Package 'performance'

April 24, 2019

Type Package

Title Assessment of Regression Models Performance

Version 0.1.0

Date 2019-04-23

Maintainer Daniel Lüdecke <d.luedecke@uke.de>

Description Utilities for computing measures to assess model quality, which are not directly provided by R's 'base' or 'stats' packages. These include e.g. measures like r-squared, intraclass correlation coefficient (Nakagawa, Johnson & Schielzeth (2017) <doi:10.1098/rsif.2017.0213>), root mean squared error or functions to check models for overdispersion, singularity or zero-inflation and more. Functions apply to a large variety of regression models, including generalized linear models, mixed effects models and Bayesian models.

URL https://easystats.github.io/performance/

BugReports https://github.com/easystats/performance/issues

Imports insight, bayestestR

Suggests brms, covr, graphics, glmmTMB, ggplot2, lme4, loo, Matrix, MASS, mlogit, nlme, psych, rmarkdown, rstanarm, rstantools, scales, testthat

License GPL-3

Encoding UTF-8

LazyData true

RoxygenNote 6.1.1

NeedsCompilation no

Author Daniel Lüdecke [aut, cre] (<https://orcid.org/0000-0002-8895-3206>), Dominique Makowski [aut, ctb] (<https://orcid.org/0000-0001-5375-9967>)

Repository CRAN

Date/Publication 2019-04-24 16:40:03 UTC

R topics documented:

`binned_residuals` *Binned residuals for logistic regression*

Description

Check model quality of logistic regression models.

Usage

```
binned_residuals(model, term = NULL, n_nins = NULL)
```

Arguments

model	A glm-object with binomial-family.
term	Name of independent variable from x. If not NULL, average residuals for the categories of term are plotted; else, average residuals for the estimated probabilities of the response are plotted.
n_nins	Numeric, the number of bins to divide the data. If n_nins = NULL, the square root of the number of observations is taken.

Details

Binned residual plots are achieved by "dividing the data into categories (bins) based on their fitted values, and then plotting the average residual versus the average fitted value for each bin." *(Gelman, Hill 2007: 97)*. If the model were true, one would expect about 95% of the residuals to fall inside the error bounds.

If term is not NULL, one can compare the residuals in relation to a specific model predictor. This may be helpful to check if a term would fit better when transformed, e.g. a rising and falling pattern of residuals along the x-axis (the pattern is indicated by a green line) is a signal to consider taking the logarithm of the predictor (cf. Gelman and Hill 2007, pp. 97ff).

Value

A data frame representing the data that is mapped to the plot, which is automatically plotted. In case all residuals are inside the error bounds, points are black. If some of the residuals are outside the error bounds (indicates by the grey-shaded area), blue points indicate residuals that are OK, while red points indicate model under- or overfitting for the related range of estimated probabilities.

References

Gelman, A., & Hill, J. (2007). Data analysis using regression and multilevel/hierarchical models. Cambridge; New York: Cambridge University Press.

Examples

```
model <- glm(vs ~ wt + mpg, data = mtcars, family = "binomial")
binned_residuals(model)
```

---

check_convergence *Convergence test for mixed effects models*

Description

check_convergence() provides an alternative convergence test for merMod-objects.

Usage

```
check_convergence(x, tolerance = 0.001)
```

Arguments

x	A merMod-object.
tolerance	Indicates up to which value the convergence result is accepted. The smaller tolerance is, the stricter the test will be.

Details

check_convergence() provides an alternative convergence test for merMod-objects, as discussed here and suggested by Ben Bolker in this comment. Further details can be found in convergence.

Value

TRUE if convergence is fine and FALSE if convergence is suspicious. Additionally, the convergence value is returned as attribute.

Examples

```
library(lme4)
data(cbpp)
set.seed(1)
cbpp$x <- rnorm(nrow(cbpp))
cbpp$x2 <- runif(nrow(cbpp))

model <- glmer(
  cbind(incidence, size - incidence) ~ period + x + x2 + (1 + x | herd),
  data = cbpp,
  family = binomial()
)

check_convergence(model)
```

---

check_overdispersion *Check overdispersion of GL(M)M's*

Description

check_overdispersion() checks generalized linear (mixed) models for overdispersion.

Usage

```
check_overdispersion(x, ...)
```

Arguments

x	Fitted model of class merMod, glmmTMB, glm, or glm.nb (package **MASS**).
...	Currently not used.

Details

A p-value < .05 indicates overdispersion. For merMod- and glmmTMB-objects, check_overdispersion() is based on the code in the GLMM FAQ, section *How can I deal with overdispersion in GLMMs?*. Note that this function only returns an *approximate* estimate of an overdispersion parameter, and is probably inaccurate for zero-inflated mixed models (fitted with glmmTMB). The same code is also used to check overdispersion for negative binomial models.

For Poisson-models, the overdispersion test is based on the code from *Gelman and Hill (2007), page 115.*

Overdispersion can be fixed by either modelling the dispersion parameter, or by choosing a different distributional family (like Quasi-Poisson, or negative binomial, see *Gelman and Hill (2007), pages 115-116*).

Value

A list with results from the overdispersion test, like chi-squared statistics, p-value or dispersion ratio.

References

- Bolker B et al. (2017): GLMM FAQ.
- Gelman, A., & Hill, J. (2007). Data analysis using regression and multilevel/hierarchical models. Cambridge; New York: Cambridge University Press.

Examples

```
library(glmmTMB)
data(Salamanders)

m <- glm(count ~ spp + mined, family = poisson, data = Salamanders)
check_overdispersion(m)

m <- glmmTMB(
  count ~ mined + spp + (1 | site),
  family = poisson,
  data = Salamanders
)
check_overdispersion(m)
```

check_singularity *Check mixed models for boundary fits*

Description

Check mixed models for boundary fits.

Usage

```
check_singularity(x, tolerance = 1e-05, ...)
```

Arguments

x	A mixed model.
tolerance	Indicates up to which value the convergence result is accepted. The larger tolerance is, the stricter the test will be.
...	Currently not used.

Details

If a model is "singular", this means that some dimensions of the variance-covariance matrix have been estimated as exactly zero. This often occurs for mixed models with complex random effects structures.

"While singular models are statistically well defined (it is theoretically sensible for the true maximum likelihood estimate to correspond to a singular fit), there are real concerns that (1) singular fits correspond to overfitted models that may have poor power; (2) chances of numerical problems and mis-convergence are higher for singular models (e.g. it may be computationally difficult to compute profile confidence intervals for such models); (3) standard inferential procedures such as Wald statistics and likelihood ratio tests may be inappropriate." (*lme4 Reference Manual*)

There is no gold-standard about how to deal with singularity and which random-effects specification to choose. Beside using fully Bayesian methods (with informative priors), proposals in a frequentist framework are:

- avoid fitting overly complex models, such that the variance-covariance matrices can be estimated precisely enough (*Matuschek et al. 2017*)
- use some form of model selection to choose a model that balances predictive accuracy and overfitting/type I error (*Bates et al. 2015*, *Matuschek et al. 2017*)
- "keep it maximal", i.e. fit the most complex model consistent with the experimental design, removing only terms required to allow a non-singular fit (*Barr et al. 2013*)

Value

TRUE if the model fit is singular.

References

- Bates D, Kliegl R, Vasishth S, Baayen H. Parsimonious Mixed Models. arXiv:1506.04967, June 2015.
- Barr DJ, Levy R, Scheepers C, Tily HJ. Random effects structure for confirmatory hypothesis testing: Keep it maximal. Journal of Memory and Language, 68(3):255-278, April 2013.
- Matuschek H, Kliegl R, Vasishth S, Baayen H, Bates D. Balancing type I error and power in linear mixed models. Journal of Memory and Language, 94:305-315, 2017.
- lme4 Reference Manual, https://cran.r-project.org/package=lme4

Examples

```
library(lme4)
data(sleepstudy)
set.seed(1)
sleepstudy$mygrp <- sample(1:5, size = 180, replace = TRUE)
sleepstudy$mysubgrp <- NA
for (i in 1:5) {
  filter_group <- sleepstudy$mygrp == i
  sleepstudy$mysubgrp[filter_group] <-
    sample(1:30, size = sum(filter_group), replace = TRUE)
}

model <- lmer(
  Reaction ~ Days + (1 | mygrp / mysubgrp) + (1 | Subject),
  data = sleepstudy
)

check_singularity(model)
```

---

check_zeroinflation *Check for zero-inflation in count models*

Description

`check_zeroinflation()` checks whether count models are over- or underfitting zeros in the outcome.

Usage

```
check_zeroinflation(x, tolerance = 0.05)
```

Arguments

`x`	Fitted model of class `merMod`, `glmmTMB`, `glm`, or `glm.nb` (package **MASS**).
`tolerance`	The tolerance for the ratio of observed and predicted zeros to considered as over- or underfitting zeros. A ratio between 1 +/- `tolerance` is considered as OK, while a ratio beyond or below this treshold would indicate over- or underfitting.

Details

If the amount of observed zeros is larger than the amount of predicted zeros, the model is underfitting zeros, which indicates a zero-inflation in the data. In such cases, it is recommended to use negative binomial or zero-inflated models.

Value

A list with information about the amount of predicted and observed zeros in the outcome, as well as the ratio between these two values.

Examples

```
library(glmmTMB)
data(Salamanders)
m <- glm(count ~ spp + mined, family = poisson, data = Salamanders)
check_zeroinflation(m)
```

---

cronbachs_alpha *Cronbach's Alpha for Items or Scales*

Description

Compute various measures of internal consistencies for tests or item-scales of questionnaires.

Usage

```
cronbachs_alpha(x)
```

Arguments

x	A matrix or a data frame.

Details

The Cronbach's Alpha value for x. A value closer to 1 indicates greater internal consistency, where usually following rule of thumb is applied to interprete the results: $\alpha < 0.5$ is unacceptable, $0.5 < \alpha < 0.6$ is poor, $0.6 < \alpha < 0.7$ is questionable, $0.7 < \alpha < 0.8$ is acceptable, and everything > 0.8 is good or excellent.

Value

The Cronbach's Alpha value for x.

Examples

```
data(mtcars)
x <- mtcars[, c("cyl", "gear", "carb", "hp")]
cronbachs_alpha(x)
```

error_rate *Error Rate for Logistic Regression Models*

Description

Compute the error rate for logistic regression models, which is a crude measure for the model fit.

Usage

```
error_rate(model)
```

Arguments

model	A glm-object with binomial-family.

Details

The error rate is a crude measure for model fit for logistic regression models. It is defined as the proportion of cases for which the deterministic prediction is wrong, i.e. the proportion where the the predicted probability is above 0.5, although y = 0 (and vice versa). In general, the error rate should be below 0.5 (i.e. 50%), the closer to zero, the better. Furthermore, the error rate of the full model should be considerably below the null model's error rate (cf. Gelman and Hill 2007, pp. 99).

Value

A list with four values: the error rate of the full and the null model, as well as the chi-squared and p-value from the Likelihood-Ratio-Test between the full and null model.

References

Gelman, A., & Hill, J. (2007). Data analysis using regression and multilevel/hierarchical models. Cambridge; New York: Cambridge University Press.

Examples

```
data(mtcars)
m <- glm(am ~ mpg + hp + cyl, data = mtcars, family= binomial)
error_rate(m)
```

`hosmer_lemeshow` *Hosmer-Lemeshow goodness-of-fit test*

Description

Check model quality of logistic regression models.

Usage

```
hosmer_lemeshow(model, n_bins = 10)
```

Arguments

`model`	A `glm`-object with binomial-family.
`n_bins`	Numeric, the number of bins to divide the data.

Details

A well-fitting model shows *no* significant difference between the model and the observed data, i.e. the reported p-value should be greater than 0.05.

Value

An object of class `hoslem_test` with following values: `chisq`, the Hosmer-Lemeshow chi-squared statistic; `df`, degrees of freedom and `p.value` the p-value for the goodness-of-fit test.

References

Hosmer, D. W., & Lemeshow, S. (2000). Applied Logistic Regression. Hoboken, NJ, USA: John Wiley & Sons, Inc. doi: 10.1002/0471722146

Examples

```
model <- glm(vs ~ wt + mpg, data = mtcars, family = "binomial")
hosmer_lemeshow(model)
```

icc *The Intraclass Correlation Coefficient (ICC) for mixed models*

Description

This function calculates the intraclass-correlation coefficient (icc) - sometimes also called *variance partition coefficient* (vpc) - for mixed effects models. The ICC is calculated for `merMod` (**lme4**), `glmmTMB` (**glmmTMB**), `MixMod` (**GLMMadpative**), `lme` (**nlme**), `mixed` (**afex**), and `stanreg` (**rstanarm**) objects and can be interpreted as "the proportion of the variance explained by the grouping structure in the population" *(Hox 2010: 15)*. For models fitted with the **brms**-package, a variance decomposition based on the posterior predictive distribution is calculated (see 'Details').

Usage

```
icc(model, ...)

## S3 method for class 'brmsfit'
icc(model, re.form = NULL, robust = TRUE,
  ci = 0.95, ...)
```

Arguments

`model`	A mixed effects model of class `merMod`, `glmmTMB`, `MixMod`, `lme`, `mixed`, `stanreg` or `brmsfit`.
`...`	Currently not used.
`re.form`	Formula containing group-level effects to be considered in the prediction. If `NULL` (default), include all group-level effects. Else, for instance for nested models, name a specific group-level effect to calculate the variance decomposition for this group-level.
`robust`	Logical, if `TRUE`, the median instead of mean is used to calculate the central tendency of the variances.
`ci`	The Credible Interval level.

Details

Calculation of the ICC

The ICC is calculated by dividing the random effect variance, σ_i^2, by the total variance, i.e. the sum of the random effect variance and the residual variance, σ_ϵ^2.

Adjusted and conditional ICC

`icc()` calculates an adjusted and conditional ICC, which take all sources of uncertainty (i.e. of *all random effects*) into account. While the adjusted ICC only relates to the random effects, the conditional ICC also takes the fixed effects variances into account (see *Nakagawa et al. 2017*). Typically, the *adjusted* ICC is of interest when the analysis of random effects is of interest. `icc()`

returns a meaningful ICC also for more complex random effects structures, like models with random slopes or nested design (more than two levels) and is applicable for models with other distributions than Gaussian. For more details on the computation of the variances, see `get_variance`.

ICC for unconditional and conditional models

Usually, the ICC is calculated for the null model ("unconditional model"). However, according to *Raudenbush and Bryk (2002)* or *Rabe-Hesketh and Skrondal (2012)* it is also feasible to compute the ICC for full models with covariates ("conditional models") and compare how much, e.g., a level-2 variable explains the portion of variation in the grouping structure (random intercept).

ICC for specific group-levels

The proportion of variance for specific levels related to each other (e.g., similarity of level-1-units within level-2-units or level-2-units within level-3-units) must be calculated manually. Use `get_variance` to get the random intercept variances (between-group-variances) and residual variance of the model, and calculate the ICC for the various level correlations.

For example, for the ICC between level 1 and 2:

```
sum(insight::get_variance_intercept(model)) /
(sum(insight::get_variance_intercept(model)) + insight::get_variance_residual(model))
```

For for the ICC between level 2 and 3:

```
insight::get_variance_intercept(model)[2] /
sum(insight::get_variance_intercept(model))
```

ICC for brms-models

If `model` is of class `brmsfit`, `icc()` calculates a variance decomposition based on the posterior predictive distribution. In this case, first, the draws from the posterior predictive distribution *not conditioned* on group-level terms (`posterior_predict(..., re.form = NA)`) are calculated as well as draws from this distribution *conditioned* on *all random effects* (by default, unless specified else in `re.form`) are taken. Then, second, the variances for each of these draws are calculated. The "ICC" is then the ratio between these two variances. This is the recommended way to analyse random-effect-variances for non-Gaussian models. It is then possible to compare variances accross models, also by specifying different group-level terms via the `re.form`-argument.

Sometimes, when the variance of the posterior predictive distribution is very large, the variance ratio in the output makes no sense, e.g. because it is negative. In such cases, it might help to use `robust = TRUE`.

Value

A list with two values, the adjusted and conditional ICC. For models of class `brmsfit`, a list with two values, the decomposed ICC as well as the credible intervals for this ICC.

References

- Hox, J. J. (2010). Multilevel analysis: techniques and applications (2nd ed). New York: Routledge.
- Nakagawa, S., Johnson, P. C. D., & Schielzeth, H. (2017). The coefficient of determination R2 and intra-class correlation coefficient from generalized linear mixed-effects models revisited and expanded. Journal of The Royal Society Interface, 14(134), 20170213. doi: 10.1098/rsif.2017.0213
- Rabe-Hesketh, S., & Skrondal, A. (2012). Multilevel and longitudinal modeling using Stata (3rd ed). College Station, Tex: Stata Press Publication.
- Raudenbush, S. W., & Bryk, A. S. (2002). Hierarchical linear models: applications and data analysis methods (2nd ed). Thousand Oaks: Sage Publications.

Examples

```
library(lme4)
model <- lme4::lmer(Sepal.Length ~ Petal.Length + (1 | Species), data = iris)
icc(model)
```

---

item_difficulty *Difficulty of Questionnaire Items*

Description

Compute various measures of internal consistencies for tests or item-scales of questionnaires.

Usage

```
item_difficulty(x)
```

Arguments

x	Depending on the function, x may be a matrix as returned by the cor-function, or a data frame with items (e.g. from a test or questionnaire).

Details

This function calculates the item difficutly, which should range between 0.2 and 0.8. Lower values are a signal for more difficult items, while higher values close to one are a sign for easier items. The ideal value for item difficulty is `p + (1 - p) / 2`, where `p = 1 / max(x)`. In most cases, the ideal item difficulty lies between 0.5 and 0.8.

Value

A data frame with three columns: The name(s) of the item(s), the item difficulties for each item, and the ideal item difficulty.

Examples

```
data(mtcars)
x <- mtcars[, c("cyl", "gear", "carb", "hp")]
item_difficulty(x)
```

`item_intercor` *Mean Inter-Item-Correlation*

Description

Compute various measures of internal consistencies for tests or item-scales of questionnaires.

Usage

```
item_intercor(x, method = c("pearson", "spearman", "kendall"))
```

Arguments

`x`	A matrix as returned by the `cor`-function, or a data frame with items (e.g. from a test or questionnaire).
`method`	Correlation computation method. May be one of `"spearman"` (default), `"pearson"` or `"kendall"`. You may use initial letter only.

Details

This function calculates a mean inter-item-correlation, i.e. a correlation matrix of `x` will be computed (unless `x` is already a matrix as returned by the `cor()`-function) and the mean of the sum of all item's correlation values is returned. Requires either a data frame or a computed `cor()`-object.

"Ideally, the average inter-item correlation for a set of items should be between .20 and .40, suggesting that while the items are reasonably homogenous, they do contain sufficiently unique variance so as to not be isomorphic with each other. When values are lower than .20, then the items may not be representative of the same content domain. If values are higher than .40, the items may be only capturing a small bandwidth of the construct." *(Piedmont 2014)*

Value

The mean inter-item-correlation value for `x`.

References

Piedmont RL. 2014. Inter-item Correlations. In: Michalos AC (eds) Encyclopedia of Quality of Life and Well-Being Research. Dordrecht: Springer, 3303-3304. doi: 10.1007/9789400707535_1493

Examples

```
data(mtcars)
x <- mtcars[, c("cyl", "gear", "carb", "hp")]
item_intercor(x)
```

`item_reliability` *Reliability Test for Items or Scales*

Description

Compute various measures of internal consistencies for tests or item-scales of questionnaires.

Usage

```
item_reliability(x, standardize = FALSE, digits = 3)
```

Arguments

`x`	A matrix or a data frame.
`standardize`	Logical, if `TRUE`, the data frame's vectors will be standardized. Recommended when the variables have different measures / scales.
`digits`	Amount of digits for returned values.

Details

This function calculates the item discriminations (corrected item-total correlations for each item of `x` with the remaining items) and the Cronbach's alpha for each item, if it was deleted from the scale. The absolute value of the item discrimination indices should be above 0.1. An index between 0.1 and 0.3 is considered as "fair", while an index above 0.3 (or below -0.3) is "good". Items with low discrimination indices are often ambiguously worded and should be examined. Items with negative indices should be examined to determine why a negative value was obtained (e.g. reversed answer categories regarding positive and negative poles).

Value

A data frame with the corrected item-total correlations (*item discrimination*, column `item_discrimination`) and Cronbach's Alpha (if item deleted, column `alpha_if_deleted`) for each item of the scale, or `NULL` if data frame had too less columns.

Examples

```
data(mtcars)
x <- mtcars[, c("cyl", "gear", "carb", "hp")]
item_reliability(x)
```

`item_split_half` *Split-Half Reliability*

Description

Compute various measures of internal consistencies for tests or item-scales of questionnaires.

Usage

```
item_split_half(x, digits = 3)
```

Arguments

`x`	A matrix or a data frame.
`digits`	Amount of digits for returned values.

Details

This function calculates the split-half reliability for items in `x`, including the Spearman-Brown adjustment. Splitting is done by selecting odd versus even columns in `x`. A value closer to 1 indicates greater internal consistency.

Value

A list with two elements: the split-half reliability `splithalf` and the Spearman-Brown corrected split-half reliability `spearmanbrown`.

References

Spearman C. 1910. Correlation calculated from faulty data. British Journal of Psychology (3): 271-295. doi: 10.1111/j.20448295.1910.tb00206.x

Brown W. 1910. Some experimental results in the correlation of mental abilities. British Journal of Psychology (3): 296-322. doi: 10.1111/j.20448295.1910.tb00207.x

Examples

```
data(mtcars)
x <- mtcars[, c("cyl", "gear", "carb", "hp")]
item_split_half(x)
```

looic *LOO-related Indices for Bayesian regressions.*

Description

Compute LOOIC (leave-one-out cross-validation (LOO) information criterion) and ELPD (expected log predictive density) for Bayesian regressions.

Usage

```
looic(model)
```

Arguments

model	A Bayesian regression model.

Value

A list with four elements, the ELPD, LOOIC and their standard errors.

Examples

```
library(rstanarm)

model <- stan_glm(mpg ~ wt + cyl, data = mtcars, chains = 1, iter = 500)
looic(model)
```

model_performance *Model Performance*

Description

See the documentation for your object's class: lm, glm, mixed models and Bayesian models. `compare_performance()` computes indices of model performance for different models.

Usage

```
model_performance(model, ...)

compare_performance(..., metrics = "all")
```

Arguments

model	Statistical model.
...	Arguments passed to or from other methods, resp. for compare_performance(), one or multiple model objects (also of different classes).
metrics	Can be "all" or a character vector of metrics to be computed. See related documentation of object's class for details.

Value

For model_performance(), a data frame (with one row) and one column per "index" (see metrics). For compare_performance(), the same data frame with one row per model.

Examples

```
library(lme4)

m1 <- lm(mpg ~ wt + cyl, data = mtcars)
model_performance(m1)

m2 <- glm(vs ~ wt + mpg, data = mtcars, family = "binomial")
m3 <- lmer(Petal.Length ~ Sepal.Length + (1 | Species), data = iris)
compare_performance(m1, m2, m3)
```

---

model_performance.lm *Performance of (Generalized) Linear Models*

Description

Compute indices of model performance for (generalized) linear models.

Usage

```
## S3 method for class 'lm'
model_performance(model, metrics = "all", ...)

## S3 method for class 'glm'
model_performance(model, metrics = "all", ...)
```

Arguments

model	Object of class lm or glm.
metrics	Can be "all" or a character vector of metrics to be computed (some of c("AIC", "BIC", "R2", "RMSE"
...	Arguments passed to or from other methods.

Value

A data frame (with one row) and one column per "index" (see `metrics`).

Examples

```
model <- lm(mpg ~ wt + cyl, data = mtcars)
model_performance(model)

model <- glm(vs ~ wt + mpg, data = mtcars, family = "binomial")
model_performance(model)
```

`model_performance.merMod`

Performance of Mixed Models

Description

Compute indices of model performance for mixed models.

Usage

```
## S3 method for class 'merMod'
model_performance(model, metrics = "all", ...)
```

Arguments

`model`	Object of class `merMod`, `glmmTMB`, `lme` or `MixMod`.
`metrics`	Can be `"all"` or a character vector of metrics to be computed (some of `c("AIC", "BIC", "R2", "ICC",`
`...`	Arguments passed to or from other methods.

Value

A data frame (with one row) and one column per "index" (see `metrics`).

Examples

```
library(lme4)
model <- lmer(Petal.Length ~ Sepal.Length + (1 | Species), data = iris)
model_performance(model)
```

`model_performance.stanreg` *Performance of Bayesian Models*

Description

Compute indices of model performance for (general) linear models.

Usage

```
## S3 method for class 'stanreg'
model_performance(model, metrics = "all", ci = 0.9,
  ...)
```

Arguments

`model`	Object of class `stanreg` or `brmsfit`.
`metrics`	Can be `"all"` or a character vector of metrics to be computed (some of `c("LOOIC", "R2", "R2_adj")`).
`ci`	The Credible Interval level for R2.
`...`	Arguments passed to or from other methods.

Value

A data frame (with one row) and one column per "index" (see `metrics`).

References

Gelman, A., Goodrich, B., Gabry, J., & Vehtari, A. (2018). R-squared for Bayesian regression models. The American Statistician, The American Statistician, 1-6.

See Also

r2_bayes

Examples

```
library(rstanarm)
model <- stan_glm(mpg ~ wt + cyl, data = mtcars, chains = 1, iter = 500)
model_performance(model)
```

mse *Mean Square Error of Linear Models*

Description

Compute mean square error of linear models.

Usage

```
mse(model)
```

Arguments

model	Linear model of class lm, merMod (**lme4**) or lme (**nlme**).

Details

The mean square error is the mean of the sum of squared residuals, i.e. it measures the average of the squares of the errors. Lower values (closer to zero) indicate better fit.

Value

Numeric, the mean square error of model.

Examples

```
data(mtcars)
m <- lm(mpg ~ hp + gear, data = mtcars)
mse(m)
```

principal_components *Principal Components Analysis*

Description

This function performs a principal component analysis and returns the loadings (of the unrotated matrix) as data frame, or returns a rotated matrix of the loadings (if rotation is not NULL).

Usage

```
principal_components(x, rotation = NULL, n_comp = NULL)
```

Arguments

x	A data frame or a prcomp-object.
rotation	Rotation of the factor loadings. May be one of "varimax", "quartimax", "promax", "oblimin", "si or "none". If rotatoin = NULL, loadings for the principal components from the unrotated matrix are returned.
n_comp	Number of components to extract. If rotation = "varmiax" and n_comp = NULL, number of components is based on the Kaiser-criteria.

Details

The print()-method has a cutoff-argument, which is a scalar between 0 and 1, indicating which (absolute) values from the *rotated* loadings (i.e. when rotation is *not* NULL) should be blank in the output. By default, all loadings between -.1 and .1 are not shown.

Value

If rotation = NULL, a data frame with all loadings of principal components. Else, a rotated loadings matrix, as data frame. Details on the variance components are saved as attributes.

Examples

```
data(iris)
principal_components(iris[, 1:4])

data(iris)
principal_components(iris[, 1:4], rotation = "varimax", n_comp = 2)

pr <- principal_components(iris[, 1:4], rotation = "varimax", n_comp = 2)

# show all
print(pr, cutoff = .001)

# show only some
print(pr, cutoff = .5)
```

---

r2 *Compute the model's R2*

Description

Calculate the R2 value for different model objects. Depending on the model, R2, pseudo-R2 or marginal / adjusted R2 values are returned.

Usage

```
r2(model, ...)
```

Arguments

model	A statistical model.
...	Currently not used.

Value

Returns a list containing values related to the most appropriate R2 for the given model. See the list below:

- Logistic models: Tjur's R2
- General linear models: Nagelkerke's R2
- Multinomial Logit: McFadden's R2
- Mixed models: Nakagawa's R2
- Bayesian models: R2 bayes

See Also

link{r2_bayes}, link{r2_coxsnell}, link{r2_kl}, link{r2_loo}, link{r2_mcfadden}, link{r2_nagelkerke}, link{r2_nakagawa} and link{r2_tjur}.

Examples

```
model <- glm(vs ~ wt + mpg, data = mtcars, family = "binomial")
r2(model)

library(lme4)
model <- lmer(Sepal.Length ~ Petal.Length + (1 | Species), data = iris)
r2(model)
```

---

r2_bayes *Bayesian R2*

Description

Compute R2 for Bayesian models. For mixed models (including a random part), it additionally computes the R2 related to the fixed effects only (marginal R2).

Usage

```
r2_bayes(model, robust = TRUE)
```

Arguments

model	A Bayesian regression model.
robust	Logical, if TRUE, the median instead of mean is used to calculate the central tendency of the variances.

Details

r2_bayes() returns an "unadjusted" R2 value. See r2_loo to calculate a LOO-adjusted R2, which comes conceptionally closer to an adjusted R2 measure.

For mixed models, the conditional and marginal R2 are returned. The marginal R2 considers only the variance of the fixed effects, while the conditional R2 takes both the fixed and random effects into account.

Value

A list with the Bayesian R2 value. For mixed models, a list with the Bayesian R2 value and the marginal Bayesian R2 value. The standard errors for the R2 values are saved as attributes.

References

Gelman, A., Goodrich, B., Gabry, J., & Vehtari, A. (2018). R-squared for Bayesian regression models. The American Statistician, 1–6. doi: 10.1080/00031305.2018.1549100

Examples

```
library(rstanarm)

model <- stan_glm(mpg ~ wt + cyl, data = mtcars, chains = 1, iter = 500)
r2_bayes(model)

model <- stan_lmer(
  Petal.Length ~ Petal.Width + (1 | Species),
  data = iris,
  chains = 1,
  iter = 500
)
r2_bayes(model)

## Not run:
library(brms)
model <- brms::brm(mpg ~ wt + cyl, data = mtcars)
r2_bayes(model)

model <- brms::brm(Petal.Length ~ Petal.Width + (1 | Species), data = iris)
r2_bayes(model)

## End(Not run)
```

---

r2_coxnell *Cox & Snell's R2*

Description

Calculates the pseudo-R2 value based on the proposal from *Cox & Snell (1989)*.

Usage

```
r2_coxnell(model)
```

Arguments

`model`	Model with binary outcome.

Details

This index was proposed by *Cox & Snell (1989, pp. 208-9)* and, apparently independently, by *Magee (1990)*; but had been suggested earlier for binary response models by *Maddala (1983)*. However, this index achieves a maximum of less than 1 for discrete models (i.e. models whose likelihood is a product of probabilities) which have a maximum of 1, instead of densities, which can become infinite *(Nagelkerke, 1991)*.

Value

A named vector with the R2 value.

References

- Cox, D. R., Snell, E. J. (1989). Analysis of binary data (Vol. 32). Monographs on Statistics and Applied Probability.
- Magee, L. (1990). R 2 measures based on Wald and likelihood ratio joint significance tests. The American Statistician, 44(3), 250-253.
- Maddala, G. S. (1986). Limited-dependent and qualitative variables in econometrics (No. 3). Cambridge university press.
- Nagelkerke, N. J. (1991). A note on a general definition of the coefficient of determination. Biometrika, 78(3), 691-692.

Examples

```
model <- glm(vs ~ wt + mpg, data = mtcars, family = "binomial")
r2_coxnell(model)
```

r2_kl *Kullback-Leibler R2*

Description

Calculates the Kullback-Leibler-divergence-based R2 for generalized linear models.

Usage

```
r2_kl(model, adjust = TRUE)
```

Arguments

model	A generalized linear model.
adjust	Logical, if TRUE (the default), the adjusted R2 value is returned.

Value

A named vector with the R2 value.

References

Cameron, A. C. and Windmeijer, A. G. (1997) An R-squared measure of goodness of fit for some common nonlinear regression models. Journal of Econometrics, 77: 329-342.

Examples

```
model <- glm(vs ~ wt + mpg, data = mtcars, family = "binomial")
r2_kl(model)
```

r2_loo *LOO-adjusted R2*

Description

Compute LOO-adjusted R2.

Usage

```
r2_loo(model)
```

Arguments

model	A Bayesian regression model.

Details

Unlike r2_bayes, which returns an "unadjusted" R2 value, r2_loo() calculates a LOO-adjusted R2, which comes conceptionally closer to an "adjusted" R2 measure.

Value

The LOO-adjusted R2 for model, as numeric value.

Examples

```
library(rstanarm)

model <- stan_glm(mpg ~ wt + cyl, data = mtcars, chains = 1, iter = 500)
r2_loo(model)
```

---

r2_mcfadden *McFadden's R2*

Description

Calculates McFadden's pseudo R2.

Usage

```
r2_mcfadden(model)
```

Arguments

model	Multinomial Logit (mlogit) Model.

Value

A named vector with the R2 value.

References

- McFadden, D. (1987). Regression-based specification tests for the multinomial logit model. Journal of econometrics, 34(1-2), 63-82.
- McFadden, D. (1973). Conditional logit analysis of qualitative choice behavior.

Examples

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

model <- mlogit(mode ~ price + catch, data = Fish)
r2_mcfadden(model)
```

r2_nagelkerke *Nagelkerke's R2*

Description

Calculate Nagelkerke's pseudo-R2.

Usage

```
r2_nagelkerke(model)
```

Arguments

model	A generalized linear model, including cumulative links resp. multinomial models.

Value

A named vector with the R2 value.

References

Nagelkerke, N. J. (1991). A note on a general definition of the coefficient of determination. Biometrika, 78(3), 691-692.

Examples

```
model <- glm(vs ~ wt + mpg, data = mtcars, family = "binomial")
r2_nagelkerke(model)
```

r2_nakagawa *Nakagawa's R2 for mixed models*

Description

Compute the marginal and conditional r-squared value for mixed effects models with complex random effects structures.

Usage

```
r2_nakagawa(model)
```

Arguments

model	A mixed effects model.

Details

Marginal and conditional r-squared values for mixed models are calculated based on *Nakagawa et al. 2017*. For more details on the computation of the variances, see `get_variance`.

The marginal r-squared considers only the variance of the fixed effects, while the conditional r-squared takes both the fixed and random effects into account. The random effect variances are actually the mean random effect variances, thus the r-squared value is also appropriate for mixed models with random slopes or nested random effects (see *Johnson 2014*).

Value

A list with the conditional and marginal R2 values.

References

- Johnson, P. C. D. (2014). Extension of Nakagawa & Schielzeth's R2 GLMM to random slopes models. Methods in Ecology and Evolution, 5(9), 944–946. doi: 10.1111/2041210X.12225
- Nakagawa, S., & Schielzeth, H. (2013). A general and simple method for obtaining R2 from generalized linear mixed-effects models. Methods in Ecology and Evolution, 4(2), 133–142. doi: 10.1111/j.2041210x.2012.00261.x
- Nakagawa, S., Johnson, P. C. D., & Schielzeth, H. (2017). The coefficient of determination R2 and intra-class correlation coefficient from generalized linear mixed-effects models revisited and expanded. Journal of The Royal Society Interface, 14(134), 20170213. doi: 10.1098/rsif.2017.0213

Examples

```
library(lme4)
model <- lmer(Sepal.Length ~ Petal.Length + (1 | Species), data = iris)
r2_nakagawa(model)
```

---

r2_tjur *Tjur's R2 - coefficient of determination (D)*

Description

This method calculates the Coefficient of Discrimination `D` (also known as Tjur's R2; *Tjur, 2009*) for generalized linear (mixed) models for binary outcomes. It is an alternative to other pseudo-R2 values like Nagelkerke's R2 or Cox-Snell R2. The Coefficient of Discrimination `D` can be read like any other (pseudo-)R2 value.

Usage

```
r2_tjur(model)
```

Arguments

model	Binomial Model.

Value

A named vector with the R2 value.

References

Tjur, T. (2009). Coefficients of determination in logistic regression models - A new proposal: The coefficient of discrimination. The American Statistician, 63(4), 366-372.

Examples

```
model <- glm(vs ~ wt + mpg, data = mtcars, family = "binomial")
r2_tjur(model)
```

---

rmse *Root Mean Squared Error of Linear Models*

Description

Compute root mean squared error for linear (mixed effects) models.

Usage

```
rmse(model, normalized = FALSE)
```

Arguments

model	Linear model of class lm, merMod (**lme4**) or lme (**nlme**).
normalized	Logical, use TRUE if normalized rmse should be returned.

Details

The RMSE is the square root of the variance of the residuals and indicates the absolute fit of the model to the data (difference between observed data to model's predicted values). It can be interpreted as the standard deviation of the unexplained variance, and is in the same units as the response variable. Lower values indicate better model fit.

The normalized RMSE is the proportion of the RMSE related to the range of the response variable. Hence, lower values indicate less residual variance.

Value

Numeric, the root mean squared error.

Examples

```
library(nlme)
m <- lme(distance ~ age, data = Orthodont)

# RMSE
rmse(m, normalized = TRUE)

# normalized RMSE
rmse(m, normalized = TRUE)
```

---

rse *Residual Standard Error for Linear Models*

Description

Compute residual standard error of linear models.

Usage

```
rse(model)
```

Arguments

model	Linear model of class lm, merMod (**lme4**) or lme (**nlme**).

Details

The residual standard error is the square root of the residual sum of squares divided by the residual degrees of freedom.

Value

Numeric, the residual standard error of model.

Examples

```
data(mtcars)
m <- lm(mpg ~ hp + gear, data = mtcars)
rse(m)
```

Index