Package ‘groupedstats’

February 28, 2020

Type Package

Title Grouped Statistical Analyses in a Tidy Way

Version 0.2.0

Maintainer Indrajeet Patil <patilindrajeet.science@gmail.com>

Description Collection of functions to run statistical tests
  across all combinations of multiple grouping variables.

License GPL-3 | file LICENSE

URL https://indrajeetpatil.github.io/groupedstats/,
https://github.com/IndrajeetPatil/groupedstats/

BugReports https://github.com/IndrajeetPatil/groupedstats/issues/

Depends R (>= 3.5.0)

Imports broomExtra, dplyr, glue, haven, ipmisc, lme4, parameters,
  purrr, rlang, rstudioapi, sjstats, skimr, stats, tibble, tidyr

Suggests ez, gapminder, ggplot2, knitr, rmarkdown, spelling, testthat

VignetteBuilder knitr

Encoding UTF-8

Language en-US

LazyData true

RoxygenNote 7.0.2.9000

NeedsCompilation no

Author Indrajeet Patil [aut, cre] (<https://orcid.org/0000-0003-1995-6531>)

Repository CRAN

Date/Publication 2020-02-28 12:10:09 UTC
R topics documented:

- groupedstats-package ................................................. 2
- grouped_aov .......................................................... 2
- grouped_glm ........................................................... 4
- grouped_glmer .......................................................... 5
- grouped_lm ............................................................ 8
- grouped_lmer ........................................................... 9
- grouped_proptest ....................................................... 12
- grouped_slr ............................................................ 13
- grouped_summary ....................................................... 14
- grouped_ttest ........................................................... 15
- grouped_wilcox ........................................................ 16
- intent_morality ......................................................... 18
- lm_effsize_ci ............................................................ 19
- lm_effsize_standardizer ............................................... 20
- movies_long ............................................................ 21
- movies_wide ............................................................ 22
- Titanic_full ............................................................. 23

Index 25

---

**groupedstats-package**

**groupedstats**

**Description**

Collection of functions to run statistical operations on multiple variables across multiple grouping variables in a dataframe. This package is retired.

**Details**

For more, see README on GitHub.

---

**grouped_aov**

*Running analysis of variance (aov) across multiple grouping variables.*

**Description**

Running analysis of variance (aov) across multiple grouping variables.
Usage

```r
grouped_aov(
  data,
  grouping.vars,  
  formula,        
  effsize = "eta",  
  output = "tidy",  
  nboot = 1000,  
  ...  
)
```

Arguments

- **data**: A data frame in which the variables specified in the formula will be found. If missing, the variables are searched for in the standard way.
- **grouping.vars**: Grouping variables.
- **formula**: A formula specifying the model.
- **effsize**: Character describing the effect size to be displayed: "eta" (default) or "omega".
- **output**: A character describing what output is expected. Two possible options: "tidy" (default), which will return the results, or "tukey", which will return results from Tukey's Honest Significant Differences method for post hoc comparisons. The "glance" method to get model summary is currently not supported for this function.
- **nboot**: Number of bootstrap samples for confidence intervals for partial eta-squared and omega-squared (Default: 500).
- **...**: Currently ignored.

Author(s)

Indrajeet Patil

Examples

```r
# uses dataset included in the `groupedstats` package
set.seed(123)
library(groupedstats)

# effect size
groupedstats::grouped_aov(
  formula = rating ~ belief * outcome * question,
  data = intent_morality,
  grouping.vars = item,
  effsize = "eta"
)

# pairwise comparisons
groupedstats::grouped_aov("
grouped_glm

Function to run generalized linear model (glm) across multiple grouping variables.

Description

Function to run generalized linear model (glm) across multiple grouping variables.

Usage

grouped_glm(
  data,
  grouping.vars,
  ...,
  output = "tidy",
  tidy.args = list(conf.int = TRUE, conf.level = 0.95),
  augment.args = list()
)

Arguments

- **data**: Dataframe (or tibble) from which variables are to be taken.
- **grouping.vars**: Grouping variables.
- **...**: Additional arguments to broom::tidy, broom::glance, or broom::augment S3 method.
- **output**: A character describing what output is expected. Two possible options: "tidy" (default), which will return the results, or "glance", which will return model summaries.
- **tidy.args**: A list of arguments to be used in the relevant S3 method.
- **augment.args**: A list of arguments to be used in the relevant S3 method.

Value

A tibble dataframe with tidy results from linear model.

Author(s)

Indrajeet Patil
grouped_glmer

See Also
grouped_lm, grouped_lmer, grouped_glmer

Examples

# to get tidy output
groupedstats::grouped_glm(
  data = groupedstats::Titanic_full,
  formula = Survived ~ Sex,
  grouping.vars = Class,
  family = stats::binomial(link = "logit")
)

# to get glance output
groupedstats::grouped_glm(
  data = groupedstats::Titanic_full,
  formula = Survived ~ Sex,
  grouping.vars = Class,
  family = stats::binomial(link = "logit"),
  output = "glance"
)

grouped_glmer

Function to run generalized linear mixed-effects model (glmer) across multiple grouping variables.

Description

Function to run generalized linear mixed-effects model (glmer) across multiple grouping variables.

Usage

grouped_glmer(
  data,
  grouping.vars,
  ..., 
  output = "tidy",
  tidy.args = list(conf.int = TRUE, conf.level = 0.95, effects = "fixed", conf.method = "Wald"),
  augment.args = list()
)

Arguments

data Dataframe (or tibble) from which variables are to be taken.
grouping.vars Grouping variables.
... Arguments passed on to lme4::glmer
formula a two-sided linear formula object describing both the fixed-effects and random-effects part of the model, with the response on the left of a ~ operator and the terms, separated by + operators, on the right. Random-effects terms are distinguished by vertical bars ("|") separating expressions for design matrices from grouping factors.

family a GLM family, see glm and family.

control a list (of correct class, resulting from lmerControl() or glmerControl() respectively) containing control parameters, including the nonlinear optimizer to be used and parameters to be passed through to the nonlinear optimizer, see the *lmerControl documentation for details.

start a named list of starting values for the parameters in the model, or a numeric vector. A numeric start argument will be used as the starting value of theta. If start is a list, the theta element (a numeric vector) is used as the starting value for the first optimization step (default=1 for diagonal elements and 0 for off-diagonal elements of the lower Cholesky factor); the fitted value of theta from the first step, plus start[['fixef']], are used as starting values for the second optimization step. If start has both fixef and theta elements, the first optimization step is skipped. For more details or finer control of optimization, see modular.

verbose integer scalar. If > 0 verbose output is generated during the optimization of the parameter estimates. If > 1 verbose output is generated during the individual penalized iteratively reweighted least squares (PIRLS) steps.

nAGQ integer scalar - the number of points per axis for evaluating the adaptive Gauss-Hermite approximation to the log-likelihood. Defaults to 1, corresponding to the Laplace approximation. Values greater than 1 produce greater accuracy in the evaluation of the log-likelihood at the expense of speed. A value of zero uses a faster but less exact form of parameter estimation for GLMMs by optimizing the random effects and the fixed-effects coefficients in the penalized iteratively reweighted least squares step. (See Details.)

subset an optional expression indicating the subset of the rows of data that should be used in the fit. This can be a logical vector, or a numeric vector indicating which observation numbers are to be included, or a character vector of the row names to be included. All observations are included by default.

weights an optional vector of 'prior weights' to be used in the fitting process. Should be NULL or a numeric vector.

na.action a function that indicates what should happen when the data contain NAs. The default action (na.omit, inherited from the 'factory fresh' value ofgetOption("na.action")) strips any observations with any missing values in any variables.

offset this can be used to specify an a priori known component to be included in the linear predictor during fitting. This should be NULL or a numeric vector of length equal to the number of cases. One or more offset terms can be included in the formula instead or as well, and if more than one is specified their sum is used. See model.offset.

contrasts an optional list. See the contrasts.arg of model.matrix.default.
mustart optional starting values on the scale of the conditional mean, as in \texttt{glm}; see there for details.

etastart optional starting values on the scale of the unbounded predictor as in \texttt{glm}; see there for details.

devFunOnly logical - return only the deviance evaluation function. Note that because the deviance function operates on variables stored in its environment, it may not return \textit{exactly} the same values on subsequent calls (but the results should always be within machine tolerance).

output A character describing what output is expected. Two possible options: "\texttt{tidy}" (default), which will return the results, or "\texttt{glance}" , which will return model summaries.

tidy.args A list of arguments to be used in the relevant S3 method.

augment.args A list of arguments to be used in the relevant S3 method.

Value

A tibble dataframe with tidy results from linear model or model summaries.

Author(s)

Indrajeet Patil

See Also

grouped_lmer

Examples

# for reproducibility
set.seed(123)

# categorical outcome; binomial family
groupedstats::grouped_glmer(
  formula = Survived ~ Age + (Age | Class),
  family = stats::binomial(link = "probit"),
  data = dplyr::sample_frac(groupedstats::Titanic_full, size = 0.3),
  grouping.vars = Sex,
  REML = FALSE,
  tidy.args = list(effects = "fixed", conf.int = TRUE, conf.level = 0.95)
)
grouped_lm  

Running linear model (lm) across multiple grouping variables.

Description

Running linear model (lm) across multiple grouping variables.

Usage

grouped_lm(
  data,
  grouping.vars,
  ..., 
  output = "tidy",
  tidy.args = list(conf.int = TRUE, conf.level = 0.95),
  augment.args = list()
)

Arguments

data  Dataframe (or tibble) from which variables are to be taken.
grouping.vars  Grouping variables.
...  Additional arguments to broom::tidy, broom::glance, or broom::augment S3 method.
output  A character describing what output is expected. Two possible options: "tidy" (default), which will return the results, or "glance", which will return model summaries.
tidy.args  A list of arguments to be used in the relevant S3 method.
augment.args  A list of arguments to be used in the relevant S3 method.

Value

A tibble dataframe with tidy results from linear model.

Author(s)

Indrajeet Patil

See Also

grouped_slr, grouped_tidy
grouped_lmer

Examples

# loading needed libraries
library(ggplot2)

# getting tidy output of results
grouped_lm(
  data = mtcars,
  grouping.vars = cyl,
  formula = mpg ~ am * wt,
  output = "tidy"
)

# getting model summaries
# diamonds dataset from ggplot2
grouped_lm(
  data = diamonds,
  grouping.vars = c(cut, color),
  formula = price ~ carat * clarity,
  output = "glance"
)

grouped_lmer

Linear mixed-effects model (lmer) across multiple grouping variables.

Description

Linear mixed-effects model (lmer) across multiple grouping variables.

Usage

grouped_lmer(
  data,
  grouping.vars,
  ...,
  output = "tidy",
  tidy.args = list(conf.int = TRUE, conf.level = 0.95, effects = "fixed", conf.method = "Wald"),
  augment.args = list()
)

Arguments

data Dataframe (or tibble) from which variables are to be taken.

+ grouping.vars Grouping variables.

+ ... Arguments passed on to \texttt{lme4::lmer}
formula a two-sided linear formula object describing both the fixed-effects and
random-effects part of the model, with the response on the left of a ~ operator and the terms, separated by + operators, on the right. Random-effects
terms are distinguished by vertical bars (|) separating expressions for design
matrices from grouping factors. Two vertical bars (||) can be used to
specify multiple uncorrelated random effects for the same grouping vari-
able. (Because of the way it is implemented, the ||-syntax works only for
design matrices containing numeric (continuous) predictors; to fit models
with independent categorical effects, see dummy or the lmer_alt function
from the afex package.)
REML logical scalar - Should the estimates be chosen to optimize the REML
criterion (as opposed to the log-likelihood)?
control a list (of correct class, resulting from lmerControl() or glmerControl()
respectively) containing control parameters, including the nonlinear opti-
mizer to be used and parameters to be passed through to the nonlinear opti-
mizer, see the *lmerControl documentation for details.
start a named list of starting values for the parameters in the model. For
lmer this can be a numeric vector or a list with one component named
"theta".
verbose integer scalar. If > 0 verbose output is generated during the optimiza-
tion of the parameter estimates. If > 1 verbose output is generated during
the individual penalized iteratively reweighted least squares (PIRLS) steps.
subset an optional expression indicating the subset of the rows of data that
should be used in the fit. This can be a logical vector, or a numeric vector
indicating which observation numbers are to be included, or a character
vector of the row names to be included. All observations are included by
default.
weights an optional vector of ‘prior weights’ to be used in the fitting process.
Should be NULL or a numeric vector. Prior weights are not normalized
or standardized in any way. In particular, the diagonal of the residual co-
variance matrix is the squared residual standard deviation parameter sigma
times the vector of inverse weights. Therefore, if the weights have rela-
tively large magnitudes, then in order to compensate, the sigma parameter
will also need to have a relatively large magnitude.
na.action a function that indicates what should happen when the data contain
NAs. The default action (na.omit, inherited from the ‘factory fresh’ value of
getOption("na.action")) strips any observations with any missing val-
ues in any variables.
offset this can be used to specify an a priori known component to be included
in the linear predictor during fitting. This should be NULL or a numeric
vector of length equal to the number of cases. One or more offset terms
can be included in the formula instead or as well, and if more than one is
specified their sum is used. See model.offset.
contrasts an optional list. See the contrasts.arg of model.matrix.default.
devFunOnly logical - return only the deviance evaluation function. Note that
because the deviance function operates on variables stored in its environ-
ment, it may not return exactly the same values on subsequent calls (but the
results should always be within machine tolerance).
output  A character describing what output is expected. Two possible options: "tidy" (default), which will return the results, or "glance", which will return model summaries.

tidy.args  A list of arguments to be used in the relevant S3 method.

augment.args  A list of arguments to be used in the relevant S3 method.

Value

A tibble dataframe with tidy results from a linear mixed-effects model. Note that $p$-value is computed using parameters::p.value.

Author(s)

Indrajeet Patil

Examples

```r
# for reproducibility
set.seed(123)

# loading libraries containing data
library(ggplot2)
library(gapminder)

# getting tidy output of results
# let's use only subset of the data
groupedstats::grouped_lmer(
  data = gapminder,
  formula = scale(lifeExp) ~ scale(gdpPercap) + (gdpPercap | continent),
  grouping.vars = year,
  REML = FALSE,
  tidy.args = list(effects = "fixed", conf.int = TRUE, conf.level = 0.95),
  output = "tidy"
)

# getting model summaries
groupedstats::grouped_lmer(
  data = ggplot2::diamonds,
  formula = scale(price) ~ scale(carat) + (carat | color),
  REML = FALSE,
  grouping.vars = c(cut, clarity),
  output = "glance"
)
```
grouped_proptest

Function to run proportion test on grouped data.

Description

Function to run proportion test on grouped data.

Usage

grouped_proptest(data, grouping.vars, measure, ...)

Arguments

data Dataframe (or tibble) from which variables are to be taken.
grouping.vars Grouping variables.
measure A variable for which proportion test needs to be carried out for each combination of levels of factors entered in grouping.vars.
... Currently ignored.

Value

Dataframe with percentages and statistical details from a proportion test.

Author(s)

Indrajeet Patil

Examples

# for reproducibility
set.seed(123)

groupedstats::grouped_proptest(
  data = mtcars,
  grouping.vars = cyl,
  measure = am
)
Running simple linear regression (slr) on multiple variables across multiple grouping variables.

**Usage**

```r
grouped_slr(data, dep.vars, indep.vars, grouping.vars)
```

**Arguments**

- `data` : Dataframe from which variables are to be taken.
- `dep.vars` : List criterion or **dependent** variables for simple linear model (y in y ~ x).
- `indep.vars` : List predictor or **independent** variables for simple linear model (x in y ~ x).
- `grouping.vars` : List of grouping variables.

**Value**

A tibble dataframe with tidy results from simple linear regression analyses. The estimates are standardized, i.e. the `lm` model used is `scale(y) ~ scale(x)`, and not `y ~ x`.

**Author(s)**

Indrajeet Patil

**See Also**

- `grouped_lm`, `grouped_tidy`

**Examples**

```r
# for reproducibility
set.seed(123)

# in case of just one grouping variable
groupedstats::grouped_slr(
  data = iris,
  dep.vars = c(Sepal.Length, Petal.Length),
  indep.vars = c(Sepal.Width, Petal.Width),
  grouping.vars = Species
)
```
grouped_summary

Descriptive statistics for multiple variables for all grouping variable levels

Description

Descriptive statistics for multiple variables for all grouping variable levels

Usage

```r
grouped_summary(
  data, 
  grouping.vars, 
  measures = NULL, 
  measures.type = "numeric", 
  topcount.long = FALSE, 
  k = 2L, 
  ...
)
```

Arguments

data
  Dataframe from which variables need to be taken.

grouping.vars
  A list of grouping variables. Please use unquoted arguments (i.e., use x and not "x").

measures
  List variables for which summary needs to computed. If not specified, all variables of type specified in the argument measures.type will be used to calculate summaries. **Don't** explicitly set measures.type = NULL in function call, which will produce an error because the function will try to find a column in a dataframe named "NULL".

measures.type
  A character indicating whether summary for numeric ("numeric") or factor/character ("factor") variables is expected (Default: measures.type = "numeric"). This function can't be used for both numeric and variables simultaneously.

topcount.long
  If measures.type = factor, you can get the top counts in long format for plotting purposes. (Default: topcount.long = FALSE).

k
  Number of digits after decimal point (should be an integer) (Default: k = 3).

... Currenty ignored.

Value

Dataframe with descriptive statistics for numeric variables (n, mean, sd, median, min, max)

Author(s)

Indrajeet Patil
Examples

# for reproducibility
set.seed(123)

# another possibility
groupedstats::grouped_summary(
  data = iris,
  grouping.vars = Species,
  measures = Sepal.Length:Petal.Width,
  measures.type = "numeric"
)

# if no measures are chosen, all relevant columns will be summarized
groupedstats::grouped_summary(
  data = ggplot2::msleep,
  grouping.vars = vore,
  measures.type = "factor"
)

# for factors, you can also convert the dataframe to long format with counts
groupedstats::grouped_summary(
  data = ggplot2::msleep,
  grouping.vars = c(vore),
  measures = c(genus:order),
  measures.type = "factor",
  topcount.long = TRUE
)

-------------------------------------------------------------------

**grouped_ttest**  
*Function to run t-test on multiple variables across multiple grouping variables.*

-------------------------------------------------------------------

Description

Function to run t-test on multiple variables across multiple grouping variables.

Usage

```r
grouped_ttest(
  data,
  dep.vars,
  indep.vars,
  grouping.vars,
  paired = FALSE,
  var.equal = FALSE
)
```
Arguments

data: Dataframe from which variables are to be taken.
dep.vars: List dependent variables for a t-test ($y \sim x$).
indep.vars: List independent variables for a t-test ($x \sim y$).
grouping.vars: List of grouping variables.
paired: A logical indicating whether you want a paired t-test (Default: paired = FALSE; independent t-test, i.e.).
var.equal: A logical variable indicating whether to treat the two variances as being equal. If TRUE, then the pooled variance is used to estimate the variance otherwise the Welch (or Satterthwaite) approximation to the degrees of freedom is used (Default: var.equal = FALSE; Welch’s t-test, i.e.).

Value

A tibble dataframe with tidy results from t-test analyses.

Author(s)

Indrajeet Patil

Examples

# for reproducibility
set.seed(123)

groupedstats::grouped_ttest(
  data = dplyr::filter(.data = ggplot2::diamonds, color == "E" | color == "J"),
  dep.vars = c(carat, price, depth),
  indep.vars = color,
  grouping.vars = clarity,
  paired = FALSE,
  var.equal = FALSE
)

---

grouped_wilcox: Function to run two-sample Wilcoxon tests on multiple variables across multiple grouping variables.

Description

Function to run two-sample Wilcoxon tests on multiple variables across multiple grouping variables.

Running Wilcoxon test across multiple grouping variables.
grouped_wilcox

Usage

grouped_wilcox(
  data,
  dep.vars,
  indep.vars,
  grouping.vars,
  paired = FALSE,
  correct = TRUE
)

Arguments

data Dataframe from which variables are to be taken.
dep.vars List dependent variables for a two-sample Wilcoxon tests (y in y ~ x).
indep.vars List independent variables for a two-sample Wilcoxon tests (x in y ~ x).
grouping.vars List of grouping variables (if NULL, the entire dataframe will be used).
paired A logical indicating whether you want a paired two-sample Wilcoxon tests (Default: paired = FALSE).
correct A logical indicating whether to apply continuity correction in the normal approximation for the p-value (Default: correct = TRUE).

Value

A tibble dataframe with tidy results from two-sample Wilcoxon tests analyses.

Author(s)

Indrajeet Patil

See Also

grouped_tidy

Examples

# for reproducibility
set.seed(123)

# only with one grouping variable
groupedstats::grouped_wilcox(
  data = dplyr::filter(.data = ggplot2::diamonds, color == "E" | color == "J"),
  dep.vars = depth:table,
  indep.vars = color,
  grouping.vars = clarity,
  paired = FALSE
)
**intent_morality**  
*Moral judgments about third-party moral behavior.*

**Description**

Moral judgments about third-party moral behavior.

**Usage**

`intent_morality`

**Format**

A data frame with 4016 rows and 8 variables

- id. Participant id.
- item. Which story/vignette participants read for a given condition.
- harm. What kind of harm was involved in the item.
- belief. What kind of belief the actor had (neutral or negative/harmful).
- outcome. What kind of outcome the actor caused (neutral or negative/harmful).
- condition. Type of harm, composed of belief and outcome.
- question. Type of moral judgment asked (*wrongess* or *punishment*).
- rating. Moral judgment rating on a scale of 1 to 7.

**Details**

This dataset contains data from a recent study about how people judge behavior of others when they unintentionally or intentionally cause harm to others.

Participants responded to four different vignettes that contains four different types of conditions:

- accidental harm. neutral belief, harmful/negative outcome
- intentional harm. harmful/negative belief, harmful/negative outcome
- attempted harm. harmful/negative belief, neutral outcome
- neutral harm. neutral belief, neutral outcome

Additionally, participants saw one of the four variants for each of the four items. Each of the item had a different type of harm.

**Source**

[https://www.nature.com/articles/s41598-017-05299-9](https://www.nature.com/articles/s41598-017-05299-9)

**Examples**

```r
dim(intent_morality)
head(intent_morality)
dplyr::glimpse(intent_morality)
```
Description

This function will convert a linear model object to a dataframe containing statistical details for all effects along with effect size measure and its confidence interval. For more details, see parameters::eta_squared and parameters::omega_squared.

Usage

```r
lm_effsize_ci(
  object,
  effsize = "eta",
  partial = TRUE,
  conf.level = 0.95,
  nboot = 500,
  method = c("dist", "quantile"),
  ...
)
```

Arguments

- **object**: The linear model object (can be of class `lm`, `aov`, `anova`, or `aovlist`).
- **effsize**: Character describing the effect size to be displayed: "eta" (default) or "omega".
- **partial**: Logical that decides if partial eta-squared or omega-squared are returned (Default: TRUE). If FALSE, eta-squared or omega-squared will be returned. Valid only for objects of class `lm`, `aov`, `anova`, or `aovlist`.
- **conf.level**: Numeric specifying Level of confidence for the confidence interval (Default: 0.95).
- **nboot**: Number of bootstrap samples for confidence intervals for partial eta-squared and omega-squared (Default: 500).
- **method**: Character vector, indicating if confidence intervals should be based on bootstrap standard error, multiplied by the value of the quantile function of the t-distribution (default), or on sample quantiles of the bootstrapped values. See 'Details' in `boot_ci()`. May be abbreviated.
- ... Currently ignored.

Value

A dataframe with results from `stats::lm()` with partial eta-squared, omega-squared, and bootstrapped confidence interval for the same.

Author(s)

Indrajeet Patil
Examples

```r
# for reproducibility
set.seed(123)

# model
mod <-
  stats::aov(
    formula = mpg ~ wt + qsec + Error(disp / am),
    data = mtcars
  )

# dataframe with effect size and confidence intervals
groupedstats::lm_effsize_ci(mod)
```

**lm_effsize_standardizer**

*Standardize a dataframe with effect sizes for aov, lm, aovlist, etc. objects.*

**Description**

The difference between `lm_effsize_ci` and `lm_effsize_standardizer` is that the former has more opinionated column naming, while the latter doesn’t. The latter can thus be more helpful in writing a wrapper around this function.

**Usage**

```r
lm_effsize_standardizer(
  object,
  effsize = "eta",
  partial = TRUE,
  conf.level = 0.95,
  nboot = 500,
  method = c("dist", "quantile")
)
```

**Arguments**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>object</code></td>
<td>The linear model object (can be of class <code>lm</code>, <code>aov</code>, <code>anova</code>, or <code>aovlist</code>).</td>
</tr>
<tr>
<td><code>effsize</code></td>
<td>Character describing the effect size to be displayed: &quot;eta&quot; (default) or &quot;omega&quot;.</td>
</tr>
<tr>
<td><code>partial</code></td>
<td>Logical that decides if partial eta-squared or omega-squared are returned (Default: <code>TRUE</code>). If <code>FALSE</code>, eta-squared or omega-squared will be returned. Valid only for objects of class <code>lm</code>, <code>aov</code>, <code>anova</code>, or <code>aovlist</code>.</td>
</tr>
<tr>
<td><code>conf.level</code></td>
<td>Numeric specifying Level of confidence for the confidence interval (Default: <code>0.95</code>).</td>
</tr>
<tr>
<td><code>nboot</code></td>
<td>Number of bootstrap samples for confidence intervals for partial eta-squared and omega-squared (Default: <code>500</code>).</td>
</tr>
</tbody>
</table>
movies_long

method

Character vector, indicating if confidence intervals should be based on bootstrap standard error, multiplied by the value of the quantile function of the t-distribution (default), or on sample quantiles of the bootstrapped values. See 'Details' in boot_ci(). May be abbreviated.

Examples

set.seed(123)
groupedstats::lm_effsize_standardizer(
  object = stats::lm(formula = brainwt ~ vore, data = ggplot2::msleep),
  effsize = "eta",
  partial = FALSE,
  conf.level = 0.99,
  nboot = 20
)

movies_long

Movie information and user ratings from IMDB.com (long format).

Description

Movie information and user ratings from IMDB.com (long format).

Usage

movies_long

Format

A data frame with 2433 rows and 8 variables

- title. Title of the movie.
- year. Year of release.
- budget. Total budget (if known) in US dollars
- length. Length in minutes.
- rating. Average IMDB user rating.
- votes. Number of IMDB users who rated this movie.
- mpaa. MPAA rating.
- genre. Different genres of movies (action, animation, comedy, drama, documentary, romance, short).
Details

Modified dataset from ggplot2movies package.


Movies were selected for inclusion if they had a known length and had been rated by at least one imdb user.

Source

https://CRAN.R-project.org/package=ggplot2movies

Examples

dim(movies_long)
head(movies_long)
dplyr::glimpse(movies_long)

movies_wide

Movie information and user ratings from IMDB.com (wide format).

Description

Movie information and user ratings from IMDB.com (wide format).

Usage

movies_wide

Format

A data frame with 1813 rows and 14 variables

• title. Title of the movie.
• year. Year of release.
• budget. Total budget (if known) in US dollars
• length. Length in minutes.
• rating. Average IMDB user rating.
• votes. Number of IMDB users who rated this movie.
• mpaa. MPAA rating.
• action, animation, comedy, drama, documentary, romance, short. Binary variables representing if movie was classified as belonging to that genre.
Titanic_full

Details

Modified dataset from ggplot2movies package.


Movies were selected for inclusion if they had a known length and had been rated by at least one imdb user.

Source

https://CRAN.R-project.org/package=ggplot2movies

Examples

dim(movies_wide)
head(movies_wide)
dplyr::glimpse(movies_wide)

Titanic_full

Titanic dataset.

Description

Titanic dataset.

Usage

Titanic_full

Format

A data frame with 2201 rows and 5 variables

- id. Dummy identity number for each person.
- Class. 1st, 2nd, 3rd, Crew.
- Sex. Male, Female.
- Age. Child, Adult.
- Survived. No, Yes.

Details

This data set provides information on the fate of passengers on the fatal maiden voyage of the ocean liner 'Titanic', summarized according to economic status (class), sex, age, and survival.

This is a modified dataset from datasets package.
Source


Examples

```r
dim(Titanic_full)
head(Titanic_full)
dplyr::glimpse(Titanic_full)
```
Index

*Topic datasets
intent_morality, 18
movies_long, 21
movies_wide, 22
Titanic_full, 23
dummy, 10
family, 6
glm, 6, 7
glmerControl, 6, 10
grouped_aov, 2
grouped_glm, 4
grouped_glmer, 5, 5
grouped_lm, 5, 8, 13
grouped_lmer, 5, 9
grouped_proptest, 12
grouped_slr, 8, 13
grouped_summary, 14
grouped_tidy, 8, 13, 17
grouped_ttest, 15
grouped_wilcox, 16
groupedstats-package, 2

intent_morality, 18
list, 10
lm_effsize_ci, 19
lm_effsize_standardizer, 20
lme4::glmer, 5
lme4::lmer, 9
lmerControl, 6, 10

model.matrix.default, 6
model.offset, 6, 10
modular, 6
movies_long, 21
movies_wide, 22
offset, 6, 10

sigma, 10
Titanic_full, 23