest, typically degrees of freedom. parameter

glance.geeglm 139

See Also

```
glance(), tseries::garch(),[]
Other garch tidiers: tidy.garch()
```

glance.geeglm

Glance at a(n) geeglm object

Description

Glance accepts a model object and returns a tibble::tibble() with exactly one row of model summaries. The summaries are typically goodness of fit measures, p-values for hypothesis tests on residuals, or model convergence information.

Glance never returns information from the original call to the modeling function. This includes the name of the modeling function or any arguments passed to the modeling function.

Glance does not calculate summary measures. Rather, it farms out these computations to appropriate methods and gathers the results together. Sometimes a goodness of fit measure will be undefined. In these cases the measure will be reported as NA.

Glance returns the same number of columns regardless of whether the model matrix is rank-deficient or not. If so, entries in columns that no longer have a well-defined value are filled in with an NA of the appropriate type.

Usage

```
## S3 method for class 'geeglm'
glance(x, ...)
```

Arguments

x A geeglm object returned from a call to geepack::geeglm().

Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in . . ., where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

Value

A tibble::tibble() with exactly one row and columns:

alpha Estimated correlation parameter for geepack::geeglm.

df.residual Residual degrees of freedom.

140 glance.glm

```
gamma Estimated scale parameter for geepack::geeglm.
max.cluster.size
Max number of elements in clusters.
n.clusters Number of clusters.
```

See Also

```
glance(), geepack::geeglm()
```

Examples

```
# load modeling library
library(geepack)

# load data
data(state)

ds <- data.frame(state.region, state.x77)

# fit model
geefit <- geeglm(Income ~ Frost + Murder,
    id = state.region,
    data = ds,
    corstr = "exchangeable"
)

# summarize model fit with tidiers
tidy(geefit)
tidy(geefit, conf.int = TRUE)</pre>
```

glance.glm

Glance at a(n) glm object

Description

Glance accepts a model object and returns a tibble::tibble() with exactly one row of model summaries. The summaries are typically goodness of fit measures, p-values for hypothesis tests on residuals, or model convergence information.

Glance never returns information from the original call to the modeling function. This includes the name of the modeling function or any arguments passed to the modeling function.

Glance does not calculate summary measures. Rather, it farms out these computations to appropriate methods and gathers the results together. Sometimes a goodness of fit measure will be undefined. In these cases the measure will be reported as NA.

Glance returns the same number of columns regardless of whether the model matrix is rank-deficient or not. If so, entries in columns that no longer have a well-defined value are filled in with an NA of the appropriate type.

glance.glm 141

Usage

```
## S3 method for class 'glm'
glance(x, ...)
```

Arguments

x A glm object returned from stats::glm().

Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in . . . , where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

Value

A tibble::tibble() with exactly one row and columns:

AIC Akaike's Information Criterion for the model.

BIC Bayesian Information Criterion for the model.

deviance Deviance of the model.

df.null Degrees of freedom used by the null model.

df.residual Residual degrees of freedom.

logLik The log-likelihood of the model. [stats::logLik()] may be a useful reference.

nobs Number of observations used.
null.deviance Deviance of the null model.

See Also

```
stats::glm()
Other Im tidiers: augment.glm(), augment.lm(), glance.lm(), glance.summary.lm(), glance.svyglm(),
tidy.glm(), tidy.lm(), tidy.lm.beta(), tidy.mlm(), tidy.summary.lm()
```

```
g <- glm(am ~ mpg, mtcars, family = "binomial")
glance(g)</pre>
```

142 glance.glmnet

glance.glmnet

Glance at a(n) glmnet object

Description

Glance accepts a model object and returns a tibble::tibble() with exactly one row of model summaries. The summaries are typically goodness of fit measures, p-values for hypothesis tests on residuals, or model convergence information.

Glance never returns information from the original call to the modeling function. This includes the name of the modeling function or any arguments passed to the modeling function.

Glance does not calculate summary measures. Rather, it farms out these computations to appropriate methods and gathers the results together. Sometimes a goodness of fit measure will be undefined. In these cases the measure will be reported as NA.

Glance returns the same number of columns regardless of whether the model matrix is rank-deficient or not. If so, entries in columns that no longer have a well-defined value are filled in with an NA of the appropriate type.

Usage

```
## S3 method for class 'glmnet'
glance(x, ...)
```

Arguments

x A glmnet object returned from glmnet::glmnet().

.. Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in . . . , where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

Value

A tibble::tibble() with exactly one row and columns:

nobs Number of observations used.

npasses Total passes over the data across all lambda values.

nulldev Null deviance.

glance.glmRob 143

See Also

```
glance(), glmnet::glmnet()
Other glmnet tidiers: glance.cv.glmnet(), tidy.cv.glmnet(), tidy.glmnet()
```

Examples

```
# load libraries for models and data
library(glmnet)
set.seed(2014)
x \leftarrow matrix(rnorm(100 * 20), 100, 20)
y <- rnorm(100)
fit1 <- glmnet(x, y)</pre>
# summarize model fit with tidiers + visualization
tidy(fit1)
glance(fit1)
library(dplyr)
library(ggplot2)
tidied <- tidy(fit1) %>% filter(term != "(Intercept)")
ggplot(tidied, aes(step, estimate, group = term)) +
  geom_line()
ggplot(tidied, aes(lambda, estimate, group = term)) +
  geom_line() +
  scale_x_log10()
ggplot(tidied, aes(lambda, dev.ratio)) +
  geom_line()
# works for other types of regressions as well, such as logistic
g2 <- sample(1:2, 100, replace = TRUE)</pre>
fit2 <- glmnet(x, g2, family = "binomial")</pre>
tidy(fit2)
```

glance.glmRob

Glance at a(n) glmRob object

Description

Glance accepts a model object and returns a tibble::tibble() with exactly one row of model summaries. The summaries are typically goodness of fit measures, p-values for hypothesis tests on residuals, or model convergence information.

144 glance.glmRob

Glance never returns information from the original call to the modeling function. This includes the name of the modeling function or any arguments passed to the modeling function.

Glance does not calculate summary measures. Rather, it farms out these computations to appropriate methods and gathers the results together. Sometimes a goodness of fit measure will be undefined. In these cases the measure will be reported as NA.

Glance returns the same number of columns regardless of whether the model matrix is rank-deficient or not. If so, entries in columns that no longer have a well-defined value are filled in with an NA of the appropriate type.

Usage

```
## S3 method for class 'glmRob'
glance(x, ...)
```

Arguments

x A glmRob object returned from robust::glmRob().

Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in . . ., where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

Value

A tibble::tibble() with exactly one row and columns:

deviance Deviance of the model.

df.residual Residual degrees of freedom.

nobs Number of observations used.

null.deviance Deviance of the null model.

sigma Estimated standard error of the residuals.

See Also

```
robust::glmRob()
```

Other robust tidiers: augment.lmRob(), glance.lmRob(), tidy.glmRob(), tidy.lmRob()

glance.gmm 145

Examples

```
# load libraries for models and data
library(robust)

# fit model
gm <- glmRob(am ~ wt, data = mtcars, family = "binomial")

# summarize model fit with tidiers
tidy(gm)
glance(gm)</pre>
```

glance.gmm

Glance at a(n) gmm object

Description

Glance accepts a model object and returns a tibble::tibble() with exactly one row of model summaries. The summaries are typically goodness of fit measures, p-values for hypothesis tests on residuals, or model convergence information.

Glance never returns information from the original call to the modeling function. This includes the name of the modeling function or any arguments passed to the modeling function.

Glance does not calculate summary measures. Rather, it farms out these computations to appropriate methods and gathers the results together. Sometimes a goodness of fit measure will be undefined. In these cases the measure will be reported as NA.

Glance returns the same number of columns regardless of whether the model matrix is rank-deficient or not. If so, entries in columns that no longer have a well-defined value are filled in with an NA of the appropriate type.

Usage

```
## S3 method for class 'gmm'
glance(x, ...)
```

Arguments

Χ

A gmm object returned from gmm::gmm().

Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in . . . , where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

146 glance.gmm

Value

df Degrees of freedom used by the model.
df.residual Residual degrees of freedom.

A tibble::tibble() with exactly one row and columns:

nobs Residual degrees of freedom.

Number of observations used.

p. value P-value corresponding to the test statistic.

statistic Test statistic.

See Also

```
glance(), gmm::gmm()
Other gmm tidiers: tidy.gmm()
```

```
# load libraries for models and data
library(gmm)
# examples come from the "gmm" package
# CAPM test with GMM
data(Finance)
r <- Finance[1:300, 1:10]
rm <- Finance[1:300, "rm"]
rf <- Finance[1:300, "rf"]
z <- as.matrix(r - rf)</pre>
t <- nrow(z)
zm <- rm - rf
h \leftarrow matrix(zm, t, 1)
res \leftarrow gmm(z \sim zm, x = h)
# tidy result
tidy(res)
tidy(res, conf.int = TRUE)
tidy(res, conf.int = TRUE, conf.level = .99)
# coefficient plot
library(ggplot2)
library(dplyr)
tidy(res, conf.int = TRUE) %>%
  mutate(variable = reorder(term, estimate)) %>%
  ggplot(aes(estimate, variable)) +
  geom_point() +
  geom_errorbarh(aes(xmin = conf.low, xmax = conf.high)) +
  geom_vline(xintercept = 0, color = "red", lty = 2)
# from a function instead of a matrix
```

glance.ivreg 147

```
g <- function(theta, x) {</pre>
  e \leftarrow x[, 2:11] - theta[1] - (x[, 1] - theta[1]) %*% matrix(theta[2:11], 1, 10)
  gmat \leftarrow cbind(e, e * c(x[, 1]))
  return(gmat)
}
x <- as.matrix(cbind(rm, r))</pre>
res_black <- gmm(g, x = x, t0 = rep(0, 11))
tidy(res_black)
tidy(res_black, conf.int = TRUE)
# APT test with Fama-French factors and GMM
f1 <- zm
f2 <- Finance[1:300, "hml"] - rf</pre>
f3 <- Finance[1:300, "smb"] - rf
h <- cbind(f1, f2, f3)
res2 <- gmm(z \sim f1 + f2 + f3, x = h)
td2 <- tidy(res2, conf.int = TRUE)
td2
# coefficient plot
td2 %>%
  mutate(variable = reorder(term, estimate)) %>%
  ggplot(aes(estimate, variable)) +
  geom_point() +
  geom_errorbarh(aes(xmin = conf.low, xmax = conf.high)) +
  geom_vline(xintercept = 0, color = "red", lty = 2)
```

glance.ivreg

Glance at a(n) ivreg object

Description

Glance accepts a model object and returns a tibble::tibble() with exactly one row of model summaries. The summaries are typically goodness of fit measures, p-values for hypothesis tests on residuals, or model convergence information.

Glance never returns information from the original call to the modeling function. This includes the name of the modeling function or any arguments passed to the modeling function.

Glance does not calculate summary measures. Rather, it farms out these computations to appropriate methods and gathers the results together. Sometimes a goodness of fit measure will be undefined. In these cases the measure will be reported as NA.

Glance returns the same number of columns regardless of whether the model matrix is rank-deficient or not. If so, entries in columns that no longer have a well-defined value are filled in with an NA of the appropriate type.

148 glance.ivreg

Usage

```
## S3 method for class 'ivreg'
glance(x, diagnostics = FALSE, ...)
```

Arguments

x An ivreg object created by a call to AER::ivreg().

diagnostics Logical indicating whether or not to return the Wu-Hausman and Sargan diag-

nostic information.

Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in . . . , where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed

using conf.level = 0.95. Two exceptions here are:tidy() methods will warn when supplied an exponentiate argument if it

will be ignored.

 augment() methods will warn when supplied a newdata argument if it will be ignored.

Details

This tidier currently only supports ivreg-classed objects outputted by the AER package. The ivreg package also outputs objects of class ivreg, and will be supported in a later release.

Value

A tibble::tibble() with exactly one row and columns:

adj.r.squared Adjusted R squared statistic, which is like the R squared statistic except taking

degrees of freedom into account.

df Degrees of freedom used by the model.

df.residual Residual degrees of freedom.

Number of observations used.

r.squared R squared statistic, or the percent of variation explained by the model. Also

known as the coefficient of determination.

sigma Estimated standard error of the residuals.

statistic Wald test statistic.

p.value P-value for the Wald test.

Note

Beginning 0.7.0, glance.ivreg returns statistics for the Wu-Hausman test for endogeneity and the Sargan test of overidentifying restrictions. Sargan test values are returned as NA if the number of instruments is not greater than the number of endogenous regressors.

glance.kmeans 149

See Also

```
glance(), AER::ivreg()
Other ivreg tidiers: augment.ivreg(), tidy.ivreg()
```

Examples

```
# load libraries for models and data
library(AER)
# load data
data("CigarettesSW", package = "AER")
# fit model
ivr <- ivreg(</pre>
 log(packs) ~ income | population,
 data = CigarettesSW,
 subset = year == "1995"
)
# summarize model fit with tidiers
tidy(ivr)
tidy(ivr, conf.int = TRUE)
tidy(ivr, conf.int = TRUE, instruments = TRUE)
augment(ivr)
augment(ivr, data = CigarettesSW)
augment(ivr, newdata = CigarettesSW)
glance(ivr)
```

glance.kmeans

Glance at a(n) kmeans object

Description

Glance accepts a model object and returns a tibble::tibble() with exactly one row of model summaries. The summaries are typically goodness of fit measures, p-values for hypothesis tests on residuals, or model convergence information.

Glance never returns information from the original call to the modeling function. This includes the name of the modeling function or any arguments passed to the modeling function.

Glance does not calculate summary measures. Rather, it farms out these computations to appropriate methods and gathers the results together. Sometimes a goodness of fit measure will be undefined. In these cases the measure will be reported as NA.

Glance returns the same number of columns regardless of whether the model matrix is rank-deficient or not. If so, entries in columns that no longer have a well-defined value are filled in with an NA of the appropriate type.

150 glance.kmeans

Usage

```
## S3 method for class 'kmeans' glance(x, ...)
```

Arguments

x A kmeans object created by stats::kmeans().

Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in . . . , where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

Value

A tibble::tibble() with exactly one row and columns:

betweenss The total between-cluster sum of squares.

iter Iterations of algorithm/fitting procedure completed.

tot.withinss The total within-cluster sum of squares.

totss The total sum of squares.

See Also

```
glance(), stats::kmeans()
Other kmeans tidiers: augment.kmeans(), tidy.kmeans()
```

```
library(cluster)
library(modeldata)
library(dplyr)

data(hpc_data)

x <- hpc_data[, 2:5]

fit <- pam(x, k = 4)

tidy(fit)
glance(fit)
augment(fit, x)</pre>
```

glance.lavaan 151

glance.lavaan Glance at a(n) lavaan object

Description

Glance accepts a model object and returns a tibble::tibble() with exactly one row of model summaries. The summaries are typically goodness of fit measures, p-values for hypothesis tests on residuals, or model convergence information.

Glance never returns information from the original call to the modeling function. This includes the name of the modeling function or any arguments passed to the modeling function.

Glance does not calculate summary measures. Rather, it farms out these computations to appropriate methods and gathers the results together. Sometimes a goodness of fit measure will be undefined. In these cases the measure will be reported as NA.

Glance returns the same number of columns regardless of whether the model matrix is rank-deficient or not. If so, entries in columns that no longer have a well-defined value are filled in with an NA of the appropriate type.

Usage

```
## S3 method for class 'lavaan'
glance(x, ...)
```

Arguments

x A lavaan object, such as those returned from lavaan::cfa(), and lavaan::sem().

Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in . . . , where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

Value

A one-row tibble::tibble with columns:

chisq Model chi squared

npar Number of parameters in the model
rmsea Root mean square error of approximation

rmsea.conf.high

95 percent upper bound on RMSEA

152 glance.lavaan

srmr	Standardised root mean residual
agfi	Adjusted goodness of fit
cfi	Comparative fit index
tli	Tucker Lewis index
AIC	Akaike information criterion
BIC	Bayesian information criterion
ngroups	Number of groups in model
nobs	Number of observations included
norig	Number of observation in the original dataset
nexcluded	Number of excluded observations
converged	Logical - Did the model converge
estimator	Estimator used
missing_method	Method for eliminating missing data

For further recommendations on reporting SEM and CFA models see Schreiber, J. B. (2017). Update to core reporting practices in structural equation modeling. Research in Social and Administrative Pharmacy, 13(3), 634-643. https://doi.org/10.1016/j.sapharm.2016.06.006

See Also

```
glance(), lavaan::cfa(), lavaan::sem(), lavaan::fitmeasures()
Other lavaan tidiers: tidy.lavaan()
```

```
library(lavaan)
# fit model
cfa.fit <- cfa(
   "F =~ x1 + x2 + x3 + x4 + x5",
   data = HolzingerSwineford1939, group = "school"
)
# summarize model fit with tidiers
glance(cfa.fit)</pre>
```

glance.lm 153

glance.lm

Glance at a(n) lm object

Description

Glance accepts a model object and returns a tibble::tibble() with exactly one row of model summaries. The summaries are typically goodness of fit measures, p-values for hypothesis tests on residuals, or model convergence information.

Glance never returns information from the original call to the modeling function. This includes the name of the modeling function or any arguments passed to the modeling function.

Glance does not calculate summary measures. Rather, it farms out these computations to appropriate methods and gathers the results together. Sometimes a goodness of fit measure will be undefined. In these cases the measure will be reported as NA.

Glance returns the same number of columns regardless of whether the model matrix is rank-deficient or not. If so, entries in columns that no longer have a well-defined value are filled in with an NA of the appropriate type.

Usage

```
## S3 method for class 'lm'
glance(x, ...)
```

Arguments

. . .

x An lm object created by stats::lm().

Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in . . . , where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

Value

A tibble::tibble() with exactly one row and columns:

adj.r.squared Adjusted R squared statistic, which is like the R squared statistic except taking

degrees of freedom into account.

AIC Akaike's Information Criterion for the model.

BIC Bayesian Information Criterion for the model.

deviance Deviance of the model.

df.residual Residual degrees of freedom.

154 glance.lm

logLik The log-likelihood of the model. [stats::logLik()] may be a useful reference.

nobs Number of observations used.

p. value P-value corresponding to the test statistic.

r.squared R squared statistic, or the percent of variation explained by the model. Also

known as the coefficient of determination.

sigma Estimated standard error of the residuals.

statistic Test statistic.

df The degrees for freedom from the numerator of the overall F-statistic. This

is new in broom 0.7.0. Previously, this reported the rank of the design matrix, which is one more than the numerator degrees of freedom of the overall

F-statistic.

See Also

```
glance(), glance.summary.lm()
Other Im tidiers: augment.glm(), augment.lm(), glance.glm(), glance.summary.lm(), glance.svyglm(), tidy.glm(), tidy.lm(), tidy.lm(), tidy.mlm(), tidy.summary.lm()
```

```
library(ggplot2)
library(dplyr)
mod <- lm(mpg ~ wt + qsec, data = mtcars)</pre>
tidy(mod)
glance(mod)
# coefficient plot
d <- tidy(mod, conf.int = TRUE)</pre>
ggplot(d, aes(estimate, term, xmin = conf.low, xmax = conf.high, height = 0)) +
  geom_point() +
  geom_vline(xintercept = 0, lty = 4) +
  geom_errorbarh()
# aside: There are tidy() and glance() methods for lm.summary objects too.
# this can be useful when you want to conserve memory by converting large lm
# objects into their leaner summary.lm equivalents.
s <- summary(mod)</pre>
tidy(s, conf.int = TRUE)
glance(s)
augment(mod)
augment(mod, mtcars, interval = "confidence")
# predict on new data
newdata <- mtcars %>%
```

glance.lmodel2 155

```
head(6) %>%
 mutate(wt = wt + 1)
augment(mod, newdata = newdata)
# ggplot2 example where we also construct 95% prediction interval
# simpler bivariate model since we're plotting in 2D
mod2 <- lm(mpg ~ wt, data = mtcars)</pre>
au <- augment(mod2, newdata = newdata, interval = "prediction")</pre>
ggplot(au, aes(wt, mpg)) +
 geom_point() +
 geom\_line(aes(y = .fitted)) +
 geom_ribbon(aes(ymin = .lower, ymax = .upper), col = NA, alpha = 0.3)
# predict on new data without outcome variable. Output does not include .resid
newdata <- newdata %>%
 select(-mpg)
augment(mod, newdata = newdata)
au <- augment(mod, data = mtcars)</pre>
ggplot(au, aes(.hat, .std.resid)) +
 geom_vline(size = 2, colour = "white", xintercept = 0) +
 geom_hline(size = 2, colour = "white", yintercept = 0) +
 geom_point() +
 geom_smooth(se = FALSE)
plot(mod, which = 6)
ggplot(au, aes(.hat, .cooksd)) +
 geom_vline(xintercept = 0, colour = NA) +
 geom_abline(slope = seq(0, 3, by = 0.5), colour = "white") +
 geom_smooth(se = FALSE) +
 geom_point()
# column-wise models
a <- matrix(rnorm(20), nrow = 10)
b <- a + rnorm(length(a))</pre>
result <- lm(b \sim a)
tidy(result)
```

156 glance.Imodel2

Description

Glance accepts a model object and returns a tibble::tibble() with exactly one row of model summaries. The summaries are typically goodness of fit measures, p-values for hypothesis tests on residuals, or model convergence information.

Glance never returns information from the original call to the modeling function. This includes the name of the modeling function or any arguments passed to the modeling function.

Glance does not calculate summary measures. Rather, it farms out these computations to appropriate methods and gathers the results together. Sometimes a goodness of fit measure will be undefined. In these cases the measure will be reported as NA.

Glance returns the same number of columns regardless of whether the model matrix is rank-deficient or not. If so, entries in columns that no longer have a well-defined value are filled in with an NA of the appropriate type.

Usage

```
## S3 method for class 'lmodel2'
glance(x, ...)
```

Arguments

. . .

x A lmodel2 object returned by lmodel2::lmodel2().

Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in . . . , where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

Value

A tibble::tibble() with exactly one row and columns:

nobs Number of observations used.

p. value P-value corresponding to the test statistic.

r.squared R squared statistic, or the percent of variation explained by the model. Also

known as the coefficient of determination.

theta Angle between OLS lines ' $lm(y \sim x)$ ' and ' $lm(x \sim y)$ '

H Statistic for computing confidence interval of major axis slope

See Also

```
glance(), lmodel2::lmodel2()
Other lmodel2 tidiers: tidy.lmodel2()
```

glance.lmRob 157

Examples

```
# load libraries for models and data
library(lmodel2)

data(mod2ex2)
Ex2.res <- lmodel2(Prey ~ Predators, data = mod2ex2, "relative", "relative", 99)
Ex2.res

# summarize model fit with tidiers + visualization
tidy(Ex2.res)
glance(Ex2.res)

# this allows coefficient plots with ggplot2
library(ggplot2)

ggplot(tidy(Ex2.res), aes(estimate, term, color = method)) +
    geom_point() +
    geom_errorbarh(aes(xmin = conf.low, xmax = conf.high)) +
    geom_errorbarh(aes(xmin = conf.low, xmax = conf.high))</pre>
```

glance.lmRob

Glance at a(n) lmRob object

Description

Glance accepts a model object and returns a tibble::tibble() with exactly one row of model summaries. The summaries are typically goodness of fit measures, p-values for hypothesis tests on residuals, or model convergence information.

Glance never returns information from the original call to the modeling function. This includes the name of the modeling function or any arguments passed to the modeling function.

Glance does not calculate summary measures. Rather, it farms out these computations to appropriate methods and gathers the results together. Sometimes a goodness of fit measure will be undefined. In these cases the measure will be reported as NA.

Glance returns the same number of columns regardless of whether the model matrix is rank-deficient or not. If so, entries in columns that no longer have a well-defined value are filled in with an NA of the appropriate type.

Usage

```
## S3 method for class 'lmRob'
glance(x, ...)
```

158 glance.lmRob

Arguments

x A lmRob object returned from robust::lmRob().

Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in . . . , where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

Value

A tibble::tibble() with exactly one row and columns:

deviance Deviance of the model.

df.residual Residual degrees of freedom.

nobs Number of observations used.

r.squared R squared statistic, or the percent of variation explained by the model. Also

known as the coefficient of determination.

sigma Estimated standard error of the residuals.

See Also

```
robust::lmRob()
Other robust tidiers: augment.lmRob(), glance.glmRob(), tidy.glmRob(), tidy.lmRob()
```

```
# load modeling library
library(robust)

# fit model
m <- lmRob(mpg ~ wt, data = mtcars)

# summarize model fit with tidiers
tidy(m)
augment(m)
glance(m)</pre>
```

glance.lmrob 159

glance.lmrob

Glance at a(n) lmrob object

Description

Glance accepts a model object and returns a tibble::tibble() with exactly one row of model summaries. The summaries are typically goodness of fit measures, p-values for hypothesis tests on residuals, or model convergence information.

Glance never returns information from the original call to the modeling function. This includes the name of the modeling function or any arguments passed to the modeling function.

Glance does not calculate summary measures. Rather, it farms out these computations to appropriate methods and gathers the results together. Sometimes a goodness of fit measure will be undefined. In these cases the measure will be reported as NA.

Glance returns the same number of columns regardless of whether the model matrix is rank-deficient or not. If so, entries in columns that no longer have a well-defined value are filled in with an NA of the appropriate type.

Usage

```
## S3 method for class 'lmrob'
glance(x, ...)
```

Arguments

x A lmrob object returned from robustbase::lmrob().

Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in . . . , where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

Details

For tidiers for robust models from the MASS package see tidy.rlm().

Value

A tibble::tibble() with exactly one row and columns:

df.residual Residual degrees of freedom.

r.squared R squared statistic, or the percent of variation explained by the model. Also

known as the coefficient of determination.

sigma Estimated standard error of the residuals.

160 glance.margins

See Also

```
robustbase::lmrob()
```

Other robustbase tidiers: augment.glmrob(), augment.lmrob(), tidy.glmrob(), tidy.lmrob()

Examples

```
if (requireNamespace("robustbase", quietly = TRUE)) {
 # load libraries for models and data
 library(robustbase)
 data(coleman)
 set.seed(0)
 m <- lmrob(Y ~ ., data = coleman)</pre>
 tidy(m)
 augment(m)
 glance(m)
 data(carrots)
 Rfit <- glmrob(cbind(success, total - success) ~ logdose + block,
    family = binomial, data = carrots, method = "Mqle",
   control = glmrobMqle.control(tcc = 1.2)
 tidy(Rfit)
 augment(Rfit)
}
```

glance.margins

Glance at a(n) margins object

Description

Glance accepts a model object and returns a tibble::tibble() with exactly one row of model summaries. The summaries are typically goodness of fit measures, p-values for hypothesis tests on residuals, or model convergence information.

Glance never returns information from the original call to the modeling function. This includes the name of the modeling function or any arguments passed to the modeling function.

Glance does not calculate summary measures. Rather, it farms out these computations to appropriate methods and gathers the results together. Sometimes a goodness of fit measure will be undefined. In these cases the measure will be reported as NA.

Glance returns the same number of columns regardless of whether the model matrix is rank-deficient or not. If so, entries in columns that no longer have a well-defined value are filled in with an NA of the appropriate type.

glance.margins 161

Usage

```
## S3 method for class 'margins'
glance(x, ...)
```

Arguments

x A margins object returned from margins::margins().

Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in . . . , where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

Value

A tibble::tibble() with exactly one row and columns:

adj.r.squared Adjusted R squared statistic, which is like the R squared statistic except taking

degrees of freedom into account.

df Degrees of freedom used by the model.

df.residual Residual degrees of freedom.

nobs Number of observations used.

p.value P-value corresponding to the test statistic.

r. squared R squared statistic, or the percent of variation explained by the model. Also

known as the coefficient of determination.

sigma Estimated standard error of the residuals.

statistic Test statistic.

```
# load libraries for models and data
library(margins)

# example 1: logit model
mod_log <- glm(am ~ cyl + hp + wt, data = mtcars, family = binomial)

# get tidied "naive" model coefficients
tidy(mod_log)

# convert to marginal effects with margins()
marg_log <- margins(mod_log)</pre>
```

162 glance.Mclust

```
# get tidied marginal effects
tidy(marg_log)
tidy(marg_log, conf.int = TRUE)
# requires running the underlying model again. quick for this example
glance(marg_log)
# augmenting `margins` outputs isn't supported, but
# you can get the same info by running on the underlying model
augment(mod_log)
# example 2: threeway interaction terms
mod_ie <- lm(mpg ~ wt * cyl * disp, data = mtcars)</pre>
# get tidied "naive" model coefficients
tidy(mod_ie)
# convert to marginal effects with margins()
marg_ie0 <- margins(mod_ie)</pre>
# get tidied marginal effects
tidy(marg_ie0)
glance(marg_ie0)
# marginal effects evaluated at specific values of a variable (here: cyl)
marg_ie1 \leftarrow margins(mod_ie, at = list(cyl = c(4,6,8)))
# summarize model fit with tidiers
tidy(marg_ie1)
# marginal effects of one interaction variable (here: wt), modulated at
# specific values of the two other interaction variables (here: cyl and drat)
marg_ie2 <- margins(mod_ie,</pre>
                    variables = "wt",
                    at = list(cyl = c(4,6,8), drat = c(3, 3.5, 4)))
# summarize model fit with tidiers
tidy(marg_ie2)
```

glance.Mclust

Glance at a(n) Mclust object

Description

Glance accepts a model object and returns a tibble::tibble() with exactly one row of model summaries. The summaries are typically goodness of fit measures, p-values for hypothesis tests on residuals, or model convergence information.

Glance never returns information from the original call to the modeling function. This includes the name of the modeling function or any arguments passed to the modeling function.

glance.Mclust 163

Glance does not calculate summary measures. Rather, it farms out these computations to appropriate methods and gathers the results together. Sometimes a goodness of fit measure will be undefined. In these cases the measure will be reported as NA.

Glance returns the same number of columns regardless of whether the model matrix is rank-deficient or not. If so, entries in columns that no longer have a well-defined value are filled in with an NA of the appropriate type.

Usage

```
## S3 method for class 'Mclust'
glance(x, ...)
```

Arguments

x An Mclust object return from mclust::Mclust().

... Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in ..., where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

Value

A tibble::tibble() with exactly one row and columns:

BIC Bayesian Information Criterion for the model.

df Degrees of freedom used by the model.

logLik The log-likelihood of the model. [stats::logLik()] may be a useful reference.

nobs Number of observations used.

model A string denoting the model type with optimal BIC

Number mixture components in optimal model

hypvol If the other model contains a noise component, the value of the hypervolume

parameter. Otherwise 'NA'.

```
# load library for models and data
library(mclust)

# load data manipulation libraries
library(dplyr)
library(tibble)
```

164 glance.mfx

```
library(purrr)
library(tidyr)
set.seed(27)
centers <- tibble(</pre>
  cluster = factor(1:3),
  # number points in each cluster
  num_points = c(100, 150, 50),
  # x1 coordinate of cluster center
  x1 = c(5, 0, -3),
  # x2 coordinate of cluster center
  x2 = c(-1, 1, -2)
points <- centers %>%
  mutate(
    x1 = map2(num\_points, x1, rnorm),
    x2 = map2(num_points, x2, rnorm)
  select(-num_points, -cluster) %>%
  unnest(c(x1, x2))
# fit model
m <- Mclust(points)</pre>
# summarize model fit with tidiers
tidy(m)
augment(m, points)
glance(m)
```

glance.mfx

Glance at a(n) mfx object

Description

Glance accepts a model object and returns a tibble::tibble() with exactly one row of model summaries. The summaries are typically goodness of fit measures, p-values for hypothesis tests on residuals, or model convergence information.

Glance never returns information from the original call to the modeling function. This includes the name of the modeling function or any arguments passed to the modeling function.

Glance does not calculate summary measures. Rather, it farms out these computations to appropriate methods and gathers the results together. Sometimes a goodness of fit measure will be undefined. In these cases the measure will be reported as NA.

Glance returns the same number of columns regardless of whether the model matrix is rank-deficient or not. If so, entries in columns that no longer have a well-defined value are filled in with an NA of the appropriate type.

glance.mfx 165

Usage

```
## S3 method for class 'mfx'
glance(x, ...)

## S3 method for class 'logitmfx'
glance(x, ...)

## S3 method for class 'negbinmfx'
glance(x, ...)

## S3 method for class 'poissonmfx'
glance(x, ...)

## S3 method for class 'probitmfx'
glance(x, ...)
```

Arguments

x A logitmfx, negbinmfx, poissonmfx, or probitmfx object. (Note that betamfx

objects receive their own set of tidiers.)

... Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in ..., where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

Details

This generic glance method wraps glance.glm() for applicable objects from the mfx package.

Value

A tibble::tibble() with exactly one row and columns:

AIC Akaike's Information Criterion for the model.

BIC Bayesian Information Criterion for the model.

deviance Deviance of the model.

df.null Degrees of freedom used by the null model.

df.residual Residual degrees of freedom.

logLik The log-likelihood of the model. [stats::logLik()] may be a useful reference.

nobs Number of observations used. null.deviance Deviance of the null model. 166 glance.mjoint

See Also

```
glance.glm(), mfx::logitmfx(), mfx::negbinmfx(), mfx::poissonmfx(), mfx::probitmfx()
Other mfx tidiers: augment.betamfx(), augment.mfx(), glance.betamfx(), tidy.betamfx(),
tidy.mfx()
```

Examples

```
# load libraries for models and data
library(mfx)

# get the marginal effects from a logit regression
mod_logmfx <- logitmfx(am ~ cyl + hp + wt, atmean = TRUE, data = mtcars)

tidy(mod_logmfx, conf.int = TRUE)

# compare with the naive model coefficients of the same logit call
tidy(
    glm(am ~ cyl + hp + wt, family = binomial, data = mtcars),
    conf.int = TRUE
)

augment(mod_logmfx)
glance(mod_logmfx)

# another example, this time using probit regression
mod_probmfx <- probitmfx(am ~ cyl + hp + wt, atmean = TRUE, data = mtcars)

tidy(mod_probmfx, conf.int = TRUE)
augment(mod_probmfx)
glance(mod_probmfx)</pre>
```

glance.mjoint

Glance at a(n) mjoint object

Description

Glance accepts a model object and returns a tibble::tibble() with exactly one row of model summaries. The summaries are typically goodness of fit measures, p-values for hypothesis tests on residuals, or model convergence information.

Glance never returns information from the original call to the modeling function. This includes the name of the modeling function or any arguments passed to the modeling function.

Glance does not calculate summary measures. Rather, it farms out these computations to appropriate methods and gathers the results together. Sometimes a goodness of fit measure will be undefined. In these cases the measure will be reported as NA.

Glance returns the same number of columns regardless of whether the model matrix is rank-deficient or not. If so, entries in columns that no longer have a well-defined value are filled in with an NA of the appropriate type.

glance.mjoint 167

Usage

```
## S3 method for class 'mjoint'
glance(x, ...)
```

Arguments

x An mjoint object returned from joineRML::mjoint().

Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in . . . , where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

Value

A tibble::tibble() with exactly one row and columns:

AIC Akaike's Information Criterion for the model.

BIC Bayesian Information Criterion for the model.

logLik The log-likelihood of the model. [stats::logLik()] may be a useful reference.

sigma2_j The square root of the estimated residual variance for the j-th longitudinal pro-

cess

See Also

```
glance(), joineRML::mjoint()
Other mjoint tidiers: tidy.mjoint()
```

```
# broom only skips running these examples because the example models take a
# while to generate—they should run just fine, though!
## Not run:

# load libraries for models and data
library(joineRML)

# fit a joint model with bivariate longitudinal outcomes
data(heart.valve)

hvd <- heart.valve[!is.na(heart.valve$log.grad) &
    !is.na(heart.valve$log.lvmi) &</pre>
```

168 glance.mlogit

```
heart.valve$num <= 50, ]
fit <- mjoint(</pre>
  formLongFixed = list(
    "grad" = log.grad ~ time + sex + hs,
    "lvmi" = log.lvmi \sim time + sex
  formLongRandom = list(
    "grad" = ~ 1 | num,
    "lvmi" = ~ time | num
  ),
  formSurv = Surv(fuyrs, status) ~ age,
  data = hvd,
  inits = list("gamma" = c(0.11, 1.51, 0.80)),
  timeVar = "time"
)
# extract the survival fixed effects
tidy(fit)
# extract the longitudinal fixed effects
tidy(fit, component = "longitudinal")
# extract the survival fixed effects with confidence intervals
tidy(fit, ci = TRUE)
# extract the survival fixed effects with confidence intervals based
# on bootstrapped standard errors
bSE <- bootSE(fit, nboot = 5, safe.boot = TRUE)</pre>
tidy(fit, boot_se = bSE, ci = TRUE)
# augment original data with fitted longitudinal values and residuals
hvd2 <- augment(fit)</pre>
# extract model statistics
glance(fit)
## End(Not run)
```

glance.mlogit

Glance at a(n) mlogit object

Description

Glance accepts a model object and returns a tibble::tibble() with exactly one row of model summaries. The summaries are typically goodness of fit measures, p-values for hypothesis tests on residuals, or model convergence information.

Glance never returns information from the original call to the modeling function. This includes the name of the modeling function or any arguments passed to the modeling function.

glance.mlogit 169

Glance does not calculate summary measures. Rather, it farms out these computations to appropriate methods and gathers the results together. Sometimes a goodness of fit measure will be undefined. In these cases the measure will be reported as NA.

Glance returns the same number of columns regardless of whether the model matrix is rank-deficient or not. If so, entries in columns that no longer have a well-defined value are filled in with an NA of the appropriate type.

Usage

```
## S3 method for class 'mlogit'
glance(x, ...)
```

Arguments

x an object returned from mlogit::mlogit().

.. Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in . . . , where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

Value

A tibble::tibble() with exactly one row and columns:

AIC Akaike's Information Criterion for the model.

BIC Bayesian Information Criterion for the model.

logLik The log-likelihood of the model. [stats::logLik()] may be a useful reference.

nobs Number of observations used.

rho2 McFadden's rho squared with respect to a market shares (constants-only) model.
rho20 McFadden's rho squared with respect to an equal shares (no information) model.

See Also

```
glance(), mlogit::mlogit()
Other mlogit tidiers: augment.mlogit(), tidy.mlogit()
```

```
# load libraries for models and data
library(mlogit)
```

170 glance.muhaz

```
data("Fishing", package = "mlogit")
Fish <- dfidx(Fishing, varying = 2:9, shape = "wide", choice = "mode")
# fit model
m <- mlogit(mode ~ price + catch | income, data = Fish)
# summarize model fit with tidiers
tidy(m)
augment(m)
glance(m)</pre>
```

glance.muhaz

Glance at a(n) muhaz object

Description

Glance accepts a model object and returns a tibble::tibble() with exactly one row of model summaries. The summaries are typically goodness of fit measures, p-values for hypothesis tests on residuals, or model convergence information.

Glance never returns information from the original call to the modeling function. This includes the name of the modeling function or any arguments passed to the modeling function.

Glance does not calculate summary measures. Rather, it farms out these computations to appropriate methods and gathers the results together. Sometimes a goodness of fit measure will be undefined. In these cases the measure will be reported as NA.

Glance returns the same number of columns regardless of whether the model matrix is rank-deficient or not. If so, entries in columns that no longer have a well-defined value are filled in with an NA of the appropriate type.

Usage

```
## S3 method for class 'muhaz'
glance(x, ...)
```

Arguments

x A muhaz object returned by muhaz::muhaz().

Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in . . ., where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

glance.multinom 171

Value

A tibble::tibble() with exactly one row and columns:

max.hazard Maximal estimated hazard.

max.time The maximum observed event or censoring time.

min.hazard Minimal estimated hazard.

min. time The minimum observed event or censoring time.

nobs Number of observations used.

See Also

```
glance(), muhaz::muhaz()
Other muhaz tidiers: tidy.muhaz()
```

Examples

```
# load libraries for models and data
library(muhaz)
library(survival)

# fit model
x <- muhaz(ovarian$futime, ovarian$fustat)

# summarize model fit with tidiers
tidy(x)
glance(x)</pre>
```

 ${\tt glance.multinom}$

Glance at a(n) multinom object

Description

Glance accepts a model object and returns a tibble::tibble() with exactly one row of model summaries. The summaries are typically goodness of fit measures, p-values for hypothesis tests on residuals, or model convergence information.

Glance never returns information from the original call to the modeling function. This includes the name of the modeling function or any arguments passed to the modeling function.

Glance does not calculate summary measures. Rather, it farms out these computations to appropriate methods and gathers the results together. Sometimes a goodness of fit measure will be undefined. In these cases the measure will be reported as NA.

Glance returns the same number of columns regardless of whether the model matrix is rank-deficient or not. If so, entries in columns that no longer have a well-defined value are filled in with an NA of the appropriate type.

172 glance.multinom

Usage

```
## S3 method for class 'multinom'
glance(x, ...)
```

Arguments

x A multinom object returned from nnet::multinom().

Additional arguments. Not used. Needed to match generic signature only. Cautionary note: Misspelled arguments will be absorbed in . . . , where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

Value

A tibble::tibble() with exactly one row and columns:

AIC Akaike's Information Criterion for the model.

deviance Deviance of the model.

edf The effective degrees of freedom.

nobs Number of observations used.

See Also

```
glance(), nnet::multinom()
Other multinom tidiers: tidy.multinom()
```

```
# load libraries for models and data
library(nnet)
library(MASS)

example(birthwt)

bwt.mu <- multinom(low ~ ., bwt)

tidy(bwt.mu)
glance(bwt.mu)

# or, for output from a multinomial logistic regression
fit.gear <- multinom(gear ~ mpg + factor(am), data = mtcars)
tidy(fit.gear)</pre>
```

glance.negbin 173

```
glance(fit.gear)
```

glance.negbin

Glance at a(n) negbin object

Description

Glance accepts a model object and returns a tibble::tibble() with exactly one row of model summaries. The summaries are typically goodness of fit measures, p-values for hypothesis tests on residuals, or model convergence information.

Glance never returns information from the original call to the modeling function. This includes the name of the modeling function or any arguments passed to the modeling function.

Glance does not calculate summary measures. Rather, it farms out these computations to appropriate methods and gathers the results together. Sometimes a goodness of fit measure will be undefined. In these cases the measure will be reported as NA.

Glance returns the same number of columns regardless of whether the model matrix is rank-deficient or not. If so, entries in columns that no longer have a well-defined value are filled in with an NA of the appropriate type.

Usage

```
## S3 method for class 'negbin'
glance(x, ...)
```

Arguments

. . .

x A negbin object returned by MASS::glm.nb().

Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in . . . , where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

Value

A tibble::tibble() with exactly one row and columns:

AIC Akaike's Information Criterion for the model.

BIC Bayesian Information Criterion for the model.

deviance Deviance of the model.

174 glance.nlrq

df.null Degrees of freedom used by the null model.

df.residual Residual degrees of freedom.

logLik The log-likelihood of the model. [stats::logLik()] may be a useful reference.

nobs Number of observations used. null.deviance Deviance of the null model.

See Also

```
glance(), MASS::glm.nb()
Other glm.nb tidiers: tidy.negbin()
```

Examples

```
# load libraries for models and data
library(MASS)

# fit model
r <- glm.nb(Days ~ Sex / (Age + Eth * Lrn), data = quine)

# summarize model fit with tidiers
tidy(r)
glance(r)</pre>
```

glance.nlrq

Glance at a(n) nlrq object

Description

Glance accepts a model object and returns a tibble::tibble() with exactly one row of model summaries. The summaries are typically goodness of fit measures, p-values for hypothesis tests on residuals, or model convergence information.

Glance never returns information from the original call to the modeling function. This includes the name of the modeling function or any arguments passed to the modeling function.

Glance does not calculate summary measures. Rather, it farms out these computations to appropriate methods and gathers the results together. Sometimes a goodness of fit measure will be undefined. In these cases the measure will be reported as NA.

Glance returns the same number of columns regardless of whether the model matrix is rank-deficient or not. If so, entries in columns that no longer have a well-defined value are filled in with an NA of the appropriate type.

Usage

```
## S3 method for class 'nlrq'
glance(x, ...)
```

glance.nlrq 175

Arguments

. . .

x A nlrq object returned from quantreg::nlrq().

Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in . . . , where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

Value

A tibble::tibble() with exactly one row and columns:

AIC Akaike's Information Criterion for the model.

BIC Bayesian Information Criterion for the model.

df.residual Residual degrees of freedom.

logLik The log-likelihood of the model. [stats::logLik()] may be a useful reference.

tau Quantile.

See Also

```
glance(), quantreg::nlrq()
Other quantreg tidiers: augment.nlrq(), augment.rq(), augment.rqs(), glance.rq(), tidy.nlrq(),
tidy.rq(), tidy.rqs()
```

```
# load modeling library
library(quantreg)

# build artificial data with multiplicative error
set.seed(1)
dat <- NULL
dat$x <- rep(1:25, 20)
dat$y <- SSlogis(dat$x, 10, 12, 2) * rnorm(500, 1, 0.1)

# fit the median using nlrq
mod <- nlrq(y ~ SSlogis(x, Asym, mid, scal),
    data = dat, tau = 0.5, trace = TRUE
)

# summarize model fit with tidiers
tidy(mod)
glance(mod)</pre>
```

176 glance.nls

augment(mod)

glance.nls

Glance at a(n) nls object

Description

Glance accepts a model object and returns a tibble::tibble() with exactly one row of model summaries. The summaries are typically goodness of fit measures, p-values for hypothesis tests on residuals, or model convergence information.

Glance never returns information from the original call to the modeling function. This includes the name of the modeling function or any arguments passed to the modeling function.

Glance does not calculate summary measures. Rather, it farms out these computations to appropriate methods and gathers the results together. Sometimes a goodness of fit measure will be undefined. In these cases the measure will be reported as NA.

Glance returns the same number of columns regardless of whether the model matrix is rank-deficient or not. If so, entries in columns that no longer have a well-defined value are filled in with an NA of the appropriate type.

Usage

```
## S3 method for class 'nls'
glance(x, ...)
```

Arguments

. . .

x An nls object returned from stats::nls().

Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in . . ., where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

Value

A tibble::tibble() with exactly one row and columns:

AIC Akaike's Information Criterion for the model.

BIC Bayesian Information Criterion for the model.

deviance Deviance of the model.

glance.orcutt 177

df.residual Residual degrees of freedom.

finTol The achieved convergence tolerance.

isConv Whether the fit successfully converged.

logLik The log-likelihood of the model. [stats::logLik()] may be a useful reference.

nobs Number of observations used.

sigma Estimated standard error of the residuals.

See Also

```
tidy, stats::nls()
Other nls tidiers: augment.nls(), tidy.nls()
```

Examples

```
# fit model
n <- nls(mpg ~ k * e^wt, data = mtcars, start = list(k = 1, e = 2))
# summarize model fit with tidiers + visualization
tidy(n)
augment(n)
glance(n)
library(ggplot2)
ggplot(augment(n), aes(wt, mpg)) +
    geom_point() +
    geom_line(aes(y = .fitted))
newdata <- head(mtcars)
newdata$wt <- newdata$wt + 1
augment(n, newdata = newdata)</pre>
```

glance.orcutt

Glance at a(n) orcutt object

Description

Glance accepts a model object and returns a tibble::tibble() with exactly one row of model summaries. The summaries are typically goodness of fit measures, p-values for hypothesis tests on residuals, or model convergence information.

Glance never returns information from the original call to the modeling function. This includes the name of the modeling function or any arguments passed to the modeling function.

178 glance.orcutt

Glance does not calculate summary measures. Rather, it farms out these computations to appropriate methods and gathers the results together. Sometimes a goodness of fit measure will be undefined. In these cases the measure will be reported as NA.

Glance returns the same number of columns regardless of whether the model matrix is rank-deficient or not. If so, entries in columns that no longer have a well-defined value are filled in with an NA of the appropriate type.

Usage

```
## S3 method for class 'orcutt'
glance(x, ...)
```

Arguments

x An orcutt object returned from orcutt::cochrane.orcutt().

Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in . . . , where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

Value

A tibble::tibble() with exactly one row and columns:

adj.r.squared Adjusted R squared statistic, which is like the R squared statistic except taking

degrees of freedom into account.

dw.original Durbin-Watson statistic of original fit.

dw. transformed Durbin-Watson statistic of transformed fit.

nobs Number of observations used.

number.interaction

Number of interactions.

p.value.original

P-value of original Durbin-Watson statistic.

p.value.transformed

P-value of autocorrelation after transformation.

r.squared R squared statistic, or the percent of variation explained by the model. Also

known as the coefficient of determination.

rho Spearman's rho autocorrelation

See Also

```
glance(), orcutt::cochrane.orcutt()
Other orcutt tidiers: tidy.orcutt()
```

glance.pam 179

Examples

```
# load libraries for models and data
library(orcutt)

# fit model and summarize results
reg <- lm(mpg ~ wt + qsec + disp, mtcars)
tidy(reg)

co <- cochrane.orcutt(reg)
tidy(co)
glance(co)</pre>
```

glance.pam

Glance at a(n) pam object

Description

Glance accepts a model object and returns a tibble::tibble() with exactly one row of model summaries. The summaries are typically goodness of fit measures, p-values for hypothesis tests on residuals, or model convergence information.

Glance never returns information from the original call to the modeling function. This includes the name of the modeling function or any arguments passed to the modeling function.

Glance does not calculate summary measures. Rather, it farms out these computations to appropriate methods and gathers the results together. Sometimes a goodness of fit measure will be undefined. In these cases the measure will be reported as NA.

Glance returns the same number of columns regardless of whether the model matrix is rank-deficient or not. If so, entries in columns that no longer have a well-defined value are filled in with an NA of the appropriate type.

Usage

```
## S3 method for class 'pam'
glance(x, ...)
```

Arguments

x An pam object returned from cluster::pam()

Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in . . ., where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

180 glance.plm

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

Value

```
A tibble::tibble() with exactly one row and columns:

avg.silhouette.width

The average silhouette width for the dataset.
```

See Also

```
glance(), cluster::pam()
Other pam tidiers: augment.pam(), tidy.pam()
```

```
# load libraries for models and data
library(dplyr)
library(ggplot2)
library(cluster)
library(modeldata)
data(hpc_data)
x <- hpc_data[, 2:5]</pre>
p \leftarrow pam(x, k = 4)
# summarize model fit with tidiers + visualization
tidy(p)
glance(p)
augment(p, x)
augment(p, x) %>%
  ggplot(aes(compounds, input_fields)) +
  geom_point(aes(color = .cluster)) +
  geom_text(aes(label = cluster), data = tidy(p), size = 10)
```

glance.plm 181

Description

Glance accepts a model object and returns a tibble::tibble() with exactly one row of model summaries. The summaries are typically goodness of fit measures, p-values for hypothesis tests on residuals, or model convergence information.

Glance never returns information from the original call to the modeling function. This includes the name of the modeling function or any arguments passed to the modeling function.

Glance does not calculate summary measures. Rather, it farms out these computations to appropriate methods and gathers the results together. Sometimes a goodness of fit measure will be undefined. In these cases the measure will be reported as NA.

Glance returns the same number of columns regardless of whether the model matrix is rank-deficient or not. If so, entries in columns that no longer have a well-defined value are filled in with an NA of the appropriate type.

Usage

```
## S3 method for class 'plm'
glance(x, ...)
```

Arguments

x A plm objected returned by plm::plm().

Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in . . ., where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

Value

A tibble::tibble() with exactly one row and columns:

adj.r.squared Adjusted R squared statistic, which is like the R squared statistic except taking

degrees of freedom into account.

deviance Deviance of the model.

df.residual Residual degrees of freedom.

Number of observations used.

p.value P-value corresponding to the test statistic.

r.squared R squared statistic, or the percent of variation explained by the model. Also

known as the coefficient of determination.

statistic F-statistic

182 glance.poLCA

See Also

```
glance(), plm::plm()
Other plm tidiers: augment.plm(), tidy.plm()
```

Examples

```
# load libraries for models and data
library(plm)

# load data
data("Produc", package = "plm")

# fit model
zz <- plm(log(gsp) ~ log(pcap) + log(pc) + log(emp) + unemp,
data = Produc, index = c("state", "year")
)

# summarize model fit with tidiers
summary(zz)

tidy(zz)
tidy(zz, conf.int = TRUE)
tidy(zz, conf.int = TRUE, conf.level = 0.9)
augment(zz)
glance(zz)</pre>
```

glance.poLCA

Glance at a(n) poLCA object

Description

Glance accepts a model object and returns a tibble::tibble() with exactly one row of model summaries. The summaries are typically goodness of fit measures, p-values for hypothesis tests on residuals, or model convergence information.

Glance never returns information from the original call to the modeling function. This includes the name of the modeling function or any arguments passed to the modeling function.

Glance does not calculate summary measures. Rather, it farms out these computations to appropriate methods and gathers the results together. Sometimes a goodness of fit measure will be undefined. In these cases the measure will be reported as NA.

Glance returns the same number of columns regardless of whether the model matrix is rank-deficient or not. If so, entries in columns that no longer have a well-defined value are filled in with an NA of the appropriate type.

glance.poLCA 183

Usage

```
## S3 method for class 'poLCA'
glance(x, ...)
```

Arguments

x A poLCA object returned from poLCA::poLCA().

Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in . . ., where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

Value

A tibble::tibble() with exactly one row and columns:

AIC Akaike's Information Criterion for the model.

BIC Bayesian Information Criterion for the model.

chi. squared The Pearson Chi-Square goodness of fit statistic for multiway tables.

df Degrees of freedom used by the model.

df.residual Residual degrees of freedom.

logLik The log-likelihood of the model. [stats::logLik()] may be a useful reference.

nobs Number of observations used.

g. squared The likelihood ratio/deviance statistic

See Also

```
glance(), poLCA::poLCA()
Other poLCA tidiers: augment.poLCA(), tidy.poLCA()
```

```
# load libraries for models and data
library(poLCA)
library(dplyr)

# generate data
data(values)

f <- cbind(A, B, C, D) ~ 1</pre>
```

184 glance.polr

```
# fit model
M1 <- poLCA(f, values, nclass = 2, verbose = FALSE)
M1
# summarize model fit with tidiers + visualization
tidy(M1)
augment(M1)
glance(M1)
library(ggplot2)
ggplot(tidy(M1), aes(factor(class), estimate, fill = factor(outcome))) +
  geom_bar(stat = "identity", width = 1) +
  facet_wrap(~variable)
# three-class model with a single covariate.
data(election)
f2a <- cbind(
  MORALG, CARESG, KNOWG, LEADG, DISHONG, INTELG,
  MORALB, CARESB, KNOWB, LEADB, DISHONB, INTELB
) ~ PARTY
nes2a <- poLCA(f2a, election, nclass = 3, nrep = 5, verbose = FALSE)</pre>
td <- tidy(nes2a)
ggplot(td, aes(outcome, estimate, color = factor(class), group = class)) +
  geom_line() +
  facet_wrap(\sim variable, nrow = 2) +
  theme(axis.text.x = element_text(angle = 90, hjust = 1))
au <- augment(nes2a)</pre>
au
count(au, .class)
# if the original data is provided, it leads to NAs in new columns
# for rows that weren't predicted
au2 <- augment(nes2a, data = election)</pre>
au2
dim(au2)
```

glance.polr 185

Description

Glance accepts a model object and returns a tibble::tibble() with exactly one row of model summaries. The summaries are typically goodness of fit measures, p-values for hypothesis tests on residuals, or model convergence information.

Glance never returns information from the original call to the modeling function. This includes the name of the modeling function or any arguments passed to the modeling function.

Glance does not calculate summary measures. Rather, it farms out these computations to appropriate methods and gathers the results together. Sometimes a goodness of fit measure will be undefined. In these cases the measure will be reported as NA.

Glance returns the same number of columns regardless of whether the model matrix is rank-deficient or not. If so, entries in columns that no longer have a well-defined value are filled in with an NA of the appropriate type.

Usage

```
## S3 method for class 'polr'
glance(x, ...)
```

Arguments

x A polr object returned from MASS::polr().

Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in . . . , where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

Value

A tibble::tibble() with exactly one row and columns:

AIC Akaike's Information Criterion for the model.

BIC Bayesian Information Criterion for the model.

deviance Deviance of the model.

df.residual Residual degrees of freedom.

edf The effective degrees of freedom.

logLik The log-likelihood of the model. [stats::logLik()] may be a useful reference.

nobs Number of observations used.

186 glance.pyears

See Also

```
tidy, MASS::polr()
Other ordinal tidiers: augment.clm(), augment.polr(), glance.clm(), glance.clmm(), glance.svyolr(),
tidy.clm(), tidy.clmm(), tidy.polr(), tidy.svyolr()
```

Examples

```
# load libraries for models and data
library(MASS)

# fit model
fit <- polr(Sat ~ Infl + Type + Cont, weights = Freq, data = housing)

# summarize model fit with tidiers
tidy(fit, exponentiate = TRUE, conf.int = TRUE)

glance(fit)
augment(fit, type.predict = "class")

fit2 <- polr(factor(gear) ~ am + mpg + qsec, data = mtcars)

tidy(fit, p.values = TRUE)</pre>
```

glance.pyears

Glance at a(n) pyears object

Description

Glance accepts a model object and returns a tibble::tibble() with exactly one row of model summaries. The summaries are typically goodness of fit measures, p-values for hypothesis tests on residuals, or model convergence information.

Glance never returns information from the original call to the modeling function. This includes the name of the modeling function or any arguments passed to the modeling function.

Glance does not calculate summary measures. Rather, it farms out these computations to appropriate methods and gathers the results together. Sometimes a goodness of fit measure will be undefined. In these cases the measure will be reported as NA.

Glance returns the same number of columns regardless of whether the model matrix is rank-deficient or not. If so, entries in columns that no longer have a well-defined value are filled in with an NA of the appropriate type.

Usage

```
## S3 method for class 'pyears'
glance(x, ...)
```

glance.pyears 187

Arguments

. . .

x A pyears object returned from survival::pyears().

Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in . . ., where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

Value

A tibble::tibble() with exactly one row and columns:

nobs Number of observations used.

total number of person-years tabulated offtable total number of person-years off table

See Also

```
glance(), survival::pyears()
Other pyears tidiers: tidy.pyears()
Other survival tidiers: augment.coxph(), augment.survreg(), glance.aareg(), glance.cch(),
glance.coxph(), glance.survdiff(), glance.survexp(), glance.survfit(), glance.survreg(),
tidy.aareg(), tidy.cch(), tidy.coxph(), tidy.pyears(), tidy.survdiff(), tidy.survexp(),
tidy.survfit(), tidy.survreg()
```

```
# load libraries for models and data
library(survival)

# generate and format data
temp.yr <- tcut(mgus$dxyr, 55:92, labels = as.character(55:91))
temp.age <- tcut(mgus$age, 34:101, labels = as.character(34:100))
ptime <- ifelse(is.na(mgus$pctime), mgus$futime, mgus$pctime)
pstat <- ifelse(is.na(mgus$pctime), 0, 1)
pfit <- pyears(Surv(ptime / 365.25, pstat) ~ temp.yr + temp.age + sex, mgus,
    data.frame = TRUE
)

# summarize model fit with tidiers
tidy(pfit)
glance(pfit)</pre>
```

188 glance.ridgelm

```
# if data.frame argument is not given, different information is present in
# output
pfit2 <- pyears(Surv(ptime / 365.25, pstat) ~ temp.yr + temp.age + sex, mgus)
tidy(pfit2)
glance(pfit2)</pre>
```

glance.ridgelm

Glance at a(n) ridgelm object

Description

Glance accepts a model object and returns a tibble::tibble() with exactly one row of model summaries. The summaries are typically goodness of fit measures, p-values for hypothesis tests on residuals, or model convergence information.

Glance never returns information from the original call to the modeling function. This includes the name of the modeling function or any arguments passed to the modeling function.

Glance does not calculate summary measures. Rather, it farms out these computations to appropriate methods and gathers the results together. Sometimes a goodness of fit measure will be undefined. In these cases the measure will be reported as NA.

Glance returns the same number of columns regardless of whether the model matrix is rank-deficient or not. If so, entries in columns that no longer have a well-defined value are filled in with an NA of the appropriate type.

Usage

```
## S3 method for class 'ridgelm'
glance(x, ...)
```

Arguments

X

A ridgelm object returned from MASS::lm.ridge().

. . .

Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in . . . , where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

Details

This is similar to the output of select.ridgelm, but it is returned rather than printed.

glance.rlm 189

Value

A tibble::tibble() with exactly one row and columns:

kHKB modified HKB estimate of the ridge constant
kLW modified L-W estimate of the ridge constant
lambdaGCV choice of lambda that minimizes GCV

See Also

```
glance(), MASS::select.ridgelm(), MASS::lm.ridge()
Other ridgelm tidiers: tidy.ridgelm()
```

```
# load libraries for models and data
library(MASS)
names(longley)[1] <- "y"</pre>
# fit model and summarizd results
fit1 <- lm.ridge(y ~ ., longley)</pre>
tidy(fit1)
fit2 <- lm.ridge(y ~ ., longley, lambda = seq(0.001, .05, .001))
td2 <- tidy(fit2)
g2 <- glance(fit2)
# coefficient plot
library(ggplot2)
ggplot(td2, aes(lambda, estimate, color = term)) +
  geom_line()
# GCV plot
ggplot(td2, aes(lambda, GCV)) +
  geom_line()
# add line for the GCV minimizing estimate
ggplot(td2, aes(lambda, GCV)) +
  geom_line() +
  geom_vline(xintercept = g2$lambdaGCV, col = "red", lty = 2)
```

190 glance.rlm

Description

Glance accepts a model object and returns a tibble::tibble() with exactly one row of model summaries. The summaries are typically goodness of fit measures, p-values for hypothesis tests on residuals, or model convergence information.

Glance never returns information from the original call to the modeling function. This includes the name of the modeling function or any arguments passed to the modeling function.

Glance does not calculate summary measures. Rather, it farms out these computations to appropriate methods and gathers the results together. Sometimes a goodness of fit measure will be undefined. In these cases the measure will be reported as NA.

Glance returns the same number of columns regardless of whether the model matrix is rank-deficient or not. If so, entries in columns that no longer have a well-defined value are filled in with an NA of the appropriate type.

Usage

```
## S3 method for class 'rlm'
glance(x, ...)
```

Arguments

x An rlm object returned by MASS::rlm().

Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in . . . , where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

Value

A tibble::tibble() with exactly one row and columns:

AIC Akaike's Information Criterion for the model.

BIC Bayesian Information Criterion for the model.

converged Logical indicating if the model fitting procedure was successful and converged.

deviance Deviance of the model.

logLik The log-likelihood of the model. [stats::logLik()] may be a useful reference.

nobs Number of observations used.

sigma Estimated standard error of the residuals.

See Also

```
glance(), MASS::rlm()
Other rlm tidiers: augment.rlm(), tidy.rlm()
```

glance.rma 191

Examples

```
# load libraries for models and data
library(MASS)

# fit model
r <- rlm(stack.loss ~ ., stackloss)

# summarize model fit with tidiers
tidy(r)
augment(r)
glance(r)</pre>
```

glance.rma

Glance at a(n) rma object

Description

Glance accepts a model object and returns a tibble::tibble() with exactly one row of model summaries. The summaries are typically goodness of fit measures, p-values for hypothesis tests on residuals, or model convergence information.

Glance never returns information from the original call to the modeling function. This includes the name of the modeling function or any arguments passed to the modeling function.

Glance does not calculate summary measures. Rather, it farms out these computations to appropriate methods and gathers the results together. Sometimes a goodness of fit measure will be undefined. In these cases the measure will be reported as NA.

Glance returns the same number of columns regardless of whether the model matrix is rank-deficient or not. If so, entries in columns that no longer have a well-defined value are filled in with an NA of the appropriate type.

Usage

```
## S3 method for class 'rma'
glance(x, ...)
```

Arguments

An rma object such as those created by metafor::rma(), metafor::rma.uni(),
metafor::rma.glmm(), metafor::rma.mh(), metafor::rma.mv(), or metafor::rma.peto().

Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in . . ., where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

192 glance.rma

• tidy() methods will warn when supplied an exponentiate argument if it will be ignored.

• augment() methods will warn when supplied a newdata argument if it will be ignored.

Value

A tibble::tibble() with exactly one row and columns:

In meta-analysis, test statistic for the Cochran's Q_e test of residual heterogenecochran.qe In meta-analysis, test statistic for the Cochran's Q_m omnibus test of coefficochran.qm cients. df.residual Residual degrees of freedom. h.squared Value of the H-Squared statistic. Value of the I-Squared statistic. i.squared measure The measure used in the meta-analysis. Which method was used. method nobs Number of observations used. p.value.cochran.qe In meta-analysis, p-value for the Cochran's Q_e test of residual heterogeneity. p.value.cochran.qm In meta-analysis, p-value for the Cochran's Q_m omnibus test of coefficients. In meta-analysis, estimated amount of residual heterogeneity. tau.squared tau.squared.se In meta-analysis, standard error of residual heterogeneity.

```
library(metafor)

df <-
    escalc(
    measure = "RR",
    ai = tpos,
    bi = tneg,
    ci = cpos,
    di = cneg,
    data = dat.bcg
)

meta_analysis <- rma(yi, vi, data = df, method = "EB")
glance(meta_analysis)</pre>
```

glance.rq 193

glance.rq

Glance at a(n) rq object

Description

Glance accepts a model object and returns a tibble::tibble() with exactly one row of model summaries. The summaries are typically goodness of fit measures, p-values for hypothesis tests on residuals, or model convergence information.

Glance never returns information from the original call to the modeling function. This includes the name of the modeling function or any arguments passed to the modeling function.

Glance does not calculate summary measures. Rather, it farms out these computations to appropriate methods and gathers the results together. Sometimes a goodness of fit measure will be undefined. In these cases the measure will be reported as NA.

Glance returns the same number of columns regardless of whether the model matrix is rank-deficient or not. If so, entries in columns that no longer have a well-defined value are filled in with an NA of the appropriate type.

Usage

```
## S3 method for class 'rq'
glance(x, ...)
```

Arguments

X

An rq object returned from quantreg::rq().

Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in . . ., where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

Details

Only models with a single tau value may be passed. For multiple values, please use a purrr::map() workflow instead, e.g.

```
taus %>%
  map(function(tau_val) rq(y ~ x, tau = tau_val)) %>%
  map_dfr(glance)
```

194 glance.rq

Value

A tibble::tibble() with exactly one row and columns:

AIC Akaike's Information Criterion for the model.

BIC Bayesian Information Criterion for the model.

df.residual Residual degrees of freedom.

logLik The log-likelihood of the model. [stats::logLik()] may be a useful reference.

tau Quantile.

See Also

```
glance(), quantreg::rq()
Other quantreg tidiers: augment.nlrq(), augment.rq(), augment.rqs(), glance.nlrq(), tidy.nlrq(),
tidy.rq(), tidy.rqs()
```

```
# load modeling library and data
library(quantreg)
data(stackloss)
# median (11) regression fit for the stackloss data.
mod1 <- rq(stack.loss ~ stack.x, .5)</pre>
# weighted sample median
mod2 \leftarrow rq(rnorm(50) \sim 1, weights = runif(50))
# summarize model fit with tidiers
tidy(mod1)
glance(mod1)
augment(mod1)
tidy(mod2)
glance(mod2)
augment(mod2)
# varying tau to generate an rqs object
mod3 \leftarrow rq(stack.loss \sim stack.x, tau = c(.25, .5))
tidy(mod3)
augment(mod3)
# glance cannot handle rqs objects like `mod3`--use a purrr
# `map`-based workflow instead
```

glance.sarlm 195

glance.sarlm	Glance at a(n) spatialreg object

Description

Glance accepts a model object and returns a tibble::tibble() with exactly one row of model summaries. The summaries are typically goodness of fit measures, p-values for hypothesis tests on residuals, or model convergence information.

Glance never returns information from the original call to the modeling function. This includes the name of the modeling function or any arguments passed to the modeling function.

Glance does not calculate summary measures. Rather, it farms out these computations to appropriate methods and gathers the results together. Sometimes a goodness of fit measure will be undefined. In these cases the measure will be reported as NA.

Glance returns the same number of columns regardless of whether the model matrix is rank-deficient or not. If so, entries in columns that no longer have a well-defined value are filled in with an NA of the appropriate type.

Usage

```
## S3 method for class 'sarlm'
glance(x, ...)
```

Arguments

x An object returned from spatialreg::lagsarlm() or spatialreg::errorsarlm().

Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in . . . , where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

Value

A tibble::tibble() with exactly one row and columns:

AIC Akaike's Information Criterion for the model.

BIC Bayesian Information Criterion for the model.

deviance Deviance of the model.

logLik The log-likelihood of the model. [stats::logLik()] may be a useful reference.

nobs Number of observations used.

196 glance.sarlm

See Also

```
glance(), spatialreg::lagsarlm(), spatialreg::errorsarlm(), spatialreg::sacsarlm()
Other spatialreg tidiers: augment.sarlm(), tidy.sarlm()
```

```
# load libraries for models and data
library(spatialreg)
library(spdep)
# load data
data(oldcol, package = "spdep")
listw <- nb2listw(COL.nb, style = "W")</pre>
# fit model
crime_sar <-</pre>
  lagsarlm(CRIME ~ INC + HOVAL,
   data = COL.OLD,
   listw = listw,
    method = "eigen"
# summarize model fit with tidiers
tidy(crime_sar)
tidy(crime_sar, conf.int = TRUE)
glance(crime_sar)
augment(crime_sar)
# fit another model
crime_sem <- errorsarlm(CRIME ~ INC + HOVAL, data = COL.OLD, listw)</pre>
# summarize model fit with tidiers
tidy(crime_sem)
tidy(crime_sem, conf.int = TRUE)
glance(crime_sem)
augment(crime_sem)
# fit another model
crime_sac <- sacsarlm(CRIME ~ INC + HOVAL, data = COL.OLD, listw)</pre>
# summarize model fit with tidiers
tidy(crime_sac)
tidy(crime_sac, conf.int = TRUE)
glance(crime_sac)
augment(crime_sac)
```

glance.smooth.spline 197

```
glance.smooth.spline Tidy \ a(n) \ smooth.spine \ object
```

Description

Tidy summarizes information about the components of a model. A model component might be a single term in a regression, a single hypothesis, a cluster, or a class. Exactly what tidy considers to be a model component varies across models but is usually self-evident. If a model has several distinct types of components, you will need to specify which components to return.

Usage

```
## S3 method for class 'smooth.spline'
glance(x, ...)
```

Arguments

x A smooth.spline object returned from stats::smooth.spline().

Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in . . ., where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

Value

A tibble::tibble() with exactly one row and columns:

crit Minimized criterion

cv.crit Cross-validation score

df Degrees of freedom used by the model.

lambda Choice of lambda corresponding to 'spar'.

nobs Number of observations used.

pen.crit Penalized criterion.

Smoothing parameter.

See Also

spar

```
augment(), stats::smooth.spline()
Other smoothing spline tidiers: augment.smooth.spline()
```

198 glance.speedglm

Examples

```
# fit model
spl <- smooth.spline(mtcars$wt, mtcars$mpg, df = 4)
# summarize model fit with tidiers
augment(spl, mtcars)
# calls original columns x and y
augment(spl)
library(ggplot2)
ggplot(augment(spl, mtcars), aes(wt, mpg)) +
    geom_point() +
    geom_line(aes(y = .fitted))</pre>
```

glance.speedglm

Glance at a(n) speedglm object

Description

Glance accepts a model object and returns a tibble::tibble() with exactly one row of model summaries. The summaries are typically goodness of fit measures, p-values for hypothesis tests on residuals, or model convergence information.

Glance never returns information from the original call to the modeling function. This includes the name of the modeling function or any arguments passed to the modeling function.

Glance does not calculate summary measures. Rather, it farms out these computations to appropriate methods and gathers the results together. Sometimes a goodness of fit measure will be undefined. In these cases the measure will be reported as NA.

Glance returns the same number of columns regardless of whether the model matrix is rank-deficient or not. If so, entries in columns that no longer have a well-defined value are filled in with an NA of the appropriate type.

Usage

```
## S3 method for class 'speedglm'
glance(x, ...)
```

Arguments

x A speedglm object returned from speedglm::speedglm().

Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in . . ., where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

glance.speedglm 199

• tidy() methods will warn when supplied an exponentiate argument if it will be ignored.

• augment() methods will warn when supplied a newdata argument if it will be ignored.

Value

A tibble::tibble() with exactly one row and columns:

AIC Akaike's Information Criterion for the model.

BIC Bayesian Information Criterion for the model.

deviance Deviance of the model.

df.null Degrees of freedom used by the null model.

df.residual Residual degrees of freedom.

logLik The log-likelihood of the model. [stats::logLik()] may be a useful reference.

nobs Number of observations used.
null.deviance Deviance of the null model.

See Also

```
speedglm::speedlm()
Other speedlm tidiers: augment.speedlm(), glance.speedlm(), tidy.speedglm(), tidy.speedlm()
```

```
# load libraries for models and data
library(speedglm)

# generate data
clotting <- data.frame(
    u = c(5, 10, 15, 20, 30, 40, 60, 80, 100),
    lot1 = c(118, 58, 42, 35, 27, 25, 21, 19, 18)
)

# fit model
fit <- speedglm(lot1 ~ log(u), data = clotting, family = Gamma(log))

# summarize model fit with tidiers
tidy(fit)
glance(fit)</pre>
```

200 glance.speedlm

glance.speedlm

Glance at a(n) speedlm object

Description

Glance accepts a model object and returns a tibble::tibble() with exactly one row of model summaries. The summaries are typically goodness of fit measures, p-values for hypothesis tests on residuals, or model convergence information.

Glance never returns information from the original call to the modeling function. This includes the name of the modeling function or any arguments passed to the modeling function.

Glance does not calculate summary measures. Rather, it farms out these computations to appropriate methods and gathers the results together. Sometimes a goodness of fit measure will be undefined. In these cases the measure will be reported as NA.

Glance returns the same number of columns regardless of whether the model matrix is rank-deficient or not. If so, entries in columns that no longer have a well-defined value are filled in with an NA of the appropriate type.

Usage

```
## S3 method for class 'speedlm'
glance(x, ...)
```

Arguments

. . .

x A speedlm object returned from speedglm::speedlm().

Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in . . . , where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

Value

A tibble::tibble() with exactly one row and columns:

adj.r.squared Adjusted R squared statistic, which is like the R squared statistic except taking

degrees of freedom into account.

AIC Akaike's Information Criterion for the model.

BIC Bayesian Information Criterion for the model.

deviance Deviance of the model.

df Degrees of freedom used by the model.

glance.summary.lm 201

df.residual Residual degrees of freedom.

logLik The log-likelihood of the model. [stats::logLik()] may be a useful reference.

nobs Number of observations used.

p. value P-value corresponding to the test statistic.

r.squared R squared statistic, or the percent of variation explained by the model. Also

known as the coefficient of determination.

statistic F-statistic.

See Also

```
speedglm::speedlm()
Other speedlm tidiers: augment.speedlm(), glance.speedglm(), tidy.speedglm(), tidy.speedlm()
```

Examples

```
# load modeling library
library(speedglm)

# fit model
mod <- speedlm(mpg ~ wt + qsec, data = mtcars, fitted = TRUE)

# summarize model fit with tidiers
tidy(mod)
glance(mod)
augment(mod)</pre>
```

glance.summary.lm

Glance at a(n) summary.lm object

Description

Glance accepts a model object and returns a tibble::tibble() with exactly one row of model summaries. The summaries are typically goodness of fit measures, p-values for hypothesis tests on residuals, or model convergence information.

Glance never returns information from the original call to the modeling function. This includes the name of the modeling function or any arguments passed to the modeling function.

Glance does not calculate summary measures. Rather, it farms out these computations to appropriate methods and gathers the results together. Sometimes a goodness of fit measure will be undefined. In these cases the measure will be reported as NA.

Glance returns the same number of columns regardless of whether the model matrix is rank-deficient or not. If so, entries in columns that no longer have a well-defined value are filled in with an NA of the appropriate type.

202 glance.summary.lm

Usage

```
## S3 method for class 'summary.lm'
glance(x, ...)
```

Arguments

. . .

x An lm object created by stats::lm().

Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in . . . , where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

Details

The glance.summary.lm() method is a potentially useful alternative to glance.lm(). For instance, if users have already converted large lm objects into their leaner summary.lm equivalents to conserve memory. Note, however, that this method does not return all of the columns of the non-summary method (e.g. AIC and BIC will be missing.)

Value

A tibble::tibble() with exactly one row and columns:

adj.r.squared Adjusted R squared statistic, which is like the R squared statistic except taking

degrees of freedom into account.

df.residual Residual degrees of freedom.

nobs Number of observations used.

p. value P-value corresponding to the test statistic.

r.squared R squared statistic, or the percent of variation explained by the model. Also

known as the coefficient of determination.

sigma Estimated standard error of the residuals.

statistic Test statistic.

df The degrees for freedom from the numerator of the overall F-statistic. This

is new in broom 0.7.0. Previously, this reported the rank of the design matrix, which is one more than the numerator degrees of freedom of the overall

F-statistic.

See Also

```
glance(), glance.summary.lm()
Other Im tidiers: augment.glm(), augment.lm(), glance.glm(), glance.lm(), glance.svyglm(),
tidy.glm(), tidy.lm(), tidy.lm.beta(), tidy.mlm(), tidy.summary.lm()
```

glance.summary.lm 203

```
library(ggplot2)
library(dplyr)
mod <- lm(mpg ~ wt + qsec, data = mtcars)</pre>
tidy(mod)
glance(mod)
# coefficient plot
d <- tidy(mod, conf.int = TRUE)</pre>
ggplot(d, aes(estimate, term, xmin = conf.low, xmax = conf.high, height = 0)) +
  geom_point() +
  geom_vline(xintercept = 0, lty = 4) +
  geom_errorbarh()
# aside: There are tidy() and glance() methods for lm.summary objects too.
# this can be useful when you want to conserve memory by converting large lm
# objects into their leaner summary.lm equivalents.
s <- summary(mod)</pre>
tidy(s, conf.int = TRUE)
glance(s)
augment(mod)
augment(mod, mtcars, interval = "confidence")
# predict on new data
newdata <- mtcars %>%
  head(6) %>%
  mutate(wt = wt + 1)
augment(mod, newdata = newdata)
# ggplot2 example where we also construct 95% prediction interval
# simpler bivariate model since we're plotting in 2D
mod2 <- lm(mpg \sim wt, data = mtcars)
au <- augment(mod2, newdata = newdata, interval = "prediction")</pre>
ggplot(au, aes(wt, mpg)) +
  geom_point() +
  geom\_line(aes(y = .fitted)) +
  geom_ribbon(aes(ymin = .lower, ymax = .upper), col = NA, alpha = 0.3)
# predict on new data without outcome variable. Output does not include .resid
newdata <- newdata %>%
  select(-mpg)
augment(mod, newdata = newdata)
```

204 glance.survdiff

```
au <- augment(mod, data = mtcars)</pre>
ggplot(au, aes(.hat, .std.resid)) +
 geom_vline(size = 2, colour = "white", xintercept = 0) +
 geom_hline(size = 2, colour = "white", yintercept = 0) +
 geom_point() +
 geom_smooth(se = FALSE)
plot(mod, which = 6)
ggplot(au, aes(.hat, .cooksd)) +
 geom_vline(xintercept = 0, colour = NA) +
 geom_abline(slope = seq(0, 3, by = 0.5), colour = "white") +
 geom_smooth(se = FALSE) +
 geom_point()
# column-wise models
a <- matrix(rnorm(20), nrow = 10)
b <- a + rnorm(length(a))</pre>
result <- lm(b \sim a)
tidy(result)
```

glance.survdiff

Glance at a(n) survdiff object

Description

Glance accepts a model object and returns a tibble::tibble() with exactly one row of model summaries. The summaries are typically goodness of fit measures, p-values for hypothesis tests on residuals, or model convergence information.

Glance never returns information from the original call to the modeling function. This includes the name of the modeling function or any arguments passed to the modeling function.

Glance does not calculate summary measures. Rather, it farms out these computations to appropriate methods and gathers the results together. Sometimes a goodness of fit measure will be undefined. In these cases the measure will be reported as NA.

Glance returns the same number of columns regardless of whether the model matrix is rank-deficient or not. If so, entries in columns that no longer have a well-defined value are filled in with an NA of the appropriate type.

Usage

```
## S3 method for class 'survdiff'
glance(x, ...)
```

glance.survdiff 205

Arguments

. . .

x An survdiff object returned from survival::survdiff().

Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in . . . , where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

Value

A tibble::tibble() with exactly one row and columns:

df Degrees of freedom used by the model.
p.value P-value corresponding to the test statistic.

statistic Test statistic.

See Also

```
glance(), survival::survdiff()
Other survdiff tidiers: tidy.survdiff()
Other survival tidiers: augment.coxph(), augment.survreg(), glance.aareg(), glance.cch(),
glance.coxph(), glance.pyears(), glance.survexp(), glance.survfit(), glance.survreg(),
tidy.aareg(), tidy.cch(), tidy.coxph(), tidy.pyears(), tidy.survdiff(), tidy.survexp(),
tidy.survfit(), tidy.survreg()
```

```
# load libraries for models and data
library(survival)

# fit model
s <- survdiff(
   Surv(time, status) ~ pat.karno + strata(inst),
   data = lung
)

# summarize model fit with tidiers
tidy(s)
glance(s)</pre>
```

206 glance.survexp

glance.survexp

Glance at a(n) survexp object

Description

Glance accepts a model object and returns a tibble::tibble() with exactly one row of model summaries. The summaries are typically goodness of fit measures, p-values for hypothesis tests on residuals, or model convergence information.

Glance never returns information from the original call to the modeling function. This includes the name of the modeling function or any arguments passed to the modeling function.

Glance does not calculate summary measures. Rather, it farms out these computations to appropriate methods and gathers the results together. Sometimes a goodness of fit measure will be undefined. In these cases the measure will be reported as NA.

Glance returns the same number of columns regardless of whether the model matrix is rank-deficient or not. If so, entries in columns that no longer have a well-defined value are filled in with an NA of the appropriate type.

Usage

```
## S3 method for class 'survexp'
glance(x, ...)
```

Arguments

x An survexp object returned from survival::survexp().

.. Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in . . . , where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

Value

A tibble::tibble() with exactly one row and columns:

n.max Maximum number of subjects at risk.n.start Initial number of subjects at risk.

timepoints Number of timepoints.

glance.survfit 207

See Also

```
glance(), survival::survexp()
Other survexp tidiers: tidy.survexp()
Other survival tidiers: augment.coxph(), augment.survreg(), glance.aareg(), glance.cch(),
glance.coxph(), glance.pyears(), glance.survdiff(), glance.survfit(), glance.survreg(),
tidy.aareg(), tidy.cch(), tidy.coxph(), tidy.pyears(), tidy.survdiff(), tidy.survexp(),
tidy.survfit(), tidy.survreg()
```

Examples

```
# load libraries for models and data
library(survival)

# fit model
sexpfit <- survexp(
  futime ~ 1,
  rmap = list(
    sex = "male",
    year = accept.dt,
    age = (accept.dt - birth.dt)
  ),
  method = "conditional",
  data = jasa
)

# summarize model fit with tidiers
tidy(sexpfit)
glance(sexpfit)</pre>
```

glance.survfit

Glance at a(n) survfit object

Description

Glance accepts a model object and returns a tibble::tibble() with exactly one row of model summaries. The summaries are typically goodness of fit measures, p-values for hypothesis tests on residuals, or model convergence information.

Glance never returns information from the original call to the modeling function. This includes the name of the modeling function or any arguments passed to the modeling function.

Glance does not calculate summary measures. Rather, it farms out these computations to appropriate methods and gathers the results together. Sometimes a goodness of fit measure will be undefined. In these cases the measure will be reported as NA.

Glance returns the same number of columns regardless of whether the model matrix is rank-deficient or not. If so, entries in columns that no longer have a well-defined value are filled in with an NA of the appropriate type.

208 glance.survfit

Usage

```
## S3 method for class 'survfit'
glance(x, ...)
```

Arguments

x An survfit object returned from survival::survfit().

... Additional arguments passed to survival::summary.survfit(). Important

arguments include rmean.

Value

A tibble::tibble() with exactly one row and columns:

events Number of events.

n.max Maximum number of subjects at risk.n.start Initial number of subjects at risk.nobs Number of observations used.

records Number of observations

rmean Restricted mean (see [survival::print.survfit()]).

rmean.std.error

Restricted mean standard error.

conf.low lower end of confidence interval on median conf.high upper end of confidence interval on median

median median survival

See Also

```
glance(), survival::survfit()
Other cch tidiers: glance.cch(), tidy.cch()
Other survival tidiers: augment.coxph(), augment.survreg(), glance.aareg(), glance.cch(),
glance.coxph(), glance.pyears(), glance.survdiff(), glance.survexp(), glance.survreg(),
tidy.aareg(), tidy.cch(), tidy.coxph(), tidy.pyears(), tidy.survdiff(), tidy.survexp(),
tidy.survfit(), tidy.survreg()
```

```
# load libraries for models and data
library(survival)

# fit model
cfit <- coxph(Surv(time, status) ~ age + sex, lung)
sfit <- survfit(cfit)

# summarize model fit with tidiers + visualization</pre>
```

glance.survreg 209

```
tidy(sfit)
glance(sfit)

library(ggplot2)

ggplot(tidy(sfit), aes(time, estimate)) +
    geom_line() +
    geom_ribbon(aes(ymin = conf.low, ymax = conf.high), alpha = .25)

# multi-state
fitCI <- survfit(Surv(stop, status * as.numeric(event), type = "mstate") ~ 1,
    data = mgus1, subset = (start == 0)
)

td_multi <- tidy(fitCI)

td_multi
ggplot(td_multi, aes(time, estimate, group = state)) +
    geom_line(aes(color = state)) +
    geom_ribbon(aes(ymin = conf.low, ymax = conf.high), alpha = .25)</pre>
```

glance.survreg

Glance at a(n) survreg object

Description

Glance accepts a model object and returns a tibble::tibble() with exactly one row of model summaries. The summaries are typically goodness of fit measures, p-values for hypothesis tests on residuals, or model convergence information.

Glance never returns information from the original call to the modeling function. This includes the name of the modeling function or any arguments passed to the modeling function.

Glance does not calculate summary measures. Rather, it farms out these computations to appropriate methods and gathers the results together. Sometimes a goodness of fit measure will be undefined. In these cases the measure will be reported as NA.

Glance returns the same number of columns regardless of whether the model matrix is rank-deficient or not. If so, entries in columns that no longer have a well-defined value are filled in with an NA of the appropriate type.

Usage

```
## S3 method for class 'survreg'
glance(x, ...)
```

210 glance.survreg

Arguments

x An survreg object returned from survival::survreg().

Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in . . . , where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

Value

A tibble::tibble() with exactly one row and columns:

AIC Akaike's Information Criterion for the model.

BIC Bayesian Information Criterion for the model.

df Degrees of freedom used by the model.

df.residual Residual degrees of freedom.

iter Iterations of algorithm/fitting procedure completed.

logLik The log-likelihood of the model. [stats::logLik()] may be a useful reference.

nobs Number of observations used.

p.value P-value corresponding to the test statistic.

statistic Chi-squared statistic.

See Also

```
glance(), survival::survreg()
Other survreg tidiers: augment.survreg(), tidy.survreg()
Other survival tidiers: augment.coxph(), augment.survreg(), glance.aareg(), glance.cch(),
glance.coxph(), glance.pyears(), glance.survdiff(), glance.survexp(), glance.survfit(),
tidy.aareg(), tidy.cch(), tidy.coxph(), tidy.pyears(), tidy.survdiff(), tidy.survexp(),
tidy.survfit(), tidy.survreg()
```

```
# load libraries for models and data
library(survival)

# fit model
sr <- survreg(
   Surv(futime, fustat) ~ ecog.ps + rx,
   ovarian,
   dist = "exponential"</pre>
```

glance.svyglm 211

```
# summarize model fit with tidiers + visualization
tidy(sr)
augment(sr, ovarian)
glance(sr)

# coefficient plot
td <- tidy(sr, conf.int = TRUE)

library(ggplot2)

ggplot(td, aes(estimate, term)) +
  geom_point() +
  geom_errorbarh(aes(xmin = conf.low, xmax = conf.high), height = 0) +
  geom_vline(xintercept = 0)</pre>
```

glance.svyglm

Glance at a(n) svyglm object

Description

Glance accepts a model object and returns a tibble::tibble() with exactly one row of model summaries. The summaries are typically goodness of fit measures, p-values for hypothesis tests on residuals, or model convergence information.

Glance never returns information from the original call to the modeling function. This includes the name of the modeling function or any arguments passed to the modeling function.

Glance does not calculate summary measures. Rather, it farms out these computations to appropriate methods and gathers the results together. Sometimes a goodness of fit measure will be undefined. In these cases the measure will be reported as NA.

Glance returns the same number of columns regardless of whether the model matrix is rank-deficient or not. If so, entries in columns that no longer have a well-defined value are filled in with an NA of the appropriate type.

Usage

```
## S3 method for class 'svyglm'
glance(x, maximal = x, ...)
```

Arguments

X

A svyglm object returned from survey::svyglm().

maximal

A svyglm object corresponding to the maximal model against which to compute the BIC. See Lumley and Scott (2015) for details. Defaults to x, which is equivalent to not using a maximal model.

212 glance.svyglm

Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in . . ., where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

Value

. . .

A tibble::tibble() with exactly one row and columns:

AIC Akaike's Information Criterion for the model.

BIC Bayesian Information Criterion for the model.

deviance Deviance of the model.

df.null Degrees of freedom used by the null model.

df.residual Residual degrees of freedom. null.deviance Deviance of the null model.

References

Lumley T, Scott A (2015). AIC and BIC for modelling with complex survey data. *Journal of Survey Statistics and Methodology*, 3(1).

See Also

```
survey::svyglm(), stats::glm(), survey::anova.svyglm
Other lm tidiers: augment.glm(), augment.lm(), glance.glm(), glance.lm(), glance.summary.lm(),
tidy.glm(), tidy.lm(), tidy.lm.beta(), tidy.mlm(), tidy.summary.lm()
```

```
# load libraries for models and data
library(survey)

set.seed(123)
data(api)

# survey design
dstrat <-
    svydesign(
    id = ~1,
        strata = ~stype,
        weights = ~pw,
        data = apistrat,
        fpc = ~fpc</pre>
```

213 glance.svyolr

```
)
# model
m <- svyglm(
 formula = sch.wide ~ ell + meals + mobility,
 design = dstrat,
 family = quasibinomial()
glance(m)
```

glance.svyolr

Glance at a(n) svyolr object

Description

Glance accepts a model object and returns a tibble::tibble() with exactly one row of model summaries. The summaries are typically goodness of fit measures, p-values for hypothesis tests on residuals, or model convergence information.

Glance never returns information from the original call to the modeling function. This includes the name of the modeling function or any arguments passed to the modeling function.

Glance does not calculate summary measures. Rather, it farms out these computations to appropriate methods and gathers the results together. Sometimes a goodness of fit measure will be undefined. In these cases the measure will be reported as NA.

Glance returns the same number of columns regardless of whether the model matrix is rank-deficient or not. If so, entries in columns that no longer have a well-defined value are filled in with an NA of the appropriate type.

Usage

```
## S3 method for class 'svyolr'
glance(x, ...)
```

Arguments Χ

A svyolr object returned from survey::svyolr().

Additional arguments. Not used. Needed to match generic signature only. Cautionary note: Misspelled arguments will be absorbed in . . . , where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf. level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

214 glance.varest

Value

```
A tibble::tibble() with exactly one row and columns:
```

df.residual Residual degrees of freedom.

edf The effective degrees of freedom.

Number of observations used.

See Also

```
tidy, survey::svyolr()
Other ordinal tidiers: augment.clm(), augment.polr(), glance.clm(), glance.clmm(), glance.polr(),
tidy.clm(), tidy.clmm(), tidy.polr(), tidy.svyolr()
```

Examples

```
library(broom)
library(survey)

data(api)
dclus1 <- svydesign(id = ~dnum, weights = ~pw, data = apiclus1, fpc = ~fpc)
dclus1 <- update(dclus1, mealcat = cut(meals, c(0, 25, 50, 75, 100)))

m <- svyolr(mealcat ~ avg.ed + mobility + stype, design = dclus1)

m

tidy(m, conf.int = TRUE)</pre>
```

glance.varest

Glance at a(n) varest object

Description

Glance accepts a model object and returns a tibble::tibble() with exactly one row of model summaries. The summaries are typically goodness of fit measures, p-values for hypothesis tests on residuals, or model convergence information.

Glance never returns information from the original call to the modeling function. This includes the name of the modeling function or any arguments passed to the modeling function.

Glance does not calculate summary measures. Rather, it farms out these computations to appropriate methods and gathers the results together. Sometimes a goodness of fit measure will be undefined. In these cases the measure will be reported as NA.

Glance returns the same number of columns regardless of whether the model matrix is rank-deficient or not. If so, entries in columns that no longer have a well-defined value are filled in with an NA of the appropriate type.

glance.varest 215

Usage

```
## S3 method for class 'varest'
glance(x, ...)
```

Arguments

A varest object produced by a call to vars::VAR().

Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in . . ., where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

Value

A tibble::tibble() with exactly one row and columns:

lag.order Lag order.

logLik The log-likelihood of the model. [stats::logLik()] may be a useful reference.

n The total number of observations.

Number of observations used.

See Also

```
glance(), vars::VAR()
```

```
# load libraries for models and data
library(vars)

# load data
data("Canada", package = "vars")

# fit models
mod <- VAR(Canada, p = 1, type = "both")

# summarize model fit with tidiers
tidy(mod)
glance(mod)</pre>
```

216 glance_optim

glance_optim

Tidy a(n) optim object masquerading as list

Description

Broom tidies a number of lists that are effectively S3 objects without a class attribute. For example, stats::optim(), svd() and interp::interp() produce consistent output, but because they do not have a class attribute, they cannot be handled by S3 dispatch.

These functions look at the elements of a list and determine if there is an appropriate tidying method to apply to the list. Those tidiers are implemented as functions of the form tidy_<function> or glance_<function> and are not exported (but they are documented!).

If no appropriate tidying method is found, they throw an error.

Usage

```
glance_optim(x, ...)
```

Arguments

x A list returned from stats::optim().

Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in . . ., where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

Value

A tibble::tibble() with exactly one row and columns:

```
convergence Convergence code.

function.count Number of calls to 'fn'.

gradient.count Number of calls to 'gr'.

value Minimized or maximized output value.
```

See Also

```
glance(), stats::optim()
Other list tidiers: list_tidiers, tidy_irlba(), tidy_optim(), tidy_svd(), tidy_xyz()
```

leveneTest_tidiers 217

Examples

```
f <- function(x) (x[1] - 2)^2 + (x[2] - 3)^2 + (x[3] - 8)^2
o <- optim(c(1, 1, 1), f)
```

leveneTest_tidiers

Tidy/glance a(n) leveneTest object

Description

For models that have only a single component, the tidy() and glance() methods are identical. Please see the documentation for both of those methods.

Usage

```
## S3 method for class 'leveneTest'
tidy(x, ...)
```

Arguments

x An object of class anova created by a call to car::leveneTest().

Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in . . . , where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

Value

A tibble::tibble() with columns:

df Degrees of freedom used by this term in the model.

p. value The two-sided p-value associated with the observed statistic.

statistic The value of a T-statistic to use in a hypothesis that the regression term is non-

zero.

df.residual Residual degrees of freedom.

See Also

```
tidy(), glance(), car::leveneTest()
Other car tidiers: durbinWatsonTest_tidiers
```

218 list_tidiers

Examples

```
# load libraries for models and data
library(car)

data(Moore)

lt <- with(Moore, leveneTest(conformity, fcategory))

tidy(lt)
glance(lt)</pre>
```

list_tidiers

Tidying methods for lists / returned values that are not S3 objects

Description

Broom tidies a number of lists that are effectively S3 objects without a class attribute. For example, stats::optim(), base::svd() and interp::interp() produce consistent output, but because they do not have a class attribute, they cannot be handled by S3 dispatch.

Usage

```
## S3 method for class 'list'
tidy(x, ...)
## S3 method for class 'list'
glance(x, ...)
```

Arguments

x A list, potentially representing an object that can be tidied.

. . . Additionally, arguments passed to the tidying function.

Details

These functions look at the elements of a list and determine if there is an appropriate tidying method to apply to the list. Those tidiers are themselves are implemented as functions of the form tidy_<function> or glance_<function> and are not exported (but they are documented!).

If no appropriate tidying method is found, throws an error.

See Also

```
Other list tidiers: glance_optim(), tidy_irlba(), tidy_optim(), tidy_svd(), tidy_xyz()
```

null_tidiers 219

null_tidiers

Tidiers for NULL inputs

Description

tidy(NULL), glance(NULL) and augment(NULL) all return an empty tibble::tibble. This empty tibble can be treated a tibble with zero rows, making it convenient to combine with other tibbles using functions like purrr::map_df() on lists of potentially NULL objects.

Usage

```
## S3 method for class '`NULL`'
tidy(x, ...)
## S3 method for class '`NULL`'
glance(x, ...)
## S3 method for class '`NULL`'
augment(x, ...)
```

Arguments

x The value NULL.

. . . Additional arguments (not used).

Value

An empty tibble::tibble.

See Also

tibble::tibble

sp_tidiers

Tidy a(n) SpatialPolygonsDataFrame object

Description

Tidy summarizes information about the components of a model. A model component might be a single term in a regression, a single hypothesis, a cluster, or a class. Exactly what tidy considers to be a model component varies across models but is usually self-evident. If a model has several distinct types of components, you will need to specify which components to return.

Note that the sf package now defines tidy spatial objects and is the recommended approach to spatial data. sp tidiers are now deprecated in favor of sf::st_as_sf() and coercion methods found in other packages. See https://r-spatial.org/r/2023/05/15/evolution4.html for more on migration from retiring spatial packages.

220 summary_tidiers

Usage

```
## S3 method for class 'SpatialPolygonsDataFrame'
tidy(x, region = NULL, ...)

## S3 method for class 'SpatialPolygons'
tidy(x, ...)

## S3 method for class 'Polygons'
tidy(x, ...)

## S3 method for class 'Polygon'
tidy(x, ...)

## S3 method for class 'SpatialLinesDataFrame'
tidy(x, ...)

## S3 method for class 'Lines'
tidy(x, ...)

## S3 method for class 'Lines'
tidy(x, ...)
```

Arguments

x A SpatialPolygonsDataFrame, SpatialPolygons, Polygon, SpatialLinesDataFrame, Lines or Line object.region name of variable used to split up regions

... not used by this method

summary_tidiers (Deprecated) Tidy summaryDefault objects

Description

Tidiers for summary Default objects have been deprecated as of broom 0.7.0 in favor of skimr::skim().

Usage

```
## S3 method for class 'summaryDefault'
tidy(x, ...)
## S3 method for class 'summaryDefault'
glance(x, ...)
```

summary_tidiers 221

Arguments

. . .

x A summaryDefault object, created by calling summary() on a vector.

Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in . . ., where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

Value

A one-row tibble::tibble with columns:

minimum Minimum value in original vector.

q1 First quartile of original vector.

median Median of original vector.

mean Mean of original vector.

q3 Third quartile of original vector.

maximum Maximum value in original vector.

na Number of NA values in original vector. Column present only when original

vector had at least one NA entry.

See Also

```
Other deprecated: bootstrap(), confint_tidy(), data.frame_tidiers, finish_glance(), fix_data_frame(), tidy.density(), tidy.dist(), tidy.ftable(), tidy.numeric()

Other deprecated: bootstrap(), confint_tidy(), data.frame_tidiers, finish_glance(), fix_data_frame(), tidy.density(), tidy.dist(), tidy.ftable(), tidy.numeric()
```

```
v <- rnorm(1000)
s <- summary(v)
s

tidy(s)
glance(s)

v2 <- c(v,NA)
tidy(summary(v2))</pre>
```

222 tidy.aareg

tidy.aareg	Tidy a(n) aareg object	

Description

Tidy summarizes information about the components of a model. A model component might be a single term in a regression, a single hypothesis, a cluster, or a class. Exactly what tidy considers to be a model component varies across models but is usually self-evident. If a model has several distinct types of components, you will need to specify which components to return.

Usage

```
## S3 method for class 'aareg' tidy(x, ...)
```

Arguments

. . .

x An aareg object returned from survival::aareg().

Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in . . . , where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

Details

robust. se is only present when x was created with dfbeta = TRUE.

Value

z

A tibble::tibble() with columns:

z score.

robust.se The value of the regression term.

The two-sided p-value associated with the observed statistic.

robust.se robust version of standard error estimate.

Statistic The value of a T-statistic to use in a hypothesis that the regression term is non-zero.

Std.error The standard error of the regression term.

The name of the regression term.

tidy.acf 223

See Also

```
tidy(), survival::aareg()
Other aareg tidiers: glance.aareg()
Other survival tidiers: augment.coxph(), augment.survreg(), glance.aareg(), glance.cch(),
glance.coxph(), glance.pyears(), glance.survdiff(), glance.survexp(), glance.survfit(),
glance.survreg(), tidy.cch(), tidy.coxph(), tidy.pyears(), tidy.survdiff(), tidy.survexp(),
tidy.survfit(), tidy.survreg()
```

Examples

```
# load libraries for models and data
library(survival)

# fit model
afit <- aareg(
   Surv(time, status) ~ age + sex + ph.ecog,
   data = lung,
   dfbeta = TRUE
)

# summarize model fit with tidiers
tidy(afit)</pre>
```

tidy.acf

Tidy a(n) acf object

Description

Tidy summarizes information about the components of a model. A model component might be a single term in a regression, a single hypothesis, a cluster, or a class. Exactly what tidy considers to be a model component varies across models but is usually self-evident. If a model has several distinct types of components, you will need to specify which components to return.

Usage

```
## S3 method for class 'acf'
tidy(x, ...)
```

Arguments

x An acf object created by stats::acf(), stats::pacf() or stats::ccf().

... Additional arguments. Not used. Needed to match generic signature only. Cautionary note: Misspelled arguments will be absorbed in ..., where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

224 tidy.anova

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

Value

```
A tibble::tibble() with columns:

acf Autocorrelation.

lag Lag values.
```

See Also

```
tidy(), stats::acf(), stats::pacf(), stats::ccf()
Other time series tidiers: tidy.spec(), tidy.ts(), tidy.zoo()
```

Examples

```
tidy(acf(lh, plot = FALSE))
tidy(ccf(mdeaths, fdeaths, plot = FALSE))
tidy(pacf(lh, plot = FALSE))
```

tidy.anova

Tidy a(n) anova object

Description

Tidy summarizes information about the components of a model. A model component might be a single term in a regression, a single hypothesis, a cluster, or a class. Exactly what tidy considers to be a model component varies across models but is usually self-evident. If a model has several distinct types of components, you will need to specify which components to return.

Usage

```
## S3 method for class 'anova' tidy(x, ...)
```

Arguments

An anova object, such as those created by stats::anova(), car::Anova(), car::leveneTest(), or car::linearHypothesis().

Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in . . . , where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

tidy.anova 225

• tidy() methods will warn when supplied an exponentiate argument if it will be ignored.

• augment() methods will warn when supplied a newdata argument if it will be ignored.

Details

The term column of an ANOVA table can come with leading or trailing whitespace, which this tidying method trims.

For documentation on the tidier for car::leveneTest() output, see tidy.leveneTest()

Value

A tibble::tibble() with columns:

df Degrees of freedom used by this term in the model.

meansq Mean sum of squares. Equal to total sum of squares divided by degrees of

freedom.

p.value The two-sided p-value associated with the observed statistic.

statistic The value of a T-statistic to use in a hypothesis that the regression term is non-

zero

sumsq Sum of squares explained by this term.
term The name of the regression term.

See Also

```
tidy(), stats::anova(), car::Anova(), car::leveneTest()
Other anova tidiers: glance.anova(), glance.aov(), tidy.TukeyHSD(), tidy.aov(), tidy.aovlist(), tidy.manova()
```

```
# fit models
a <- lm(mpg ~ wt + qsec + disp, mtcars)
b <- lm(mpg ~ wt + qsec, mtcars)

mod <- anova(a, b)

# summarize model fit with tidiers
tidy(mod)
glance(mod)

# car::linearHypothesis() example
library(car)
mod_lht <- linearHypothesis(a, "wt - disp")
tidy(mod_lht)
glance(mod_lht)</pre>
```

226 tidy.aov

tidy.aov

Tidy a(n) aov object

Description

Tidy summarizes information about the components of a model. A model component might be a single term in a regression, a single hypothesis, a cluster, or a class. Exactly what tidy considers to be a model component varies across models but is usually self-evident. If a model has several distinct types of components, you will need to specify which components to return.

Usage

```
## S3 method for class 'aov'
tidy(x, intercept = FALSE, ...)
```

Arguments

Х

An aov object, such as those created by stats::aov().

intercept

A logical indicating whether information on the intercept ought to be included. Passed to stats::summary.aov().

Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in . . . , where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

Details

The term column of an ANOVA table can come with leading or trailing whitespace, which this tidying method trims.

For documentation on the tidier for car::leveneTest() output, see tidy.leveneTest()

See Also

```
tidy(), stats::aov()
Other anova tidiers: glance.anova(), glance.aov(), tidy.TukeyHSD(), tidy.anova(), tidy.aovlist(),
tidy.manova()
```

```
a <- aov(mpg ~ wt + qsec + disp, mtcars)
tidy(a)</pre>
```

tidy.aovlist 227

tidy.aovlist	Tidy $a(n)$ aovlist object

Description

Tidy summarizes information about the components of a model. A model component might be a single term in a regression, a single hypothesis, a cluster, or a class. Exactly what tidy considers to be a model component varies across models but is usually self-evident. If a model has several distinct types of components, you will need to specify which components to return.

Usage

```
## S3 method for class 'aovlist'
tidy(x, ...)
```

Arguments

. . .

x An aovlist objects, such as those created by stats::aov().

Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in . . . , where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

Details

The term column of an ANOVA table can come with leading or trailing whitespace, which this tidying method trims.

For documentation on the tidier for car::leveneTest() output, see tidy.leveneTest()

Value

A tibble::tibble() with columns:

df Degrees of freedom used by this term in the model.

meansq Mean sum of squares. Equal to total sum of squares divided by degrees of freedom.

p.value The two-sided p-value associated with the observed statistic.

statistic The value of a T-statistic to use in a hypothesis that the regression term is non-

zero.

stratum The error stratum.

sumsq Sum of squares explained by this term.
term The name of the regression term.

228 tidy.Arima

See Also

```
tidy(), stats::aov()
Other anova tidiers: glance.anova(), glance.aov(), tidy.TukeyHSD(), tidy.anova(), tidy.aov(), tidy.manova()
```

Examples

```
a <- aov(mpg ~ wt + qsec + Error(disp / am), mtcars)
tidy(a)</pre>
```

tidy.Arima

Tidy a(n) Arima object

Description

Tidy summarizes information about the components of a model. A model component might be a single term in a regression, a single hypothesis, a cluster, or a class. Exactly what tidy considers to be a model component varies across models but is usually self-evident. If a model has several distinct types of components, you will need to specify which components to return.

Usage

```
## S3 method for class 'Arima'
tidy(x, conf.int = FALSE, conf.level = 0.95, ...)
```

Arguments

x An object of class Arima created by stats::arima().

conf.int Logical indicating whether or not to include a confidence interval in the tidied output. Defaults to FALSE.

output. Detautts to FALSE

conf.level The confidence level to use for the confidence interval if conf.int = TRUE. Must be strictly greater than 0 and less than 1. Defaults to 0.95, which corresponds to

a 95 percent confidence interval.

Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in . . . , where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

tidy.betamfx 229

Value

```
A tibble::tibble() with columns:
```

conf.high Upper bound on the confidence interval for the estimate.

conf.low Lower bound on the confidence interval for the estimate.

estimate The estimated value of the regression term.

std.error The standard error of the regression term.

term The name of the regression term.

See Also

```
stats::arima()
Other Arima tidiers: glance.Arima()
```

Examples

```
# fit model
fit <- arima(lh, order = c(1, 0, 0))
# summarize model fit with tidiers
tidy(fit)
glance(fit)</pre>
```

tidy.betamfx

Tidy a(n) betamfx object

Description

Tidy summarizes information about the components of a model. A model component might be a single term in a regression, a single hypothesis, a cluster, or a class. Exactly what tidy considers to be a model component varies across models but is usually self-evident. If a model has several distinct types of components, you will need to specify which components to return.

Usage

```
## S3 method for class 'betamfx'
tidy(x, conf.int = FALSE, conf.level = 0.95, ...)
```

Arguments

x A betamfx object.

conf.int Logical indicating whether or not to include a confidence interval in the tidied

output. Defaults to FALSE.

230 tidy.betamfx

conf.level

The confidence level to use for the confidence interval if conf.int = TRUE. Must be strictly greater than 0 and less than 1. Defaults to 0.95, which corresponds to a 95 percent confidence interval.

Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in . . . , where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

Details

The mfx package provides methods for calculating marginal effects for various generalized linear models (GLMs). Unlike standard linear models, estimated model coefficients in a GLM cannot be directly interpreted as marginal effects (i.e., the change in the response variable predicted after a one unit change in one of the regressors). This is because the estimated coefficients are multiplicative, dependent on both the link function that was used for the estimation and any other variables that were included in the model. When calculating marginal effects, users must typically choose whether they want to use i) the average observation in the data, or ii) the average of the sample marginal effects. See vignette("mfxarticle") from the mfx package for more details.

Value

A tibble::tibble() with columns:

conf.high	Upper bound on the confidence interval for the estimate.
conf.low	Lower bound on the confidence interval for the estimate.
estimate	The estimated value of the regression term.
p.value	The two-sided p-value associated with the observed statistic.
statistic	The value of a T-statistic to use in a hypothesis that the regression term is non-zero.
std.error	The standard error of the regression term.
term	The name of the regression term.
atmean	TRUE if the marginal effects were originally calculated as the partial effects for the average observation. If FALSE, then these were instead calculated as average partial effects.

See Also

```
tidy.betareg(), mfx::betamfx()
Other mfx tidiers: augment.betamfx(), augment.mfx(), glance.betamfx(), glance.mfx(),
tidy.mfx()
```

tidy.betareg 231

Examples

```
library(mfx)
# Simulate some data
set.seed(12345)
n <- 1000
x <- rnorm(n)</pre>
# Beta outcome
y < - rbeta(n, shape1 = plogis(1 + 0.5 * x), shape2 = (abs(0.2 * x)))
# Use Smithson and Verkuilen correction
y \leftarrow (y * (n - 1) + 0.5) / n
d <- data.frame(y, x)</pre>
mod\_betamfx \leftarrow betamfx(y \sim x \mid x, data = d)
tidy(mod_betamfx, conf.int = TRUE)
# Compare with the naive model coefficients of the equivalent betareg call (not run)
# tidy(betamfx(y \sim x | x, data = d), conf.int = TRUE)
augment(mod_betamfx)
glance(mod_betamfx)
```

tidy.betareg

Tidy a(n) betareg object

Description

Tidy summarizes information about the components of a model. A model component might be a single term in a regression, a single hypothesis, a cluster, or a class. Exactly what tidy considers to be a model component varies across models but is usually self-evident. If a model has several distinct types of components, you will need to specify which components to return.

Usage

```
## S3 method for class 'betareg'
tidy(x, conf.int = FALSE, conf.level = 0.95, ...)
```

Arguments

X	A betareg object produced by a call to betareg::betareg().
conf.int	Logical indicating whether or not to include a confidence interval in the tidied output. Defaults to FALSE.
conf.level	The confidence level to use for the confidence interval if conf.int = TRUE. Must be strictly greater than 0 and less than 1. Defaults to 0.95, which corresponds to a 95 percent confidence interval.

232 tidy.betareg

Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in . . . , where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

Details

. . .

The tibble has one row for each term in the regression. The component column indicates whether a particular term was used to model either the "mean" or "precision". Here the precision is the inverse of the variance, often referred to as phi. At least one term will have been used to model the precision phi.

Value

A tibble::tibble() with columns:

conf.high	Upper bound on the confidence interval for the estimate.
conf.low	Lower bound on the confidence interval for the estimate.
estimate	The estimated value of the regression term.
p.value	The two-sided p-value associated with the observed statistic.
statistic	The value of a T-statistic to use in a hypothesis that the regression term is non-zero.
std.error	The standard error of the regression term.
term	The name of the regression term.
component	Whether a particular term was used to model the mean or the precision in the regression. See details.

See Also

```
tidy(), betareg::betareg()
```

```
# load libraries for models and data
library(betareg)

# load dats
data("GasolineYield", package = "betareg")

# fit model
mod <- betareg(yield ~ batch + temp, data = GasolineYield)</pre>
```

tidy.biglm 233

```
mod

# summarize model fit with tidiers
tidy(mod)
tidy(mod, conf.int = TRUE)
tidy(mod, conf.int = TRUE, conf.level = .99)
augment(mod)
glance(mod)
```

tidy.biglm

Tidy a(n) biglm object

Description

Tidy summarizes information about the components of a model. A model component might be a single term in a regression, a single hypothesis, a cluster, or a class. Exactly what tidy considers to be a model component varies across models but is usually self-evident. If a model has several distinct types of components, you will need to specify which components to return.

Usage

```
## S3 method for class 'biglm'
tidy(x, conf.int = FALSE, conf.level = 0.95, exponentiate = FALSE, ...)
```

Arguments

x A biglm object created by a call to biglm::biglm() or biglm::bigglm().

conf. int Logical indicating whether or not to include a confidence interval in the tidied

output. Defaults to FALSE.

 ${\tt conf.level} \qquad {\tt The confidence \ level \ to \ use \ for \ the \ confidence \ interval \ if \ conf. \ int = TRUE. \ Must}$

be strictly greater than 0 and less than 1. Defaults to 0.95, which corresponds to

a 95 percent confidence interval.

exponentiate Logical indicating whether or not to exponentiate the the coefficient estimates.

This is typical for logistic and multinomial regressions, but a bad idea if there is

no log or logit link. Defaults to FALSE.

Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in . . . , where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

234 tidy.biglm

Value

A tibble::tibble() with columns:

conf.high Upper bound on the confidence interval for the estimate.

conf.low Lower bound on the confidence interval for the estimate.

estimate The estimated value of the regression term.

p.value The two-sided p-value associated with the observed statistic.

statistic The value of a T-statistic to use in a hypothesis that the regression term is non-zero.

std.error The standard error of the regression term.

term The name of the regression term.

See Also

```
tidy(), biglm::biglm(), biglm::bigglm()
Other biglm tidiers: glance.biglm()
```

```
# load modeling library
library(biglm)
# fit model -- linear regression
bfit <- biglm(mpg ~ wt + disp, mtcars)</pre>
# summarize model fit with tidiers
tidy(bfit)
tidy(bfit, conf.int = TRUE)
tidy(bfit, conf.int = TRUE, conf.level = .9)
glance(bfit)
# fit model -- logistic regression
bgfit <- bigglm(am ~ mpg, mtcars, family = binomial())</pre>
# summarize model fit with tidiers
tidy(bgfit)
tidy(bgfit, exponentiate = TRUE)
tidy(bgfit, conf.int = TRUE)
tidy(bgfit, conf.int = TRUE, conf.level = .9)
tidy(bgfit, conf.int = TRUE, conf.level = .9, exponentiate = TRUE)
glance(bgfit)
```

tidy.binDesign 235

tidy.binDesign

Tidy a(n) binDesign object

Description

Tidy summarizes information about the components of a model. A model component might be a single term in a regression, a single hypothesis, a cluster, or a class. Exactly what tidy considers to be a model component varies across models but is usually self-evident. If a model has several distinct types of components, you will need to specify which components to return.

Usage

```
## S3 method for class 'binDesign'
tidy(x, ...)
```

Arguments

x A binGroup::binDesign() object.

Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in . . . , where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

Value

```
A tibble::tibble() with columns:
```

Number of trials in given iteration.powerPower achieved for given value of n.

See Also

```
tidy(), binGroup::binDesign()
Other bingroup tidiers: glance.binDesign(), tidy.binWidth()
```

```
library(binGroup)
des <- binDesign(
  nmax = 300, delta = 0.06,
  p.hyp = 0.1, power = .8</pre>
```

236 tidy.binWidth

```
)
glance(des)
tidy(des)
# the ggplot2 equivalent of plot(des)
library(ggplot2)
ggplot(tidy(des), aes(n, power)) +
 geom_line()
```

tidy.binWidth

Tidy a(n) binWidth object

Description

Tidy summarizes information about the components of a model. A model component might be a single term in a regression, a single hypothesis, a cluster, or a class. Exactly what tidy considers to be a model component varies across models but is usually self-evident. If a model has several distinct types of components, you will need to specify which components to return.

Usage

```
## S3 method for class 'binWidth'
tidy(x, ...)
```

Arguments Х

A binGroup::binWidth() object.

Additional arguments. Not used. Needed to match generic signature only. Cautionary note: Misspelled arguments will be absorbed in . . . , where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

Value

n

A tibble::tibble() with columns:

Alternative hypothesis (character). alternative ci.width Expected width of confidence interval. True proportion. р

Total sample size

tidy.boot 237

See Also

```
tidy(), binGroup::binWidth()
Other bingroup tidiers: glance.binDesign(), tidy.binDesign()
```

Examples

```
# load libraries
library(binGroup)

# fit model
bw <- binWidth(100, .1)

bw

# summarize model fit with tidiers
tidy(bw)</pre>
```

tidy.boot

Tidy a(n) boot object

Description

Tidy summarizes information about the components of a model. A model component might be a single term in a regression, a single hypothesis, a cluster, or a class. Exactly what tidy considers to be a model component varies across models but is usually self-evident. If a model has several distinct types of components, you will need to specify which components to return.

Usage

```
## S3 method for class 'boot'
tidy(
    x,
    conf.int = FALSE,
    conf.level = 0.95,
    conf.method = c("perc", "bca", "basic", "norm"),
    exponentiate = FALSE,
    ...
)
```

Arguments

```
x A boot::boot() object.
```

conf.int Logical indicating whether or not to include a confidence interval in the tidied output. Defaults to FALSE.

238 tidy.boot

conf.level The confidence level to use for the confidence interval if conf.int = TRUE. Must be strictly greater than 0 and less than 1. Defaults to 0.95, which corresponds to

a 95 percent confidence interval.

conf.method Passed to the type argument of boot::boot.ci(). Defaults to "perc". The al-

lowed types are "perc", "basic", "bca", and "norm". Does not support "stud"

or "all".

exponentiate Logical indicating whether or not to exponentiate the the coefficient estimates.

This is typical for logistic and multinomial regressions, but a bad idea if there is

no log or logit link. Defaults to FALSE.

Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in . . ., where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

Details

If weights were provided to the boot function, an estimate column is included showing the weighted bootstrap estimate, and the standard error is of that estimate.

If there are no original statistics in the "boot" object, such as with a call to tsboot with orig.t = FALSE, the original and statistic columns are omitted, and only estimate and std.error columns shown.

Value

A tibble::tibble() with columns:

bias Bias of the statistic.

std.error The standard error of the regression term.

term The name of the regression term. statistic Original value of the statistic.

See Also

```
tidy(), boot::boot(), boot::tsboot(), boot::boot.ci(), rsample::bootstraps()
```

```
# load modeling library
library(boot)

clotting <- data.frame(
    u = c(5, 10, 15, 20, 30, 40, 60, 80, 100),</pre>
```

tidy.btergm 239

```
lot1 = c(118, 58, 42, 35, 27, 25, 21, 19, 18),
lot2 = c(69, 35, 26, 21, 18, 16, 13, 12, 12)
)

# fit models
g1 <- glm(lot2 ~ log(u), data = clotting, family = Gamma)
bootfun <- function(d, i) {
   coef(update(g1, data = d[i, ]))
}
bootres <- boot(clotting, bootfun, R = 999)

# summarize model fits with tidiers
tidy(g1, conf.int = TRUE)
tidy(bootres, conf.int = TRUE)</pre>
```

tidy.btergm

Tidy a(n) btergm object

Description

Tidy summarizes information about the components of a model. A model component might be a single term in a regression, a single hypothesis, a cluster, or a class. Exactly what tidy considers to be a model component varies across models but is usually self-evident. If a model has several distinct types of components, you will need to specify which components to return.

This method tidies the coefficients of a bootstrapped temporal exponential random graph model estimated with the **xergm**. It simply returns the coefficients and their confidence intervals.

Usage

```
## S3 method for class 'btergm'
tidy(x, conf.level = 0.95, exponentiate = FALSE, ...)
```

Arguments

x A btergm::btergm() object.

conf.level Confidence level for confidence in

conf.level Confidence level for confidence intervals. Defaults to 0.95. exponentiate Logical indicating whether or not to exponentiate the the confidence level for confidence intervals.

Logical indicating whether or not to exponentiate the the coefficient estimates. This is typical for logistic and multinomial regressions, but a bad idea if there is no log or logit link. Defaults to EALSE

no log or logit link. Defaults to FALSE.

Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in . . . , where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

240 tidy.btergm

• tidy() methods will warn when supplied an exponentiate argument if it will be ignored.

• augment() methods will warn when supplied a newdata argument if it will be ignored.

Value

A tibble::tibble() with columns:

conf.high Upper bound on the confidence interval for the estimate.

conf.low Lower bound on the confidence interval for the estimate.

estimate The estimated value of the regression term.

term The name of the regression term.

See Also

```
tidy(), btergm::btergm()
```

```
library(btergm)
library(network)
set.seed(5)
# create 10 random networks with 10 actors
networks <- list()</pre>
for (i in 1:10) {
  mat <- matrix(rbinom(100, 1, .25), nrow = 10, ncol = 10)
  diag(mat) <- 0
  nw <- network(mat)</pre>
  networks[[i]] <- nw</pre>
}
# create 10 matrices as covariates
covariates <- list()</pre>
for (i in 1:10) {
  mat <- matrix(rnorm(100), nrow = 10, ncol = 10)
  covariates[[i]] <- mat</pre>
# fit the model
mod <- btergm(networks ~ edges + istar(2) + edgecov(covariates), R = 100)</pre>
# summarize model fit with tidiers
tidy(mod)
```

tidy.cch 241

Description

Tidy summarizes information about the components of a model. A model component might be a single term in a regression, a single hypothesis, a cluster, or a class. Exactly what tidy considers to be a model component varies across models but is usually self-evident. If a model has several distinct types of components, you will need to specify which components to return.

Usage

```
## S3 method for class 'cch'
tidy(x, conf.level = 0.95, ...)
```

Arguments

An cch object returned from survival::cch().

conf.level confidence level for CI

Additional arguments. Not used. Needed to match generic signature only. Cautionary note: Misspelled arguments will be absorbed in . . . , where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

Value

A tibble::tibble() with columns:

conf.high	Upper bound on the confidence interval for the estimate.
conf.low	Lower bound on the confidence interval for the estimate.
estimate	The estimated value of the regression term.
p.value	The two-sided p-value associated with the observed statistic.
statistic	The value of a T-statistic to use in a hypothesis that the regression term is non-zero.
std.error	The standard error of the regression term.
term	The name of the regression term.

242 tidy.cld

See Also

```
tidy(), survival::cch()
Other cch tidiers: glance.cch(), glance.survfit()
Other survival tidiers: augment.coxph(), augment.survreg(), glance.aareg(), glance.cch(),
glance.coxph(), glance.pyears(), glance.survdiff(), glance.survexp(), glance.survfit(),
glance.survreg(), tidy.aareg(), tidy.coxph(), tidy.pyears(), tidy.survdiff(), tidy.survexp(),
tidy.survfit(), tidy.survreg()
```

```
# load libraries for models and data
library(survival)
# examples come from cch documentation
subcoh <- nwtco$in.subcohort</pre>
selccoh <- with(nwtco, rel == 1 | subcoh == 1)</pre>
ccoh.data <- nwtco[selccoh, ]</pre>
ccoh.data$subcohort <- subcoh[selccoh]</pre>
# central-lab histology
ccoh.data$histol <- factor(ccoh.data$histol, labels = c("FH", "UH"))</pre>
# tumour stage
ccoh.data$stage <- factor(ccoh.data$stage, labels = c("I", "II", "III", "IV"))</pre>
ccoh.data$age <- ccoh.data$age / 12 # age in years</pre>
# fit model
fit.ccP <- cch(Surv(edrel, rel) ~ stage + histol + age,</pre>
 data = ccoh.data,
 subcoh = ~subcohort, id = ~seqno, cohort.size = 4028
)
# summarize model fit with tidiers + visualization
tidy(fit.ccP)
# coefficient plot
library(ggplot2)
ggplot(tidy(fit.ccP), aes(x = estimate, y = term)) +
 geom_point() +
 geom_errorbarh(aes(xmin = conf.low, xmax = conf.high), height = 0) +
 geom_vline(xintercept = 0)
```

tidy.cld 243

Description

Tidy summarizes information about the components of a model. A model component might be a single term in a regression, a single hypothesis, a cluster, or a class. Exactly what tidy considers to be a model component varies across models but is usually self-evident. If a model has several distinct types of components, you will need to specify which components to return.

Usage

```
## S3 method for class 'cld' tidy(x, ...)
```

Arguments

A cld object created by calling multcomp::cld() on a glht, confint.glht() or summary.glht() object.

Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in . . ., where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

Value

```
A tibble::tibble() with columns:

contrast Levels being compared.

letters Compact letter display denoting all pair-wise comparisons.
```

See Also

```
tidy(), multcomp::cld(), multcomp::summary.glht(), multcomp::confint.glht(), multcomp::glht()
Other multcomp tidiers: tidy.confint.glht(), tidy.glht(), tidy.summary.glht()
```

```
# load libraries for models and data
library(multcomp)
library(ggplot2)

amod <- aov(breaks ~ wool + tension, data = warpbreaks)
wht <- glht(amod, linfct = mcp(tension = "Tukey"))
tidy(wht)</pre>
```

244 tidy.clm

```
ggplot(wht, aes(lhs, estimate)) +
   geom_point()

CI <- confint(wht)

tidy(CI)

ggplot(CI, aes(lhs, estimate, ymin = lwr, ymax = upr)) +
   geom_pointrange()

tidy(summary(wht))

ggplot(mapping = aes(lhs, estimate)) +
   geom_linerange(aes(ymin = lwr, ymax = upr), data = CI) +
   geom_point(aes(size = p), data = summary(wht)) +
   scale_size(trans = "reverse")

cld <- cld(wht)
tidy(cld)</pre>
```

tidy.clm

Tidy a(n) clm object

Description

Tidy summarizes information about the components of a model. A model component might be a single term in a regression, a single hypothesis, a cluster, or a class. Exactly what tidy considers to be a model component varies across models but is usually self-evident. If a model has several distinct types of components, you will need to specify which components to return.

Usage

```
## S3 method for class 'clm'
tidy(
    x,
    conf.int = FALSE,
    conf.level = 0.95,
    conf.type = c("profile", "Wald"),
    exponentiate = FALSE,
    ...
)
```

Arguments

x A clm object returned from ordinal::clm().

conf.int Logical indicating whether or not to include a confidence interval in the tidied output. Defaults to FALSE.

tidy.clm 245

conf.level The confidence level to use for the confidence interval if conf.int = TRUE. Must
 be strictly greater than 0 and less than 1. Defaults to 0.95, which corresponds to
 a 95 percent confidence interval.

conf.type Whether to use "profile" or "Wald" confidence intervals, passed to the type
 argument of ordinal::confint.clm(). Defaults to "profile".

Logical indicating whether or not to exponentiate the coefficient estimates. This is typical for logistic and multinomial regressions, but a bad idea if there is no log or logit link. Defaults to FALSE.

Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in . . . , where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

Details

exponentiate

In broom 0.7.0 the coefficient_type column was renamed to coef.type, and the contents were changed as well.

Note that intercept type coefficients correspond to alpha parameters, location type coefficients correspond to beta parameters, and scale type coefficients correspond to zeta parameters.

Value

A tibble::tibble() with columns:

conf.high	Upper bound on the confidence interval for the estimate.
conf.low	Lower bound on the confidence interval for the estimate.
estimate	The estimated value of the regression term.
p.value	The two-sided p-value associated with the observed statistic.
statistic	The value of a T-statistic to use in a hypothesis that the regression term is non-zero.
std.error	The standard error of the regression term.
term	The name of the regression term.

See Also

```
tidy, ordinal::clm(), ordinal::confint.clm()
Other ordinal tidiers: augment.clm(), augment.polr(), glance.clm(), glance.clmm(), glance.polr(),
glance.svyolr(), tidy.clmm(), tidy.polr(), tidy.svyolr()
```

246 tidy.clmm

Examples

```
# load libraries for models and data
library(ordinal)

# fit model
fit <- clm(rating ~ temp * contact, data = wine)

# summarize model fit with tidiers
tidy(fit)
tidy(fit, conf.int = TRUE, conf.level = 0.9)
tidy(fit, conf.int = TRUE, conf.type = "Wald", exponentiate = TRUE)

glance(fit)
augment(fit, type.predict = "prob")
augment(fit, type.predict = "class")

# ...and again with another model specification
fit2 <- clm(rating ~ temp, nominal = ~contact, data = wine)

tidy(fit2)
glance(fit2)</pre>
```

tidy.clmm

Tidy a(n) clmm object

Description

Tidy summarizes information about the components of a model. A model component might be a single term in a regression, a single hypothesis, a cluster, or a class. Exactly what tidy considers to be a model component varies across models but is usually self-evident. If a model has several distinct types of components, you will need to specify which components to return.

Usage

```
## S3 method for class 'clmm'
tidy(x, conf.int = FALSE, conf.level = 0.95, exponentiate = FALSE, ...)
```

Arguments

x	A clmm object returned from ordinal::clmm().
conf.int	Logical indicating whether or not to include a confidence interval in the tidied output. Defaults to FALSE.
conf.level	The confidence level to use for the confidence interval if conf.int = TRUE. Must be strictly greater than 0 and less than 1. Defaults to 0.95, which corresponds to a 95 percent confidence interval.

tidy.clmm 247

exponentiate

Logical indicating whether or not to exponentiate the the coefficient estimates. This is typical for logistic and multinomial regressions, but a bad idea if there is no log or logit link. Defaults to FALSE.

. . .

Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in . . . , where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

Value

A tibble::tibble() with columns:

conf.high	Upper bound on the confidence interval for the estimate.
conf.low	Lower bound on the confidence interval for the estimate.
estimate	The estimated value of the regression term.
p.value	The two-sided p-value associated with the observed statistic.
statistic	The value of a T-statistic to use in a hypothesis that the regression term is non-zero.
std.error	The standard error of the regression term.
term	The name of the regression term.

Note

In broom 0.7.0 the coefficient_type column was renamed to coef.type, and the contents were changed as well.

Note that intercept type coefficients correspond to alpha parameters, location type coefficients correspond to beta parameters, and scale type coefficients correspond to zeta parameters.

See Also

```
tidy, ordinal::clmm(), ordinal::confint.clm()
Other ordinal tidiers: augment.clm(), augment.polr(), glance.clm(), glance.clmm(), glance.polr(),
glance.svyolr(), tidy.clm(), tidy.polr(), tidy.svyolr()
```

```
# load libraries for models and data
library(ordinal)

# fit model
fit <- clmm(rating ~ temp + contact + (1 | judge), data = wine)</pre>
```

248 tidy.coeftest

```
# summarize model fit with tidiers
tidy(fit)
tidy(fit, conf.int = TRUE, conf.level = 0.9)
tidy(fit, conf.int = TRUE, exponentiate = TRUE)
glance(fit)
# ...and again with another model specification
fit2 <- clmm(rating ~ temp + (1 | judge), nominal = ~contact, data = wine)</pre>
tidy(fit2)
glance(fit2)
```

tidy.coeftest

Tidy a(n) coeftest object

Description

Tidy summarizes information about the components of a model. A model component might be a single term in a regression, a single hypothesis, a cluster, or a class. Exactly what tidy considers to be a model component varies across models but is usually self-evident. If a model has several distinct types of components, you will need to specify which components to return.

Usage

```
## S3 method for class 'coeftest'
tidy(x, conf.int = FALSE, conf.level = 0.95, ...)
```

Arguments

conf.level

A coeftest object returned from lmtest::coeftest(). Х

conf.int Logical indicating whether or not to include a confidence interval in the tidied output. Defaults to FALSE.

The confidence level to use for the confidence interval if conf. int = TRUE. Must be strictly greater than 0 and less than 1. Defaults to 0.95, which corresponds to

a 95 percent confidence interval.

Additional arguments. Not used. Needed to match generic signature only. Cautionary note: Misspelled arguments will be absorbed in ..., where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

tidy.coeftest 249

Value

A tibble::tibble() with columns:

conf.high Upper bound on the confidence interval for the estimate.

Lower bound on the confidence interval for the estimate.

The estimated value of the regression term.

The two-sided p-value associated with the observed statistic.

The value of a T-statistic to use in a hypothesis that the regression term is non-zero.

Std.error The standard error of the regression term.

term The name of the regression term.

See Also

```
tidy(), lmtest::coeftest()
```

```
# load libraries for models and data
library(lmtest)
m <- lm(dist ~ speed, data = cars)</pre>
coeftest(m)
tidy(coeftest(m))
tidy(coeftest(m, conf.int = TRUE))
# a very common workflow is to combine lmtest::coeftest with alternate
# variance-covariance matrices via the sandwich package. The lmtest
# tidiers support this workflow too, enabling you to adjust the standard
# errors of your tidied models on the fly.
library(sandwich)
# "HC3" (default) robust SEs
tidy(coeftest(m, vcov = vcovHC))
# "HC2" robust SEs
tidy(coeftest(m, vcov = vcovHC, type = "HC2"))
# N-W HAC robust SEs
tidy(coeftest(m, vcov = NeweyWest))
# the columns of the returned tibble for glance.coeftest() will vary
# depending on whether the coeftest object retains the underlying model.
# Users can control this with the "save = TRUE" argument of coeftest().
glance(coeftest(m))
glance(coeftest(m, save = TRUE))
```

250 tidy.confint.glht

tidy.confint.glht $Tidy \ a(n) \ confint.glht \ object$

Description

Tidy summarizes information about the components of a model. A model component might be a single term in a regression, a single hypothesis, a cluster, or a class. Exactly what tidy considers to be a model component varies across models but is usually self-evident. If a model has several distinct types of components, you will need to specify which components to return.

Usage

```
## S3 method for class 'confint.glht'
tidy(x, ...)
```

Arguments

x A confint.glht object created by calling multcomp::confint.glht() on a glht object created with multcomp::glht().

Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in . . . , where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

Value

A tibble::tibble() with columns:

conf.high Upper bound on the confidence interval for the estimate.

conf.low Lower bound on the confidence interval for the estimate.

contrast Levels being compared.

estimate The estimated value of the regression term.

See Also

```
tidy(), multcomp::confint.glht(), multcomp::glht()
Other multcomp tidiers: tidy.cld(), tidy.glht(), tidy.summary.glht()
```

tidy.confusionMatrix 251

Examples

```
# load libraries for models and data
library(multcomp)
library(ggplot2)
amod <- aov(breaks ~ wool + tension, data = warpbreaks)</pre>
wht <- glht(amod, linfct = mcp(tension = "Tukey"))</pre>
tidy(wht)
ggplot(wht, aes(lhs, estimate)) +
  geom_point()
CI <- confint(wht)</pre>
tidy(CI)
ggplot(CI, aes(lhs, estimate, ymin = lwr, ymax = upr)) +
  geom_pointrange()
tidy(summary(wht))
ggplot(mapping = aes(lhs, estimate)) +
  geom_linerange(aes(ymin = lwr, ymax = upr), data = CI) +
  geom_point(aes(size = p), data = summary(wht)) +
  scale_size(trans = "reverse")
cld <- cld(wht)</pre>
tidy(cld)
```

 ${\tt tidy.confusionMatrix} \quad \textit{Tidy a(n) confusionMatrix object}$

Description

Tidy summarizes information about the components of a model. A model component might be a single term in a regression, a single hypothesis, a cluster, or a class. Exactly what tidy considers to be a model component varies across models but is usually self-evident. If a model has several distinct types of components, you will need to specify which components to return.

Usage

```
## S3 method for class 'confusionMatrix'
tidy(x, by_class = TRUE, ...)
```

252 tidy.confusionMatrix

Arguments

x An object of class confusionMatrix created by a call to caret::confusionMatrix().

by_class Logical indicating whether or not to show performance measures broken down

by class. Defaults to TRUE. When by class = FALSE only returns a tibble with

accuracy, kappa, and McNemar statistics.

Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in . . . , where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

Value

A tibble::tibble() with columns:

class The class under consideration.

conf.high Upper bound on the confidence interval for the estimate.

conf.low Lower bound on the confidence interval for the estimate.

estimate The estimated value of the regression term.

term The name of the regression term.

p.value P-value for accuracy and kappa statistics.

See Also

```
tidy(), caret::confusionMatrix()
```

```
# load libraries for models and data
library(caret)

set.seed(27)

# generate data
two_class_sample1 <- as.factor(sample(letters[1:2], 100, TRUE))
two_class_sample2 <- as.factor(sample(letters[1:2], 100, TRUE))

two_class_cm <- confusionMatrix(
   two_class_sample1,
   two_class_sample2
)

# summarize model fit with tidiers</pre>
```

tidy.coxph 253

```
tidy(two_class_cm)
tidy(two_class_cm, by_class = FALSE)

# multiclass example
six_class_sample1 <- as.factor(sample(letters[1:6], 100, TRUE))
six_class_sample2 <- as.factor(sample(letters[1:6], 100, TRUE))

six_class_cm <- confusionMatrix(
    six_class_sample1,
    six_class_sample2
)

# summarize model fit with tidiers
tidy(six_class_cm)
tidy(six_class_cm, by_class = FALSE)</pre>
```

tidy.coxph

Tidy a(n) coxph object

Description

Tidy summarizes information about the components of a model. A model component might be a single term in a regression, a single hypothesis, a cluster, or a class. Exactly what tidy considers to be a model component varies across models but is usually self-evident. If a model has several distinct types of components, you will need to specify which components to return.

Usage

```
## S3 method for class 'coxph'
tidy(x, exponentiate = FALSE, conf.int = FALSE, conf.level = 0.95, ...)
```

Arguments

Х	A coxph object returned from survival::coxph().
exponentiate	Logical indicating whether or not to exponentiate the the coefficient estimates. This is typical for logistic and multinomial regressions, but a bad idea if there is no log or logit link. Defaults to FALSE.
conf.int	Logical indicating whether or not to include a confidence interval in the tidied output. Defaults to FALSE.
conf.level	The confidence level to use for the confidence interval if $conf.int = TRUE$. Must be strictly greater than 0 and less than 1. Defaults to 0.95, which corresponds to a 95 percent confidence interval.
	For tidy(), additional arguments passed to summary(x ,). Otherwise ignored.

254 tidy.coxph

Value

```
A tibble::tibble() with columns:
```

estimate The estimated value of the regression term.

p.value The two-sided p-value associated with the observed statistic.

statistic The value of a T-statistic to use in a hypothesis that the regression term is non-

zero.

std.error The standard error of the regression term.

See Also

```
tidy(), survival::coxph()
Other coxph tidiers: augment.coxph(), glance.coxph()
Other survival tidiers: augment.coxph(), augment.survreg(), glance.aareg(), glance.cch(),
glance.coxph(), glance.pyears(), glance.survdiff(), glance.survexp(), glance.survfit(),
glance.survreg(), tidy.aareg(), tidy.cch(), tidy.pyears(), tidy.survdiff(), tidy.survexp(),
tidy.survfit(), tidy.survreg()
```

```
# load libraries for models and data
library(survival)
# fit model
cfit <- coxph(Surv(time, status) ~ age + sex, lung)</pre>
# summarize model fit with tidiers
tidy(cfit)
tidy(cfit, exponentiate = TRUE)
lp <- augment(cfit, lung)</pre>
risks <- augment(cfit, lung, type.predict = "risk")
expected <- augment(cfit, lung, type.predict = "expected")</pre>
glance(cfit)
# also works on clogit models
resp <- levels(logan$occupation)</pre>
n <- nrow(logan)</pre>
indx <- rep(1:n, length(resp))</pre>
logan2 <- data.frame(</pre>
  logan[indx, ],
  id = indx,
  tocc = factor(rep(resp, each = n))
)
logan2$case <- (logan2$occupation == logan2$tocc)</pre>
cl <- clogit(case ~ tocc + tocc:education + strata(id), logan2)</pre>
```

tidy.crr 255

```
tidy(cl)
glance(cl)

library(ggplot2)

ggplot(lp, aes(age, .fitted, color = sex)) +
    geom_point()

ggplot(risks, aes(age, .fitted, color = sex)) +
    geom_point()

ggplot(expected, aes(time, .fitted, color = sex)) +
    geom_point()
```

tidy.crr

Tidy a(n) cmprsk object

Description

Tidy summarizes information about the components of a model. A model component might be a single term in a regression, a single hypothesis, a cluster, or a class. Exactly what tidy considers to be a model component varies across models but is usually self-evident. If a model has several distinct types of components, you will need to specify which components to return.

Usage

```
## S3 method for class 'crr'
tidy(x, exponentiate = FALSE, conf.int = FALSE, conf.level = 0.95, ...)
```

Arguments

x	A crr object returned from cmprsk::crr().
exponentiate	Logical indicating whether or not to exponentiate the the coefficient estimates. This is typical for logistic and multinomial regressions, but a bad idea if there is no log or logit link. Defaults to FALSE.
conf.int	Logical indicating whether or not to include a confidence interval in the tidied output. Defaults to FALSE.
conf.level	The confidence level to use for the confidence interval if conf. int = TRUE. Must be strictly greater than 0 and less than 1. Defaults to 0.95 , which corresponds to a 95 percent confidence interval.
	Additional arguments. Not used. Needed to match generic signature only. Cautionary note: Misspelled arguments will be absorbed in , where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

256 tidy.crr

• tidy() methods will warn when supplied an exponentiate argument if it will be ignored.

• augment() methods will warn when supplied a newdata argument if it will be ignored.

Value

A tibble::tibble() with columns:

```
conf.high Upper bound on the confidence interval for the estimate.

conf.low Lower bound on the confidence interval for the estimate.

estimate The estimated value of the regression term.

p.value The two-sided p-value associated with the observed statistic.

Statistic The value of a T-statistic to use in a hypothesis that the regression term is non-zero.

std.error The standard error of the regression term.
```

See Also

```
tidy(), cmprsk::crr()
Other cmprsk tidiers: glance.crr()
```

```
library(cmprsk)

# time to loco-regional failure (lrf)
lrf_time <- rexp(100)
lrf_event <- sample(0:2, 100, replace = TRUE)
trt <- sample(0:1, 100, replace = TRUE)
strt <- sample(1:2, 100, replace = TRUE)

# fit model
x <- crr(lrf_time, lrf_event, cbind(trt, strt))

# summarize model fit with tidiers
tidy(x, conf.int = TRUE)
glance(x)</pre>
```

tidy.cv.glmnet 257

tidy.cv.glmnet	Tidy a(n) cv.glmnet object

Description

Tidy summarizes information about the components of a model. A model component might be a single term in a regression, a single hypothesis, a cluster, or a class. Exactly what tidy considers to be a model component varies across models but is usually self-evident. If a model has several distinct types of components, you will need to specify which components to return.

Usage

```
## S3 method for class 'cv.glmnet'
tidy(x, ...)
```

Arguments

A cv.glmnet object returned from glmnet::cv.glmnet(). Х

> Additional arguments. Not used. Needed to match generic signature only. Cautionary note: Misspelled arguments will be absorbed in . . . , where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

Value

A tibble::tibble() with columns:

lambda	Value of penalty parameter lambda.
nzero	Number of non-zero coefficients for the given lambda.
std.error	The standard error of the regression term.
conf.low	lower bound on confidence interval for cross-validation estimated loss.
conf.high	upper bound on confidence interval for cross-validation estimated loss.
estimate	Median loss across all cross-validation folds for a given lamdba

See Also

```
tidy(), glmnet::cv.glmnet()
Other glmnet tidiers: glance.cv.glmnet(), glance.glmnet(), tidy.glmnet()
```

258 tidy.cv.glmnet

```
# load libraries for models and data
library(glmnet)
set.seed(27)
nobs <- 100
nvar <- 50
real <- 5
x <- matrix(rnorm(nobs * nvar), nobs, nvar)</pre>
beta <- c(rnorm(real, 0, 1), rep(0, nvar - real))
y \leftarrow c(t(beta) %*% t(x)) + rnorm(nvar, sd = 3)
cvfit1 <- cv.glmnet(x, y)</pre>
tidy(cvfit1)
glance(cvfit1)
library(ggplot2)
tidied_cv <- tidy(cvfit1)</pre>
glance_cv <- glance(cvfit1)</pre>
# plot of MSE as a function of lambda
g <- ggplot(tidied_cv, aes(lambda, estimate)) +</pre>
  geom_line() +
  scale_x_log10()
# plot of MSE as a function of lambda with confidence ribbon
g <- g + geom_ribbon(aes(ymin = conf.low, ymax = conf.high), alpha = .25)
# plot of MSE as a function of lambda with confidence ribbon and choices
# of minimum lambda marked
  geom_vline(xintercept = glance_cv$lambda.min) +
  geom_vline(xintercept = glance_cv$lambda.1se, lty = 2)
# plot of number of zeros for each choice of lambda
ggplot(tidied_cv, aes(lambda, nzero)) +
  geom_line() +
  scale_x_log10()
# coefficient plot with min lambda shown
tidied <- tidy(cvfit1$glmnet.fit)</pre>
ggplot(tidied, aes(lambda, estimate, group = term)) +
  scale_x_log10() +
```

tidy.density 259

```
geom_line() +
geom_vline(xintercept = glance_cv$lambda.min) +
geom_vline(xintercept = glance_cv$lambda.1se, lty = 2)
```

tidy.density

(Deprecated) Tidy density objects

Description

(Deprecated) Tidy density objects

Usage

```
## S3 method for class 'density' tidy(x, ...)
```

Arguments

. . .

x A density object returned from stats::density().

Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in . . . , where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

Value

A tibble::tibble with two columns: points x where the density is estimated, and estimated density y.

See Also

```
Other deprecated: bootstrap(), confint_tidy(), data.frame_tidiers, finish_glance(), fix_data_frame(), summary_tidiers, tidy.dist(), tidy.ftable(), tidy.numeric()
```

260 tidy.dist

tidy.dist

(Deprecated) Tidy dist objects

Description

(Deprecated) Tidy dist objects

Usage

```
## S3 method for class 'dist'
tidy(x, diagonal = attr(x, "Diag"), upper = attr(x, "Upper"), ...)
```

Arguments

x A dist object returned from stats::dist().

diagonal Logical indicating whether or not to tidy the diagonal elements of the distance

matrix. Defaults to whatever was based to the diag argument of stats::dist().

upper Logical indicating whether or not to tidy the upper half of the distance matrix.

Defaults to whatever was based to the upper argument of stats::dist().

... Additional arguments. Not used. Needed to match generic signature only. Cautionary note: Misspelled arguments will be absorbed in ..., where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed

using conf.level = 0.95. Two exceptions here are:

 tidy() methods will warn when supplied an exponentiate argument if it will be ignored.

• augment() methods will warn when supplied a newdata argument if it will be ignored.

Details

If the distance matrix does not include an upper triangle and/or diagonal, the tidied version will not either.

Value

A tibble::tibble with one row for each pair of items in the distance matrix, with columns:

item1 First item item2 Second item

distance Distance between items

See Also

```
Other deprecated: bootstrap(), confint_tidy(), data.frame_tidiers, finish_glance(), fix_data_frame(), summary_tidiers, tidy.density(), tidy.ftable(), tidy.numeric()
```

tidy.drc 261

Examples

```
cars_dist <- dist(t(mtcars[, 1:4]))
cars_dist

tidy(cars_dist)
tidy(cars_dist, upper = TRUE)
tidy(cars_dist, diagonal = TRUE)</pre>
```

tidy.drc

Tidy a(n) drc object

Description

Tidy summarizes information about the components of a model. A model component might be a single term in a regression, a single hypothesis, a cluster, or a class. Exactly what tidy considers to be a model component varies across models but is usually self-evident. If a model has several distinct types of components, you will need to specify which components to return.

Usage

```
## S3 method for class 'drc'
tidy(x, conf.int = FALSE, conf.level = 0.95, ...)
```

Arguments

x A drc object produced by a call to drc::drm().

conf.int Logical indicating whether or not to include a confidence interval in the tidied

output. Defaults to FALSE.

conf. level The confidence level to use for the confidence interval if conf. int = TRUE. Must be strictly greater than 0 and less than 1. Defaults to 0.95, which corresponds to

a 95 percent confidence interval.

Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in . . . , where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

Details

The tibble has one row for each curve and term in the regression. The curveid column indicates the curve.

262 tidy.emmGrid

Value

A tibble::tibble() with columns:

Upper bound on the confidence interval for the estimate. conf.high conf.low Lower bound on the confidence interval for the estimate. The estimated value of the regression term. estimate p.value The two-sided p-value associated with the observed statistic. statistic The value of a T-statistic to use in a hypothesis that the regression term is nonzero. std.error The standard error of the regression term. term The name of the regression term.

curve Index identifying the curve.

See Also

```
tidy(), drc::drm()
Other drc tidiers: augment.drc(), glance.drc()
```

Examples

```
# load libraries for models and data
library(drc)
# fit model
mod <- drm(dead / total ~ conc, type,</pre>
  weights = total, data = selenium, fct = LL.2(), type = "binomial"
# summarize model fit with tidiers
tidy(mod)
tidy(mod, conf.int = TRUE)
glance(mod)
augment(mod, selenium)
```

tidy.emmGrid

Tidy a(n) emmGrid object

Description

Tidy summarizes information about the components of a model. A model component might be a single term in a regression, a single hypothesis, a cluster, or a class. Exactly what tidy considers to be a model component varies across models but is usually self-evident. If a model has several distinct types of components, you will need to specify which components to return.

tidy.emmGrid 263

Usage

```
## S3 method for class 'emmGrid'
tidy(x, conf.int = FALSE, conf.level = 0.95, ...)
```

Arguments

Х	An emmGrid object.
conf.int	Logical indicating whether or not to include a confidence interval in the tidied output. Defaults to FALSE.
conf.level	The confidence level to use for the confidence interval if conf.int = TRUE. Must be strictly greater than 0 and less than 1. Defaults to 0.95, which corresponds to a 95 percent confidence interval.
	Additional arguments passed to emmeans::summary.emmGrid() or lsmeans::summary.ref.grid(). Cautionary note: misspecified arguments may be silently ignored!

Details

Returns a data frame with one observation for each estimated marginal mean, and one column for each combination of factors. When the input is a contrast, each row will contain one estimated contrast.

There are a large number of arguments that can be passed on to emmeans::summary.emmGrid() or lsmeans::summary.ref.grid().

Value

A tibble::tibble() with columns:

conf.high	Upper bound on the confidence interval for the estimate.
conf.low	Lower bound on the confidence interval for the estimate.
df	Degrees of freedom used by this term in the model.
p.value	The two-sided p-value associated with the observed statistic.
std.error	The standard error of the regression term.
estimate	Expected marginal mean
statistic	T-ratio statistic

See Also

```
tidy(), emmeans::ref_grid(), emmeans::emmeans(), emmeans::contrast()
Other emmeans tidiers: tidy.lsmobj(), tidy.ref.grid(), tidy.summary_emm()
```

264 tidy.epi.2by2

```
# load libraries for models and data
library(emmeans)
# linear model for sales of oranges per day
oranges_lm1 <- lm(sales1 ~ price1 + price2 + day + store, data = oranges)</pre>
# reference grid; see vignette("basics", package = "emmeans")
oranges_rg1 <- ref_grid(oranges_lm1)</pre>
td <- tidy(oranges_rg1)</pre>
td
# marginal averages
marginal <- emmeans(oranges_rg1, "day")</pre>
tidy(marginal)
# contrasts
tidy(contrast(marginal))
tidy(contrast(marginal, method = "pairwise"))
# plot confidence intervals
library(ggplot2)
ggplot(tidy(marginal, conf.int = TRUE), aes(day, estimate)) +
  geom_point() +
  geom_errorbar(aes(ymin = conf.low, ymax = conf.high))
# by multiple prices
by_price <- emmeans(oranges_lm1, "day",</pre>
 by = "price2",
  at = list(
    price1 = 50, price2 = c(40, 60, 80),
    day = c("2", "3", "4")
  )
)
by_price
tidy(by_price)
ggplot(tidy(by_price, conf.int = TRUE), aes(price2, estimate, color = day)) +
  geom_line() +
  geom_errorbar(aes(ymin = conf.low, ymax = conf.high))
# joint_tests
tidy(joint_tests(oranges_lm1))
```

tidy.epi.2by2 265

Description

Tidy summarizes information about the components of a model. A model component might be a single term in a regression, a single hypothesis, a cluster, or a class. Exactly what tidy considers to be a model component varies across models but is usually self-evident. If a model has several distinct types of components, you will need to specify which components to return.

Usage

```
## S3 method for class 'epi.2by2'
tidy(x, parameters = c("moa", "stat"), ...)
```

Arguments

A epi.2by2 object produced by a call to epiR::epi.2by2()

Return measures of association (moa) or test statistics (stat), default is moa (measures of association)

Additional arguments. Not used. Needed to match generic signature only. Cautionary note: Misspelled arguments will be absorbed in . . ., where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

Details

The tibble has a column for each of the measures of association or tests contained in massoc or massoc.detail when epiR::epi.2by2() is called.

Value

```
A tibble::tibble() with columns:
```

conf.high Upper bound on the confidence interval for the estimate.

Lower bound on the confidence interval for the estimate.

Degrees of freedom used by this term in the model.

p.value The two-sided p-value associated with the observed statistic.

Statistic The value of a T-statistic to use in a hypothesis that the regression term is non-zero.

The name of the regression term.

Estimated measure of association

See Also

estimate

```
tidy(), epiR::epi.2by2()
```

266 tidy.ergm

Examples

```
# load libraries for models and data
library(epiR)

# generate data
dat <- matrix(c(13, 2163, 5, 3349), nrow = 2, byrow = TRUE)

rownames(dat) <- c("DF+", "DF-")
colnames(dat) <- c("FUS+", "FUS-")

# fit model
fit <- epi.2by2(
   dat = as.table(dat), method = "cross.sectional",
   conf.level = 0.95, units = 100, outcome = "as.columns"
)

# summarize model fit with tidiers
tidy(fit, parameters = "moa")
tidy(fit, parameters = "stat")</pre>
```

tidy.ergm

Tidy a(n) ergm object

Description

Tidy summarizes information about the components of a model. A model component might be a single term in a regression, a single hypothesis, a cluster, or a class. Exactly what tidy considers to be a model component varies across models but is usually self-evident. If a model has several distinct types of components, you will need to specify which components to return.

The methods should work with any model that conforms to the **ergm** class, such as those produced from weighted networks by the **ergm.count** package.

Usage

```
## S3 method for class 'ergm'
tidy(x, conf.int = FALSE, conf.level = 0.95, exponentiate = FALSE, ...)
```

Arguments

X	An ergm object returned from a call to ergm::ergm().
conf.int	Logical indicating whether or not to include a confidence interval in the tidied output. Defaults to FALSE.
conf.level	The confidence level to use for the confidence interval if conf.int = TRUE. Must be strictly greater than 0 and less than 1. Defaults to 0.95, which corresponds to a 95 percent confidence interval.

tidy.ergm 267

Logical indicating whether or not to exponentiate the the coefficient estimates. This is typical for logistic and multinomial regressions, but a bad idea if there is no log or logit link. Defaults to FALSE.
 Additional arguments to pass to ergm::summary(). Cautionary note: Misspecified arguments may be silently ignored.

Value

A tibble::tibble with one row for each coefficient in the exponential random graph model, with columns:

term The term in the model being estimated and tested

estimate The estimated coefficient
std.error The standard error
mcmc.error The MCMC error
p.value The two-sided p-value

References

Hunter DR, Handcock MS, Butts CT, Goodreau SM, Morris M (2008b). **ergm**: A Package to Fit, Simulate and Diagnose Exponential-Family Models for Networks. *Journal of Statistical Software*, 24(3). https://www.jstatsoft.org/v24/i03/.

See Also

```
tidy(), ergm::ergm(), ergm::control.ergm(), ergm::summary()
Other ergm tidiers: glance.ergm()
```

```
# load libraries for models and data
library(ergm)

# load the Florentine marriage network data
data(florentine)

# fit a model where the propensity to form ties between
# families depends on the absolute difference in wealth
gest <- ergm(flomarriage ~ edges + absdiff("wealth"))

# show terms, coefficient estimates and errors
tidy(gest)

# show coefficients as odds ratios with a 99% CI
tidy(gest, exponentiate = TRUE, conf.int = TRUE, conf.level = 0.99)

# take a look at likelihood measures and other
# control parameters used during MCMC estimation</pre>
```

268 tidy.factanal

```
glance(gest)
glance(gest, deviance = TRUE)
glance(gest, mcmc = TRUE)
```

tidy.factanal

Tidy a(n) factanal object

Description

Tidy summarizes information about the components of a model. A model component might be a single term in a regression, a single hypothesis, a cluster, or a class. Exactly what tidy considers to be a model component varies across models but is usually self-evident. If a model has several distinct types of components, you will need to specify which components to return.

Usage

```
## S3 method for class 'factanal' tidy(x, ...)
```

Arguments

x A factanal object created by stats::factanal().

Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in . . ., where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

Value

A tibble::tibble() with columns:

variable Variable under consideration.

uniqueness Proportion of residual, or unexplained variance

f1X Factor loading for level X.

See Also

```
tidy(), stats::factanal()
Other factanal tidiers: augment.factanal(), glance.factanal()
```

tidy.felm 269

Examples

```
set.seed(123)
# generate data
library(dplyr)
library(purrr)
m1 <- tibble(
  v1 = c(1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 3, 3, 3, 3, 3, 4, 5, 6),
  v2 = c(1, 2, 1, 1, 1, 1, 2, 1, 2, 1, 3, 4, 3, 3, 3, 4, 6, 5),
  v3 = c(3, 3, 3, 3, 3, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 5, 4, 6),
  v4 = c(3, 3, 4, 3, 3, 1, 1, 2, 1, 1, 1, 1, 2, 1, 1, 5, 6, 4),
  v5 = c(1, 1, 1, 1, 1, 3, 3, 3, 3, 1, 1, 1, 1, 1, 6, 4, 5),
  v6 = c(1, 1, 1, 2, 1, 3, 3, 3, 4, 3, 1, 1, 1, 2, 1, 6, 5, 4)
)
# new data
m2 <- map_dfr(m1, rev)</pre>
# factor analysis objects
fit1 <- factanal(m1, factors = 3, scores = "Bartlett")</pre>
fit2 <- factanal(m1, factors = 3, scores = "regression")</pre>
# tidying the object
tidy(fit1)
tidy(fit2)
# augmented dataframe
augment(fit1)
augment(fit2)
# augmented dataframe (with new data)
augment(fit1, data = m2)
augment(fit2, data = m2)
```

tidy.felm

Tidy a(n) felm object

Description

Tidy summarizes information about the components of a model. A model component might be a single term in a regression, a single hypothesis, a cluster, or a class. Exactly what tidy considers to be a model component varies across models but is usually self-evident. If a model has several distinct types of components, you will need to specify which components to return.

Usage

```
## S3 method for class 'felm'
```

270 tidy.felm

```
tidy(
  Х,
  conf.int = FALSE,
  conf.level = 0.95,
  fe = FALSE,
  se.type = c("default", "iid", "robust", "cluster"),
)
```

Arguments

A felm object returned from lfe::felm(). Х

conf.int Logical indicating whether or not to include a confidence interval in the tidied

output. Defaults to FALSE.

conf.level The confidence level to use for the confidence interval if conf.int = TRUE. Must

be strictly greater than 0 and less than 1. Defaults to 0.95, which corresponds to

a 95 percent confidence interval.

fe Logical indicating whether or not to include estimates of fixed effects. Defaults

to FALSE.

se.type Character indicating the type of standard errors. Defaults to using those of the

> underlying felm() model object, e.g. clustered errors for models that were provided a cluster specification. Users can override these defaults by specifying an appropriate alternative: "iid" (for homoskedastic errors), "robust" (for Eicker-Huber-White robust errors), or "cluster" (for clustered standard errors; if the

model object supports it).

Additional arguments. Not used. Needed to match generic signature only. Cautionary note: Misspelled arguments will be absorbed in . . . , where they will be ignored. If the misspelled argument has a default value, the default value will be

used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

• tidy() methods will warn when supplied an exponentiate argument if it

will be ignored.

• augment() methods will warn when supplied a newdata argument if it will be ignored.

Value

A tibble::tibble() with columns:

conf.high Upper bound on the confidence interval for the estimate. conf.low Lower bound on the confidence interval for the estimate.

estimate The estimated value of the regression term.

p.value The two-sided p-value associated with the observed statistic.

statistic The value of a T-statistic to use in a hypothesis that the regression term is non-

std.error The standard error of the regression term.

The name of the regression term. term

tidy.fitdistr 271

See Also

```
tidy(), lfe::felm()
Other felm tidiers: augment.felm()
```

```
# load libraries for models and data
library(lfe)
# use built-in `airquality` dataset
head(airquality)
# no FEs; same as lm()
est0 <- felm(Ozone ~ Temp + Wind + Solar.R, airquality)</pre>
# summarize model fit with tidiers
tidy(est0)
augment(est0)
# add month fixed effects
est1 <- felm(Ozone ~ Temp + Wind + Solar.R | Month, airquality)</pre>
# summarize model fit with tidiers
tidy(est1)
tidy(est1, fe = TRUE)
augment(est1)
glance(est1)
# the "se.type" argument can be used to switch out different standard errors
# types on the fly. In turn, this can be useful exploring the effect of
# different error structures on model inference.
tidy(est1, se.type = "iid")
tidy(est1, se.type = "robust")
# add clustered SEs (also by month)
est2 <- felm(Ozone ~ Temp + Wind + Solar.R | Month | 0 | Month, airquality)</pre>
# summarize model fit with tidiers
tidy(est2, conf.int = TRUE)
tidy(est2, conf.int = TRUE, se.type = "cluster")
tidy(est2, conf.int = TRUE, se.type = "robust")
tidy(est2, conf.int = TRUE, se.type = "iid")
```

272 tidy.fitdistr

Description

Tidy summarizes information about the components of a model. A model component might be a single term in a regression, a single hypothesis, a cluster, or a class. Exactly what tidy considers to be a model component varies across models but is usually self-evident. If a model has several distinct types of components, you will need to specify which components to return.

Usage

```
## S3 method for class 'fitdistr'
tidy(x, ...)
```

Arguments

x A fitdistr object returned by MASS::fitdistr().

Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in . . . , where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

Value

A tibble::tibble() with columns:

estimate The estimated value of the regression term.

std.error The standard error of the regression term.

term The name of the regression term.

See Also

```
tidy(), MASS::fitdistr()
Other fitdistr tidiers: glance.fitdistr()
```

```
# load libraries for models and data
library(MASS)

# generate data
set.seed(2015)
x <- rnorm(100, 5, 2)

# fit models</pre>
```

tidy.fixest 273

```
fit <- fitdistr(x, dnorm, list(mean = 3, sd = 1))
# summarize model fit with tidiers
tidy(fit)
glance(fit)</pre>
```

tidy.fixest

Tidy a(n) fixest object

Description

Tidy summarizes information about the components of a model. A model component might be a single term in a regression, a single hypothesis, a cluster, or a class. Exactly what tidy considers to be a model component varies across models but is usually self-evident. If a model has several distinct types of components, you will need to specify which components to return.

Usage

```
## S3 method for class 'fixest'
tidy(x, conf.int = FALSE, conf.level = 0.95, ...)
```

Arguments

X	A fixest object returned from any of the fixest estimators
conf.int	Logical indicating whether or not to include a confidence interval in the tidied output. Defaults to FALSE.
conf.level	The confidence level to use for the confidence interval if conf.int = TRUE. Must be strictly greater than 0 and less than 1. Defaults to 0.95, which corresponds to a 95 percent confidence interval.
	Additional arguments passed to summary and confint. Important arguments are se and cluster. Other arguments are dof, exact_dof, forceCovariance, and keepBounded. See summary.fixest.

Details

The fixest package provides a family of functions for estimating models with arbitrary numbers of fixed-effects, in both an OLS and a GLM context. The package also supports robust (i.e. White) and clustered standard error reporting via the generic summary.fixest() command. In a similar vein, the tidy() method for these models allows users to specify a desired standard error correction either 1) implicitly via the supplied fixest object, or 2) explicitly as part of the tidy call. See examples below.

Note that fixest confidence intervals are calculated assuming a normal distribution – this assumes infinite degrees of freedom for the CI. (This assumption is distinct from the degrees of freedom used to calculate the standard errors. For more on degrees of freedom with clusters and fixed effects, see https://github.com/lrberge/fixest/issues/6 and https://github.com/sgaure/lfe/issues/1#issuecomment-530646990)

274 tidy.fixest

Value

```
A tibble::tibble() with columns:
```

```
conf.high Upper bound on the confidence interval for the estimate.

conf.low Lower bound on the confidence interval for the estimate.

estimate The estimated value of the regression term.

p.value The two-sided p-value associated with the observed statistic.

The value of a T-statistic to use in a hypothesis that the regression term is non-zero.

std.error The standard error of the regression term.

The name of the regression term.
```

See Also

```
tidy(), fixest::feglm(), fixest::fenegbin(), fixest::feNmlm(), fixest::femlm(), fixest
```

```
# load libraries for models and data
library(fixest)
gravity <-
 feols(
   log(Euros) ~ log(dist_km) | Origin + Destination + Product + Year, trade
tidy(gravity)
glance(gravity)
augment(gravity, trade)
# to get robust or clustered SEs, users can either:
# 1) specify the arguments directly in the `tidy()` call
tidy(gravity, conf.int = TRUE, cluster = c("Product", "Year"))
tidy(gravity, conf.int = TRUE, se = "threeway")
# 2) or, feed tidy() a summary.fixest object that has already accepted
# these arguments
gravity_summ <- summary(gravity, cluster = c("Product", "Year"))</pre>
tidy(gravity_summ, conf.int = TRUE)
# approach (1) is preferred.
```

tidy.ftable 275

tidy.ftable

(Deprecated) Tidy ftable objects

Description

This function is deprecated. Please use tibble::as_tibble() instead.

Usage

```
## S3 method for class 'ftable' tidy(x, ...)
```

Arguments

Х

An ftable object returned from stats::ftable().

. . .

Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in . . . , where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

Value

An ftable contains a "flat" contingency table. This melts it into a tibble::tibble with one column for each variable, then a Freq column.

See Also

```
Other deprecated: bootstrap(), confint_tidy(), data.frame_tidiers, finish_glance(), fix_data_frame(), summary_tidiers, tidy.density(), tidy.dist(), tidy.numeric()
```

tidy.Gam

Tidy a(n) Gam object

Description

Tidy summarizes information about the components of a model. A model component might be a single term in a regression, a single hypothesis, a cluster, or a class. Exactly what tidy considers to be a model component varies across models but is usually self-evident. If a model has several distinct types of components, you will need to specify which components to return.

276 tidy.Gam

Usage

```
## S3 method for class 'Gam' tidy(x, ...)
```

Arguments

x A Gam object returned from a call to gam::gam().

Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in . . ., where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

Details

Tidy gam objects created by calls to mgcv::gam() with tidy.gam().

Value

```
A tibble::tibble() with columns:
```

df Degrees of freedom used by this term in the model.

meansq Mean sum of squares. Equal to total sum of squares divided by degrees of

freedom.

p.value The two-sided p-value associated with the observed statistic.

statistic The value of a T-statistic to use in a hypothesis that the regression term is non-

zero.

sumsq Sum of squares explained by this term.

term The name of the regression term.

See Also

```
tidy(), gam::gam(), tidy.anova(), tidy.gam()
Other gam tidiers: glance.Gam()
```

```
# load libraries for models and data
library(gam)

# fit model
g <- gam(mpg ~ s(hp, 4) + am + qsec, data = mtcars)</pre>
```

tidy.gam 277

```
# summarize model fit with tidiers
tidy(g)
glance(g)
```

tidy.gam

Tidy a(n) gam object

Description

Tidy summarizes information about the components of a model. A model component might be a single term in a regression, a single hypothesis, a cluster, or a class. Exactly what tidy considers to be a model component varies across models but is usually self-evident. If a model has several distinct types of components, you will need to specify which components to return.

Usage

```
## S3 method for class 'gam'
tidy(
    x,
    parametric = FALSE,
    conf.int = FALSE,
    conf.level = 0.95,
    exponentiate = FALSE,
    ...
)
```

Arguments

X	A gam object returned from a call to mgcv::gam().
parametric	Logical indicating if parametric or smooth terms should be tidied. Defaults to FALSE, meaning that smooth terms are tidied by default.
conf.int	Logical indicating whether or not to include a confidence interval in the tidied output. Defaults to FALSE.
conf.level	The confidence level to use for the confidence interval if conf.int = TRUE. Must be strictly greater than 0 and less than 1. Defaults to 0.95, which corresponds to a 95 percent confidence interval.
exponentiate	Logical indicating whether or not to exponentiate the the coefficient estimates. This is typical for logistic and multinomial regressions, but a bad idea if there is no log or logit link. Defaults to FALSE.
	Additional arguments. Not used. Needed to match generic signature only. Cautionary note: Misspelled arguments will be absorbed in , where they will be ignored. If the misspelled argument has a default value, the default value will be

using conf.level = 0.95. Two exceptions here are:

used. For example, if you pass conf.lvel = 0.9, all computation will proceed

278 tidy.gam

• tidy() methods will warn when supplied an exponentiate argument if it will be ignored.

• augment() methods will warn when supplied a newdata argument if it will be ignored.

Details

When parametric = FALSE return columns edf and ref.df rather than estimate and std.error.

Value

A tibble::tibble() with columns:

ref.df

The estimated value of the regression term.

The two-sided p-value associated with the observed statistic.

The value of a T-statistic to use in a hypothesis that the regression term is non-zero.

The standard error of the regression term.

The name of the regression term.

The effective degrees of freedom. Only reported when 'parametric = FALSE'

The reference degrees of freedom. Only reported when 'parametric = FALSE'

See Also

```
tidy(), mgcv::gam()
Other mgcv tidiers: glance.gam()
```

```
# load libraries for models and data
library(mgcv)

# fit model
g <- gam(mpg ~ s(hp) + am + qsec, data = mtcars)

# summarize model fit with tidiers
tidy(g)
tidy(g, parametric = TRUE)
glance(g)
augment(g)</pre>
```

tidy.garch 279

tidy.garch	Tidy a(n) garch object

Description

Tidy summarizes information about the components of a model. A model component might be a single term in a regression, a single hypothesis, a cluster, or a class. Exactly what tidy considers to be a model component varies across models but is usually self-evident. If a model has several distinct types of components, you will need to specify which components to return.

Usage

```
## S3 method for class 'garch'
tidy(x, conf.int = FALSE, conf.level = 0.95, ...)
```

Arguments

X	A garch object returned by tseries::garch().
conf.int	Logical indicating whether or not to include a confidence interval in the tidied output. Defaults to FALSE.
conf.level	The confidence level to use for the confidence interval if conf.int = TRUE. Must be strictly greater than 0 and less than 1. Defaults to 0.95, which corresponds to a 95 percent confidence interval.
	Additional arguments. Not used. Needed to match generic signature only. Cautionary note: Misspelled arguments will be absorbed in , where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

Value

A tibble::tibble() with columns:

conf.high	Upper bound on the confidence interval for the estimate.
conf.low	Lower bound on the confidence interval for the estimate.
estimate	The estimated value of the regression term.
p.value	The two-sided p-value associated with the observed statistic.
statistic	The value of a T-statistic to use in a hypothesis that the regression term is non-
	zero.
std.error	The standard error of the regression term.
term	The name of the regression term.

280 tidy.geeglm

See Also

```
tidy(), tseries::garch()
Other garch tidiers: glance.garch()
```

Examples

```
# load libraries for models and data
library(tseries)

# load data
data(EuStockMarkets)

# fit model
dax <- diff(log(EuStockMarkets))[, "DAX"]
dax.garch <- garch(dax)
dax.garch

# summarize model fit with tidiers
tidy(dax.garch)
glance(dax.garch)</pre>
```

tidy.geeglm

Tidy a(n) geeglm object

Description

Tidy summarizes information about the components of a model. A model component might be a single term in a regression, a single hypothesis, a cluster, or a class. Exactly what tidy considers to be a model component varies across models but is usually self-evident. If a model has several distinct types of components, you will need to specify which components to return.

Usage

```
## S3 method for class 'geeglm'
tidy(x, conf.int = FALSE, conf.level = 0.95, exponentiate = FALSE, ...)
```

Arguments

X	A geeglm object returned from a call to geepack::geeglm().
conf.int	Logical indicating whether or not to include a confidence interval in the tidied output. Defaults to FALSE.
conf.level	The confidence level to use for the confidence interval if conf.int = TRUE. Must be strictly greater than 0 and less than 1. Defaults to 0.95, which corresponds to a 95 percent confidence interval.

tidy.geeglm 281

exponentiate

Logical indicating whether or not to exponentiate the the coefficient estimates. This is typical for logistic and multinomial regressions, but a bad idea if there is no log or logit link. Defaults to FALSE.

Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in . . . , where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

Details

If conf.int = TRUE, the confidence interval is computed with the an internal confint.geeglm() function.

If you have missing values in your model data, you may need to refit the model with na.action = na.exclude or deal with the missingness in the data beforehand.

Value

A tibble::tibble() with columns:

conf.high	Upper bound on the confidence interval for the estimate.
conf.low	Lower bound on the confidence interval for the estimate.
estimate	The estimated value of the regression term.
p.value	The two-sided p-value associated with the observed statistic.
statistic	The value of a T-statistic to use in a hypothesis that the regression term is non-zero.
std.error	The standard error of the regression term.
term	The name of the regression term.

See Also

```
tidy(), geepack::geeglm()
```

```
# load modeling library
library(geepack)
# load data
data(state)
```

282 tidy.glht

```
ds <- data.frame(state.region, state.x77)
# fit model
geefit <- geeglm(Income ~ Frost + Murder,
   id = state.region,
   data = ds,
   corstr = "exchangeable"
)
# summarize model fit with tidiers
tidy(geefit)
tidy(geefit, conf.int = TRUE)</pre>
```

tidy.glht

Tidy a(n) glht object

Description

Tidy summarizes information about the components of a model. A model component might be a single term in a regression, a single hypothesis, a cluster, or a class. Exactly what tidy considers to be a model component varies across models but is usually self-evident. If a model has several distinct types of components, you will need to specify which components to return.

Usage

```
## S3 method for class 'glht'
tidy(x, conf.int = FALSE, conf.level = 0.95, ...)
```

Arguments

X	A glht object returned by multcomp::glht().
conf.int	Logical indicating whether or not to include a confidence interval in the tidied output. Defaults to FALSE.
conf.level	The confidence level to use for the confidence interval if $conf.int = TRUE$. Must be strictly greater than 0 and less than 1. Defaults to 0.95, which corresponds to a 95 percent confidence interval.
	Additional arguments passed to summary() and tidy.summary.glht().

Value

A tibble::tibble() with columns:

contrast Levels being compared.

estimate The estimated value of the regression term.

null.value Value to which the estimate is compared.

tidy.glm 283

See Also

```
tidy(), multcomp::glht()
Other multcomp tidiers: tidy.cld(), tidy.confint.glht(), tidy.summary.glht()
```

Examples

```
# load libraries for models and data
library(multcomp)
library(ggplot2)
amod <- aov(breaks ~ wool + tension, data = warpbreaks)</pre>
wht <- glht(amod, linfct = mcp(tension = "Tukey"))</pre>
tidy(wht)
ggplot(wht, aes(lhs, estimate)) +
 geom_point()
CI <- confint(wht)
tidy(CI)
ggplot(CI, aes(lhs, estimate, ymin = lwr, ymax = upr)) +
 geom_pointrange()
tidy(summary(wht))
ggplot(mapping = aes(lhs, estimate)) +
 geom_linerange(aes(ymin = lwr, ymax = upr), data = CI) +
 geom_point(aes(size = p), data = summary(wht)) +
 scale_size(trans = "reverse")
cld <- cld(wht)</pre>
tidy(cld)
```

tidy.glm

Tidy a(n) glm object

Description

Tidy summarizes information about the components of a model. A model component might be a single term in a regression, a single hypothesis, a cluster, or a class. Exactly what tidy considers to be a model component varies across models but is usually self-evident. If a model has several distinct types of components, you will need to specify which components to return.

Usage

```
## S3 method for class 'glm'
tidy(x, conf.int = FALSE, conf.level = 0.95, exponentiate = FALSE, ...)
```

284 tidy.glmnet

Arguments

X	A glm object returned from stats::glm().
conf.int	Logical indicating whether or not to include a confidence interval in the tidied output. Defaults to FALSE.
conf.level	The confidence level to use for the confidence interval if conf.int = TRUE. Must be strictly greater than 0 and less than 1. Defaults to 0.95, which corresponds to a 95 percent confidence interval.
exponentiate	Logical indicating whether or not to exponentiate the the coefficient estimates. This is typical for logistic and multinomial regressions, but a bad idea if there is no log or logit link. Defaults to FALSE.
	Additional arguments. Not used. Needed to match generic signature only. Cautionary note: Misspelled arguments will be absorbed in , where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

See Also

```
stats::glm()
Other Im tidiers: augment.glm(), augment.lm(), glance.glm(), glance.lm(), glance.summary.lm(),
glance.svyglm(), tidy.lm(), tidy.lm.beta(), tidy.mlm(), tidy.summary.lm()
```

tidy.glmnet

Tidy a(n) glmnet object

Description

Tidy summarizes information about the components of a model. A model component might be a single term in a regression, a single hypothesis, a cluster, or a class. Exactly what tidy considers to be a model component varies across models but is usually self-evident. If a model has several distinct types of components, you will need to specify which components to return.

Usage

```
## S3 method for class 'glmnet'
tidy(x, return_zeros = FALSE, ...)
```

tidy.glmnet 285

Arguments

x A glmnet object returned from glmnet::glmnet().

in the results. Defaults to FALSE.

Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in . . ., where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

Details

Note that while this representation of GLMs is much easier to plot and combine than the default structure, it is also much more memory-intensive. Do not use for large, sparse matrices.

No augment method is yet provided even though the model produces predictions, because the input data is not tidy (it is a matrix that may be very wide) and therefore combining predictions with it is not logical. Furthermore, predictions make sense only with a specific choice of lambda.

Value

```
A tibble::tibble() with columns:
```

dev.ratio Fraction of null deviance explained at each value of lambda.

estimate The estimated value of the regression term.

lambda Value of penalty parameter lambda. step Which step of lambda choices was used.

term The name of the regression term.

See Also

```
tidy(), glmnet::glmnet()
Other glmnet tidiers: glance.cv.glmnet(), glance.glmnet(), tidy.cv.glmnet()
```

```
# load libraries for models and data
library(glmnet)

set.seed(2014)
x <- matrix(rnorm(100 * 20), 100, 20)
y <- rnorm(100)
fit1 <- glmnet(x, y)</pre>
```

286 tidy.glmRob

```
# summarize model fit with tidiers + visualization
tidy(fit1)
glance(fit1)
library(dplyr)
library(ggplot2)
tidied <- tidy(fit1) %>% filter(term != "(Intercept)")
ggplot(tidied, aes(step, estimate, group = term)) +
 geom_line()
ggplot(tidied, aes(lambda, estimate, group = term)) +
 geom_line() +
 scale_x_log10()
ggplot(tidied, aes(lambda, dev.ratio)) +
 geom_line()
# works for other types of regressions as well, such as logistic
g2 <- sample(1:2, 100, replace = TRUE)</pre>
fit2 <- glmnet(x, g2, family = "binomial")</pre>
tidy(fit2)
```

tidy.glmRob

Tidy a(n) glmRob object

Description

Tidy summarizes information about the components of a model. A model component might be a single term in a regression, a single hypothesis, a cluster, or a class. Exactly what tidy considers to be a model component varies across models but is usually self-evident. If a model has several distinct types of components, you will need to specify which components to return.

Usage

```
## S3 method for class 'glmRob'
tidy(x, ...)
```

Arguments

. . .

x A glmRob object returned from robust::glmRob().

Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in . . . , where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

tidy.glmrob 287

• tidy() methods will warn when supplied an exponentiate argument if it will be ignored.

• augment() methods will warn when supplied a newdata argument if it will be ignored.

Details

For tidiers for robust models from the MASS package see tidy.rlm().

See Also

```
robust::glmRob()
Other robust tidiers: augment.lmRob(), glance.glmRob(), glance.lmRob(), tidy.lmRob()
```

Examples

```
# load libraries for models and data
library(robust)

# fit model
gm <- glmRob(am ~ wt, data = mtcars, family = "binomial")

# summarize model fit with tidiers
tidy(gm)
glance(gm)</pre>
```

tidy.glmrob

Tidy a(n) glmrob object

Description

Tidy summarizes information about the components of a model. A model component might be a single term in a regression, a single hypothesis, a cluster, or a class. Exactly what tidy considers to be a model component varies across models but is usually self-evident. If a model has several distinct types of components, you will need to specify which components to return.

Usage

```
## S3 method for class 'glmrob'
tidy(x, conf.int = FALSE, conf.level = 0.95, ...)
```

288 tidy.glmrob

Arguments

A glmrob object returned from robustbase::glmrob().

Logical indicating whether or not to include a confidence interval in the tidied output. Defaults to FALSE.

The confidence level to use for the confidence interval if conf.int = TRUE. Must be strictly greater than 0 and less than 1. Defaults to 0.95, which corresponds to a 95 percent confidence interval.

Additional arguments. Not used. Needed to match generic signature only. Cautionary note: Misspelled arguments will be absorbed in . . ., where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

Details

For tidiers for robust models from the MASS package see tidy.rlm().

Value

```
A tibble::tibble() with columns:
```

conf.high	Upper bound on the confidence interval for the estimate.
conf.low	Lower bound on the confidence interval for the estimate.
estimate	The estimated value of the regression term.
p.value	The two-sided p-value associated with the observed statistic.
statistic	The value of a T-statistic to use in a hypothesis that the regression term is non-zero.
std.error	The standard error of the regression term.
term	The name of the regression term.

See Also

```
robustbase::glmrob()
Other robustbase tidiers: augment.glmrob(), augment.lmrob(), glance.lmrob(), tidy.lmrob()
```

```
if (requireNamespace("robustbase", quietly = TRUE)) {
  # load libraries for models and data
  library(robustbase)

data(coleman)
```

tidy.gmm 289

```
set.seed(0)

m <- lmrob(Y ~ ., data = coleman)
tidy(m)
augment(m)
glance(m)

data(carrots)

Rfit <- glmrob(cbind(success, total - success) ~ logdose + block,
  family = binomial, data = carrots, method = "Mqle",
  control = glmrobMqle.control(tcc = 1.2)
)

tidy(Rfit)
augment(Rfit)
}</pre>
```

tidy.gmm

Tidy a(n) gmm object

Description

Tidy summarizes information about the components of a model. A model component might be a single term in a regression, a single hypothesis, a cluster, or a class. Exactly what tidy considers to be a model component varies across models but is usually self-evident. If a model has several distinct types of components, you will need to specify which components to return.

Usage

```
## S3 method for class 'gmm'
tidy(x, conf.int = FALSE, conf.level = 0.95, exponentiate = FALSE, ...)
```

Arguments

x	A gmm object returned from gmm::gmm().
conf.int	Logical indicating whether or not to include a confidence interval in the tidied output. Defaults to FALSE.
conf.level	The confidence level to use for the confidence interval if conf.int = TRUE. Must be strictly greater than 0 and less than 1. Defaults to 0.95, which corresponds to a 95 percent confidence interval.
exponentiate	Logical indicating whether or not to exponentiate the the coefficient estimates. This is typical for logistic and multinomial regressions, but a bad idea if there is no log or logit link. Defaults to FALSE.
	Additional arguments. Not used. Needed to match generic signature only. Cautionary note: Misspelled arguments will be absorbed in , where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

290 tidy.gmm

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

Value

A tibble::tibble() with columns:

```
conf.high Upper bound on the confidence interval for the estimate.

Lower bound on the confidence interval for the estimate.

The estimated value of the regression term.

P.value The two-sided p-value associated with the observed statistic.

The value of a T-statistic to use in a hypothesis that the regression term is non-zero.

Std.error The standard error of the regression term.

The name of the regression term.
```

See Also

```
tidy(), gmm::gmm()
Other gmm tidiers: glance.gmm()
```

```
# load libraries for models and data
library(gmm)
# examples come from the "gmm" package
# CAPM test with GMM
data(Finance)
r <- Finance[1:300, 1:10]
rm <- Finance[1:300, "rm"]
rf <- Finance[1:300, "rf"]
z <- as.matrix(r - rf)</pre>
t <- nrow(z)
zm <- rm - rf
h <- matrix(zm, t, 1)</pre>
res <- gmm(z \sim zm, x = h)
# tidy result
tidy(res)
tidy(res, conf.int = TRUE)
tidy(res, conf.int = TRUE, conf.level = .99)
# coefficient plot
library(ggplot2)
```

tidy.htest 291

```
library(dplyr)
tidy(res, conf.int = TRUE) %>%
  mutate(variable = reorder(term, estimate)) %>%
  ggplot(aes(estimate, variable)) +
  geom_point() +
  geom_errorbarh(aes(xmin = conf.low, xmax = conf.high)) +
  geom_vline(xintercept = 0, color = "red", lty = 2)
# from a function instead of a matrix
g <- function(theta, x) {</pre>
  e \leftarrow x[, 2:11] - theta[1] - (x[, 1] - theta[1]) %*% matrix(theta[2:11], 1, 10)
  gmat \leftarrow cbind(e, e * c(x[, 1]))
  return(gmat)
}
x <- as.matrix(cbind(rm, r))</pre>
res_black <- gmm(g, x = x, t0 = rep(0, 11))
tidy(res_black)
tidy(res_black, conf.int = TRUE)
# APT test with Fama-French factors and GMM
f1 <- zm
f2 <- Finance[1:300, "hml"] - rf</pre>
f3 <- Finance[1:300, "smb"] - rf
h <- cbind(f1, f2, f3)
res2 <- gmm(z \sim f1 + f2 + f3, x = h)
td2 <- tidy(res2, conf.int = TRUE)
td2
# coefficient plot
td2 %>%
  mutate(variable = reorder(term, estimate)) %>%
  ggplot(aes(estimate, variable)) +
  geom_point() +
  geom_errorbarh(aes(xmin = conf.low, xmax = conf.high)) +
  geom_vline(xintercept = 0, color = "red", lty = 2)
```

tidy.htest

Tidy/glance a(n) htest object

Description

For models that have only a single component, the tidy() and glance() methods are identical. Please see the documentation for both of those methods.

292 tidy.htest

Usage

```
## S3 method for class 'htest'
tidy(x, ...)
## S3 method for class 'htest'
glance(x, ...)
```

Arguments

Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in . . ., where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

Value

A tibble::tibble() with columns:

alternative	Alternative hypothesis (character).
conf.high	Upper bound on the confidence interval for the estimate.
conf.low	Lower bound on the confidence interval for the estimate.
estimate	The estimated value of the regression term.
estimate1	Sometimes two estimates are computed, such as in a two-sample t-test.
estimate2	Sometimes two estimates are computed, such as in a two-sample t-test.
method	Method used.
p.value	The two-sided p-value associated with the observed statistic.
parameter	The parameter being modeled.
statistic	The value of a T-statistic to use in a hypothesis that the regression term is non-zero.

See Also

```
tidy(), stats::cor.test(), stats::t.test(), stats::wilcox.test(), stats::chisq.test()
Other htest tidiers: augment.htest(), tidy.pairwise.htest(), tidy.power.htest()
```

tidy.ivreg 293

Examples

```
tt <- t.test(rnorm(10))
tidy(tt)

# the glance output will be the same for each of the below tests
glance(tt)

tt <- t.test(mpg ~ am, data = mtcars)
tidy(tt)

wt <- wilcox.test(mpg ~ am, data = mtcars, conf.int = TRUE, exact = FALSE)
tidy(wt)

ct <- cor.test(mtcars$wt, mtcars$mpg)
tidy(ct)
chit <- chisq.test(xtabs(Freq ~ Sex + Class, data = as.data.frame(Titanic)))
tidy(chit)
augment(chit)</pre>
```

tidy.ivreg

Tidy a(n) ivreg object

Description

Tidy summarizes information about the components of a model. A model component might be a single term in a regression, a single hypothesis, a cluster, or a class. Exactly what tidy considers to be a model component varies across models but is usually self-evident. If a model has several distinct types of components, you will need to specify which components to return.

Usage

```
## S3 method for class 'ivreg'
tidy(x, conf.int = FALSE, conf.level = 0.95, instruments = FALSE, ...)
```

Arguments

x	An ivreg object created by a call to AER::ivreg().
conf.int	Logical indicating whether or not to include a confidence interval in the tidied output. Defaults to FALSE.
conf.level	The confidence level to use for the confidence interval if conf.int = TRUE. Must be strictly greater than 0 and less than 1. Defaults to 0.95, which corresponds to a 95 percent confidence interval.

294 tidy.ivreg

instruments Logical indicating whether to return coefficients from the second-stage or diagnostics tests for each endogenous regressor (F-statistics). Defaults to FALSE.

Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in . . . , where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

Details

This tidier currently only supports ivreg-classed objects outputted by the AER package. The ivreg package also outputs objects of class ivreg, and will be supported in a later release.

Value

```
A tibble::tibble() with columns:
conf.high
                  Upper bound on the confidence interval for the estimate.
                  Lower bound on the confidence interval for the estimate.
conf.low
estimate
                  The estimated value of the regression term.
p.value
                  The two-sided p-value associated with the observed statistic.
p.value.Sargan p-value for Sargan test of overidentifying restrictions.
p.value.weakinst
                  p-value for weak instruments test.
p.value.Wu.Hausman
                  p-value for Wu-Hausman weak instruments test for endogeneity.
                  The value of a T-statistic to use in a hypothesis that the regression term is non-
statistic
statistic.Sargan
                  Statistic for Sargan test of overidentifying restrictions.
statistic.weakinst
                  Statistic for Wu-Hausman test.
statistic.Wu.Hausman
                  Statistic for Wu-Hausman weak instruments test for endogeneity.
                  The standard error of the regression term.
std.error
                  The name of the regression term.
term
```

See Also

```
tidy(), AER::ivreg()
Other ivreg tidiers: augment.ivreg(), glance.ivreg()
```

tidy.kappa 295

Examples

```
# load libraries for models and data
library(AER)
# load data
data("CigarettesSW", package = "AER")
# fit model
ivr <- ivreg(</pre>
 log(packs) ~ income | population,
 data = CigarettesSW,
 subset = year == "1995"
)
# summarize model fit with tidiers
tidy(ivr, conf.int = TRUE)
tidy(ivr, conf.int = TRUE, instruments = TRUE)
augment(ivr)
augment(ivr, data = CigarettesSW)
augment(ivr, newdata = CigarettesSW)
glance(ivr)
```

tidy.kappa

Tidy a(n) kappa object

Description

Tidy summarizes information about the components of a model. A model component might be a single term in a regression, a single hypothesis, a cluster, or a class. Exactly what tidy considers to be a model component varies across models but is usually self-evident. If a model has several distinct types of components, you will need to specify which components to return.

Usage

```
## S3 method for class 'kappa' tidy(x, ...)
```

Arguments

x A kappa object returned from psych::cohen.kappa().

... Additional arguments. Not used. Needed to match generic signature only. Cautionary note: Misspelled arguments will be absorbed in ..., where they will be ignored. If the misspelled argument has a default value, the default value will be

296 tidy.kappa

> used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

Details

Note that confidence level (alpha) for the confidence interval cannot be set in tidy. Instead you must set the alpha argument to psych::cohen.kappa() when creating the kappa object.

Value

```
A tibble::tibble() with columns:
```

conf.high Upper bound on the confidence interval for the estimate. conf.low Lower bound on the confidence interval for the estimate. The estimated value of the regression term. estimate Either 'weighted' or 'unweighted'.

See Also

type

```
tidy(), psych::cohen.kappa()
```

```
# load libraries for models and data
library(psych)
# generate example data
rater1 <- 1:9
rater2 <- c(1, 3, 1, 6, 1, 5, 5, 6, 7)
# fit model
ck <- cohen.kappa(cbind(rater1, rater2))</pre>
# summarize model fit with tidiers + visualization
tidy(ck)
# graph the confidence intervals
library(ggplot2)
ggplot(tidy(ck), aes(estimate, type)) +
  geom_point() +
  geom_errorbarh(aes(xmin = conf.low, xmax = conf.high))
```

tidy.kde 297

tidy.kde	Tidy a(n) kde object
----------	----------------------

Description

Tidy summarizes information about the components of a model. A model component might be a single term in a regression, a single hypothesis, a cluster, or a class. Exactly what tidy considers to be a model component varies across models but is usually self-evident. If a model has several distinct types of components, you will need to specify which components to return.

Usage

```
## S3 method for class 'kde' tidy(x, ...)
```

Arguments

x A kde object returned from ks::kde().

Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in . . . , where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

Details

Returns a data frame in long format with four columns. Use tidyr::pivot_wider(..., names_from = variable, values_from = value) on the output to return to a wide format.

Value

```
A tibble::tibble() with columns:
```

estimate The estimated value of the regression term.

obs weighted observed number of events in each group.

value The value/estimate of the component. Results from data reshaping.

variable Variable under consideration.

See Also

```
tidy(), ks::kde()
```

298 tidy.Kendall

Examples

```
# load libraries for models and data
library(ks)
# generate data
dat <- replicate(2, rnorm(100))</pre>
k <- kde(dat)
# summarize model fit with tidiers + visualization
td <- tidy(k)
td
library(ggplot2)
library(dplyr)
library(tidyr)
td %>%
  pivot_wider(c(obs, estimate),
    names_from = variable,
    values_from = value
  ggplot(aes(x1, x2, fill = estimate)) +
  geom_tile() +
  theme_void()
# also works with 3 dimensions
dat3 <- replicate(3, rnorm(100))</pre>
k3 <- kde(dat3)
td3 \leftarrow tidy(k3)
td3
```

tidy.Kendall

Tidy a(n) Kendall object

Description

Tidy summarizes information about the components of a model. A model component might be a single term in a regression, a single hypothesis, a cluster, or a class. Exactly what tidy considers to be a model component varies across models but is usually self-evident. If a model has several distinct types of components, you will need to specify which components to return.

Usage

```
## S3 method for class 'Kendall' tidy(x, ...)
```

tidy.Kendall 299

Arguments

. . .

x A Kendall object returned from a call to Kendall::Kendall(), Kendall::MannKendall(), or Kendall::SeasonalMannKendall().

Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in . . ., where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

Value

```
A tibble::tibble() with columns:
```

kendall_score Kendall score.

p.value The two-sided p-value associated with the observed statistic.

var_kendall_score

Variance of the kendall_score.

statistic Kendall's tau statistic

denominator The denominator, which is tau=kendall_score/denominator.

See Also

```
tidy(), Kendall::Kendall(), Kendall::MannKendall(), Kendall::SeasonalMannKendall()
```

```
# load libraries for models and data
library(Kendall)

A <- c(2.5, 2.5, 2.5, 2.5, 5, 6.5, 6.5, 10, 10, 10, 10, 10, 14, 14, 14, 16, 17)
B <- c(1, 1, 1, 1, 2, 1, 1, 2, 1, 1, 1, 1, 1, 1, 2, 2, 2)

# fit models and summarize results
f_res <- Kendall(A, B)
tidy(f_res)

s_res <- MannKendall(B)
tidy(s_res)

t_res <- SeasonalMannKendall(ts(A))
tidy(t_res)</pre>
```

300 tidy.kmeans

tidy.kmeans

Tidy a(n) kmeans object

Description

Tidy summarizes information about the components of a model. A model component might be a single term in a regression, a single hypothesis, a cluster, or a class. Exactly what tidy considers to be a model component varies across models but is usually self-evident. If a model has several distinct types of components, you will need to specify which components to return.

Usage

```
## S3 method for class 'kmeans'
tidy(x, col.names = colnames(x$centers), ...)
```

Arguments

x A kmeans object created by stats::kmeans().

col.names Dimension names. Defaults to the names of the variables in x. Set to NULL to

get names x1, x2,

Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in . . . , where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

Value

A tibble::tibble() with columns:

cluster A factor describing the cluster from 1:k.
size Number of points assigned to cluster.
withinss The within-cluster sum of squares.

See Also

```
tidy(), stats::kmeans()
Other kmeans tidiers: augment.kmeans(), glance.kmeans()
```

tidy.lavaan 301

Examples

```
library(cluster)
library(modeldata)
library(dplyr)

data(hpc_data)

x <- hpc_data[, 2:5]

fit <- pam(x, k = 4)

tidy(fit)
glance(fit)
augment(fit, x)</pre>
```

tidy.lavaan

Tidy a(n) lavaan object

Description

Tidy summarizes information about the components of a model. A model component might be a single term in a regression, a single hypothesis, a cluster, or a class. Exactly what tidy considers to be a model component varies across models but is usually self-evident. If a model has several distinct types of components, you will need to specify which components to return.

Usage

```
## S3 method for class 'lavaan'
tidy(x, conf.int = FALSE, conf.level = 0.95, ...)
```

Arguments

x	A lavaan object, such as those returned from lavaan::cfa(), and lavaan::sem().
conf.int	Logical indicating whether or not to include a confidence interval in the tidied output. Defaults to FALSE.
conf.level	The confidence level to use for the confidence interval if conf.int = TRUE. Must be strictly greater than 0 and less than 1. Defaults to 0.95, which corresponds to a 95 percent confidence interval.
• • •	Additional arguments passed to lavaan::parameterEstimates(). Cautionary note: Misspecified arguments may be silently ignored.

302 tidy.lavaan

Value

A tibble::tibble() with one row for each estimated parameter and columns:

term	The result of paste(lhs, op, rhs)
ор	The operator in the model syntax (e.g. \sim for covariances, or \sim for regression parameters)
group	The group (if specified) in the lavaan model
estimate	The parameter estimate (may be standardized)
std.error	
statistic	The z value returned by lavaan::parameterEstimates()
p.value	
conf.low	
conf.high	
std.lv	Standardized estimates based on the variances of the (continuous) latent variables only
std.all	Standardized estimates based on both the variances of both (continuous) observed and latent variables.
std.nox	Standardized estimates based on both the variances of both (continuous) observed and latent variables, but not the variances of exogenous covariates.

See Also

```
tidy(), lavaan::cfa(), lavaan::sem(), lavaan::parameterEstimates()
Other lavaan tidiers: glance.lavaan()
```

```
# load libraries for models and data
library(lavaan)

cfa.fit <- cfa("F =~ x1 + x2 + x3 + x4 + x5 + x6 + x7 + x8 + x9",
    data = HolzingerSwineford1939, group = "school"
)

tidy(cfa.fit)</pre>
```

tidy.lm 303

tidy.lm

Tidy a(n) lm object

Description

Tidy summarizes information about the components of a model. A model component might be a single term in a regression, a single hypothesis, a cluster, or a class. Exactly what tidy considers to be a model component varies across models but is usually self-evident. If a model has several distinct types of components, you will need to specify which components to return.

Usage

```
## S3 method for class 'lm'
tidy(x, conf.int = FALSE, conf.level = 0.95, exponentiate = FALSE, ...)
```

Arguments

An lm object created by stats::lm().

conf.int

Logical indicating whether or not to include a confidence interval in the tidied output. Defaults to FALSE.

conf.level

The confidence level to use for the confidence interval if conf.int = TRUE. Must be strictly greater than 0 and less than 1. Defaults to 0.95, which corresponds to a 95 percent confidence interval.

exponentiate

Logical indicating whether or not to exponentiate the the coefficient estimates. This is typical for logistic and multinomial regressions, but a bad idea if there is no log or logit link. Defaults to FALSE.

Additional arguments. Not used. Needed to match generic signature only. Cautionary note: Misspelled arguments will be absorbed in . . . , where they will be

Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in . . . , where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

Details

If the linear model is an mlm object (multiple linear model), there is an additional column response. See tidy.mlm().

Value

```
A tibble::tibble() with columns:
```

conf.high Upper bound on the confidence interval for the estimate.

304 tidy.lm

conf.low Lower bound on the confidence interval for the estimate.

estimate The estimated value of the regression term.

p.value The two-sided p-value associated with the observed statistic.

statistic The value of a T-statistic to use in a hypothesis that the regression term is non-zero.

std.error The standard error of the regression term.

term The name of the regression term.

See Also

```
tidy(), stats::summary.lm()
Other Im tidiers: augment.glm(), augment.lm(), glance.glm(), glance.lm(), glance.summary.lm(),
glance.svyglm(), tidy.glm(), tidy.lm.beta(), tidy.mlm(), tidy.summary.lm()
```

```
library(ggplot2)
library(dplyr)
mod <- lm(mpg ~ wt + qsec, data = mtcars)</pre>
tidy(mod)
glance(mod)
# coefficient plot
d <- tidy(mod, conf.int = TRUE)</pre>
ggplot(d, aes(estimate, term, xmin = conf.low, xmax = conf.high, height = 0)) +
  geom_point() +
  geom_vline(xintercept = 0, lty = 4) +
  geom_errorbarh()
# aside: There are tidy() and glance() methods for lm.summary objects too.
# this can be useful when you want to conserve memory by converting large lm
# objects into their leaner summary.lm equivalents.
s <- summary(mod)</pre>
tidy(s, conf.int = TRUE)
glance(s)
augment(mod)
augment(mod, mtcars, interval = "confidence")
# predict on new data
newdata <- mtcars %>%
  head(6) %>%
  mutate(wt = wt + 1)
augment(mod, newdata = newdata)
# ggplot2 example where we also construct 95% prediction interval
```

tidy.lm.beta 305

```
# simpler bivariate model since we're plotting in 2D
mod2 <- lm(mpg ~ wt, data = mtcars)</pre>
au <- augment(mod2, newdata = newdata, interval = "prediction")</pre>
ggplot(au, aes(wt, mpg)) +
  geom_point() +
  geom\_line(aes(y = .fitted)) +
  geom_ribbon(aes(ymin = .lower, ymax = .upper), col = NA, alpha = 0.3)
# predict on new data without outcome variable. Output does not include .resid
newdata <- newdata %>%
  select(-mpg)
augment(mod, newdata = newdata)
au <- augment(mod, data = mtcars)</pre>
ggplot(au, aes(.hat, .std.resid)) +
  geom_vline(size = 2, colour = "white", xintercept = 0) +
  geom_hline(size = 2, colour = "white", yintercept = 0) +
  geom_point() +
  geom_smooth(se = FALSE)
plot(mod, which = 6)
ggplot(au, aes(.hat, .cooksd)) +
  geom_vline(xintercept = 0, colour = NA) +
  geom_abline(slope = seq(0, 3, by = 0.5), colour = "white") +
  geom_smooth(se = FALSE) +
  geom_point()
# column-wise models
a <- matrix(rnorm(20), nrow = 10)
b <- a + rnorm(length(a))</pre>
result <- lm(b \sim a)
tidy(result)
```

tidy.lm.beta

Tidy a(n) lm.beta object

Description

Tidy summarizes information about the components of a model. A model component might be a single term in a regression, a single hypothesis, a cluster, or a class. Exactly what tidy considers to be a model component varies across models but is usually self-evident. If a model has several distinct types of components, you will need to specify which components to return.

306 tidy.lm.beta

Usage

```
## S3 method for class 'lm.beta'
tidy(x, conf.int = FALSE, conf.level = 0.95, ...)
```

Arguments

An lm.beta object created by lm.beta::lm.beta.

Logical indicating whether or not to include a confidence interval in the tidied output. Defaults to FALSE.

The confidence level to use for the confidence interval if conf.int = TRUE. Must be strictly greater than 0 and less than 1. Defaults to 0.95, which corresponds to a 95 percent confidence interval.

Additional arguments. Not used. Needed to match generic signature only. Cautionary note: Misspelled arguments will be absorbed in . . . , where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

Details

If the linear model is an mlm object (multiple linear model), there is an additional column response. If you have missing values in your model data, you may need to refit the model with na.action = na.exclude.

Value

A tibble::tibble() with columns:

conf.high	Upper bound on the confidence interval for the estimate.
conf.low	Lower bound on the confidence interval for the estimate.
estimate	The estimated value of the regression term.
p.value	The two-sided p-value associated with the observed statistic.
statistic	The value of a T-statistic to use in a hypothesis that the regression term is non-zero.
std.error	The standard error of the regression term.
term	The name of the regression term.

See Also

```
Other Im tidiers: augment.glm(), augment.lm(), glance.glm(), glance.lm(), glance.summary.lm(), glance.svyglm(), tidy.glm(), tidy.lm(), tidy.mlm(), tidy.summary.lm()
```

tidy.lmodel2 307

Examples

```
# load libraries for models and data
library(lm.beta)
# fit models
mod <- stats::lm(speed ~ ., data = cars)</pre>
std <- lm.beta(mod)</pre>
# summarize model fit with tidiers
tidy(std, conf.int = TRUE)
# generate data
ctl <- c(4.17, 5.58, 5.18, 6.11, 4.50, 4.61, 5.17, 4.53, 5.33, 5.14)
trt <- c(4.81, 4.17, 4.41, 3.59, 5.87, 3.83, 6.03, 4.89, 4.32, 4.69)
group <- gl(2, 10, 20, labels = c("Ctl", "Trt"))</pre>
weight <- c(ctl, trt)</pre>
# fit models
mod2 <- lm(weight ~ group)</pre>
std2 <- lm.beta(mod2)</pre>
# summarize model fit with tidiers
tidy(std2, conf.int = TRUE)
```

tidy.lmodel2

Tidy a(n) lmodel2 object

Description

Tidy summarizes information about the components of a model. A model component might be a single term in a regression, a single hypothesis, a cluster, or a class. Exactly what tidy considers to be a model component varies across models but is usually self-evident. If a model has several distinct types of components, you will need to specify which components to return.

Usage

```
## S3 method for class 'lmodel2' tidy(x, ...)
```

Arguments

x A lmodel2 object returned by lmodel2::lmodel2().

Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in . . . , where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

308 tidy.lmodel2

• tidy() methods will warn when supplied an exponentiate argument if it will be ignored.

• augment() methods will warn when supplied a newdata argument if it will be ignored.

Details

There are always only two terms in an lmodel2: "Intercept" and "Slope". These are computed by four methods: OLS (ordinary least squares), MA (major axis), SMA (standard major axis), and RMA (ranged major axis).

The returned p-value is one-tailed and calculated via a permutation test. A permutational test is used because distributional assumptions may not be valid. More information can be found in vignette("mod2user", package = "lmodel2").

Value

A tibble::tibble() with columns:

conf.high Upper bound on the confidence interval for the estimate.

conf.low Lower bound on the confidence interval for the estimate.

estimate The estimated value of the regression term.

p.value The two-sided p-value associated with the observed statistic.

term The name of the regression term.
method Either OLS/MA/SMA/RMA

See Also

```
tidy(), lmodel2::lmodel2()
Other lmodel2 tidiers: glance.lmodel2()
```

```
# load libraries for models and data
library(lmodel2)

data(mod2ex2)
Ex2.res <- lmodel2(Prey ~ Predators, data = mod2ex2, "relative", "relative", 99)
Ex2.res

# summarize model fit with tidiers + visualization
tidy(Ex2.res)
glance(Ex2.res)

# this allows coefficient plots with ggplot2
library(ggplot2)

ggplot(tidy(Ex2.res), aes(estimate, term, color = method)) +
    geom_point() +</pre>
```

tidy.lmRob 309

```
geom_errorbarh(aes(xmin = conf.low, xmax = conf.high)) +
geom_errorbarh(aes(xmin = conf.low, xmax = conf.high))
```

tidy.lmRob

Tidy a(n) lmRob object

Description

Tidy summarizes information about the components of a model. A model component might be a single term in a regression, a single hypothesis, a cluster, or a class. Exactly what tidy considers to be a model component varies across models but is usually self-evident. If a model has several distinct types of components, you will need to specify which components to return.

Usage

```
## S3 method for class 'lmRob' tidy(x, ...)
```

Arguments

x A lmRob object returned from robust::lmRob().

Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in . . . , where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

Details

For tidiers for robust models from the MASS package see tidy.rlm().

See Also

```
robust::lmRob()
Other robust tidiers: augment.lmRob(), glance.glmRob(), glance.lmRob(), tidy.glmRob()
```

310 tidy.lmrob

Examples

```
# load modeling library
library(robust)

# fit model
m <- lmRob(mpg ~ wt, data = mtcars)

# summarize model fit with tidiers
tidy(m)
augment(m)
glance(m)</pre>
```

tidy.lmrob

Tidy a(n) lmrob object

Description

Tidy summarizes information about the components of a model. A model component might be a single term in a regression, a single hypothesis, a cluster, or a class. Exactly what tidy considers to be a model component varies across models but is usually self-evident. If a model has several distinct types of components, you will need to specify which components to return.

Usage

```
## S3 method for class 'lmrob'
tidy(x, conf.int = FALSE, conf.level = 0.95, ...)
```

Arguments

x A lmrob object returned from robustbase::lmrob().

conf.int Logical indicating whether or not to include a confidence interval in the tidied output. Defaults to FALSE.

output. Defaults to FALSE.

The confidence level to use for the confidence interval if conf.int = TRUE. Must be strictly greater than 0 and less than 1. Defaults to 0.95, which corresponds to

a 95 percent confidence interval.

Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in . . . , where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

tidy.lsmobj 311

Details

For tidiers for robust models from the MASS package see tidy.rlm().

See Also

```
robustbase::lmrob()
Other robustbase tidiers: augment.glmrob(), augment.lmrob(), glance.lmrob(), tidy.glmrob()
```

Examples

```
if (requireNamespace("robustbase", quietly = TRUE)) {
 # load libraries for models and data
 library(robustbase)
 data(coleman)
 set.seed(0)
 m <- lmrob(Y ~ ., data = coleman)</pre>
 tidy(m)
 augment(m)
 glance(m)
 data(carrots)
 Rfit <- glmrob(cbind(success, total - success) ~ logdose + block,
   family = binomial, data = carrots, method = "Mqle",
   control = glmrobMqle.control(tcc = 1.2)
 tidy(Rfit)
 augment(Rfit)
}
```

tidy.lsmobj

Tidy a(n) lsmobj object

Description

Tidy summarizes information about the components of a model. A model component might be a single term in a regression, a single hypothesis, a cluster, or a class. Exactly what tidy considers to be a model component varies across models but is usually self-evident. If a model has several distinct types of components, you will need to specify which components to return.

Usage

```
## S3 method for class 'lsmobj'
tidy(x, conf.int = FALSE, conf.level = 0.95, ...)
```

312 tidy.lsmobj

Arguments

X	An 1smobj object.
conf.int	Logical indicating whether or not to include a confidence interval in the tidied output. Defaults to FALSE.
conf.level	The confidence level to use for the confidence interval if conf.int = TRUE. Must be strictly greater than 0 and less than 1. Defaults to 0.95, which corresponds to a 95 percent confidence interval.
	Additional arguments passed to emmeans::summary.emmGrid() or lsmeans::summary.ref.grid(). Cautionary note: misspecified arguments may be silently ignored!

Details

Returns a data frame with one observation for each estimated marginal mean, and one column for each combination of factors. When the input is a contrast, each row will contain one estimated contrast.

There are a large number of arguments that can be passed on to emmeans::summary.emmGrid() or lsmeans::summary.ref.grid().

Value

```
A tibble::tibble() with columns:
                  Upper bound on the confidence interval for the estimate.
conf.high
conf.low
                  Lower bound on the confidence interval for the estimate.
                  Levels being compared.
contrast
df
                  Degrees of freedom used by this term in the model.
null.value
                  Value to which the estimate is compared.
p.value
                  The two-sided p-value associated with the observed statistic.
std.error
                  The standard error of the regression term.
                  Expected marginal mean
estimate
statistic
                  T-ratio statistic
```

See Also

```
tidy(), emmeans::ref_grid(), emmeans::emmeans(), emmeans::contrast()
Other emmeans tidiers: tidy.emmGrid(), tidy.ref.grid(), tidy.summary_emm()
```

```
# load libraries for models and data
library(emmeans)

# linear model for sales of oranges per day
oranges_lm1 <- lm(sales1 ~ price1 + price2 + day + store, data = oranges)</pre>
```

tidy.manova 313

```
# reference grid; see vignette("basics", package = "emmeans")
oranges_rg1 <- ref_grid(oranges_lm1)</pre>
td <- tidy(oranges_rg1)</pre>
td
# marginal averages
marginal <- emmeans(oranges_rg1, "day")</pre>
tidy(marginal)
# contrasts
tidy(contrast(marginal))
tidy(contrast(marginal, method = "pairwise"))
# plot confidence intervals
library(ggplot2)
ggplot(tidy(marginal, conf.int = TRUE), aes(day, estimate)) +
  geom_point() +
  geom_errorbar(aes(ymin = conf.low, ymax = conf.high))
# by multiple prices
by_price <- emmeans(oranges_lm1, "day",</pre>
  by = "price2",
  at = list(
    price1 = 50, price2 = c(40, 60, 80),
    day = c("2", "3", "4")
)
by_price
tidy(by_price)
ggplot(tidy(by_price, conf.int = TRUE), aes(price2, estimate, color = day)) +
  geom_line() +
  geom_errorbar(aes(ymin = conf.low, ymax = conf.high))
# joint_tests
tidy(joint_tests(oranges_lm1))
```

tidy.manova

Tidy a(n) manova object

Description

Tidy summarizes information about the components of a model. A model component might be a single term in a regression, a single hypothesis, a cluster, or a class. Exactly what tidy considers to be a model component varies across models but is usually self-evident. If a model has several distinct types of components, you will need to specify which components to return.

314 tidy.manova

Usage

```
## S3 method for class 'manova'
tidy(x, test = "Pillai", ...)
```

Arguments

Χ	A manova object return from stats::manova().
test	One of "Pillai" (Pillai's trace), "Wilks" (Wilk's lambda), "Hotelling-Lawley" (Hotelling-Lawley trace) or "Roy" (Roy's greatest root) indicating which test statistic should be used. Defaults to "Pillai".
	Arguments passed on to stats::summary.manova
	object An object of class "manova" or an aov object with multiple responses. intercept logical. If TRUE, the intercept term is included in the table.
	to tolerance to be used in deciding if the residuals are rank-deficient; see gr.

Details

Depending on which test statistic is specified only one of pillai, wilks, hl or roy is included.

Value

```
A tibble::tibble() with columns:
```

den.df Degrees of freedom of the denominator.

num. df Degrees of freedom.

p.value The two-sided p-value associated with the observed statistic.

statistic The value of a T-statistic to use in a hypothesis that the regression term is non-

zero.

term The name of the regression term.

pillai Pillai's trace. wilks Wilk's lambda.

hl Hotelling-Lawley trace.
roy Roy's greatest root.

See Also

```
tidy(), stats::summary.manova()
Other anova tidiers: glance.anova(), glance.aov(), tidy.TukeyHSD(), tidy.anova(), tidy.aov(), tidy.aovlist()
```

```
npk2 <- within(npk, foo <- rnorm(24))
m <- manova(cbind(yield, foo) \sim block + N * P * K, npk2)
tidy(m)
```

tidy.map 315

tidy.map

Tidy a(n) map object

Description

Tidy summarizes information about the components of a model. A model component might be a single term in a regression, a single hypothesis, a cluster, or a class. Exactly what tidy considers to be a model component varies across models but is usually self-evident. If a model has several distinct types of components, you will need to specify which components to return.

Usage

```
## S3 method for class 'map' tidy(x, ...)
```

Arguments

x A map object returned from maps::map().

Additional arguments. Not used. Needed to match generic signature only. Cautionary note: Misspelled arguments will be absorbed in . . . , where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

Value

A tibble::tibble() with columns:

term The name of the regression term.

long Longitude.

Latitude.

Remaining columns give information on geographic attributes and depend on the inputted map object. See ?maps::map for more information.

See Also

```
tidy(), maps::map()
```

316 tidy.margins

Examples

```
# load libraries for models and data
library(maps)
library(ggplot2)

ca <- map("county", "ca", plot = FALSE, fill = TRUE)

tidy(ca)

qplot(long, lat, data = ca, geom = "polygon", group = group)

tx <- map("county", "texas", plot = FALSE, fill = TRUE)
tidy(tx)
qplot(long, lat,
    data = tx, geom = "polygon", group = group,
    colour = I("white")
)</pre>
```

tidy.margins

Tidy a(n) margins object

Description

Tidy summarizes information about the components of a model. A model component might be a single term in a regression, a single hypothesis, a cluster, or a class. Exactly what tidy considers to be a model component varies across models but is usually self-evident. If a model has several distinct types of components, you will need to specify which components to return.

Usage

```
## S3 method for class 'margins'
tidy(x, conf.int = FALSE, conf.level = 0.95, ...)
```

Arguments

X	A margins object returned from margins::margins().
conf.int	Logical indicating whether or not to include a confidence interval in the tidied output. Defaults to FALSE.
conf.level	The confidence level to use for the confidence interval if conf.int = TRUE. Must be strictly greater than 0 and less than 1. Defaults to 0.95, which corresponds to a 95 percent confidence interval.
	Additional arguments. Not used. Needed to match generic signature only. Cautionary note: Misspelled arguments will be absorbed in , where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

tidy.margins 317

• tidy() methods will warn when supplied an exponentiate argument if it will be ignored.

• augment() methods will warn when supplied a newdata argument if it will be ignored.

Details

The margins package provides a way to obtain coefficient marginal effects for a variety of (non-linear) models, such as logit or models with multiway interaction terms. Note that the glance.margins() method requires rerunning the underlying model again, which can take some time. Similarly, an augment.margins() method is not currently supported, but users can simply run the underlying model to obtain the same information.

Value

A tibble::tibble() with columns:

conf.high	Upper bound on the confidence interval for the estimate.
conf.low	Lower bound on the confidence interval for the estimate.
estimate	The estimated value of the regression term.
p.value	The two-sided p-value associated with the observed statistic.
statistic	The value of a T-statistic to use in a hypothesis that the regression term is non-zero.
std.error	The standard error of the regression term.
term	The name of the regression term.

See Also

```
tidy(), margins::margins()
```

```
# load libraries for models and data
library(margins)

# example 1: logit model
mod_log <- glm(am ~ cyl + hp + wt, data = mtcars, family = binomial)

# get tidied "naive" model coefficients
tidy(mod_log)

# convert to marginal effects with margins()
marg_log <- margins(mod_log)

# get tidied marginal effects
tidy(marg_log)
tidy(marg_log, conf.int = TRUE)</pre>
```

318 tidy.Mclust

```
# requires running the underlying model again. quick for this example
glance(marg_log)
# augmenting `margins` outputs isn't supported, but
# you can get the same info by running on the underlying model
augment(mod_log)
# example 2: threeway interaction terms
mod_ie <- lm(mpg ~ wt * cyl * disp, data = mtcars)</pre>
# get tidied "naive" model coefficients
tidy(mod_ie)
# convert to marginal effects with margins()
marg_ie0 <- margins(mod_ie)</pre>
# get tidied marginal effects
tidy(marg_ie0)
glance(marg_ie0)
# marginal effects evaluated at specific values of a variable (here: cyl)
marg_ie1 \leftarrow margins(mod_ie, at = list(cyl = c(4,6,8)))
# summarize model fit with tidiers
tidy(marg_ie1)
# marginal effects of one interaction variable (here: wt), modulated at
# specific values of the two other interaction variables (here: cyl and drat)
marg_ie2 <- margins(mod_ie,</pre>
                    variables = "wt",
                    at = list(cyl = c(4,6,8), drat = c(3, 3.5, 4)))
# summarize model fit with tidiers
tidy(marg_ie2)
```

tidy.Mclust

Tidy a(n) Mclust object

Description

Tidy summarizes information about the components of a model. A model component might be a single term in a regression, a single hypothesis, a cluster, or a class. Exactly what tidy considers to be a model component varies across models but is usually self-evident. If a model has several distinct types of components, you will need to specify which components to return.

Usage

```
## S3 method for class 'Mclust' tidy(x, ...)
```

tidy.Mclust 319

Arguments

. . .

x An Mclust object return from mclust::Mclust().

Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in . . . , where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

Value

A tibble::tibble() with columns:

proportion The mixing proportion of each component size Number of points assigned to cluster.

mean The mean for each component. In case of 2+ dimensional models, a column

with the mean is added for each dimension. NA for noise component

variance In case of one-dimensional and spherical models, the variance for each compo-

nent, omitted otherwise. NA for noise component

component Cluster id as a factor.

See Also

```
tidy(), mclust::Mclust()
Other mclust tidiers: augment.Mclust()
```

```
# load library for models and data
library(mclust)

# load data manipulation libraries
library(dplyr)
library(tibble)
library(purrr)
library(tidyr)

set.seed(27)

centers <- tibble(
   cluster = factor(1:3),
    # number points in each cluster
   num_points = c(100, 150, 50),
    # x1 coordinate of cluster center</pre>
```

320 tidy.mediate

```
x1 = c(5, 0, -3),
# x2 coordinate of cluster center
x2 = c(-1, 1, -2)
)

points <- centers %>%
  mutate(
    x1 = map2(num_points, x1, rnorm),
    x2 = map2(num_points, x2, rnorm)
) %>%
  select(-num_points, -cluster) %>%
  unnest(c(x1, x2))

# fit model
m <- Mclust(points)

# summarize model fit with tidiers
tidy(m)
augment(m, points)
glance(m)</pre>
```

tidy.mediate

Tidy a(n) mediate object

Description

Tidy summarizes information about the components of a model. A model component might be a single term in a regression, a single hypothesis, a cluster, or a class. Exactly what tidy considers to be a model component varies across models but is usually self-evident. If a model has several distinct types of components, you will need to specify which components to return.

Usage

```
## S3 method for class 'mediate'
tidy(x, conf.int = FALSE, conf.level = 0.95, ...)
```

Arguments

x	A mediate object produced by a call to mediation::mediate().
conf.int	Logical indicating whether or not to include a confidence interval in the tidied output. Defaults to FALSE.
conf.level	The confidence level to use for the confidence interval if conf.int = TRUE. Must be strictly greater than 0 and less than 1. Defaults to 0.95, which corresponds to a 95 percent confidence interval.
	Additional arguments. Not used. Needed to match generic signature only. Cautionary note: Misspelled arguments will be absorbed in , where they will be ignored. If the misspelled argument has a default value, the default value will be

tidy.mediate 321

used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

Details

The tibble has four rows. The first two indicate the mediated effect in the control and treatment groups, respectively. And the last two the direct effect in each group.

Value

A tibble::tibble() with columns:

conf.high	Upper bound on the confidence interval for the estimate.
conf.low	Lower bound on the confidence interval for the estimate.
estimate	The estimated value of the regression term.
p.value	The two-sided p-value associated with the observed statistic.
statistic	The value of a T-statistic to use in a hypothesis that the regression term is non-zero.
std.error	The standard error of the regression term.
term	The name of the regression term.

See Also

```
tidy(), mediation::mediate()
```

```
# load libraries for models and data
library(mediation)

data(jobs)

# fit models
b <- lm(job_seek ~ treat + econ_hard + sex + age, data = jobs)
c <- lm(depress2 ~ treat + job_seek + econ_hard + sex + age, data = jobs)
mod <- mediate(b, c, sims = 50, treat = "treat", mediator = "job_seek")

# summarize model fit with tidiers
tidy(mod)
tidy(mod, conf.int = TRUE)
tidy(mod, conf.int = TRUE, conf.level = .99)</pre>
```

322 tidy.mfx

tidy.mfx $Tidy \ a(n) \ mfx \ object$

Description

Tidy summarizes information about the components of a model. A model component might be a single term in a regression, a single hypothesis, a cluster, or a class. Exactly what tidy considers to be a model component varies across models but is usually self-evident. If a model has several distinct types of components, you will need to specify which components to return.

The particular functions below provide generic tidy methods for objects returned by the mfx package, preserving the calculated marginal effects instead of the naive model coefficients. The returned tidy tibble will also include an additional "atmean" column indicating how the marginal effects were originally calculated (see Details below).

Usage

```
## S3 method for class 'mfx'
tidy(x, conf.int = FALSE, conf.level = 0.95, ...)
## S3 method for class 'logitmfx'
tidy(x, conf.int = FALSE, conf.level = 0.95, ...)
## S3 method for class 'negbinmfx'
tidy(x, conf.int = FALSE, conf.level = 0.95, ...)
## S3 method for class 'poissonmfx'
tidy(x, conf.int = FALSE, conf.level = 0.95, ...)
## S3 method for class 'probitmfx'
tidy(x, conf.int = FALSE, conf.level = 0.95, ...)
```

Arguments

X	A logitmfx, negbinmfx, poissonmfx, or probitmfx object. (Note that betamfx objects receive their own set of tidiers.)
conf.int	Logical indicating whether or not to include a confidence interval in the tidied output. Defaults to FALSE.
conf.level	The confidence level to use for the confidence interval if conf.int = TRUE. Must be strictly greater than 0 and less than 1. Defaults to 0.95, which corresponds to a 95 percent confidence interval.
	Additional arguments. Not used. Needed to match generic signature only. Cautionary note: Misspelled arguments will be absorbed in , where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

tidy.mfx 323

 tidy() methods will warn when supplied an exponentiate argument if it will be ignored.

 augment() methods will warn when supplied a newdata argument if it will be ignored.

Details

The mfx package provides methods for calculating marginal effects for various generalized linear models (GLMs). Unlike standard linear models, estimated model coefficients in a GLM cannot be directly interpreted as marginal effects (i.e., the change in the response variable predicted after a one unit change in one of the regressors). This is because the estimated coefficients are multiplicative, dependent on both the link function that was used for the estimation and any other variables that were included in the model. When calculating marginal effects, users must typically choose whether they want to use i) the average observation in the data, or ii) the average of the sample marginal effects. See vignette("mfxarticle") from the mfx package for more details.

Value

A tibble::tibble() with columns:

conf.high	Upper bound on the confidence interval for the estimate.
conf.low	Lower bound on the confidence interval for the estimate.
estimate	The estimated value of the regression term.
p.value	The two-sided p-value associated with the observed statistic.
statistic	The value of a T-statistic to use in a hypothesis that the regression term is non-
	zero.
std.error	The standard error of the regression term.
term	The name of the regression term.
atmean	TRUE if the marginal effects were originally calculated as the partial effects for the average observation. If FALSE, then these were instead calculated as average partial effects.

See Also

```
tidy(), mfx::logitmfx(), mfx::negbinmfx(), mfx::poissonmfx(), mfx::probitmfx()
Other mfx tidiers: augment.betamfx(), augment.mfx(), glance.betamfx(), glance.mfx(),
tidy.betamfx()
```

```
# load libraries for models and data
library(mfx)

# get the marginal effects from a logit regression
mod_logmfx <- logitmfx(am ~ cyl + hp + wt, atmean = TRUE, data = mtcars)
tidy(mod_logmfx, conf.int = TRUE)</pre>
```

324 tidy.mjoint

```
# compare with the naive model coefficients of the same logit call
tidy(
   glm(am ~ cyl + hp + wt, family = binomial, data = mtcars),
   conf.int = TRUE
)

augment(mod_logmfx)
glance(mod_logmfx)

# another example, this time using probit regression
mod_probmfx <- probitmfx(am ~ cyl + hp + wt, atmean = TRUE, data = mtcars)

tidy(mod_probmfx, conf.int = TRUE)
augment(mod_probmfx)
glance(mod_probmfx)</pre>
```

tidy.mjoint

Tidy a(n) mjoint object

Description

Tidy summarizes information about the components of a model. A model component might be a single term in a regression, a single hypothesis, a cluster, or a class. Exactly what tidy considers to be a model component varies across models but is usually self-evident. If a model has several distinct types of components, you will need to specify which components to return.

Usage

```
## $3 method for class 'mjoint'
tidy(
    x,
    component = "survival",
    conf.int = FALSE,
    conf.level = 0.95,
    boot_se = NULL,
    ...
)
```

Arguments

x An mjoint object returned from joineRML::mjoint().

component Character specifying whether to tidy the survival or the longitudinal component of the model. Must be either "survival" or "longitudinal". Defaults to

"survival".

conf. int Logical indicating whether or not to include a confidence interval in the tidied

output. Defaults to FALSE.

tidy.mjoint 325

conf.level The confidence level to use for the confidence interval if conf.int = TRUE. Must be strictly greater than 0 and less than 1. Defaults to 0.95, which corresponds to

a 95 percent confidence interval.

boot_se Optionally a bootSE object from joineRML::bootSE(). If specified, calculates

confidence intervals via the bootstrap. Defaults to NULL, in which case standard

errors are calculated from the empirical information matrix.

Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in . . . , where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed

using conf.level = 0.95. Two exceptions here are:

• tidy() methods will warn when supplied an exponentiate argument if it will be ignored.

• augment() methods will warn when supplied a newdata argument if it will be ignored.

Value

A tibble::tibble() with columns:

conf.high Upper bound on the confidence interval for the estimate.

conf.low Lower bound on the confidence interval for the estimate.

estimate The estimated value of the regression term.

p.value The two-sided p-value associated with the observed statistic.

statistic The value of a T-statistic to use in a hypothesis that the regression term is non-

zero.

std.error The standard error of the regression term.

term The name of the regression term.

See Also

```
tidy(), joineRML::mjoint(), joineRML::bootSE()
Other mjoint tidiers: glance.mjoint()
```

```
# broom only skips running these examples because the example models take a
# while to generate—they should run just fine, though!
## Not run:

# load libraries for models and data
library(joineRML)

# fit a joint model with bivariate longitudinal outcomes
data(heart.valve)
```

326 tidy.mle2

```
hvd <- heart.valve[!is.na(heart.valve$log.grad) &</pre>
  !is.na(heart.valve$log.lvmi) &
  heart.valve$num <= 50, ]
fit <- mjoint(</pre>
  formLongFixed = list(
    "grad" = log.grad ~ time + sex + hs,
    "lvmi" = log.lvmi ~ time + sex
  ),
  formLongRandom = list(
    "grad" = ~ 1 | num,
    "lvmi" = ~ time | num
  ),
  formSurv = Surv(fuyrs, status) ~ age,
  data = hvd,
  inits = list("gamma" = c(0.11, 1.51, 0.80)),
  timeVar = "time"
)
# extract the survival fixed effects
tidy(fit)
# extract the longitudinal fixed effects
tidy(fit, component = "longitudinal")
# extract the survival fixed effects with confidence intervals
tidy(fit, ci = TRUE)
# extract the survival fixed effects with confidence intervals based
# on bootstrapped standard errors
bSE <- bootSE(fit, nboot = 5, safe.boot = TRUE)</pre>
tidy(fit, boot_se = bSE, ci = TRUE)
# augment original data with fitted longitudinal values and residuals
hvd2 <- augment(fit)</pre>
# extract model statistics
glance(fit)
## End(Not run)
```

tidy.mle2

Tidy a(n) mle2 object

Description

Tidy summarizes information about the components of a model. A model component might be a single term in a regression, a single hypothesis, a cluster, or a class. Exactly what tidy considers

tidy.mle2 327

to be a model component varies across models but is usually self-evident. If a model has several distinct types of components, you will need to specify which components to return.

Usage

```
## S3 method for class 'mle2'
tidy(x, conf.int = FALSE, conf.level = 0.95, ...)
```

Arguments

An mle2 object created by a call to bbmle::mle2().

Logical indicating whether or not to include a confidence interval in the tidied output. Defaults to FALSE.

The confidence level to use for the confidence interval if conf.int = TRUE. Must be strictly greater than 0 and less than 1. Defaults to 0.95, which corresponds to a 95 percent confidence interval.

Additional arguments. Not used. Needed to match generic signature only. Cautionary note: Misspelled arguments will be absorbed in . . . , where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

Value

A tibble::tibble() with columns:

conf.high	Upper bound on the confidence interval for the estimate.
conf.low	Lower bound on the confidence interval for the estimate.
estimate	The estimated value of the regression term.
p.value	The two-sided p-value associated with the observed statistic.
statistic	The value of a T-statistic to use in a hypothesis that the regression term is non-zero.
std.error	The standard error of the regression term.
term	The name of the regression term.

See Also

```
tidy(), bbmle::mle2(), tidy_optim()
```

328 tidy.mlm

Examples

```
# load libraries for models and data
library(bbmle)

# generate data
x <- 0:10
y <- c(26, 17, 13, 12, 20, 5, 9, 8, 5, 4, 8)
d <- data.frame(x, y)

# fit model
fit <- mle2(y ~ dpois(lambda = ymean),
    start = list(ymean = mean(y)), data = d
)

# summarize model fit with tidiers
tidy(fit)</pre>
```

tidy.mlm

Tidy a(n) mlm object

Description

Tidy summarizes information about the components of a model. A model component might be a single term in a regression, a single hypothesis, a cluster, or a class. Exactly what tidy considers to be a model component varies across models but is usually self-evident. If a model has several distinct types of components, you will need to specify which components to return.

Usage

```
## S3 method for class 'mlm'
tidy(x, conf.int = FALSE, conf.level = 0.95, ...)
```

Arguments

X	An mlm object created by stats::lm() with a matrix as the response.
conf.int	Logical indicating whether or not to include a confidence interval in the tidied output. Defaults to FALSE.
conf.level	The confidence level to use for the confidence interval if conf.int = TRUE. Must be strictly greater than 0 and less than 1. Defaults to 0.95, which corresponds to a 95 percent confidence interval.
	Additional arguments. Not used. Needed to match generic signature only. Cautionary note: Misspelled arguments will be absorbed in , where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

tidy.mlm 329

• tidy() methods will warn when supplied an exponentiate argument if it will be ignored.

• augment() methods will warn when supplied a newdata argument if it will be ignored.

Details

In contrast to 1m object (simple linear model), tidy output for m1m (multiple linear model) objects contain an additional column response.

If you have missing values in your model data, you may need to refit the model with na.action = na.exclude.

Value

A tibble::tibble() with columns:

conf.high	Upper bound on the confidence interval for the estimate.
conf.low	Lower bound on the confidence interval for the estimate.
estimate	The estimated value of the regression term.
p.value	The two-sided p-value associated with the observed statistic.
statistic	The value of a T-statistic to use in a hypothesis that the regression term is non-zero.
std.error	The standard error of the regression term.
term	The name of the regression term.

See Also

```
tidy()
```

```
Other Im tidiers: augment.glm(), augment.lm(), glance.glm(), glance.lm(), glance.summary.lm(), glance.svyglm(), tidy.glm(), tidy.lm(), tidy.lm.beta(), tidy.summary.lm()
```

```
# fit model
mod <- lm(cbind(mpg, disp) ~ wt, mtcars)
# summarize model fit with tidiers
tidy(mod, conf.int = TRUE)</pre>
```

330 tidy.mlogit

tidy.mlogit	Tidying methods for logit models	

Description

These methods tidy the coefficients of mnl and nl models generated by the functions of the mlogit package.

Usage

```
## S3 method for class 'mlogit'
tidy(x, conf.int = FALSE, conf.level = 0.95, ...)
```

Arguments

an object returned from mlogit::mlogit().
conf.int Logical indicating whether or not to include a confidence interval in the tidied output. Defaults to FALSE.

conf.level The confidence level to use for the confidence interval if conf.int = TRUE. Must be strictly greater than 0 and less than 1. Defaults to 0.95, which corresponds to a 95 percent confidence interval.

Additional arguments. Not used. Needed to match generic signature only. Cautionary note: Misspelled arguments will be absorbed in . . . , where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

Value

A tibble::tibble() with columns:

conf.high	Upper bound on the confidence interval for the estimate.
conf.low	Lower bound on the confidence interval for the estimate.
estimate	The estimated value of the regression term.
p.value	The two-sided p-value associated with the observed statistic.
statistic	The value of a T-statistic to use in a hypothesis that the regression term is non-zero.
std.error	The standard error of the regression term.
term	The name of the regression term.

tidy.muhaz 331

See Also

```
tidy(), mlogit::mlogit()
Other mlogit tidiers: augment.mlogit(), glance.mlogit()
```

Examples

```
# load libraries for models and data
library(mlogit)

data("Fishing", package = "mlogit")
Fish <- dfidx(Fishing, varying = 2:9, shape = "wide", choice = "mode")

# fit model
m <- mlogit(mode ~ price + catch | income, data = Fish)

# summarize model fit with tidiers
tidy(m)
augment(m)
glance(m)</pre>
```

tidy.muhaz

Tidy a(n) muhaz object

Description

Tidy summarizes information about the components of a model. A model component might be a single term in a regression, a single hypothesis, a cluster, or a class. Exactly what tidy considers to be a model component varies across models but is usually self-evident. If a model has several distinct types of components, you will need to specify which components to return.

Usage

```
## S3 method for class 'muhaz' tidy(x, ...)
```

Arguments

x A muhaz object returned by muhaz::muhaz().

Additional arguments. Not used. Needed to match generic signature only. Cautionary note: Misspelled arguments will be absorbed in . . ., where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

332 tidy.multinom

Value

```
A tibble::tibble() with columns:

time Point in time.

estimate Estimated hazard rate.
```

See Also

```
tidy(), muhaz::muhaz()
Other muhaz tidiers: glance.muhaz()
```

Examples

```
# load libraries for models and data
library(muhaz)
library(survival)

# fit model
x <- muhaz(ovarian$futime, ovarian$fustat)

# summarize model fit with tidiers
tidy(x)
glance(x)</pre>
```

tidy.multinom

Tidying methods for multinomial logistic regression models

Description

These methods tidy the coefficients of multinomial logistic regression models generated by multinom of the nnet package.

Usage

```
## S3 method for class 'multinom'
tidy(x, conf.int = FALSE, conf.level = 0.95, exponentiate = FALSE, ...)
```

Arguments

object returned from nnet::multinom().
object returned from nnet::multinom@

conf.int Logical indicating whether or not to include a confidence interval in the tidied

output. Defaults to FALSE.

conf.level The confidence level to use for the confidence interval if conf.int = TRUE. Must

be strictly greater than 0 and less than 1. Defaults to 0.95, which corresponds to

a 95 percent confidence interval.

tidy.multinom 333

exponentiate

Logical indicating whether or not to exponentiate the the coefficient estimates. This is typical for logistic and multinomial regressions, but a bad idea if there is no log or logit link. Defaults to FALSE.

Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in . . . , where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

Value

A tibble::tibble() with columns:

conf.high Upper bound on the confidence interval for the estimate. conf.low Lower bound on the confidence interval for the estimate. estimate The estimated value of the regression term. p.value The two-sided p-value associated with the observed statistic. statistic The value of a T-statistic to use in a hypothesis that the regression term is nonstd.error The standard error of the regression term. The name of the regression term. term y.value The response level.

See Also

```
tidy(), nnet::multinom()
Other multinom tidiers: glance.multinom()
```

```
# load libraries for models and data
library(nnet)
library(MASS)

example(birthwt)

bwt.mu <- multinom(low ~ ., bwt)

tidy(bwt.mu)
glance(bwt.mu)

# or, for output from a multinomial logistic regression</pre>
```

334 tidy.negbin

```
fit.gear <- multinom(gear ~ mpg + factor(am), data = mtcars)
tidy(fit.gear)
glance(fit.gear)</pre>
```

tidy.negbin

 $Tidy \ a(n) \ negbin \ object$

Description

Tidy summarizes information about the components of a model. A model component might be a single term in a regression, a single hypothesis, a cluster, or a class. Exactly what tidy considers to be a model component varies across models but is usually self-evident. If a model has several distinct types of components, you will need to specify which components to return.

Usage

```
## S3 method for class 'negbin'
tidy(x, conf.int = FALSE, conf.level = 0.95, exponentiate = FALSE, ...)
```

Arguments

X	A glm.nb object returned by MASS::glm.nb().
conf.int	Logical indicating whether or not to include a confidence interval in the tidied output. Defaults to FALSE.
conf.level	The confidence level to use for the confidence interval if conf.int = TRUE. Must be strictly greater than 0 and less than 1. Defaults to 0.95, which corresponds to a 95 percent confidence interval.
exponentiate	Logical indicating whether or not to exponentiate the the coefficient estimates. This is typical for logistic and multinomial regressions, but a bad idea if there is no log or logit link. Defaults to FALSE.
	For tidy(), additional arguments passed to summary(). Otherwise ignored.

See Also

```
MASS::glm.nb()
Other glm.nb tidiers: glance.negbin()
```

```
# load libraries for models and data
library(MASS)
# fit model
r <- glm.nb(Days ~ Sex / (Age + Eth * Lrn), data = quine)</pre>
```

tidy.nlrq 335

```
# summarize model fit with tidiers
tidy(r)
glance(r)
```

tidy.nlrq

Tidy a(n) nlrq object

Description

Tidy summarizes information about the components of a model. A model component might be a single term in a regression, a single hypothesis, a cluster, or a class. Exactly what tidy considers to be a model component varies across models but is usually self-evident. If a model has several distinct types of components, you will need to specify which components to return.

Usage

```
## S3 method for class 'nlrq'
tidy(x, conf.int = FALSE, conf.level = 0.95, ...)
```

Arguments

A nlrq object returned from quantreg::nlrq().

Logical indicating whether or not to include a confidence interval in the tidied output. Defaults to FALSE.

The confidence level to use for the confidence interval if conf.int = TRUE. Must be strictly greater than 0 and less than 1. Defaults to 0.95, which corresponds to a 95 percent confidence interval.

Additional arguments. Not used. Needed to match generic signature only. Cautionary note: Misspelled arguments will be absorbed in . . . , where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

• tidy() methods will warn when supplied an exponent interval in the tidied output.

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

Value

A tibble::tibble() with columns:

```
conf.high Upper bound on the confidence interval for the estimate.

conf.low Lower bound on the confidence interval for the estimate.

estimate The estimated value of the regression term.

p.value The two-sided p-value associated with the observed statistic.
```

336 tidy.nls

statistic The value of a T-statistic to use in a hypothesis that the regression term is non-

zero.

std.error The standard error of the regression term.

term The name of the regression term.

See Also

```
tidy(), quantreg::nlrq()
Other quantreg tidiers: augment.nlrq(), augment.rq(), augment.rqs(), glance.nlrq(), glance.rq(),
tidy.rq(), tidy.rqs()
```

Examples

```
# load modeling library
library(quantreg)

# build artificial data with multiplicative error
set.seed(1)
dat <- NULL
dat$x <- rep(1:25, 20)
dat$y <- SSlogis(dat$x, 10, 12, 2) * rnorm(500, 1, 0.1)

# fit the median using nlrq
mod <- nlrq(y ~ SSlogis(x, Asym, mid, scal),
    data = dat, tau = 0.5, trace = TRUE
)

# summarize model fit with tidiers
tidy(mod)
glance(mod)
augment(mod)</pre>
```

tidy.nls

Tidy a(n) nls object

Description

Tidy summarizes information about the components of a model. A model component might be a single term in a regression, a single hypothesis, a cluster, or a class. Exactly what tidy considers to be a model component varies across models but is usually self-evident. If a model has several distinct types of components, you will need to specify which components to return.

Usage

```
## S3 method for class 'nls'
tidy(x, conf.int = FALSE, conf.level = 0.95, ...)
```

tidy.nls 337

Arguments

An nls object returned from stats::nls().

Logical indicating whether or not to include a confidence interval in the tidied output. Defaults to FALSE.

Conf.level The confidence level to use for the confidence interval if conf.int = TRUE. Must be strictly greater than 0 and less than 1. Defaults to 0.95, which corresponds to a 95 percent confidence interval.

Additional arguments. Not used. Needed to match generic signature only. Cautionary note: Misspelled arguments will be absorbed in . . . , where they will be ignored. If the misspelled argument has a default value, the default value will be

used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:
tidy() methods will warn when supplied an exponentiate argument if it

 augment() methods will warn when supplied a newdata argument if it will be ignored.

Value

A tibble::tibble() with columns:

will be ignored.

```
conf.high Upper bound on the confidence interval for the estimate.

Lower bound on the confidence interval for the estimate.

The estimated value of the regression term.

The two-sided p-value associated with the observed statistic.

The value of a T-statistic to use in a hypothesis that the regression term is non-zero.

The standard error of the regression term.

The name of the regression term.
```

See Also

```
tidy, stats::nls(), stats::summary.nls()
Other nls tidiers: augment.nls(), glance.nls()
```

```
# fit model
n <- nls(mpg ~ k * e^wt, data = mtcars, start = list(k = 1, e = 2))
# summarize model fit with tidiers + visualization
tidy(n)
augment(n)
glance(n)</pre>
```

338 tidy.numeric

```
library(ggplot2)
ggplot(augment(n), aes(wt, mpg)) +
  geom_point() +
  geom_line(aes(y = .fitted))

newdata <- head(mtcars)
newdata$wt <- newdata$wt + 1

augment(n, newdata = newdata)</pre>
```

tidy.numeric

Tidy atomic vectors

Description

Vector tidiers are deprecated and will be removed from an upcoming release of broom.

Usage

```
## S3 method for class 'numeric'
tidy(x, ...)
## S3 method for class 'character'
tidy(x, ...)
## S3 method for class 'logical'
tidy(x, ...)
```

Arguments

x An object of class "numeric", "integer", "character", or "logical". Most likely a named vector

... Extra arguments (not used)

Details

Turn atomic vectors into data frames, where the names of the vector (if they exist) are a column and the values of the vector are a column.

See Also

```
Other deprecated: bootstrap(), confint_tidy(), data.frame_tidiers, finish_glance(), fix_data_frame(), summary_tidiers, tidy.density(), tidy.dist(), tidy.ftable()

Other deprecated: bootstrap(), confint_tidy(), data.frame_tidiers, finish_glance(), fix_data_frame(), summary_tidiers, tidy.density(), tidy.dist(), tidy.ftable()

Other deprecated: bootstrap(), confint_tidy(), data.frame_tidiers, finish_glance(), fix_data_frame(), summary_tidiers, tidy.density(), tidy.dist(), tidy.ftable()
```

tidy.orcutt 339

Examples

```
## Not run:
x <- 1:5
names(x) <- letters[1:5]
tidy(x)
## End(Not run)</pre>
```

tidy.orcutt

Tidy a(n) orcutt object

Description

Tidy summarizes information about the components of a model. A model component might be a single term in a regression, a single hypothesis, a cluster, or a class. Exactly what tidy considers to be a model component varies across models but is usually self-evident. If a model has several distinct types of components, you will need to specify which components to return.

Usage

```
## S3 method for class 'orcutt' tidy(x, ...)
```

Arguments

x An orcutt object returned from orcutt::cochrane.orcutt().

Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in . . . , where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

Value

A tibble::tibble() with columns:

rhe estimated value of the regression term.

p.value
The two-sided p-value associated with the observed statistic.

The value of a T-statistic to use in a hypothesis that the regression term is non-zero.

Std.error
The standard error of the regression term.

The name of the regression term.

340 tidy.pairwise.htest

See Also

```
orcutt::cochrane.orcutt()
Other orcutt tidiers: glance.orcutt()
```

Examples

```
# load libraries for models and data
library(orcutt)

# fit model and summarize results
reg <- lm(mpg ~ wt + qsec + disp, mtcars)
tidy(reg)

co <- cochrane.orcutt(reg)
tidy(co)
glance(co)</pre>
```

tidy.pairwise.htest

Tidy a(n) pairwise.htest object

Description

Tidy summarizes information about the components of a model. A model component might be a single term in a regression, a single hypothesis, a cluster, or a class. Exactly what tidy considers to be a model component varies across models but is usually self-evident. If a model has several distinct types of components, you will need to specify which components to return.

Usage

```
## S3 method for class 'pairwise.htest'
tidy(x, ...)
```

Arguments

Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in . . . , where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

tidy.pam 341

Details

Note that in one-sided tests, the alternative hypothesis of each test can be stated as "group1 is greater/less than group2".

Note also that the columns of group1 and group2 will always be a factor, even if the original input is (e.g.) numeric.

Value

```
A tibble::tibble() with columns:
```

group1 First group being compared.
group2 Second group being compared.

p.value The two-sided p-value associated with the observed statistic.

See Also

```
stats::pairwise.t.test(), stats::pairwise.wilcox.test(), tidy()
Other htest tidiers: augment.htest(), tidy.htest(), tidy.power.htest()
```

Examples

```
attach(airquality)
Month <- factor(Month, labels = month.abb[5:9])
ptt <- pairwise.t.test(Ozone, Month)
tidy(ptt)

library(modeldata)
data(hpc_data)
attach(hpc_data)
ptt2 <- pairwise.t.test(compounds, class)
tidy(ptt2)

tidy(pairwise.t.test(compounds, class, alternative = "greater"))
tidy(pairwise.t.test(compounds, class, alternative = "less"))

tidy(pairwise.wilcox.test(compounds, class))</pre>
```

tidy.pam

Tidy a(n) pam object

Description

Tidy summarizes information about the components of a model. A model component might be a single term in a regression, a single hypothesis, a cluster, or a class. Exactly what tidy considers to be a model component varies across models but is usually self-evident. If a model has several distinct types of components, you will need to specify which components to return.

342 tidy.pam

Usage

```
## S3 method for class 'pam'
tidy(x, col.names = paste0("x", 1:ncol(x$medoids)), ...)
```

Arguments

An pam object returned from cluster::pam()

Col.names

Column names in the input data frame. Defaults to the names of the variables in x.

Additional arguments. Not used. Needed to match generic signature only. Cautionary note: Misspelled arguments will be absorbed in . . . , where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

Details

For examples, see the pam vignette.

Value

A tibble::tibble() with columns:

size	Size of each cluster.
max.diss	Maximal dissimilarity between the observations in the cluster and that cluster's medoid.
avg.diss	Average dissimilarity between the observations in the cluster and that cluster's medoid.
diameter	Diameter of the cluster.
separation	Separation of the cluster.
avg.width	Average silhouette width of the cluster.
cluster	A factor describing the cluster from 1:k.

See Also

```
tidy(), cluster::pam()
Other pam tidiers: augment.pam(), glance.pam()
```

tidy.plm 343

Examples

```
# load libraries for models and data
library(dplyr)
library(ggplot2)
library(cluster)
library(modeldata)
data(hpc_data)
x <- hpc_data[, 2:5]</pre>
p \leftarrow pam(x, k = 4)
# summarize model fit with tidiers + visualization
tidy(p)
glance(p)
augment(p, x)
augment(p, x) %>%
  ggplot(aes(compounds, input_fields)) +
  geom_point(aes(color = .cluster)) +
  geom_text(aes(label = cluster), data = tidy(p), size = 10)
```

tidy.plm

Tidy a(n) plm object

Description

Tidy summarizes information about the components of a model. A model component might be a single term in a regression, a single hypothesis, a cluster, or a class. Exactly what tidy considers to be a model component varies across models but is usually self-evident. If a model has several distinct types of components, you will need to specify which components to return.

Usage

```
## S3 method for class 'plm'
tidy(x, conf.int = FALSE, conf.level = 0.95, ...)
```

Arguments

X	A plm objected returned by plm::plm().
conf.int	Logical indicating whether or not to include a confidence interval in the tidied output. Defaults to FALSE.
conf.level	The confidence level to use for the confidence interval if conf.int = TRUE. Must be strictly greater than 0 and less than 1. Defaults to 0.95, which corresponds to a 95 percent confidence interval.

344 tidy.plm

Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in . . ., where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

Value

. . .

A tibble::tibble() with columns:

conf.high Upper bound on the confidence interval for the estimate.

conf.low Lower bound on the confidence interval for the estimate.

estimate The estimated value of the regression term.

p.value The two-sided p-value associated with the observed statistic.

statistic The value of a T-statistic to use in a hypothesis that the regression term is non-zero.

std.error The standard error of the regression term.

term The name of the regression term.

See Also

```
tidy(), plm::plm(), tidy.lm()
Other plm tidiers: augment.plm(), glance.plm()
```

```
# load libraries for models and data
library(plm)

# load data
data("Produc", package = "plm")

# fit model
zz <- plm(log(gsp) ~ log(pcap) + log(pc) + log(emp) + unemp,
data = Produc, index = c("state", "year")
)

# summarize model fit with tidiers
summary(zz)

tidy(zz)
tidy(zz, conf.int = TRUE)
tidy(zz, conf.int = TRUE, conf.level = 0.9)</pre>
```

tidy.poLCA 345

```
augment(zz)
glance(zz)
```

tidy.poLCA

Tidy a(n) poLCA object

Description

Tidy summarizes information about the components of a model. A model component might be a single term in a regression, a single hypothesis, a cluster, or a class. Exactly what tidy considers to be a model component varies across models but is usually self-evident. If a model has several distinct types of components, you will need to specify which components to return.

Usage

```
## S3 method for class 'poLCA' tidy(x, ...)
```

Arguments

. . .

x A poLCA object returned from poLCA::poLCA().

Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in . . . , where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

Value

A tibble::tibble() with columns:

class The class under consideration.
outcome Outcome of manifest variable.

std.error The standard error of the regression term.

variable Manifest variable

estimate Estimated class-conditional response probability

See Also

```
tidy(), poLCA::poLCA()
Other poLCA tidiers: augment.poLCA(), glance.poLCA()
```

346 tidy.poLCA

```
# load libraries for models and data
library(poLCA)
library(dplyr)
# generate data
data(values)
f \leftarrow cbind(A, B, C, D) \sim 1
# fit model
M1 <- poLCA(f, values, nclass = 2, verbose = FALSE)
# summarize model fit with tidiers + visualization
tidy(M1)
augment(M1)
glance(M1)
library(ggplot2)
ggplot(tidy(M1), aes(factor(class), estimate, fill = factor(outcome))) +
  geom_bar(stat = "identity", width = 1) +
  facet_wrap(~variable)
# three-class model with a single covariate.
data(election)
f2a <- cbind(
  MORALG, CARESG, KNOWG, LEADG, DISHONG, INTELG,
  MORALB, CARESB, KNOWB, LEADB, DISHONB, INTELB
) ~ PARTY
nes2a <- poLCA(f2a, election, nclass = 3, nrep = 5, verbose = FALSE)</pre>
td <- tidy(nes2a)
td
ggplot(td, aes(outcome, estimate, color = factor(class), group = class)) +
  geom_line() +
  facet_wrap(\sim variable, nrow = 2) +
  theme(axis.text.x = element_text(angle = 90, hjust = 1))
au <- augment(nes2a)</pre>
au
count(au, .class)
# if the original data is provided, it leads to NAs in new columns
```

tidy.polr 347

```
# for rows that weren't predicted
au2 <- augment(nes2a, data = election)
au2
dim(au2)</pre>
```

tidy.polr

Tidy a(n) polr object

Description

Tidy summarizes information about the components of a model. A model component might be a single term in a regression, a single hypothesis, a cluster, or a class. Exactly what tidy considers to be a model component varies across models but is usually self-evident. If a model has several distinct types of components, you will need to specify which components to return.

Usage

```
## $3 method for class 'polr'
tidy(
    x,
    conf.int = FALSE,
    conf.level = 0.95,
    exponentiate = FALSE,
    p.values = FALSE,
    ...
)
```

Arguments

X	A polr object returned from MASS::polr().
conf.int	Logical indicating whether or not to include a confidence interval in the tidied output. Defaults to FALSE.
conf.level	The confidence level to use for the confidence interval if conf.int = TRUE. Must be strictly greater than 0 and less than 1. Defaults to 0.95, which corresponds to a 95 percent confidence interval.
exponentiate	Logical indicating whether or not to exponentiate the the coefficient estimates. This is typical for logistic and multinomial regressions, but a bad idea if there is no log or logit link. Defaults to FALSE.
p.values	Logical. Should p-values be returned, based on chi-squared tests from MASS::dropterm(). Defaults to FALSE.
	Additional arguments. Not used. Needed to match generic signature only. Cautionary note: Misspelled arguments will be absorbed in , where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

348 tidy.polr

• tidy() methods will warn when supplied an exponentiate argument if it will be ignored.

• augment() methods will warn when supplied a newdata argument if it will be ignored.

Details

In broom 0.7.0 the coefficient_type column was renamed to coef.type, and the contents were changed as well. Now the contents are coefficient and scale, rather than coefficient and zeta.

Calculating p.values with the dropterm() function is the approach suggested by the MASS package author. This approach is computationally intensive so that p.values are only returned if requested explicitly. Additionally, it only works for models containing no variables with more than two categories. If this condition is not met, a message is shown and NA is returned instead of p-values.

Value

A tibble::tibble() with columns:

conf.high	Upper bound on the confidence interval for the estimate.
conf.low	Lower bound on the confidence interval for the estimate.
estimate	The estimated value of the regression term.
p.value	The two-sided p-value associated with the observed statistic.
statistic	The value of a T-statistic to use in a hypothesis that the regression term is non-zero.
std.error	The standard error of the regression term.
term	The name of the regression term.

See Also

```
tidy, MASS::polr()
Other ordinal tidiers: augment.clm(), augment.polr(), glance.clm(), glance.clmm(), glance.polr(),
glance.svyolr(), tidy.clm(), tidy.clmm(), tidy.svyolr()
```

```
# load libraries for models and data
library(MASS)

# fit model
fit <- polr(Sat ~ Infl + Type + Cont, weights = Freq, data = housing)

# summarize model fit with tidiers
tidy(fit, exponentiate = TRUE, conf.int = TRUE)

glance(fit)
augment(fit, type.predict = "class")</pre>
```

tidy.power.htest 349

```
fit2 <- polr(factor(gear) ~ am + mpg + qsec, data = mtcars)
tidy(fit, p.values = TRUE)</pre>
```

tidy.power.htest

 $Tidy \ a(n) \ power.htest \ object$

Description

Tidy summarizes information about the components of a model. A model component might be a single term in a regression, a single hypothesis, a cluster, or a class. Exactly what tidy considers to be a model component varies across models but is usually self-evident. If a model has several distinct types of components, you will need to specify which components to return.

Usage

```
## S3 method for class 'power.htest'
tidy(x, ...)
```

Arguments

x A power.htest object such as those returned from stats::power.t.test().

Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in . . ., where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

Value

A tibble::tibble() with columns:

delta True difference in means.

Number of observations by component.powerPower achieved for given value of n.

sd Standard deviation.

sig.level Significance level (Type I error probability).

350 tidy.prcomp

See Also

```
stats::power.t.test()
Other htest tidiers: augment.htest(), tidy.htest(), tidy.pairwise.htest()
```

Examples

```
ptt <- power.t.test(n = 2:30, delta = 1)
tidy(ptt)
library(ggplot2)
ggplot(tidy(ptt), aes(n, power)) +
  geom_line()</pre>
```

tidy.prcomp

Tidy a(n) prcomp object

Description

Tidy summarizes information about the components of a model. A model component might be a single term in a regression, a single hypothesis, a cluster, or a class. Exactly what tidy considers to be a model component varies across models but is usually self-evident. If a model has several distinct types of components, you will need to specify which components to return.

Usage

```
## S3 method for class 'prcomp'
tidy(x, matrix = "u", ...)
```

Arguments

х

A prcomp object returned by stats::prcomp().

matrix

Character specifying which component of the PCA should be tidied.

- "u", "samples", "scores", or "x": returns information about the map from the original space into principle components space.
- "v", "rotation", "loadings" or "variables": returns information about the map from principle components space back into the original space.
- "d", "eigenvalues" or "pcs": returns information about the eigenvalues.

. . .

Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in . . . , where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

tidy.prcomp 351

• tidy() methods will warn when supplied an exponentiate argument if it will be ignored.

• augment() methods will warn when supplied a newdata argument if it will be ignored.

Details

See https://stats.stackexchange.com/questions/134282/relationship-between-svd-and-pca-how-to-use-svd-to-perform-pca for information on how to interpret the various tidied matrices. Note that SVD is only equivalent to PCA on centered data.

Value

A tibble::tibble with columns depending on the component of PCA being tidied.

If matrix is "u", "samples", "scores", or "x" each row in the tidied output corresponds to the original data in PCA space. The columns are:

row ID of the original observation (i.e. rowname from original data).

PC Integer indicating a principal component.

value The score of the observation for that particular principal component. That is, the

location of the observation in PCA space.

If matrix is "v", "rotation", "loadings" or "variables", each row in the tidied output corresponds to information about the principle components in the original space. The columns are:

row The variable labels (colnames) of the data set on which PCA was performed.

PC An integer vector indicating the principal component.

value The value of the eigenvector (axis score) on the indicated principal component.

If matrix is "d", "eigenvalues" or "pcs", the columns are:

PC An integer vector indicating the principal component.

std.dev Standard deviation explained by this PC.

percent Fraction of variation explained by this component (a numeric value between 0

and 1).

cumulative Cumulative fraction of variation explained by principle components up to this

component (a numeric value between 0 and 1).

See Also

```
stats::prcomp(), svd_tidiers
```

Other svd tidiers: augment.prcomp(), tidy_irlba(), tidy_svd()

352 tidy.pyears

Examples

```
pc <- prcomp(USArrests, scale = TRUE)</pre>
# information about rotation
tidy(pc)
# information about samples (states)
tidy(pc, "samples")
# information about PCs
tidy(pc, "pcs")
# state map
library(dplyr)
library(ggplot2)
library(maps)
pc %>%
  tidy(matrix = "samples") %>%
  mutate(region = tolower(row)) %>%
  inner_join(map_data("state"), by = "region") %>%
  ggplot(aes(long, lat, group = group, fill = value)) +
  geom_polygon() +
  facet_wrap(~PC) +
  theme_void() +
  ggtitle("Principal components of arrest data")
au <- augment(pc, data = USArrests)</pre>
au
ggplot(au, aes(.fittedPC1, .fittedPC2)) +
  geom_point() +
  geom_text(aes(label = .rownames), vjust = 1, hjust = 1)
```

tidy.pyears

Tidy a(n) pyears object

Description

Tidy summarizes information about the components of a model. A model component might be a single term in a regression, a single hypothesis, a cluster, or a class. Exactly what tidy considers to be a model component varies across models but is usually self-evident. If a model has several distinct types of components, you will need to specify which components to return.

Usage

```
## S3 method for class 'pyears' tidy(x, ...)
```

tidy.pyears 353

Arguments

. . .

x A pyears object returned from survival::pyears().

Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in . . ., where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

Details

expected is only present in the output when if a ratetable term is present.

If the data. frame = TRUE argument is supplied to pyears, this is simply the contents of x\$data.

Value

```
A tibble::tibble() with columns:
```

expected Expected number of events.

pyears Person-years of exposure.

n number of subjects contributing time

event observed number of events

See Also

```
tidy(), survival::pyears()
Other pyears tidiers: glance.pyears()
Other survival tidiers: augment.coxph(), augment.survreg(), glance.aareg(), glance.cch(),
glance.coxph(), glance.pyears(), glance.survdiff(), glance.survexp(), glance.survfit(),
glance.survreg(), tidy.aareg(), tidy.cch(), tidy.coxph(), tidy.survdiff(), tidy.survexp(),
tidy.survfit(), tidy.survreg()
```

```
# load libraries for models and data
library(survival)

# generate and format data
temp.yr <- tcut(mgus$dxyr, 55:92, labels = as.character(55:91))
temp.age <- tcut(mgus$age, 34:101, labels = as.character(34:100))
ptime <- ifelse(is.na(mgus$pctime), mgus$futime, mgus$pctime)
pstat <- ifelse(is.na(mgus$pctime), 0, 1)
pfit <- pyears(Surv(ptime / 365.25, pstat) ~ temp.yr + temp.age + sex, mgus,</pre>
```

354 tidy.rcorr

```
data.frame = TRUE
)

# summarize model fit with tidiers
tidy(pfit)
glance(pfit)

# if data.frame argument is not given, different information is present in
# output
pfit2 <- pyears(Surv(ptime / 365.25, pstat) ~ temp.yr + temp.age + sex, mgus)
tidy(pfit2)
glance(pfit2)</pre>
```

tidy.rcorr

Tidy a(n) rcorr object

Description

Tidy summarizes information about the components of a model. A model component might be a single term in a regression, a single hypothesis, a cluster, or a class. Exactly what tidy considers to be a model component varies across models but is usually self-evident. If a model has several distinct types of components, you will need to specify which components to return.

Usage

```
## S3 method for class 'rcorr'
tidy(x, diagonal = FALSE, ...)
```

Arguments

Y

An rcorr object returned from Hmisc::rcorr().

diagonal

Logical indicating whether or not to include diagonal elements of the correlation matrix, or the correlation of a column with itself. For the elements, estimate is always 1 and p.value is always NA. Defaults to FALSE.

. . .

Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in . . . , where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

tidy.rcorr 355

Details

Suppose the original data has columns A and B. In the correlation matrix from rcorr there may be entries for both the cor(A, B) and cor(B, A). Only one of these pairs will ever be present in the tidy output.

Value

A tibble::tibble() with columns:

column1	Name or index of the first column being described.
column2	Name or index of the second column being described.
estimate	The estimated value of the regression term.
p.value	The two-sided p-value associated with the observed statistic.
n	Number of observations used to compute the correlation

See Also

```
tidy(), Hmisc::rcorr()
```

```
# load libraries for models and data
library(Hmisc)
mat <- replicate(52, rnorm(100))</pre>
# add some NAs
mat[sample(length(mat), 2000)] <- NA</pre>
# also, column names
colnames(mat) <- c(LETTERS, letters)</pre>
# fit model
rc <- rcorr(mat)</pre>
# summarize model fit with tidiers + visualization
td <- tidy(rc)
td
library(ggplot2)
ggplot(td, aes(p.value)) +
  geom\_histogram(binwidth = .1)
ggplot(td, aes(estimate, p.value)) +
  geom_point() +
  scale_y_log10()
```

356 tidy.ref.grid

tray it (ii) tray a tra	tidy.ref.grid	Tidy a(n) ref.grid object	
--	---------------	---------------------------	--

Description

Tidy summarizes information about the components of a model. A model component might be a single term in a regression, a single hypothesis, a cluster, or a class. Exactly what tidy considers to be a model component varies across models but is usually self-evident. If a model has several distinct types of components, you will need to specify which components to return.

Usage

```
## S3 method for class 'ref.grid'
tidy(x, conf.int = FALSE, conf.level = 0.95, ...)
```

Arguments

X	A ref.grid object created by emmeans::ref_grid().
conf.int	Logical indicating whether or not to include a confidence interval in the tidied output. Defaults to FALSE.
conf.level	The confidence level to use for the confidence interval if conf.int = TRUE. Must be strictly greater than 0 and less than 1. Defaults to 0.95, which corresponds to a 95 percent confidence interval.
• • •	Additional arguments passed to emmeans::summary.emmGrid() or lsmeans::summary.ref.grid(). Cautionary note: misspecified arguments may be silently ignored!

Details

Returns a data frame with one observation for each estimated marginal mean, and one column for each combination of factors. When the input is a contrast, each row will contain one estimated contrast.

There are a large number of arguments that can be passed on to emmeans::summary.emmGrid() or lsmeans::summary.ref.grid().

Value

A tibble::tibble() with columns:

conf.high	Upper bound on the confidence interval for the estimate.
conf.low	Lower bound on the confidence interval for the estimate.
df	Degrees of freedom used by this term in the model.
p.value	The two-sided p-value associated with the observed statistic.
std.error	The standard error of the regression term.
estimate	Expected marginal mean
statistic	T-ratio statistic

tidy.ref.grid 357

See Also

```
tidy(), emmeans::ref_grid(), emmeans::emmeans(), emmeans::contrast()
Other emmeans tidiers: tidy.emmGrid(), tidy.lsmobj(), tidy.summary_emm()
```

```
# load libraries for models and data
library(emmeans)
# linear model for sales of oranges per day
oranges_lm1 <- lm(sales1 ~ price1 + price2 + day + store, data = oranges)</pre>
# reference grid; see vignette("basics", package = "emmeans")
oranges_rg1 <- ref_grid(oranges_lm1)</pre>
td <- tidy(oranges_rg1)</pre>
td
# marginal averages
marginal <- emmeans(oranges_rg1, "day")</pre>
tidy(marginal)
# contrasts
tidy(contrast(marginal))
tidy(contrast(marginal, method = "pairwise"))
# plot confidence intervals
library(ggplot2)
ggplot(tidy(marginal, conf.int = TRUE), aes(day, estimate)) +
  geom_point() +
  geom_errorbar(aes(ymin = conf.low, ymax = conf.high))
# by multiple prices
by_price <- emmeans(oranges_lm1, "day",</pre>
  by = "price2",
 at = list(
    price1 = 50, price2 = c(40, 60, 80),
    day = c("2", "3", "4")
  )
)
by_price
tidy(by_price)
ggplot(tidy(by_price, conf.int = TRUE), aes(price2, estimate, color = day)) +
  geom_line() +
  geom_errorbar(aes(ymin = conf.low, ymax = conf.high))
# joint_tests
tidy(joint_tests(oranges_lm1))
```

358 tidy.regsubsets

tidy.regsubsets

Tidy a(n) regsubsets object

Description

Tidy summarizes information about the components of a model. A model component might be a single term in a regression, a single hypothesis, a cluster, or a class. Exactly what tidy considers to be a model component varies across models but is usually self-evident. If a model has several distinct types of components, you will need to specify which components to return.

Usage

```
## S3 method for class 'regsubsets' tidy(x, ...)
```

Arguments

Χ

0

A regsubsets object created by leaps::regsubsets().

Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in . . . , where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

Value

A tibble::tibble() with columns:

r.squared R squared statistic, or the percent of variation explained by the model.

adj.r.squared Adjusted R squared statistic

BIC Bayesian information criterion for the component.

mallows_cp Mallow's Cp statistic.

See Also

```
tidy(), leaps::regsubsets()
```

tidy.ridgelm 359

Examples

```
# load libraries for models and data
library(leaps)

# fit model
all_fits <- regsubsets(hp ~ ., mtcars)

# summarize model fit with tidiers
tidy(all_fits)</pre>
```

tidy.ridgelm

Tidy a(n) ridgelm object

Description

Tidy summarizes information about the components of a model. A model component might be a single term in a regression, a single hypothesis, a cluster, or a class. Exactly what tidy considers to be a model component varies across models but is usually self-evident. If a model has several distinct types of components, you will need to specify which components to return.

Usage

```
## S3 method for class 'ridgelm'
tidy(x, ...)
```

Arguments

. . .

x A ridgelm object returned from MASS::lm.ridge().

Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in . . ., where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

Value

A tibble::tibble() with columns:

GCV Generalized cross validation error estimate.

lambda Value of penalty parameter lambda. term The name of the regression term. 360 tidy.rlm

estimate estimate of scaled coefficient using this lambda scale Scaling factor of estimated coefficient

See Also

```
tidy(), MASS::lm.ridge()
Other ridgelm tidiers: glance.ridgelm()
```

Examples

```
# load libraries for models and data
library(MASS)
names(longley)[1] <- "y"</pre>
# fit model and summarizd results
fit1 <- lm.ridge(y ~ ., longley)</pre>
tidy(fit1)
fit2 <- lm.ridge(y \sim ., longley, lambda = seq(0.001, .05, .001))
td2 <- tidy(fit2)
g2 <- glance(fit2)
# coefficient plot
library(ggplot2)
ggplot(td2, aes(lambda, estimate, color = term)) +
 geom_line()
# GCV plot
ggplot(td2, aes(lambda, GCV)) +
 geom_line()
# add line for the GCV minimizing estimate
ggplot(td2, aes(lambda, GCV)) +
 geom_line() +
 geom_vline(xintercept = g2$lambdaGCV, col = "red", lty = 2)
```

tidy.rlm

Tidy a(n) rlm object

Description

Tidy summarizes information about the components of a model. A model component might be a single term in a regression, a single hypothesis, a cluster, or a class. Exactly what tidy considers to be a model component varies across models but is usually self-evident. If a model has several distinct types of components, you will need to specify which components to return.

tidy.rma 361

Usage

```
## S3 method for class 'rlm'
tidy(x, conf.int = FALSE, conf.level = 0.95, ...)
```

Arguments

x An rlm object returned by MASS::rlm().

conf.int Logical indicating whether or not to include a confidence interval in the tidied

output. Defaults to FALSE.

conf.level The confidence level to use for the confidence interval if conf.int = TRUE. Must

be strictly greater than 0 and less than 1. Defaults to 0.95, which corresponds to

a 95 percent confidence interval.

Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in . . . , where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

See Also

```
MASS::rlm()
Other rlm tidiers: augment.rlm(), glance.rlm()
```

tidy.rma

Tidy a(n) rma object

Description

Tidy summarizes information about the components of a model. A model component might be a single term in a regression, a single hypothesis, a cluster, or a class. Exactly what tidy considers to be a model component varies across models but is usually self-evident. If a model has several distinct types of components, you will need to specify which components to return.

Usage

```
## S3 method for class 'rma'
tidy(
    x,
    conf.int = FALSE,
    conf.level = 0.95,
    exponentiate = FALSE,
```

362 tidy.rma

```
include_studies = FALSE,
  measure = "GEN",
    ...
)
```

Arguments

x An rma object such as those created by metafor::rma(), metafor::rma.uni(),

metafor::rma.glmm(), metafor::rma.mh(), metafor::rma.mv(), or metafor::rma.peto().

conf.int Logical indicating whether or not to include a confidence interval in the tidied

output. Defaults to FALSE.

conf.level The confidence level to use for the confidence interval if conf.int = TRUE. Must

be strictly greater than 0 and less than 1. Defaults to 0.95, which corresponds to

a 95 percent confidence interval.

exponentiate Logical indicating whether or not to exponentiate the the coefficient estimates.

This is typical for logistic and multinomial regressions, but a bad idea if there is

no log or logit link. Defaults to FALSE.

include_studies

Logical. Should individual studies be included in the output? Defaults to FALSE.

measure Measure type. See metafor::escalc()

Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in . . ., where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed

using conf.level = 0.95. Two exceptions here are:

• tidy() methods will warn when supplied an exponentiate argument if it

will be ignored.

• augment() methods will warn when supplied a newdata argument if it will

be ignored.

Value

conf.high

A tibble::tibble() with columns:

conf.low Lower bound on the confidence interval for the estimate.

estimate The estimated value of the regression term.

p.value The two-sided p-value associated with the observed statistic.

statistic The value of a T-statistic to use in a hypothesis that the regression term is non-

The value of a 1-statistic to use in a hypothesis that the regression term is non-

Upper bound on the confidence interval for the estimate.

zero.

std.error The standard error of the regression term.

term The name of the individual study

type The estimate type (summary vs individual study)

tidy.roc 363

Examples

```
# load libraries for models and data
library(metafor)

df <-
    escalc(
        measure = "RR",
        ai = tpos,
        bi = tneg,
        ci = cpos,
        di = cneg,
        data = dat.bcg
)

meta_analysis <- rma(yi, vi, data = df, method = "EB")

tidy(meta_analysis)</pre>
```

tidy.roc

Tidy a(n) roc object

Description

Tidy summarizes information about the components of a model. A model component might be a single term in a regression, a single hypothesis, a cluster, or a class. Exactly what tidy considers to be a model component varies across models but is usually self-evident. If a model has several distinct types of components, you will need to specify which components to return.

Usage

```
## S3 method for class 'roc'
tidy(x, ...)
```

Arguments

x

An roc object returned from a call to AUC::roc().

. . .

Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in . . . , where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

364 tidy.roc

Value

A tibble::tibble() with columns:

cutoff The cutoff used for classification. Observations with predicted probabilities

above this value were assigned class 1, and observations with predicted prob-

abilities below this value were assigned class 0.

fpr False positive rate.

tpr The true positive rate at the given cutoff.

See Also

```
tidy(), AUC::roc()
```

```
# load libraries for models and data
library(AUC)
# load data
data(churn)
# fit model
r <- roc(churn$predictions, churn$labels)</pre>
# summarize with tidiers + visualization
td <- tidy(r)
td
library(ggplot2)
ggplot(td, aes(fpr, tpr)) +
  geom_line()
# compare the ROC curves for two prediction algorithms
library(dplyr)
library(tidyr)
rocs <- churn %>%
  pivot_longer(contains("predictions"),
   names_to = "algorithm",
    values_to = "value"
  nest(data = -algorithm) %>%
  mutate(tidy_roc = purrr::map(data, ~ tidy(roc(.x$value, .x$labels)))) %>%
  unnest(tidy_roc)
ggplot(rocs, aes(fpr, tpr, color = algorithm)) +
  geom_line()
```

tidy.rq 365

tidy.rq	Tidy a(n) rq object	

Description

Tidy summarizes information about the components of a model. A model component might be a single term in a regression, a single hypothesis, a cluster, or a class. Exactly what tidy considers to be a model component varies across models but is usually self-evident. If a model has several distinct types of components, you will need to specify which components to return.

Usage

```
## S3 method for class 'rq'
tidy(x, se.type = NULL, conf.int = FALSE, conf.level = 0.95, ...)
```

Arguments

X	An rq object returned from quantreg::rq().
se.type	Character specifying the method to use to calculate standard errors. Passed to quantreg::summary.rq() se argument. Defaults to "rank" if the sample size is less than 1000, otherwise defaults to "nid".
conf.int	Logical indicating whether or not to include a confidence interval in the tidied output. Defaults to FALSE.
conf.level	The confidence level to use for the confidence interval if conf.int = TRUE. Must be strictly greater than 0 and less than 1. Defaults to 0.95, which corresponds to a 95 percent confidence interval.
	Additional arguments passed to quantreg::summary.rq().

Details

If se.type = "rank" confidence intervals are calculated by summary.rq and statistic and p.value values are not returned. When only a single predictor is included in the model, no confidence intervals are calculated and the confidence limits are set to NA.

Value

A tibble::tibble() with columns:

conf.high	Upper bound on the confidence interval for the estimate.
conf.low	Lower bound on the confidence interval for the estimate.
estimate	The estimated value of the regression term.
p.value	The two-sided p-value associated with the observed statistic.
statistic	The value of a T-statistic to use in a hypothesis that the regression term is non-zero.
std.error	The standard error of the regression term.
term	The name of the regression term.

366 tidy.rqs

See Also

```
tidy(), quantreg::rq()
Other quantreg tidiers: augment.nlrq(), augment.rq(), augment.rqs(), glance.nlrq(), glance.rq(),
tidy.nlrq(), tidy.rqs()
```

Examples

```
# load modeling library and data
library(quantreg)
data(stackloss)
# median (11) regression fit for the stackloss data.
mod1 <- rq(stack.loss ~ stack.x, .5)</pre>
# weighted sample median
mod2 \leftarrow rq(rnorm(50) \sim 1, weights = runif(50))
# summarize model fit with tidiers
tidy(mod1)
glance(mod1)
augment(mod1)
tidy(mod2)
glance(mod2)
augment(mod2)
# varying tau to generate an rqs object
mod3 \leftarrow rq(stack.loss \sim stack.x, tau = c(.25, .5))
tidy(mod3)
augment(mod3)
# glance cannot handle rqs objects like `mod3`--use a purrr
# `map`-based workflow instead
```

tidy.rqs

Tidy a(n) rqs object

Description

Tidy summarizes information about the components of a model. A model component might be a single term in a regression, a single hypothesis, a cluster, or a class. Exactly what tidy considers to be a model component varies across models but is usually self-evident. If a model has several distinct types of components, you will need to specify which components to return.

tidy.rqs 367

Usage

```
## S3 method for class 'rqs'
tidy(x, se.type = "rank", conf.int = FALSE, conf.level = 0.95, ...)
```

Arguments

x	An rqs object returned from quantreg::rq().
se.type	Character specifying the method to use to calculate standard errors. Passed to quantreg::summary.rq() se argument. Defaults to "rank".
conf.int	Logical indicating whether or not to include a confidence interval in the tidied output. Defaults to FALSE.
conf.level	The confidence level to use for the confidence interval if conf.int = TRUE. Must be strictly greater than 0 and less than 1. Defaults to 0.95, which corresponds to a 95 percent confidence interval.
	Additional arguments passed to quantreg::summary.rqs()

Details

If se.type = "rank" confidence intervals are calculated by summary.rq. When only a single predictor is included in the model, no confidence intervals are calculated and the confidence limits are set to NA.

Value

A tibble::tibble() with columns:

conf.high	Upper bound on the confidence interval for the estimate.
conf.low	Lower bound on the confidence interval for the estimate.
estimate	The estimated value of the regression term.
p.value	The two-sided p-value associated with the observed statistic.
statistic	The value of a T-statistic to use in a hypothesis that the regression term is non-zero.
std.error	The standard error of the regression term.
term	The name of the regression term.
quantile	Linear conditional quantile.

See Also

```
tidy(), quantreg::rq()
Other quantreg tidiers: augment.nlrq(), augment.rq(), augment.rqs(), glance.nlrq(), glance.rq(),
tidy.nlrq(), tidy.rq()
```

368 tidy.sarlm

Examples

```
# load modeling library and data
library(quantreg)
data(stackloss)
# median (11) regression fit for the stackloss data.
mod1 <- rq(stack.loss ~ stack.x, .5)</pre>
# weighted sample median
mod2 <- rq(rnorm(50) \sim 1, weights = runif(50))
# summarize model fit with tidiers
tidy(mod1)
glance(mod1)
augment(mod1)
tidy(mod2)
glance(mod2)
augment(mod2)
# varying tau to generate an rqs object
mod3 \leftarrow rq(stack.loss \sim stack.x, tau = c(.25, .5))
tidy(mod3)
augment(mod3)
# glance cannot handle rqs objects like `mod3`--use a purrr
# `map`-based workflow instead
```

tidy.sarlm

Tidying methods for spatially autoregressive models

Description

These methods tidy the coefficients of spatial autoregression models generated by functions in the spatialreg package.

Usage

```
## S3 method for class 'sarlm'
tidy(x, conf.int = FALSE, conf.level = 0.95, ...)
```

Arguments

x An object returned from spatialreg::lagsarlm() or spatialreg::errorsarlm().

369 tidy.sarlm

conf.int Logical indicating whether or not to include a confidence interval in the tidied output. Defaults to FALSE. conf.level The confidence level to use for the confidence interval if conf.int = TRUE. Must be strictly greater than 0 and less than 1. Defaults to 0.95, which corresponds to a 95 percent confidence interval. Additional arguments. Not used. Needed to match generic signature only. Cautionary note: Misspelled arguments will be absorbed in . . ., where they will be ignored. If the misspelled argument has a default value, the default value will be

used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

Value

A tibble::tibble() with columns:

conf.high Upper bound on the confidence interval for the estimate. conf.low Lower bound on the confidence interval for the estimate. estimate The estimated value of the regression term. p.value The two-sided p-value associated with the observed statistic. statistic The value of a T-statistic to use in a hypothesis that the regression term is nonstd.error The standard error of the regression term. term The name of the regression term.

See Also

```
tidy(), spatialreg::lagsarlm(), spatialreg::errorsarlm(), spatialreg::sacsarlm()
Other spatialreg tidiers: augment.sarlm(), glance.sarlm()
```

```
# load libraries for models and data
library(spatialreg)
library(spdep)
# load data
data(oldcol, package = "spdep")
listw <- nb2listw(COL.nb, style = "W")</pre>
# fit model
```

370 tidy.spec

```
crime_sar <-</pre>
  lagsarlm(CRIME ~ INC + HOVAL,
    data = COL.OLD,
    listw = listw,
    method = "eigen"
  )
# summarize model fit with tidiers
tidy(crime_sar)
tidy(crime_sar, conf.int = TRUE)
glance(crime_sar)
augment(crime_sar)
# fit another model
crime_sem <- errorsarlm(CRIME ~ INC + HOVAL, data = COL.OLD, listw)</pre>
# summarize model fit with tidiers
tidy(crime_sem)
tidy(crime_sem, conf.int = TRUE)
glance(crime_sem)
augment(crime_sem)
# fit another model
crime_sac <- sacsarlm(CRIME ~ INC + HOVAL, data = COL.OLD, listw)</pre>
# summarize model fit with tidiers
tidy(crime_sac)
tidy(crime_sac, conf.int = TRUE)
glance(crime_sac)
augment(crime_sac)
```

tidy.spec

Tidy a(n) spec object

Description

Tidy summarizes information about the components of a model. A model component might be a single term in a regression, a single hypothesis, a cluster, or a class. Exactly what tidy considers to be a model component varies across models but is usually self-evident. If a model has several distinct types of components, you will need to specify which components to return.

Usage

```
## S3 method for class 'spec' tidy(x, ...)
```

tidy.speedglm 371

Arguments

x A spec object created by stats::spectrum().

Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in . . . , where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

Value

A tibble::tibble() with columns:

freq Vector of frequencies at which the spectral density is estimated.

spec Vector (for univariate series) or matrix (for multivariate series) of estimates of

the spectral density at frequencies corresponding to freq.

See Also

```
tidy(), stats::spectrum()
Other time series tidiers: tidy.acf(), tidy.ts(), tidy.zoo()
```

Examples

```
spc <- spectrum(lh)
tidy(spc)

library(ggplot2)
ggplot(tidy(spc), aes(freq, spec)) +
  geom_line()</pre>
```

tidy.speedglm

Tidy a(n) speedglm object

Description

Tidy summarizes information about the components of a model. A model component might be a single term in a regression, a single hypothesis, a cluster, or a class. Exactly what tidy considers to be a model component varies across models but is usually self-evident. If a model has several distinct types of components, you will need to specify which components to return.

372 tidy.speedglm

Usage

```
## S3 method for class 'speedglm'
tidy(x, conf.int = FALSE, conf.level = 0.95, exponentiate = FALSE, ...)
```

Arguments

A speedglm object returned from speedglm::speedglm(). Χ conf.int Logical indicating whether or not to include a confidence interval in the tidied output. Defaults to FALSE. conf.level The confidence level to use for the confidence interval if conf. int = TRUE. Must be strictly greater than 0 and less than 1. Defaults to 0.95, which corresponds to a 95 percent confidence interval. Logical indicating whether or not to exponentiate the the coefficient estimates. exponentiate This is typical for logistic and multinomial regressions, but a bad idea if there is no log or logit link. Defaults to FALSE. Additional arguments. Not used. Needed to match generic signature only. Cautionary note: Misspelled arguments will be absorbed in . . . , where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

Value

A tibble::tibble() with columns:

conf.high	Upper bound on the confidence interval for the estimate.
conf.low	Lower bound on the confidence interval for the estimate.
estimate	The estimated value of the regression term.
p.value	The two-sided p-value associated with the observed statistic.
statistic	The value of a T-statistic to use in a hypothesis that the regression term is non-zero.
std.error	The standard error of the regression term.
term	The name of the regression term.

See Also

```
speedglm::speedglm()
Other speedlm tidiers: augment.speedlm(), glance.speedglm(), glance.speedlm(), tidy.speedlm()
```

tidy.speedlm 373

Examples

```
# load libraries for models and data
library(speedglm)

# generate data
clotting <- data.frame(
    u = c(5, 10, 15, 20, 30, 40, 60, 80, 100),
    lot1 = c(118, 58, 42, 35, 27, 25, 21, 19, 18)
)

# fit model
fit <- speedglm(lot1 ~ log(u), data = clotting, family = Gamma(log))

# summarize model fit with tidiers
tidy(fit)
glance(fit)</pre>
```

tidy.speedlm

Tidy a(n) speedlm object

Description

Tidy summarizes information about the components of a model. A model component might be a single term in a regression, a single hypothesis, a cluster, or a class. Exactly what tidy considers to be a model component varies across models but is usually self-evident. If a model has several distinct types of components, you will need to specify which components to return.

Usage

```
## S3 method for class 'speedlm'
tidy(x, conf.int = FALSE, conf.level = 0.95, ...)
```

Arguments

Х	A speedlm object returned from speedglm::speedlm().
conf.int	Logical indicating whether or not to include a confidence interval in the tidied output. Defaults to FALSE.
conf.level	The confidence level to use for the confidence interval if conf.int = TRUE. Must be strictly greater than 0 and less than 1. Defaults to 0.95, which corresponds to a 95 percent confidence interval.
	Additional arguments. Not used. Needed to match generic signature only. Cautionary note: Misspelled arguments will be absorbed in , where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed

using conf.level = 0.95. Two exceptions here are:

374 tidy.summary.glht

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

Value

A tibble::tibble() with columns:

```
conf.high Upper bound on the confidence interval for the estimate.

Lower bound on the confidence interval for the estimate.

The estimate of the regression term.

The two-sided p-value associated with the observed statistic.

The value of a T-statistic to use in a hypothesis that the regression term is non-zero.

The standard error of the regression term.

The name of the regression term.
```

See Also

```
speedglm::speedlm(), tidy.lm()
Other speedlm tidiers: augment.speedlm(), glance.speedglm(), glance.speedlm(), tidy.speedglm()
```

Examples

```
# load modeling library
library(speedglm)

# fit model
mod <- speedlm(mpg ~ wt + qsec, data = mtcars, fitted = TRUE)

# summarize model fit with tidiers
tidy(mod)
glance(mod)
augment(mod)</pre>
```

tidy.summary.glht

Tidy a(n) summary.glht object

Description

Tidy summarizes information about the components of a model. A model component might be a single term in a regression, a single hypothesis, a cluster, or a class. Exactly what tidy considers to be a model component varies across models but is usually self-evident. If a model has several distinct types of components, you will need to specify which components to return.

tidy.summary.glht 375

Usage

```
## S3 method for class 'summary.glht'
tidy(x, ...)
```

Arguments

A summary.glht object created by calling multcomp::summary.glht() on a glht object created with multcomp::glht().

Additional arguments. Not used. Needed to match generic signature only. Cautionary note: Misspelled arguments will be absorbed in . . ., where they will be ignored. If the misspelled argument has a default value, the default value will be

used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

• tidy() methods will warn when supplied an exponentiate argument if it will be ignored.

 augment() methods will warn when supplied a newdata argument if it will be ignored.

Value

A tibble::tibble() with columns:

contrast Levels being compared.

estimate The estimated value of the regression term.

null.value Value to which the estimate is compared.

p.value The two-sided p-value associated with the observed statistic.

statistic The value of a T-statistic to use in a hypothesis that the regression term is non-zero.

std.error The standard error of the regression term.

See Also

```
tidy(), multcomp::summary.glht(), multcomp::glht()
Other multcomp tidiers: tidy.cld(), tidy.confint.glht(), tidy.glht()
```

```
# load libraries for models and data
library(multcomp)
library(ggplot2)

amod <- aov(breaks ~ wool + tension, data = warpbreaks)
wht <- glht(amod, linfct = mcp(tension = "Tukey"))
tidy(wht)</pre>
```

376 tidy.summary.lm

```
ggplot(wht, aes(lhs, estimate)) +
    geom_point()

CI <- confint(wht)

tidy(CI)

ggplot(CI, aes(lhs, estimate, ymin = lwr, ymax = upr)) +
    geom_pointrange()

tidy(summary(wht))
ggplot(mapping = aes(lhs, estimate)) +
    geom_linerange(aes(ymin = lwr, ymax = upr), data = CI) +
    geom_point(aes(size = p), data = summary(wht)) +
    scale_size(trans = "reverse")

cld <- cld(wht)
tidy(cld)</pre>
```

tidy.summary.lm

Tidy a(n) summary.lm object

Description

Tidy summarizes information about the components of a model. A model component might be a single term in a regression, a single hypothesis, a cluster, or a class. Exactly what tidy considers to be a model component varies across models but is usually self-evident. If a model has several distinct types of components, you will need to specify which components to return.

Usage

```
## S3 method for class 'summary.lm'
tidy(x, conf.int = FALSE, conf.level = 0.95, ...)
```

Arguments

X	A summary.lm object created by stats::summary.lm().
conf.int	Logical indicating whether or not to include a confidence interval in the tidied output. Defaults to FALSE.
conf.level	The confidence level to use for the confidence interval if conf.int = TRUE. Must be strictly greater than 0 and less than 1. Defaults to 0.95, which corresponds to a 95 percent confidence interval.
	Additional arguments. Not used. Needed to match generic signature only. Cautionary note: Misspelled arguments will be absorbed in , where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

tidy.summary.lm 377

• tidy() methods will warn when supplied an exponentiate argument if it will be ignored.

• augment() methods will warn when supplied a newdata argument if it will be ignored.

Details

The tidy.summary.lm() method is a potentially useful alternative to tidy.lm(). For instance, if users have already converted large lm objects into their leaner summary.lm equivalents to conserve memory.

Value

A tibble::tibble() with columns:

conf.high	Upper bound on the confidence interval for the estimate.
conf.low	Lower bound on the confidence interval for the estimate.
estimate	The estimated value of the regression term.
p.value	The two-sided p-value associated with the observed statistic.
statistic	The value of a T-statistic to use in a hypothesis that the regression term is non-zero.
std.error	The standard error of the regression term.
term	The name of the regression term.

See Also

```
tidy(), stats::summary.lm()
Other Im tidiers: augment.glm(), augment.lm(), glance.glm(), glance.lm(), glance.summary.lm(),
glance.svyglm(), tidy.glm(), tidy.lm(), tidy.lm.beta(), tidy.mlm()
```

```
# fit model
mod <- lm(mpg ~ wt + qsec, data = mtcars)
modsumm <- summary(mod)

# summarize model fit with tidiers
tidy(mod, conf.int = TRUE)

# equivalent to the above
tidy(modsumm, conf.int = TRUE)
glance(mod)

# mostly the same, except for a few missing columns
glance(modsumm)</pre>
```

378 tidy.summary_emm

object

Description

Tidy summarizes information about the components of a model. A model component might be a single term in a regression, a single hypothesis, a cluster, or a class. Exactly what tidy considers to be a model component varies across models but is usually self-evident. If a model has several distinct types of components, you will need to specify which components to return.

Usage

```
## S3 method for class 'summary_emm'
tidy(x, null.value = NULL, ...)
```

Arguments

```
A summary_emm object.
Х
null.value
                 Value to which estimate is compared.
                 Additional arguments passed to emmeans::summary.emmGrid() or lsmeans::summary.ref.grid().
                 Cautionary note: misspecified arguments may be silently ignored!
```

Details

Returns a data frame with one observation for each estimated marginal mean, and one column for each combination of factors. When the input is a contrast, each row will contain one estimated contrast.

There are a large number of arguments that can be passed on to emmeans::summary.emmGrid() or lsmeans::summary.ref.grid().

Value

A tibble::tibble() with columns:

conf.high Upper bound on the confidence interval for the estimate. conf.low Lower bound on the confidence interval for the estimate. contrast Levels being compared. den.df Degrees of freedom of the denominator. df Degrees of freedom used by this term in the model. null.value Value to which the estimate is compared. num.df Degrees of freedom.

p.value The two-sided p-value associated with the observed statistic.

std.error The standard error of the regression term. level1 One level of the factor being contrasted

tidy.summary_emm 379

```
level2The other level of the factor being contrastedtermModel term in joint testsestimateExpected marginal meanstatisticT-ratio statistic or F-ratio statistic
```

See Also

```
tidy(), emmeans::ref_grid(), emmeans::emmeans(), emmeans::contrast()
Other emmeans tidiers: tidy.emmGrid(), tidy.lsmobj(), tidy.ref.grid()
```

```
# load libraries for models and data
library(emmeans)
# linear model for sales of oranges per day
oranges_lm1 <- lm(sales1 ~ price1 + price2 + day + store, data = oranges)</pre>
# reference grid; see vignette("basics", package = "emmeans")
oranges_rg1 <- ref_grid(oranges_lm1)</pre>
td <- tidy(oranges_rg1)</pre>
td
# marginal averages
marginal <- emmeans(oranges_rg1, "day")</pre>
tidy(marginal)
# contrasts
tidy(contrast(marginal))
tidy(contrast(marginal, method = "pairwise"))
# plot confidence intervals
library(ggplot2)
ggplot(tidy(marginal, conf.int = TRUE), aes(day, estimate)) +
  geom_point() +
  geom_errorbar(aes(ymin = conf.low, ymax = conf.high))
# by multiple prices
by_price <- emmeans(oranges_lm1, "day",</pre>
  by = "price2",
  at = list(
    price1 = 50, price2 = c(40, 60, 80),
    day = c("2", "3", "4")
by_price
tidy(by_price)
```

380 tidy.survdiff

```
ggplot(tidy(by_price, conf.int = TRUE), aes(price2, estimate, color = day)) +
  geom_line() +
  geom_errorbar(aes(ymin = conf.low, ymax = conf.high))
# joint_tests
tidy(joint_tests(oranges_lm1))
```

tidy.survdiff

Tidy a(n) survdiff object

Description

Tidy summarizes information about the components of a model. A model component might be a single term in a regression, a single hypothesis, a cluster, or a class. Exactly what tidy considers to be a model component varies across models but is usually self-evident. If a model has several distinct types of components, you will need to specify which components to return.

Usage

```
## S3 method for class 'survdiff' tidy(x, ...)
```

Arguments

x An survdiff object returned from survival::survdiff().

Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in . . . , where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

Value

obs

A tibble::tibble() with columns:

exp Weighted expected number of events in each group.

N Number of subjects in each group.

weighted observed number of events in each group.

tidy.survexp 381

See Also

```
tidy(), survival::survdiff()
Other survdiff tidiers: glance.survdiff()
Other survival tidiers: augment.coxph(), augment.survreg(), glance.aareg(), glance.cch(),
glance.coxph(), glance.pyears(), glance.survdiff(), glance.survexp(), glance.survfit(),
glance.survreg(), tidy.aareg(), tidy.cch(), tidy.coxph(), tidy.pyears(), tidy.survexp(),
tidy.survfit(), tidy.survreg()
```

Examples

```
# load libraries for models and data
library(survival)

# fit model
s <- survdiff(
   Surv(time, status) ~ pat.karno + strata(inst),
   data = lung
)

# summarize model fit with tidiers
tidy(s)
glance(s)</pre>
```

tidy.survexp

Tidy a(n) survexp object

Description

Tidy summarizes information about the components of a model. A model component might be a single term in a regression, a single hypothesis, a cluster, or a class. Exactly what tidy considers to be a model component varies across models but is usually self-evident. If a model has several distinct types of components, you will need to specify which components to return.

Usage

```
## S3 method for class 'survexp'
tidy(x, ...)
```

Arguments

x An survexp object returned from survival::survexp().

Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in . . . , where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

382 tidy.survexp

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

Value

A tibble::tibble() with columns:

n.risk Number of individuals at risk at time zero.

time Point in time.
estimate Estimate survival

See Also

```
tidy(), survival::survexp()
Other survexp tidiers: glance.survexp()
Other survival tidiers: augment.coxph(), augment.survreg(), glance.aareg(), glance.cch(),
glance.coxph(), glance.pyears(), glance.survdiff(), glance.survexp(), glance.survfit(),
glance.survreg(), tidy.aareg(), tidy.cch(), tidy.coxph(), tidy.pyears(), tidy.survdiff(),
tidy.survfit(), tidy.survreg()
```

```
# load libraries for models and data
library(survival)

# fit model
sexpfit <- survexp(
  futime ~ 1,
  rmap = list(
    sex = "male",
    year = accept.dt,
    age = (accept.dt - birth.dt)
  ),
  method = "conditional",
  data = jasa
)

# summarize model fit with tidiers
tidy(sexpfit)
glance(sexpfit)</pre>
```

tidy.survfit 383

Description

Tidy summarizes information about the components of a model. A model component might be a single term in a regression, a single hypothesis, a cluster, or a class. Exactly what tidy considers to be a model component varies across models but is usually self-evident. If a model has several distinct types of components, you will need to specify which components to return.

Usage

```
## S3 method for class 'survfit'
tidy(x, ...)
```

Arguments

x An survfit object returned from survival::survfit().

For glance.survfit(), additional arguments passed to summary(). Otherwise ignored.

Value

```
A tibble::tibble() with columns:
```

conf.high Upper bound on the confidence interval for the estimate.

conf.low Lower bound on the confidence interval for the estimate.

n.censor Number of censored events.

Number of events at time t.

n.risk Number of individuals at risk at time zero.std.error The standard error of the regression term.

time Point in time.

estimate estimate of survival or cumulative incidence rate when multistate

state state if multistate survfit object input strata strata if stratified survfit object input

See Also

```
tidy(), survival::survfit()
```

```
Other survival tidiers: augment.coxph(), augment.survreg(), glance.aareg(), glance.cch(), glance.coxph(), glance.pyears(), glance.survdiff(), glance.survexp(), glance.survfit(), glance.survreg(), tidy.aareg(), tidy.cch(), tidy.coxph(), tidy.pyears(), tidy.survdiff(), tidy.survexp(), tidy.survreg()
```

384 tidy.survreg

Examples

```
# load libraries for models and data
library(survival)
# fit model
cfit <- coxph(Surv(time, status) ~ age + sex, lung)</pre>
sfit <- survfit(cfit)</pre>
# summarize model fit with tidiers + visualization
tidy(sfit)
glance(sfit)
library(ggplot2)
ggplot(tidy(sfit), aes(time, estimate)) +
  geom_line() +
  geom_ribbon(aes(ymin = conf.low, ymax = conf.high), alpha = .25)
# multi-state
fitCI <- survfit(Surv(stop, status * as.numeric(event), type = "mstate") ~ 1,</pre>
  data = mgus1, subset = (start == 0)
td_multi <- tidy(fitCI)</pre>
td_multi
ggplot(td_multi, aes(time, estimate, group = state)) +
  geom_line(aes(color = state)) +
  geom_ribbon(aes(ymin = conf.low, ymax = conf.high), alpha = .25)
```

tidy.survreg

Tidy a(n) survreg object

Description

Tidy summarizes information about the components of a model. A model component might be a single term in a regression, a single hypothesis, a cluster, or a class. Exactly what tidy considers to be a model component varies across models but is usually self-evident. If a model has several distinct types of components, you will need to specify which components to return.

Usage

```
## S3 method for class 'survreg'
tidy(x, conf.level = 0.95, conf.int = FALSE, ...)
```

tidy.survreg 385

Arguments

An survreg object returned from survival::survreg().

The confidence level to use for the confidence interval if conf.int = TRUE. Must be strictly greater than 0 and less than 1. Defaults to 0.95, which corresponds to a 95 percent confidence interval.

Logical indicating whether or not to include a confidence interval in the tidied output. Defaults to FALSE.

Additional arguments. Not used. Needed to match generic signature only. Cautionary note: Misspelled arguments will be absorbed in . . . , where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

Value

A tibble::tibble() with columns:

conf.high	Upper bound on the confidence interval for the estimate.
conf.low	Lower bound on the confidence interval for the estimate.
estimate	The estimated value of the regression term.
p.value	The two-sided p-value associated with the observed statistic.
statistic	The value of a T-statistic to use in a hypothesis that the regression term is non-zero.
std.error	The standard error of the regression term.
term	The name of the regression term.

See Also

```
tidy(), survival::survreg()
Other survreg tidiers: augment.survreg(), glance.survreg()
Other survival tidiers: augment.coxph(), augment.survreg(), glance.aareg(), glance.cch(),
glance.coxph(), glance.pyears(), glance.survdiff(), glance.survexp(), glance.survfit(),
glance.survreg(), tidy.aareg(), tidy.cch(), tidy.coxph(), tidy.pyears(), tidy.survdiff(),
tidy.survexp(), tidy.survfit()
```

```
# load libraries for models and data
library(survival)
```

386 tidy.svyglm

```
# fit model
sr <- survreg(</pre>
 Surv(futime, fustat) ~ ecog.ps + rx,
 ovarian,
 dist = "exponential"
)
# summarize model fit with tidiers + visualization
tidy(sr)
augment(sr, ovarian)
glance(sr)
# coefficient plot
td <- tidy(sr, conf.int = TRUE)</pre>
library(ggplot2)
ggplot(td, aes(estimate, term)) +
 geom_point() +
 geom_errorbarh(aes(xmin = conf.low, xmax = conf.high), height = 0) +
 geom_vline(xintercept = 0)
```

tidy.svyglm

 $Tidy\ a(n)\ svyglm\ object$

Description

Tidy summarizes information about the components of a model. A model component might be a single term in a regression, a single hypothesis, a cluster, or a class. Exactly what tidy considers to be a model component varies across models but is usually self-evident. If a model has several distinct types of components, you will need to specify which components to return.

Usage

```
## S3 method for class 'svyglm'
tidy(x, conf.int = FALSE, conf.level = 0.95, exponentiate = FALSE, ...)
```

Arguments

X	A svyglm object returned from survey::svyglm().
conf.int	Logical indicating whether or not to include a confidence interval in the tidied output. Defaults to FALSE.
conf.level	The confidence level to use for the confidence interval if conf.int = TRUE. Must be strictly greater than 0 and less than 1. Defaults to 0.95, which corresponds to a 95 percent confidence interval.
exponentiate	Logical indicating whether or not to exponentiate the coefficient estimates. This is typical for logistic and multinomial regressions, but a bad idea if there is no log or logit link. Defaults to FALSE.

tidy.svyolr 387

. . .

Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in . . . , where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

See Also

```
survey::svyglm(), stats::glm()
```

tidy.svyolr

Tidy a(n) svyolr object

Description

Tidy summarizes information about the components of a model. A model component might be a single term in a regression, a single hypothesis, a cluster, or a class. Exactly what tidy considers to be a model component varies across models but is usually self-evident. If a model has several distinct types of components, you will need to specify which components to return.

Usage

```
## S3 method for class 'svyolr'
tidy(x, conf.int = FALSE, conf.level = 0.95, exponentiate = FALSE, ...)
```

Arguments

X	A svyolr object returned from survey::svyolr().
conf.int	Logical indicating whether or not to include a confidence interval in the tidied output. Defaults to FALSE.
conf.level	The confidence level to use for the confidence interval if conf.int = TRUE. Must be strictly greater than 0 and less than 1. Defaults to 0.95, which corresponds to a 95 percent confidence interval.
exponentiate	Logical indicating whether or not to exponentiate the the coefficient estimates. This is typical for logistic and multinomial regressions, but a bad idea if there is no log or logit link. Defaults to FALSE.
	Additional arguments. Not used. Needed to match generic signature only. Cautionary note: Misspelled arguments will be absorbed in , where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

388 tidy.svyolr

• tidy() methods will warn when supplied an exponentiate argument if it will be ignored.

• augment() methods will warn when supplied a newdata argument if it will be ignored.

Details

The tidy.svyolr() tidier is a light wrapper around tidy.polr(). However, the implementation for p-value calculation in tidy.polr() is both computationally intensive and specific to that model, so the p.values argument to tidy.svyolr() is currently ignored, and will raise a warning when passed.

Value

A tibble::tibble() with columns:

conf.high	Upper bound on the confidence interval for the estimate.
conf.low	Lower bound on the confidence interval for the estimate.
estimate	The estimated value of the regression term.
p.value	The two-sided p-value associated with the observed statistic.
statistic	The value of a T-statistic to use in a hypothesis that the regression term is non-zero.
std.error	The standard error of the regression term.
term	The name of the regression term.

See Also

```
tidy, survey::svyolr()
Other ordinal tidiers: augment.clm(), augment.polr(), glance.clm(), glance.clmm(), glance.polr(),
glance.svyolr(), tidy.clm(), tidy.clmm(), tidy.polr()
```

```
library(broom)
library(survey)

data(api)
dclus1 <- svydesign(id = ~dnum, weights = ~pw, data = apiclus1, fpc = ~fpc)
dclus1 <- update(dclus1, mealcat = cut(meals, c(0, 25, 50, 75, 100)))

m <- svyolr(mealcat ~ avg.ed + mobility + stype, design = dclus1)

m

tidy(m, conf.int = TRUE)</pre>
```

tidy.systemfit 389

|--|

Description

Tidy summarizes information about the components of a model. A model component might be a single term in a regression, a single hypothesis, a cluster, or a class. Exactly what tidy considers to be a model component varies across models but is usually self-evident. If a model has several distinct types of components, you will need to specify which components to return.

Usage

```
## S3 method for class 'systemfit'
tidy(x, conf.int = TRUE, conf.level = 0.95, ...)
```

Arguments

x	A systemfit object produced by a call to systemfit::systemfit().
conf.int	Logical indicating whether or not to include a confidence interval in the tidied output. Defaults to FALSE.
conf.level	The confidence level to use for the confidence interval if conf.int = TRUE. Must be strictly greater than 0 and less than 1. Defaults to 0.95, which corresponds to a 95 percent confidence interval.
	Additional arguments. Not used. Needed to match generic signature only. Cautionary note: Misspelled arguments will be absorbed in , where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

Details

This tidy method works with any model objects of class systemfit. Default returns a tibble of six columns.

Value

A tibble::tibble() with columns:

conf.high	Upper bound on the confidence interval for the estimate.
conf.low	Lower bound on the confidence interval for the estimate.
estimate	The estimated value of the regression term.

390 tidy.table

p. value The two-sided p-value associated with the observed statistic.

std.error The standard error of the regression term.

term The name of the regression term.

See Also

```
tidy(), systemfit::systemfit()
```

Examples

```
set.seed(27)
# load libraries for models and data
library(systemfit)

# generate data
df <- data.frame(
    X = rnorm(100),
    Y = rnorm(100),
    Z = rnorm(100),
    W = rnorm(100)
)

# fit model
fit <- systemfit(formula = list(Y ~ Z, W ~ X), data = df, method = "SUR")

# summarize model fit with tidiers
tidy(fit)
tidy(fit, conf.int = TRUE)</pre>
```

tidy.table

 $Tidy\ a(n)\ table\ object$

Description

Tidy summarizes information about the components of a model. A model component might be a single term in a regression, a single hypothesis, a cluster, or a class. Exactly what tidy considers to be a model component varies across models but is usually self-evident. If a model has several distinct types of components, you will need to specify which components to return.

Deprecated. Please use tibble::as_tibble() instead.

Usage

```
## S3 method for class 'table' tidy(x, ...)
```

tidy.ts 391

Arguments

x A base::table object.

Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in . . ., where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

Details

```
Directly calls tibble::as_tibble() on a base::table object.
```

Value

A tibble::tibble in long-form containing frequency information for the table in a Freq column. The result is much like what you get from tidyr::pivot_longer().

See Also

```
tibble::as_tibble.table()
```

tidy.ts

Tidy a(n) ts object

Description

Tidy summarizes information about the components of a model. A model component might be a single term in a regression, a single hypothesis, a cluster, or a class. Exactly what tidy considers to be a model component varies across models but is usually self-evident. If a model has several distinct types of components, you will need to specify which components to return.

Usage

```
## S3 method for class 'ts' tidy(x, ...)
```

392 tidy.ts

Arguments

x A univariate or multivariate ts times series object.

Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in . . . , where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

Details

series column is only present for multivariate ts objects.

Value

```
A tibble::tibble() with columns:
```

index Index (i.e. date or time) for a 'ts' or 'zoo' object.

series Name of the series (present only for multivariate time series).

value The value/estimate of the component. Results from data reshaping.

See Also

```
tidy(), stats::ts()
Other time series tidiers: tidy.acf(), tidy.spec(), tidy.zoo()
```

```
set.seed(678)

tidy(ts(1:10, frequency = 4, start = c(1959, 2)))

z <- ts(matrix(rnorm(300), 100, 3), start = c(1961, 1), frequency = 12)
colnames(z) <- c("Aa", "Bb", "Cc")

tidy(z)</pre>
```

tidy.TukeyHSD 393

tidy.TukeyHSD Tidy a(r	ı) TukeyHSD object
------------------------	--------------------

Description

Tidy summarizes information about the components of a model. A model component might be a single term in a regression, a single hypothesis, a cluster, or a class. Exactly what tidy considers to be a model component varies across models but is usually self-evident. If a model has several distinct types of components, you will need to specify which components to return.

Usage

```
## S3 method for class 'TukeyHSD'
tidy(x, ...)
```

Arguments

x A TukeyHSD object return from stats::TukeyHSD().

Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in . . . , where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

Value

A tibble::tibble() with columns:

```
adj.p.value P-value adjusted for multiple comparisons.

Conf.high Upper bound on the confidence interval for the estimate.

Conf.low Lower bound on the confidence interval for the estimate.

Contrast Levels being compared.

Estimate The estimated value of the regression term.

Value to which the estimate is compared.

The name of the regression term.
```

See Also

```
tidy(), stats::TukeyHSD()
Other anova tidiers: glance.anova(), glance.aov(), tidy.anova(), tidy.aov(), tidy.aov(), tidy.aov(), tidy.aov()
```

394 tidy.varest

Examples

```
fm1 <- aov(breaks ~ wool + tension, data = warpbreaks)
thsd <- TukeyHSD(fm1, "tension", ordered = TRUE)
tidy(thsd)

# may include comparisons on multiple terms
fm2 <- aov(mpg ~ as.factor(gear) * as.factor(cyl), data = mtcars)
tidy(TukeyHSD(fm2))</pre>
```

tidy.varest

Tidy a(n) varest object

Description

Tidy summarizes information about the components of a model. A model component might be a single term in a regression, a single hypothesis, a cluster, or a class. Exactly what tidy considers to be a model component varies across models but is usually self-evident. If a model has several distinct types of components, you will need to specify which components to return.

Usage

```
## S3 method for class 'varest'
tidy(x, conf.int = FALSE, conf.level = 0.95, ...)
```

Arguments

x	A varest object produced by a call to vars::VAR().
conf.int	Logical indicating whether or not to include a confidence interval in the tidied output. Defaults to FALSE.
conf.level	The confidence level to use for the confidence interval if conf.int = TRUE. Must be strictly greater than 0 and less than 1. Defaults to 0.95, which corresponds to a 95 percent confidence interval.
	For glance(), additional arguments passed to summary(). Otherwise ignored.

Details

The tibble has one row for each term in the regression. The component column indicates whether a particular term was used to model either the "mean" or "precision". Here the precision is the inverse of the variance, often referred to as phi. At least one term will have been used to model the precision phi.

The vars package does not include a confint method and does not report confidence intervals for varest objects. Setting the tidy argument conf.int = TRUE will return a warning.

tidy.zoo 395

Value

A tibble::tibble() with columns:

conf.high	Upper bound on the confidence interval for the estimate.
conf.low	Lower bound on the confidence interval for the estimate.
estimate	The estimated value of the regression term.
p.value	The two-sided p-value associated with the observed statistic.
statistic	The value of a T-statistic to use in a hypothesis that the regression term is non-zero.
std.error	The standard error of the regression term.
term	The name of the regression term.
component	Whether a particular term was used to model the mean or the precision in the regression. See details.

See Also

```
tidy(), vars::VAR()
```

Examples

```
# load libraries for models and data
library(vars)

# load data
data("Canada", package = "vars")

# fit models
mod <- VAR(Canada, p = 1, type = "both")

# summarize model fit with tidiers
tidy(mod)
glance(mod)</pre>
```

tidy.zoo

Tidy a(n) zoo object

Description

Tidy summarizes information about the components of a model. A model component might be a single term in a regression, a single hypothesis, a cluster, or a class. Exactly what tidy considers to be a model component varies across models but is usually self-evident. If a model has several distinct types of components, you will need to specify which components to return.

396 tidy.zoo

Usage

```
## S3 method for class 'zoo' tidy(x, ...)
```

Arguments

A zoo object such as those created by zoo::zoo().

Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in . . . , where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

Value

```
A tibble::tibble() with columns:
```

index Index (i.e. date or time) for a 'ts' or 'zoo' object.

series Name of the series (present only for multivariate time series).

value The value/estimate of the component. Results from data reshaping.

See Also

```
tidy(), zoo::zoo()
Other time series tidiers: tidy.acf(), tidy.spec(), tidy.ts()
```

```
# load libraries for models and data
library(zoo)
library(ggplot2)

set.seed(1071)

# generate data
Z.index <- as.Date(sample(12450:12500, 10))
Z.data <- matrix(rnorm(30), ncol = 3)
colnames(Z.data) <- c("Aa", "Bb", "Cc")
Z <- zoo(Z.data, Z.index)

# summarize model fit with tidiers + visualization
tidy(Z)

ggplot(tidy(Z), aes(index, value, color = series)) +</pre>
```

tidy_irlba 397

```
geom_line()
ggplot(tidy(Z), aes(index, value)) +
  geom_line() +
  facet_wrap(~series, ncol = 1)

Zrolled <- rollmean(Z, 5)
ggplot(tidy(Zrolled), aes(index, value, color = series)) +
  geom_line()</pre>
```

tidy_irlba

Tidy a(n) irlba object masquerading as list

Description

Broom tidies a number of lists that are effectively S3 objects without a class attribute. For example, stats::optim(), svd() and interp::interp() produce consistent output, but because they do not have a class attribute, they cannot be handled by S3 dispatch.

These functions look at the elements of a list and determine if there is an appropriate tidying method to apply to the list. Those tidiers are implemented as functions of the form tidy_<function> or glance_<function> and are not exported (but they are documented!).

If no appropriate tidying method is found, they throw an error.

Usage

```
tidy_irlba(x, ...)
```

Arguments

. . .

x A list returned from irlba::irlba().

Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in . . . , where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

Details

A very thin wrapper around tidy_svd().

398 tidy_irlba

Value

A tibble::tibble with columns depending on the component of PCA being tidied.

If matrix is "u", "samples", "scores", or "x" each row in the tidied output corresponds to the original data in PCA space. The columns are:

row ID of the original observation (i.e. rowname from original data).

PC Integer indicating a principal component.

value The score of the observation for that particular principal component. That is, the

location of the observation in PCA space.

If matrix is "v", "rotation", "loadings" or "variables", each row in the tidied output corresponds to information about the principle components in the original space. The columns are:

row The variable labels (colnames) of the data set on which PCA was performed.

PC An integer vector indicating the principal component.

value The value of the eigenvector (axis score) on the indicated principal component.

If matrix is "d", "eigenvalues" or "pcs", the columns are:

PC An integer vector indicating the principal component.

std.dev Standard deviation explained by this PC.

percent Fraction of variation explained by this component (a numeric value between 0

and 1).

cumulative Cumulative fraction of variation explained by principle components up to this

component (a numeric value between 0 and 1).

See Also

```
tidy(), irlba::irlba()
Other list tidiers: glance_optim(), list_tidiers, tidy_optim(), tidy_svd(), tidy_xyz()
Other svd tidiers: augment.prcomp(), tidy.prcomp(), tidy_svd()
```

Examples

```
library(modeldata)
data(hpc_data)

mat <- scale(as.matrix(hpc_data[, 2:5]))
s <- svd(mat)

tidy_u <- tidy(s, matrix = "u")
tidy_u
tidy_d <- tidy(s, matrix = "d")
tidy_d</pre>
```

tidy_optim 399

```
tidy_v <- tidy(s, matrix = "v")
tidy_v

library(ggplot2)
library(dplyr)

ggplot(tidy_d, aes(PC, percent)) +
  geom_point() +
  ylab("% of variance explained")

tidy_u %>%
  mutate(class = hpc_data$class[row]) %>%
  ggplot(aes(class, value)) +
  geom_boxplot() +
  facet_wrap(~PC, scale = "free_y")
```

tidy_optim

Tidy a(n) optim object masquerading as list

Description

Broom tidies a number of lists that are effectively S3 objects without a class attribute. For example, stats::optim(), svd() and interp::interp() produce consistent output, but because they do not have a class attribute, they cannot be handled by S3 dispatch.

These functions look at the elements of a list and determine if there is an appropriate tidying method to apply to the list. Those tidiers are implemented as functions of the form tidy_<function> or glance_<function> and are not exported (but they are documented!).

If no appropriate tidying method is found, they throw an error.

Usage

```
tidy_optim(x, ...)
```

Arguments

Х

A list returned from stats::optim().

Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in . . . , where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

400 tidy_svd

Value

A tibble::tibble() with columns:

parameter The parameter being modeled.

std.error The standard error of the regression term.

value The value/estimate of the component. Results from data reshaping.

std.error is only provided as a column if the Hessian is calculated.

Note

This function assumes that the provided objective function is a negative log-likelihood function. Results will be invalid if an incorrect function is supplied.

```
tidy(o) glance(o)
```

See Also

```
tidy(), stats::optim()
Other list tidiers: glance_optim(), list_tidiers, tidy_irlba(), tidy_svd(), tidy_xyz()
```

Examples

```
f <- function(x) (x[1] - 2)^2 + (x[2] - 3)^2 + (x[3] - 8)^2
o <- optim(c(1, 1, 1), f)
```

tidy_svd

Tidy a(n) svd object masquerading as list

Description

Broom tidies a number of lists that are effectively S3 objects without a class attribute. For example, stats::optim(), svd() and interp::interp() produce consistent output, but because they do not have a class attribute, they cannot be handled by S3 dispatch.

These functions look at the elements of a list and determine if there is an appropriate tidying method to apply to the list. Those tidiers are implemented as functions of the form tidy_<function> or glance_<function> and are not exported (but they are documented!).

If no appropriate tidying method is found, they throw an error.

Usage

```
tidy_svd(x, matrix = "u", ...)
```

tidy_svd 401

Arguments

A list with components u, d, v returned by base::svd().

matrix Character specifying which component of the PCA should be tidied.

• "u", "samples", "scores", or "x": returns information about the map from the original space into principle components space.

- "v", "rotation", "loadings" or "variables": returns information about the map from principle components space back into the original space.
- "d", "eigenvalues" or "pcs": returns information about the eigenvalues.

Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in . . ., where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

Details

See https://stats.stackexchange.com/questions/134282/relationship-between-svd-and-pca-how-to-use-svd-to-perform-pca for information on how to interpret the various tidied matrices. Note that SVD is only equivalent to PCA on centered data.

Value

A tibble::tibble with columns depending on the component of PCA being tidied.

If matrix is "u", "samples", "scores", or "x" each row in the tidied output corresponds to the original data in PCA space. The columns are:

row ID of the original observation (i.e. rowname from original data).

PC Integer indicating a principal component.

value The score of the observation for that particular principal component. That is, the

location of the observation in PCA space.

If matrix is "v", "rotation", "loadings" or "variables", each row in the tidied output corresponds to information about the principle components in the original space. The columns are:

row The variable labels (colnames) of the data set on which PCA was performed.

PC An integer vector indicating the principal component.

value The value of the eigenvector (axis score) on the indicated principal component.

If matrix is "d", "eigenvalues" or "pcs", the columns are:

PC An integer vector indicating the principal component.

std.dev Standard deviation explained by this PC.

402 tidy_xyz

percent Fraction of variation explained by this component (a numeric value between 0

and 1).

cumulative Cumulative fraction of variation explained by principle components up to this

component (a numeric value between 0 and 1).

See Also

```
base::svd()
Other svd tidiers: augment.prcomp(), tidy.prcomp(), tidy_irlba()
Other list tidiers: glance_optim(), list_tidiers, tidy_irlba(), tidy_optim(), tidy_xyz()
```

Examples

```
library(modeldata)
data(hpc_data)
mat <- scale(as.matrix(hpc_data[, 2:5]))</pre>
s <- svd(mat)
tidy_u <- tidy(s, matrix = "u")</pre>
tidy_u
tidy_d <- tidy(s, matrix = "d")</pre>
tidy_d
tidy_v <- tidy(s, matrix = "v")</pre>
tidy_v
library(ggplot2)
library(dplyr)
ggplot(tidy_d, aes(PC, percent)) +
  geom_point() +
  ylab("% of variance explained")
tidy_u %>%
  mutate(class = hpc_data$class[row]) %>%
  ggplot(aes(class, value)) +
  geom_boxplot() +
  facet_wrap(~PC, scale = "free_y")
```

tidy_xyz 403

Description

Broom tidies a number of lists that are effectively S3 objects without a class attribute. For example, stats::optim(), svd() and interp::interp() produce consistent output, but because they do not have a class attribute, they cannot be handled by S3 dispatch.

These functions look at the elements of a list and determine if there is an appropriate tidying method to apply to the list. Those tidiers are implemented as functions of the form tidy_<function> or glance_<function> and are not exported (but they are documented!).

If no appropriate tidying method is found, they throw an error.

xyz lists (lists where x and y are vectors of coordinates and z is a matrix of values) are typically used by functions such as graphics::persp() or graphics::image() and returned by interpolation functions such as interp::interp().

Usage

```
tidy_xyz(x, ...)
```

Arguments

Х

A list with component x, y and z, where x and y are vectors and z is a matrix. The length of x must equal the number of rows in z and the length of y must equal the number of columns in z.

. . .

Additional arguments. Not used. Needed to match generic signature only. **Cautionary note:** Misspelled arguments will be absorbed in . . . , where they will be ignored. If the misspelled argument has a default value, the default value will be used. For example, if you pass conf.lvel = 0.9, all computation will proceed using conf.level = 0.95. Two exceptions here are:

- tidy() methods will warn when supplied an exponentiate argument if it will be ignored.
- augment() methods will warn when supplied a newdata argument if it will be ignored.

Value

A tibble::tibble with vector columns x, y and z.

See Also

```
tidy(), graphics::persp(), graphics::image(), interp::interp()
Other list tidiers: glance_optim(), list_tidiers, tidy_irlba(), tidy_optim(), tidy_svd()
```

Examples

```
A <- list(x = 1:5, y = 1:3, z = matrix(runif(5 \star 3), nrow = 5)) image(A) tidy(A)
```

Index

* Arima tidiers	glance.coxph, 119
glance.Arima, 104	tidy.coxph, 253
tidy.Arima, 228	* crr tidiers
* aareg tidiers	glance.crr, 121
glance.aareg,99	* decompose tidiers
tidy.aareg, 222	augment.decomposed.ts, 17
* anova tidiers	augment.stl,88
glance.anova, 101	* deprecated
glance.aov, 102	bootstrap, 93
tidy.anova, 224	confint_tidy, 94
tidy.aov, <u>226</u>	data.frame_tidiers, 95
tidy.aovlist,227	finish_glance, 98
tidy.manova, 313	<pre>fix_data_frame, 99</pre>
tidy.TukeyHSD,393	summary_tidiers, 220
* betareg tidiers	tidy.density,259
tidy.betareg, 231	tidy.dist,260
* biglm tidiers	tidy.ftable, 275
glance.biglm, 108	tidy.numeric,338
tidy.biglm, 233	* drc tidiers
* bingroup tidiers	augment.drc, 19
glance.binDesign, 110	glance.drc, 125
tidy.binDesign, 235	tidy.drc, 261
tidy.binWidth, 236	* emmeans tidiers
* car tidiers	tidy.emmGrid, 262
${\tt durbinWatsonTest_tidiers}, 97$	tidy.lsmobj, 311
<pre>leveneTest_tidiers, 217</pre>	tidy.ref.grid,356
* cch tidiers	tidy.summary_emm,378
glance.cch, 111	* epiR tidiers
glance.survfit, 207	tidy.epi.2by2, 264
tidy.cch, 241	* ergm tidiers
* cmprsk tidiers	glance.ergm, 126
glance.crr, 121	tidy.ergm, 266
tidy.crr, 255	* factanal tidiers
* coeftest tidiers	augment.factanal, 22
tidy.coeftest, 248	glance.factanal, 128
* coeftest_tidiers	tidy.factanal, 268
glance.coeftest, 117	* felm tidiers
* coxph tidiers	augment.felm, 23
augment.coxph, 14	tidy.felm, 269

* fitdistr tidiers	* lm tidiers
glance.fitdistr, 131	augment.glm, 30
tidy.fitdistr, 271	augment.lm, 41
* fixest tidiers	glance.glm, 140
augment.fixest, 25	glance.lm, 153
tidy.fixest, 273	glance.summary.lm, 201
* gam tidiers	glance.svyglm, 211
glance.Gam, 135	tidy.glm, 283
tidy.Gam, 275	tidy.1m, 303
* garch tidiers	tidy.lm.beta, 305
glance.garch, 138	tidy.mlm, 328
tidy.garch, 279	tidy.summary.lm, 376
* geepack tidiers	* lmodel2 tidiers
glance.geeglm, 139	glance.lmodel2, 155
* glm.nb tidiers	tidy.lmodel2,307
glance.negbin, 173	* margins tidiers
tidy.negbin, 334	tidy.margins, 316
* glmnet tidiers	* mclust tidiers
glance.cv.glmnet, 123	augment.Mclust, 51
glance.glmnet, 142	tidy.Mclust, 318
tidy.cv.glmnet,257	* mediate tidiers
tidy.glmnet, 284	tidy.mediate, 320
* gmm tidiers	* mfx tidiers
glance.gmm, 145	augment.betamfx, 8
tidy.gmm, 289	augment.mfx, 53
* htest tidiers	glance.betamfx, 105
augment.htest, 35	glance.mfx, 164
tidy.htest, 291	tidy.betamfx, 229
tidy.pairwise.htest, 340	tidy.mfx, 322
tidy.power.htest, 349	* mgcv tidiers
* ivreg tidiers	glance.gam, 136
augment.ivreg, 37	tidy.gam, 277
glance.ivreg, 147	* mjoint tidiers
tidy.ivreg, 293	glance.mjoint, 166
* kmeans tidiers	tidy.mjoint, 324
augment.kmeans, 39	* mlogit tidiers
glance.kmeans, 149	augment.mlogit,59
tidy.kmeans, 300	glance.mlogit, 168
* lavaan tidiers	tidy.mlogit, 330
glance.lavaan, 151	* muhaz tidiers
tidy.lavaan, 301	glance.muhaz, 170
* list tidiers	tidy.muhaz, 331
glance_optim, 216	* multcomp tidiers
list_tidiers, 218	tidy.cld, 242
tidy_irlba, 397	tidy.confint.glht, 250
tidy_optim, 399	tidy.glht, 282
tidy_svd, 400	tidy.summary.glht, 374
tidy_svd, 400 tidy_xyz, 402	* multinom tidiers
014y_xy2, 102	* mumom nuicis

glance.multinom, 171	augment.rlm,74
tidy.multinom, 332	glance.rlm, 189
* nls tidiers	tidy.rlm, 360
augment.nls, 62	* robust tidiers
glance.nls, 176	augment.lmRob,45
tidy.nls, 336	glance.glmRob, 143
* orcutt tidiers	glance.lmRob, 157
glance.orcutt, 177	tidy.glmRob, 286
tidy.orcutt,339	tidy.lmRob, 309
* ordinal tidiers	* robustbase tidiers
augment.clm, 12	augment.glmrob,33
augment.polr, 70	augment.lmrob,47
glance.clm, 113	glance.lmrob, 159
glance.clmm, 115	tidy.glmrob, 287
glance.polr, 184	tidy.lmrob, 310
glance.svyolr, 213	* smoothing spline tidiers
tidy.clm, 244	augment.smooth.spline,85
tidy.clmm, 246	glance.smooth.spline, 197
tidy.polr,347	* spatialreg tidiers
tidy.svyolr,387	augment.sarlm,83
* pam tidiers	glance.sarlm, 195
augment.pam,64	tidy.sarlm,368
glance.pam, 179	* speedlm tidiers
tidy.pam, 341	augment.speedlm,86
* plm tidiers	glance.speedglm, 198
augment.plm,66	glance.speedlm, 200
glance.plm, 180	tidy.speedglm, 371
tidy.plm, 343	tidy.speedlm, 373
* poLCA tidiers	* survdiff tidiers
augment.poLCA, 68	glance.survdiff, 204
glance.poLCA, 182	tidy.survdiff,380
tidy.poLCA, 345	* survexp tidiers
* pyears tidiers	glance.survexp, 206
glance.pyears, 186	tidy.survexp,381
tidy.pyears, 352	* survey tidiers
* quantreg tidiers	tidy.svyglm, 386
augment.nlrq,61	* survfit tidiers
augment.rq,78	tidy.survfit,383
augment.rqs, 80	* survival tidiers
glance.nlrq, 174	augment.coxph, 14
glance.rq, 193	augment.survreg, 89
tidy.nlrq,335	${\tt glance.aareg,99}$
tidy.rq,365	glance.cch, 111
tidy.rqs,366	glance.coxph, 119
* ridgelm tidiers	glance.pyears, 186
glance.ridgelm, 188	glance.survdiff, 204
tidy.ridgelm,359	glance.survexp, 206
* rlm tidiers	glance.survfit, 207

glance.survreg, 209	augment.drc, 19, 126, 262
tidy.aareg,222	augment.factanal, 22, 129, 268
tidy.cch, 241	augment.felm, 23, 271
tidy.coxph, 253	augment.fixest, 25, 274
tidy.pyears, 352	augment.gam, 28
tidy.survdiff,380	augment.glm, 30, 43, 141, 154, 202, 212, 284
tidy.survexp,381	304, 306, 329, 377
tidy.survfit,383	augment.glm(), 55, 56
tidy.survreg,384	augment.glmRob, 32
* survreg tidiers	augment.glmrob, 33, 48, 160, 288, 311
augment.survreg, 89	augment.htest, 35, 292, 341, 350
glance.survreg, 209	augment.ivreg, 37, 149, 294
tidy.survreg,384	augment.kmeans, 39, 150, 300
* svd tidiers	augment.1m, 32, 41, 141, 154, 202, 212, 284,
augment.prcomp, 72	304, 306, 329, 377
tidy.prcomp, 350	augment.1mRob, 45, 144, 158, 287, 309
tidy_irlba, 397	augment.1mrob, 35, 47, 160, 288, 311
$tidy_svd, 400$	augment.loess, 49
* systemfit tidiers	augment.logitmfx (augment.mfx), 53
tidy.systemfit, 389	augment.Mclust, 51, 319
* time series tidiers	augment.mfx, 9, 53, 106, 166, 230, 323
tidy.acf, 223	augment.mjoint, 56
tidy.spec, 370	augment.mjoint, 50 augment.mlogit, 59, 169, 331
tidy.ts,391	augment.miogit, 39, 109, 331 augment.negbinmfx (augment.mfx), 53
tidy.zoo,395	augment.nlrq, 61, 80, 82, 175, 194, 336, 366
* vars tidiers	367
tidy.varest,394	augment.nls, 62, 177, 337
	augment.NULL (null_tidiers), 219
aareg_tidiers(tidy.aareg),222	- '
AER::ivreg(), 38, 39, 148, 149, 293, 294	augment.pam, 64, 180, 342
aer_tidiers(tidy.ivreg), 293	<pre>augment.plm, 66, 182, 344 augment.poissonmfx (augment.mfx), 53</pre>
Arima_tidiers(tidy.Arima),228	
AUC::roc(), 363, 364	augment.poLCA, 68, 183, 345
auc_tidiers(tidy.roc),363	augment.polr, 14, 70, 114, 116, 186, 214,
augment, 80, 82	245, 247, 348, 388
augment(), 12, 16, 18, 21, 23, 25, 27, 30, 37,	augment.prcomp, 72, 351, 398, 402
39, 41, 43, 50, 52, 60, 61, 65, 67, 69,	augment.probitmfx (augment.mfx), 53
84, 86, 89, 91, 92, 197	augment.rlm, 74, 190, 361
augment.betamfx, 8, 56, 106, 166, 230, 323	augment.rma, 76
augment.betareg, 10	augment.rq, 61, 78, 82, 175, 194, 336, 366,
augment.betareg(), 9	367
augment.clm, 12, 72, 114, 116, 186, 214, 245, 247, 348, 388	augment.rqs, 61, 80, 80, 175, 194, 336, 366, 367
augment.coxph, 14, 91, 100, 113, 120, 187,	augment.sarlm, 83, 196, 369
205, 207, 208, 210, 223, 242, 254,	augment.smooth.spline, 85, 197
353, 381–383, 385	augment.speedlm, 86, 199, 201, 372, 374
augment.data.frame	augment.stl, 18,88
(data.frame_tidiers), 95	augment.survreg, 16, 89, 100, 113, 120, 187
augment.decomposed.ts, 17, 89	205, 207, 208, 210, 223, 242, 254,

353, 381–383, 385	confint_tidy, 93, 94, 96, 98, 99, 221, 259,
<pre>augment_columns, 92</pre>	260, 275, 338
	confusionMatrix_tidiers
base::data.frame, 9, 11, 13, 15, 20, 23, 24,	(tidy.confusionMatrix), 251
26, 29, 31, 34, 38, 40, 42, 46, 47, 49,	<pre>coxph_tidiers (tidy.coxph), 253</pre>
51, 55, 57, 61, 63, 65, 67, 68, 71, 73,	
75, 79, 81, 85, 87, 90	data.frame_tidiers, 93, 94, 95, 98, 99, 221,
base::data.frame(), 9, 11, 13, 15, 21, 27,	259, 260, 275, 338
29, 31, 34, 38, 42, 46, 47, 49, 55, 61,	decompose_tidiers
63, 71, 73, 75, 79, 81, 87, 91	(augment.decomposed.ts), 17
base::svd(), 218, 401, 402	drc::drm(), 20, 21, 125, 126, 261, 262
base::table, <i>391</i>	<pre>drc_tidiers(tidy.drc), 261</pre>
bbmle::mle2(), <i>327</i>	durbinWatsonTest_tidiers, 97, 217
bbmle_tidiers(tidy.mle2), 326	
betareg::betareg(), 11, 12, 107, 108, 231,	emmeans::contrast(), 263, 312, 357, 379
232	emmeans::emmeans(), 263, 312, 357, 379
betareg::predict.betareg(),9	emmeans::ref_grid(), 263, 312, 356, 357,
betareg::residuals.betareg(),9	379
betareg_tidiers (tidy.betareg), 231	<pre>emmeans::summary.emmGrid(), 263, 312,</pre>
biglm::bigglm(), 109, 233, 234	356, 378
biglm::biglm(), 109, 233, 234	<pre>emmeans_tidiers(tidy.lsmobj), 311</pre>
<pre>bindesign_tidiers(tidy.binDesign), 235</pre>	epiR::epi.2by2(), <u>265</u>
binGroup::binDesign, 110	epiR_tidiers(tidy.epi.2by2), 264
binGroup::binDesign(), 111, 235	ergm::control.ergm(), 267
binGroup::binWidth(), 236, 237	ergm::ergm(), 127, 266, 267
binwidth_tidiers(tidy.binWidth), 236	ergm::summary(), 127, 267
boot::boot(), 237, 238	ergm::summary.ergm(), 127
boot::boot.ci(), 238	ergm_tidiers(tidy.ergm), 266
boot::tsboot(), 238	
<pre>boot_tidiers(tidy.boot), 237</pre>	factanal_tidiers(tidy.factanal), 268
bootstrap, 93, 94, 96, 98, 99, 221, 259, 260,	<pre>felm_tidiers (tidy.felm), 269</pre>
275, 338	finish_glance, 93, 94, 96, 98, 99, 221, 259,
btergm::btergm(), 239, 240	260, 275, 338
btergm_tidiers (tidy.btergm), 239	<pre>fitdistr_tidiers(tidy.fitdistr), 271</pre>
	fix_data_frame, 93, 94, 96, 98, 99, 221, 259,
car::Anova(), 101, 224, 225	260, 275, 338
<pre>car::durbinWatsonTest(), 97</pre>	fixest::feglm(), 27, 274
car::leveneTest(), <i>101</i> , <i>217</i> , <i>224</i> – <i>227</i>	fixest::femlm(), 27, 274
<pre>car::linearHypothesis(), 101, 224</pre>	fixest::fenegbin(), 27, 274
<pre>caret::confusionMatrix(), 252</pre>	fixest::feNmlm(), 27, 274
<pre>caret_tidiers(tidy.confusionMatrix),</pre>	fixest::feols(), 27, 274
251	fixest::fepois(), 27, 274
cch_tidiers(tidy.cch), 241	
cfa_tidiers(tidy.lavaan), 301	gam::gam(), 135, 136, 276
cluster::pam(), 65, 179, 180, 342	<pre>Gam_tidiers(tidy.Gam), 275</pre>
cmprsk::crr(), <i>121</i> , <i>122</i> , <i>255</i> , <i>256</i>	<pre>gam_tidiers(tidy.gam), 277</pre>
<pre>cmprsk_tidiers(tidy.crr), 255</pre>	<pre>garch_tidiers (tidy.garch), 279</pre>
<pre>coeftest_tidiers(tidy.coeftest), 248</pre>	<pre>geeglm_tidiers(tidy.geeglm), 280</pre>
confint(), <i>94</i>	geepack::geeglm(), 139, 140, 280, 281

geepack_tidiers(tidy.geeglm), 280	glance.glm, 32, 43, 140, 154, 202, 212, 284,
glance(), 97, 100, 102, 103, 108, 109, 111,	304, 306, 329, 377
113, 118, 120, 122, 124, 126, 127,	glance.glm(), <i>165</i> , <i>166</i>
129, 136, 137, 139, 140, 143, 146,	glance.glmnet, <i>124</i> , 142, <i>257</i> , <i>285</i>
149, 150, 152, 154, 156, 167, 169,	glance.glmRob, 46, 143, 158, 287, 309
171, 172, 174, 175, 178, 180, 182,	glance.gmm, 145, 290
183, 187, 189, 190, 194, 196, 202,	glance.htest (tidy.htest), 291
205, 207, 208, 210, 215–217, 291	glance.ivreg, <i>39</i> , 147, 294
glance.aareg, 16, 91, 99, 113, 120, 187, 205,	glance.kmeans, 41, 149, 300
207, 208, 210, 223, 242, 254, 353,	glance.lavaan, 151, 302
381–383, 385	<pre>glance.list(list_tidiers), 218</pre>
glance.anova, 101, 103, 225, 226, 228, 314,	glance.lm, 32, 43, 141, 153, 202, 212, 284,
393	304, 306, 329, 377
glance.aov, 102, 102, 225, 226, 228, 314, 393	glance.lm(), 202
glance.Arima, 104, 229	glance.lmodel2, 155, 308
glance.betamfx, 9, 56, 105, 166, 230, 323	glance.lmRob, 46, 144, 157, 287, 309
glance.betareg, 107	glance.lmrob, 35, 48, 159, 288, 311
glance.betareg(), 106	glance.logitmfx(glance.mfx), 164
glance.biglm, 108, 234	glance.margins, 160
glance.binDesign, 110, 235, 237	glance.Mclust, 162
glance.cch, 16, 91, 100, 111, 120, 187, 205,	glance.mfx, 9, 56, 106, 164, 230, 323
207, 208, 210, 223, 242, 254, 353,	glance.mjoint, 166, 325
381–383, 385	glance.mlogit, 60, 168, 331
glance.clm, 14, 72, 113, 116, 186, 214, 245,	glance.muhaz, 170, 332
247, 348, 388	glance.multinom, 171, 333
glance.clmm, 14, 72, 114, 115, 186, 214, 245,	glance.negbin, 173, 334
247, 348, 388	glance.negbinmfx(glance.mfx), 164
glance.coeftest, 117	glance.nlrq, 61, 80, 82, 174, 194, 336, 366,
glance.coxph, 16, 91, 100, 113, 119, 187,	367
205, 207, 208, 210, 223, 242, 254,	glance.nls, 63, 176, 337
353, 381–383, 385	glance.NULL (null_tidiers), 219
glance.crr, 121, 256	glance.optim (glance_optim), 216
glance.cv.glmnet, 123, 143, 257, 285	glance.orcutt, 177, 340
<pre>glance.data.frame(data.frame_tidiers),</pre>	glance.pam, 65, 179, 342
95	glance.plm, 67, 180, 344
glance.drc, 21, 125, 262	glance.poissonmfx(glance.mfx), 164
<pre>glance.durbinWatsonTest</pre>	glance.poLCA, 69, 182, 345
(durbinWatsonTest_tidiers), 97	glance.polr, 14, 72, 114, 116, 184, 214, 245,
glance.ergm, 126, 267	247, 348, 388
glance.factanal, 23, 128, 268	glance.probitmfx(glance.mfx), 164
glance.felm, 130	glance.pyears, 16, 91, 100, 113, 120, 186,
glance.fitdistr, 131, 272	205, 207, 208, 210, 223, 242, 254,
glance.fixest, 133	353, 381–383, 385
glance.Gam, 135, 276	glance.ridgelm, 188, 360
glance.gam, 136, 278	glance.rlm, 75, 189, 361
glance.gam(), <i>135</i>	glance.rma, 191
glance.garch, 138, 280	glance.rq, 61, 80, 82, 175, 193, 336, 366, 367
glance.geeglm.139	glance.sarlm. 84, 195, 369

glance.smooth.spline, 86, 197	<pre>joinerml_tidiers(tidy.mjoint), 324</pre>
glance.speedglm, 88, 198, 201, 372, 374	
glance.speedlm, 88, 199, 200, 372, 374	kappa_tidiers(tidy.kappa),295
glance.summary.lm, 32, 43, 141, 154, 201,	kde_tidiers(tidy.kde),297
212, 284, 304, 306, 329, 377	Kendall::Kendall(), 299
glance.summary.lm(), 154, 202	Kendall::MannKendall(), 299
glance.summaryDefault	<pre>Kendall::SeasonalMannKendall(), 299</pre>
(summary_tidiers), 220	<pre>Kendall_tidiers(tidy.Kendall), 298</pre>
glance.survdiff, 16, 91, 100, 113, 120, 187,	kendall_tidiers(tidy.Kendall), 298
204, 207, 208, 210, 223, 242, 254,	kmeans_tidiers(tidy.kmeans), 300
353, 381–383, 385	ks::kde(), 297
glance.survexp, 16, 91, 100, 113, 120, 187,	ks_tidiers(tidy.kde), 297
205, 206, 208, 210, 223, 242, 254,	• • •
353, 381–383, 385	lavaan::cfa(), 151, 152, 301, 302
glance.survfit, 16, 91, 100, 113, 120, 187,	lavaan::fitmeasures(), 152
205, 207, 207, 210, 223, 242, 254,	lavaan::parameterEstimates(), 301, 302
353, 381–383, 385	lavaan::sem(), 151, 152, 301, 302
glance.survreg, 16, 91, 100, 113, 120, 187,	lavaan_tidiers (tidy.lavaan), 301
205, 207, 208, 209, 223, 242, 254,	leaps::regsubsets(), 358
353, 381–383, 385	leaps_tidiers (tidy.regsubsets), 358
glance.svyglm, 32, 43, 141, 154, 202, 211,	leveneTest_tidiers, 97, 217
284, 304, 306, 329, 377	lfe::felm(), 24, 25, 130, 270, 271
	lfe_tidiers(tidy.felm), 269
glance.svyolr, 14, 72, 114, 116, 186, 213,	list_tidiers, 216, 218, 398, 400, 402, 403
245, 247, 348, 388	lm.beta::lm.beta, 306
glance.varest, 214	lm_tidiers (tidy.lm), 303
glance_optim, 216, 218, 398, 400, 402, 403	lmodel2::lmodel2(), 156, 307, 308
glm.nb_tidiers (glance.negbin), 173	lmode12_tidiers (tidy.lmode12), 307
glmnet::cv.glmnet(), 123, 124, 257	Imtest::coeftest(), 117, 118, 248, 249
glmnet::glmnet(), 142, 143, 285	lmtest_tidiers (tidy.coeftest), 248
glmnet_tidiers(tidy.glmnet), 284	
gmm::gmm(), 145, 146, 289, 290	loess_tidiers (augment.loess), 49
<pre>gmm_tidiers(tidy.gmm), 289</pre>	lsmeans::summary.ref.grid(), 263, 312,
graphics::image(),403	356, 378
<pre>graphics::persp(), 403</pre>	() 315
	maps::map(), 315
Hmisc::rcorr(), 354, 355	maps_tidiers (tidy.map), 315
Hmisc_tidiers(tidy.rcorr), 354	margins::margins(), 161, 316, 317
htest_tidiers(tidy.htest), 291	margins_tidiers (tidy.margins), 316
	MASS::dropterm(), 347
interp::interp(), 216, 218, 397, 399, 400,	MASS::fitdistr(), 132, 272
403	MASS::glm.nb(), 173, 174, 334
irlba::irlba(), <i>397</i> , <i>398</i>	MASS::lm.ridge(), 188, 189, 359, 360
irlba_tidiers(tidy_irlba),397	MASS::polr(), 71, 72, 185, 186, 347, 348
<pre>ivreg_tidiers(tidy.ivreg), 293</pre>	MASS::rlm(), 75, 190, 361
	MASS::select.ridgelm(), 189
<pre>joineRML::bootSE(), 325</pre>	mclust::Mclust(), <i>51</i> , <i>52</i> , <i>163</i> , <i>319</i>
<pre>joineRML::fitted.mjoint(), 57</pre>	<pre>mclust_tidiers(tidy.Mclust), 318</pre>
joineRML::mjoint(), <i>57</i> , <i>167</i> , <i>324</i> , <i>325</i>	mean, <i>95</i>
<pre>joineRML::residuals.mjoint(), 57</pre>	<pre>mediate_tidiers(tidy.mediate), 320</pre>

mediation::mediate(), 320, 321	<pre>prcomp_tidiers (tidy.prcomp), 350</pre>
metafor::escalc(), 362	predict.fixest, 27
metafor::rma(), 77, 191, 362	psych::cohen.kappa(), 295, 296
metafor::rma.glmm(), 77, 191, 362	psych_tidiers(tidy.kappa), 295
metafor::rma.mh(), 77, 191, 362	purrr::map(), <i>193</i>
metafor::rma.mv(), 77, 191, 362	purrr::map_df(), 219
metafor::rma.peto(), 77, 191, 362	pyears_tidiers (tidy.pyears), 352
metafor::rma.uni(), 77, 191, 362	13 - (3 13 //
mfx::betamfx(), 9, 106, 230	qr, <i>314</i>
mfx::logitmfx(), 56, 166, 323	quantreg::nlrq(), 61, 175, 335, 336
mfx::negbinmfx(), 56, 166, 323	quantreg::predict.rq,79,81
mfx::poissonmfx(), 56, 166, 323	quantreg::predict.rq(), 80
mfx::probitmfx(), 56, 166, 323	quantreg::predict.rqs(),82
mgcv::gam(), 29, 30, 135, 137, 276–278	quantreg::rq(), 78, 80-82, 193, 194,
mgcv_tidiers(tidy.gam), 277	365–367
mjoint_tidiers (tidy.mjoint), 324	quantreg::summary.rq(), 365, 367
mle2_tidiers (tidy.mle2), 326	quantreg::summary.rqs(), 367
mlogit::mlogit(), 60, 169, 330, 331	<pre>quantreg_tidiers (tidy.rq), 365</pre>
mlogit_tidiers(tidy.mlogit), 330	
muhaz::muhaz(), 170, 171, 331, 332	<pre>rcorr_tidiers (tidy.rcorr), 354</pre>
<pre>muhaz_tidiers(tidy.muhaz), 331</pre>	ridgelm_tidiers(tidy.ridgelm),359
multcomp::cld(), 243	rlm_tidiers(glance.rlm), 189
multcomp::confint.glht(), 243, 250	robust::glmRob(), <i>144</i> , <i>286</i> , <i>287</i>
multcomp::glht(), 243, 250, 282, 283, 375	robust::1mRob(), 46, 158, 309
multcomp::summary.glht(), 243, 375	<pre>robust_tidiers(tidy.lmRob), 309</pre>
multcomp_tidiers (tidy.glht), 282	robustbase::glmrob(), <i>34</i> , <i>35</i> , <i>288</i>
multinom_tidiers (tidy.multinom), 332	robustbase::lmrob(), 47, 48, 159, 160, 310
marernom_erarers (eray imarernomy, 552	311
nlrq_tidiers(tidy.nlrq), 335	<pre>robustbase_tidiers(tidy.lmrob), 310</pre>
nls_tidiers (tidy.nls), 336	<pre>roc_tidiers (tidy.roc), 363</pre>
nnet::multinom(), 172, 332, 333	rq_tidiers(tidy.rq), 365
<pre>nnet_tidiers (tidy.multinom), 332</pre>	rqs_tidiers (tidy.rqs), 366
null_tidiers, 219	rsample::bootstraps(), 238
	, , , , , ,
optim_tidiers(tidy_optim), 399	sem_tidiers(tidy.lavaan),301
orcutt::cochrane.orcutt(), 178, 339, 340	<pre>sexpfit_tidiers(tidy.survexp), 381</pre>
orcutt_tidiers (tidy.orcutt), 339	<pre>smooth.spline_tidiers</pre>
ordinal::clm(), 13, 14, 114, 244, 245	(augment.smooth.spline), 85
ordinal::clmm(), 116, 246, 247	sp_tidiers, 219
ordinal::confint.clm(), 245, 247	spatialreg::errorsarlm(), 83, 195, 196,
ordinal::predict.clm(), 13, 14	<i>368, 369</i>
ordinal_tidiers(tidy.clm), 244	spatialreg::lagsarlm(), 83, 195, 196, 368 369
pam_tidiers(tidy.pam), 341	spatialreg::sacsarlm(), 196, 369
plm::plm(), 67, 181, 182, 343, 344	spatialreg_tidiers (tidy.sarlm), 368
plm_tidiers (tidy.plm), 343	speedglm::speedglm(), 198, 372
poLCA::poLCA(), 68, 69, 183, 345	speedglm::speedglm(), 778, 372 speedglm::speedlm(), 87, 88, 199–201, 373
poLCA_tidiers (tidy.poLCA), 345	374
polr_tidiers (tidy.polcA), 347	speedglm_tidiers(tidy.speedglm), 371
por _ crute 3 (cruy . por 1), 341	specugin_truters (truy.specugin), 3/1

speedlm_tidiers(tidy.speedlm), 373	stats::summary. $lm(), 304, 376, 377$
splines::ns(), 8, 10, 13, 15, 18, 20, 22, 24,	stats::summary.manova, 314
26, 28, 31, 33, 36, 38, 40, 42, 45, 47,	stats::summary.manova(), 314
51, 53, 57, 59, 62, 64, 66, 68, 71, 73,	stats::summary.nls(), 337
74, 76, 78, 81, 83, 87, 88, 90	stats::t.test(), 36, 292
stats::acf(), 223, 224	stats::ts(), <i>392</i>
stats::anova(), <i>101</i> , <i>224</i> , <i>225</i>	stats::TukeyHSD(), 393
stats::aov(), 103, 226-228	stats::wilcox.test(), 36, 292
stats::arima(), 104, 105, 228, 229	summary(), 221, 282, 334, 383, 394
stats::ccf(), 223, 224	summary.fixest, 27, 133, 273
stats::chisq.test(), 36, 37, 292	summary_tidiers, 93, 94, 96, 98, 99, 220,
stats::cooks.distance(), 11, 29	259, 260, 275, 338
stats::cor.test(), 36, 292	$survdiff_tidiers(tidy.survdiff), 380$
stats::decompose(), 18	<pre>survexp_tidiers(tidy.survexp), 381</pre>
stats::density(), 259	survey::anova.svyglm,212
stats::dist(), 260	survey::svyglm(), 211, 212, 386, 387
stats::factanal(), 23, 128, 129, 268	survey::svyolr(), 213, 214, 387, 388
stats::ftable(), 275	<pre>survfit_tidiers(tidy.survfit), 383</pre>
stats::glm(), 31, 32, 141, 212, 284, 387	survival::aareg(), 100, 222, 223
stats::kmeans(), 40, 41, 150, 300	survival::cch(), 112, 113, 241, 242
stats::lm(), 23, 42, 153, 202, 303, 328	survival::coxph(), 15, 16, 119, 120, 253,
stats::loess(), 49, 50	254
stats::manova(), <i>314</i>	survival::pyears(), <i>187</i> , <i>353</i>
stats::na.action, <i>16</i> , <i>43</i> , <i>50</i>	<pre>survival::summary.survfit(), 208</pre>
stats::nls(), 63, 176, 177, 337	survival::Surv(), 8, 10, 13, 15, 18, 20, 22,
stats::optim(), 216, 218, 397, 399, 400, 403	24, 26, 28, 31, 33, 36, 38, 40, 42, 45
stats::pacf(), 223, 224	47, 51, 53, 57, 59, 62, 64, 66, 68, 71
stats::pairwise.t.test(), <i>340</i> , <i>341</i>	73, 74, 76, 78, 81, 83, 87, 88, 90
stats::pairwise.wilcox.test(), 340, 341	survival::survdiff(), 205, 380, 381
stats::poly(), 8, 10, 13, 15, 18, 20, 22, 24,	survival::survexp(), 206, 207, 381, 382
26, 28, 31, 33, 36, 38, 40, 42, 45, 47,	survival::survfit(), 208, 383
51, 53, 57, 59, 62, 64, 66, 68, 71, 73,	survival::survreg(), 90, 91, 210, 385
74, 76, 78, 81, 83, 87, 88, 90	<pre>survreg_tidiers(tidy.survreg), 384</pre>
stats::power.t.test(), <i>349</i> , <i>350</i>	svd(), 216, 397, 399, 400, 403
stats::prcomp(), 73, 74, 350, 351	svd_tidiers, <i>74</i> , <i>351</i>
stats::predict(), 11, 15, 29, 34, 91	svd_tidiers(tidy_svd),400
stats::predict.glm(), 31 , 55	svyolr_tidiers(tidy.svyolr),387
stats::predict.lm(), 43	systemfit::systemfit(), 389, 390
stats::predict.loess(), 50	<pre>systemfit_tidiers(tidy.systemfit), 389</pre>
stats::predict.nls(), 63	
stats::predict.smooth.spline(), 86	tibble::as_tibble(), 275, 390, 391
stats::residuals(), <i>11</i> , <i>15</i> , <i>29</i> , <i>34</i> , <i>91</i>	tibble::as_tibble.table(),391
stats::residuals.glm(), 31,55	tibble::tibble, 8, 10, 13, 15, 18, 20, 22, 24
stats::rstandard.glm(), $31,55$	26, 28, 30, 33, 36, 38, 40, 41, 45, 47
stats::smooth.spline(), 85, 86, 197	51, 53, 57, 59, 62, 64, 66, 68, 71, 73
stats::spectrum(), 371	74, 76, 78, 81, 83, 87–90, 151, 219,
stats::stl(), 89	221, 259, 260, 267, 275, 351, 391,
stats::summarv.aov(), 226	<i>398</i> , <i>401</i> , <i>403</i>

tibble::tibble(), 9, 11, 13, 15, 16, 20, 21,	tidy.betareg, 231
23, 24, 26, 27, 29, 31, 32, 34–36,	tidy.betareg(), 230
38–40, 42, 43, 46–52, 55, 57, 58, 60,	tidy.biglm, 109, 233
61, 63, 65, 67–69, 71, 73, 75, 77, 79,	tidy.binDesign, 111, 235, 237
81, 84–87, 90, 91, 97, 99–112,	tidy.binWidth, <i>111</i> , <i>235</i> , 236
114–117, 119–123, 125, 126, 128,	tidy.boot, 237
130, 132–151, 153, 156–183,	tidy.btergm, 239
185–195, 197–202, 204–217, 222,	tidy.cch, 16, 91, 100, 113, 120, 187, 205,
224, 225, 227, 229, 230, 232,	207, 208, 210, 223, 241, 254, 353,
234–236, 238, 240, 241, 243, 245,	381–383, 385
247, 249, 250, 252, 254, 256, 257,	tidy.character(tidy.numeric), 338
262, 263, 265, 268, 270, 272, 274,	tidy.cld, 242, 250, 283, 375
276, 278, 279, 281, 282, 285, 288,	tidy.clm, 14, 72, 114, 116, 186, 214, 244,
290, 292, 294, 296, 297, 299, 300,	247, 348, 388
302, 303, 306, 308, 312, 314, 315,	tidy.clmm, 14, 72, 114, 116, 186, 214, 245,
317, 319, 321, 323, 325, 327, 329,	246, <i>348</i> , <i>388</i>
330, 332, 333, 335, 337, 339, 341,	tidy.coeftest, 248
342, 344, 345, 348, 349, 353, 355,	tidy.confint.glht, 243, 250, 283, 375
356, 358, 359, 362, 364, 365, 367,	tidy.confusionMatrix, 251
369, 371, 372, 374, 375, 377, 378,	tidy.coxph, 16, 91, 100, 113, 120, 187, 205,
380, 382, 383, 385, 388, 389, 392,	207, 208, 210, 223, 242, 253, 353,
393, 395, 396, 400	381–383, 385
tidy, 14, 63, 114, 116, 177, 186, 214, 245,	tidy.crr, 122, 255
247, 337, 348, 388	tidy.cv.glmnet, 124, 143, 257, 285
tidy(), 72, 97, 132, 217, 223–226, 228, 232,	tidy.data.frame(data.frame_tidiers), 95
234, 235, 237, 238, 240, 242, 243,	tidy.density, 93, 94, 96, 98, 99, 221, 259,
249, 250, 252, 254, 256, 257, 262,	260, 275, 338
263, 265, 267, 268, 271, 272, 274,	tidy.dist, 93, 94, 96, 98, 99, 221, 259, 260,
276, 278, 280, 281, 283, 285,	275, 338
290–292, 294, 296, 297, 299, 300,	tidy.drc, 21, 126, 261
302, 304, 308, 312, 314, 315, 317,	tidy.durbinWatsonTest
319, 321, 323, 325, 327, 329,	(durbinWatsonTest_tidiers), 97
331–333, 336, 341, 342, 344, 345,	tidy.emmGrid, 262, 312, 357, 379
353, 355, 357, 358, 360, 364, 366,	tidy.epi.2by2, 264
367, 369, 371, 375, 377, 379,	tidy.ergm, <i>127</i> , 266
381–383, 385, 390, 392, 393, 395,	tidy.factanal, 23, 129, 268
396, 398, 400, 403	tidy.felm, 25, 269
tidy.aareg, 16, 91, 100, 113, 120, 187, 205,	tidy.fitdistr, <i>132</i> , 271
207, 208, 210, 222, 242, 254, 353,	tidy.fixest, 27, 273
381–383, 385	tidy.ftable, 93, 94, 96, 98, 99, 221, 259,
tidy.acf, 223, 371, 392, 396	260, 275, 338
tidy.anova, 102, 103, 224, 226, 228, 314, 393	tidy.Gam, <i>136</i> , 275
tidy.anova(), <u>276</u>	tidy.gam, <i>137</i> , 277
tidy.aov, 102, 103, 225, 226, 228, 314, 393	tidy.gam(), 276
tidy.aovlist, 102, 103, 225, 226, 227, 314,	tidy.garch, <i>139</i> , 279
393	tidy.geeglm, 280
tidy.Arima, <i>105</i> , 228	tidy.glht, 243, 250, 282, 375
tidv.betamfx. 9. 56, 106, 166, 229, 323	tidy.glm.32,43,141,154,202,212,283.

304, 306, 329, 377	tidy.nls, <i>63</i> , <i>177</i> , 336
tidy.glmnet, 124, 143, 257, 284	tidy.NULL(null_tidiers),219
tidy.glmRob, 46, 144, 158, 286, 309	tidy.numeric, 93, 94, 96, 98, 99, 221, 259,
tidy.glmrob, 35, 48, 160, 287, 311	260, 275, 338
tidy.gmm, <i>146</i> , 289	tidy.optim(tidy_optim), 399
tidy.htest, <i>37</i> , 291, <i>341</i> , <i>350</i>	tidy.orcutt, <i>178</i> , 339
tidy.irlba(tidy_irlba), 397	tidy.pairwise.htest, 37, 292, 340, 350
tidy.ivreg, 39, 149, 293	tidy.pam, 65, 180, 341
tidy.kappa, 295	tidy.plm, 67, 182, 343
tidy.kde, 297	tidy.poissonmfx(tidy.mfx), 322
tidy.Kendall, 298	tidy.poLCA, 69, 183, 345
tidy.kmeans, 41, 150, 300	tidy.polr, 14, 72, 114, 116, 186, 214, 245,
tidy.lavaan, 152, 301	247, 347, 388
<pre>tidy.leveneTest (leveneTest_tidiers),</pre>	tidy.polr(), 388
217	tidy.Polygon(sp_tidiers),219
tidy.leveneTest(), 225-227	tidy.Polygons(sp_tidiers),219
tidy.Line(sp_tidiers), 219	tidy.power.htest, 37, 292, 341, 349
tidy.Lines (sp_tidiers), 219	tidy.prcomp, 74, 350, 398, 402
tidy.list(list_tidiers), 218	tidy.probitmfx(tidy.mfx), 322
tidy.lm, 32, 43, 141, 154, 202, 212, 284, 303,	tidy.pyears, 16, 91, 100, 113, 120, 187, 205
306, 329, 377	207, 208, 210, 223, 242, 254, 352,
tidy.lm(), <i>344</i> , <i>374</i> , <i>377</i>	381–383, 385
tidy.lm.beta, 32, 43, 141, 154, 202, 212,	tidy.rcorr, 354
284, 304, 305, 329, 377	tidy.ref.grid, 263, 312, 356, 379
tidy.lmodel2, <i>156</i> , 307	tidy.regsubsets,358
tidy.1mRob, 46, 144, 158, 287, 309	tidy.ridgelm, <i>189</i> , 359
tidy.lmrob, 35, 48, 160, 288, 310	tidy.rlm, 75, 190, 360
tidy.logical(tidy.numeric),338	tidy.rlm(), 34, 46, 48, 159, 287, 288, 309,
tidy.logitmfx(tidy.mfx), 322	311
tidy.lsmobj, 263, 311, 357, 379	tidy.rma,361
tidy.manova, 102, 103, 225, 226, 228, 313,	tidy.roc,363
393	tidy.rq, 61, 80, 82, 175, 194, 336, 365, 367
tidy.map, 315	tidy.rqs, 61, 80, 82, 175, 194, 336, 366, 366
tidy.margins, 316	tidy.sarlm, <i>84</i> , <i>196</i> ,368
tidy.Mclust, <i>52</i> , 318	tidy.SpatialLinesDataFrame
tidy.mediate, 320	(sp_tidiers), 219
tidy.mfx, 9, 56, 106, 166, 230, 322	tidy.SpatialPolygons(sp_tidiers),219
tidy.mjoint, <i>167</i> , 324	tidy.SpatialPolygonsDataFrame
tidy.mle2, 326	(sp_tidiers), 219
tidy.mlm, 32, 43, 141, 154, 202, 212, 284,	tidy.spec, 224, 370, 392, 396
304, 306, 328, 377	tidy.speedglm, 88, 199, 201, 371, 374
tidy.mlm(), 303	tidy.speedlm, 88, 199, 201, 372, 373
tidy.mlogit, 60, 169, 330	tidy.summary.glht, 243, 250, 283, 374
tidy.muhaz, <i>171</i> , 331	tidy.summary.glht(),282
tidy.multinom, <i>172</i> , 332	tidy.summary.lm, 32, 43, 141, 154, 202, 212
tidy.negbin, <i>174</i> , 334	284, 304, 306, 329, 376
tidy.negbinmfx(tidy.mfx), 322	tidy.summary_emm, 263, 312, 357, 378
tidy.nlrq, 61, 80, 82, 175, 194, 335, 366, 367	<pre>tidy.summaryDefault(summary_tidiers),</pre>

```
220
tidy.survdiff, 16, 91, 100, 113, 120, 187,
         205, 207, 208, 210, 223, 242, 254,
         353, 380, 382, 383, 385
tidy.survexp, 16, 91, 100, 113, 120, 187,
         205, 207, 208, 210, 223, 242, 254,
         353, 381, 381, 383, 385
tidy.survfit, 16, 91, 100, 113, 120, 187,
         205, 207, 208, 210, 223, 242, 254,
         353, 381, 382, 383, 385
tidy.survreg, 16, 91, 100, 113, 120, 187,
         205, 207, 208, 210, 223, 242, 254,
         353, 381–383, 384
tidy.svyglm, 386
tidy.svyolr, 14, 72, 114, 116, 186, 214, 245,
         247, 348, 387
tidy.systemfit, 389
tidy.table, 390
tidy.ts, 224, 371, 391, 396
tidy. TukeyHSD, 102, 103, 225, 226, 228, 314,
         393
tidy.varest, 394
tidy.zoo, 224, 371, 392, 395
tidy_irlba, 74, 216, 218, 351, 397, 400, 402,
         403
tidy_optim, 216, 218, 398, 399, 402, 403
tidy_optim(), 327
tidy_svd, 74, 216, 218, 351, 398, 400, 400,
         403
tidy_svd(), 397
tidy_xyz, 216, 218, 398, 400, 402, 402
tidyr::pivot_longer(), 391
tseries::garch(), 138, 139, 279, 280
vars::VAR(), 215, 394, 395
vars_tidiers(tidy.varest), 394
xyz_tidiers(tidy_xyz), 402
zoo::zoo(), 396
zoo_tidiers (tidy.zoo), 395
```