Package 'relevance'

October 14, 2022

Type Package

Title Calculate Relevance

Version 1.3

Date 2021-07-29

Author Werner A. Stahel

Maintainer Werner A. Stahel <stahel@stat.math.ethz.ch>

Depends R (>= 3.5.0)

Imports stats, utils, graphics

Suggests MASS, survival, knitr

VignetteBuilder knitr

Description Calculates relevance and significance values for
simple models and for many types of regression models.
These are introduced in
'Stahel, Werner A.' (2021)
``New relevance and significance measures to replace p-values."
<https://stat.ethz.ch/~stahel/relevance/stahel-relevance2103.pdf>.

License GPL-2

NeedsCompilation no

Repository CRAN

Date/Publication 2021-08-10 08:30:01 UTC

R topics documented:

`relevance-package` *Calculate Relevance*

Description

Calculates relevance and significance values for simple models and for many types of regression models. These are introduced in 'Stahel, Werner A.' (2021) "New relevance and significance measures to replace p-values." <https://stat.ethz.ch/~stahel/relevance/stahel-relevance2103.pdf>.

Details

The DESCRIPTION file:

Package:	relevance
Type:	Package
Title:	Calculate Relevance
Version:	1.3
Date:	2021-07-29
Author:	Werner A. Stahel
Maintainer:	Werner A. Stahel <stahel@stat.math.ethz.ch>
Depends:	R (>= 3.5.0)
Imports:	stats, utils, graphics
Suggests:	MASS, survival, knitr
VignetteBuilder:	knitr
Description:	Calculates relevance and significance values for simple models and for many types of regression models. T
License:	GPL-2

Index of help topics:

Further information is available in the following vignettes:

relevance-descr	'Package relevance for calculating Relevance and Significance Measures' (source, pdf)

Relevance is a measure that expresses the (scientific) relevance of an effect. The simplest case is a single sample of supposedly normally distributed observations, where interest lies in the expectation, estimated by the mean of the observations. There is a threshold for the expectation, below which an effect is judged too small to be of interest.

The estimated relevance 'Rle' is then simply the estimated effect divided by the threshold. If it is larger than 1, the effect is thus judged relevant. The two other values that characterize the relevance are the limits of the confidence interval for the true value of the relevance, called the

secured relevance 'Rls' and the potential relevance 'Rlp'.

If $Rle > 1$, then one might say that the effect is "significantly relevant".

Another useful measure, meant to replace the p-value, is the "significance" 'Sg0'. In the simple case, it divides the estimated effect by the critical value of the (t-) test statistic. Thus, the statistical test of the null hypothesis of zero expectation is significant if 'Sg0' is larger than one, $Sg0 > 1$.

These measures are also calculated for the comparison of two groups, for proportions, and most importantly for regression models. For models with linear predictors, relevances are obtained for standardized coefficients as well as for the effect of dropping terms and the effect on prediction.

The most important functions are

`twosamples()`: calculate the measures for two paired or unpaired sampless or a simple mean. This function calls

`inference()`: calculates the confidence interval and siginificance based on an estimate and a standard error, and adds relevance for a standardized effect.

`termtable()`: deals with fits of regression models with a linear predictor. It calculates confidence intervals and significances for the coefficients of terms with a single degree of freedom. It includes the effect of dropping each term (based on the `drop1` function) and the respective significance and relevance measures.

`termeffects()`: calculates the relevances for the coefficients related to each term. These differ from the enties of `termtable` only for terms with more than one degree of freedom.

Author(s)

Werner A. Stahel

Maintainer: Werner A. Stahel <stahel@stat.math.ethz.ch>

References

Stahel, Werner A. (2021). New relevance and significance measures to replace p-values. To appear in PLoS ONE

See Also

Package **regr**, avaiable from https://regdevelop.r-forge.r-project.org

Examples

```
data(swiss)
rr <- lm(Fertility ~ . , data = swiss)
termtable(rr)
```

asinp *arc sine Transformation*

Description

Calculates the sqrt arc sine of x/100, rescaled to be in the unit interval.
This transformation is useful for analyzing percentages or proportions of any kind.

Usage

```
asinp(x)
```

Arguments

x	vector of data values

Value

vector of transformed values

Note

This very simple function is provided in order to simplify formulas. It has an attribute `"inverse"` that contains the inverse function, see example.

Author(s)

Werner A. Stahel, ETH Zurich

Examples

```
asinp(seq(0,100,10))
( y <- asinp(c(1,50,90,95,99)) )
attr(asinp, "inverse")(y)
```

confintF *Confidence Interval for the Non-Central F and Chisquare Distribution*

Description

Confidence Interval for the Non-Central F and Chisquare Distribution

Usage

```
confintF(f, df1, df2, testlevel = 0.05)
```

Arguments

f	observed F value(s)
df1	degrees of freedom for the numerator of the F distribution
df2	degrees of freedom for the denominator of the F distribution
testlevel	level of the (two-sided) test that determines the confidence interval, 1 - confidence level

Details

The confidence interval is calculated by solving the two implicit equations `qf(f, df1, df2, x) = testlevel/2` and `... = 1 - testlevel/2`. For `f>100`, the usual `f +- standard error` interval is used as a rather crude approximation.

A confidence interval for the non-centrality of the Chisquare distribution is obtained by setting `df2` to `Inf` (the default) and `f=x2/df1` if `x2` is the observed Chisquare value.

Value

vector of lower and upper limit of the confidence interval, or, if any of the arguments has length `>1`, matrix containing the intervals as rows.

Author(s)

Werner A. Stahel

See Also

qf

Examples

```
confintF(5, 3, 200)
## [1] 2.107 31.95
confintF(1:5, 5, 20)   ## lower limit is 0 for the first 3 f values
```

---

correlation *Correlation with Relevance and Significance Measures*

Description

Inference for a correlation coefficient: Collect quantities, including Relevance and Significance measures

Usage

```
correlation(x, y = NULL, method = c("pearson", "spearman"),
  hypothesis = 0, testlevel=getOption("testlevel"),
  rlv.threshold=getOption("rlv.threshold"), ...)
```

Arguments

x	data for the first variable, or matrix or data.frame containing both variables
y	data for the second variable
hypothesis	the null effect to be tested, and anchor for the relevance
method	type of correlation, either "pearson" for the ordinary Pearson product moment correlation, or "spearman" for the nonparametric measures
testlevel	level for the test, also determining the confidence level
rlv.threshold	Relevance threshold, or a vector of thresholds from which the element corr is taken
...	further arguments, ignored

Value

an object of class 'inference', a vector with components

effect: correlation, transformed with Fisher's z transformation

ciLow, ciUp: confidence interval for the effect

Rle, Rls, Rlp: relevance measures: estimated, secured, potential

Sig0: significance measure for test or 0 effect

Sigth: significance measure for test of effect == relevance threshold

p.value: p value for test against 0

In addition, it has attributes

method: type of correlation

effectname: label for the effect

hypothesis: the null effect

n: number(s) of observations

estimate: estimated correlation

conf.int: confidence interval on correlation scale

statistic: test statistic

data: data.frame containing the two variables

rlv.threshold: relevance threshold

Author(s)

Werner A. Stahel

References

see those in relevance-package.

See Also

cor.test

Examples

```
correlation(iris[1:50,1:2])
```

---

d.blast *Blasting for a tunnel*

Description

Blasting causes tremor in buildings, which can lead to damages. This dataset shows the relation between tremor and distance and charge of blasting.

Usage

```
data("d.blast")
```

Format

A data frame with 388 observations on the following 7 variables.

`date` date in Date format

`location` Code for location of the building, `loc1` to `loc8`

`device` Number of measuring device, 1 to 4

`distance` Distance between blasting and location of measurement

`charge` Charge of blast

`tremor` Tremor energy (target variable)

Details

The charge of the blasting should be controled in order to avoid tremors that exceed a threshold. This dataset can be used to establish the suitable rule: For a given `distance`, how large can `charge` be in order to avoid exceedance of the threshold?

Source

Basler and Hoffmann AG, Zurich

Examples

```
data(d.blast)

summary(lm(log10(tremor)~location+log10(distance)+log10(charge),
           data=d.blast))
```

`d.everest` *Data of an 'anchoring' experiment in psychology*

Description

Are answers to questions influenced by providing partial information?

Students were asked to guesstimate the height of Mount Everest. One group was 'anchored' by telling them that it was more than 2000 feet, the other group was told that it was less than 45,500 feet. The hypothesis was that respondents would be influenced by their 'anchor,' such that the first group would produce smaller numbers than the second. The true height is 29,029 feet.

The data is taken from the 'many labs' replication study (see 'source'). The first 20 values from PSU university are used here.

Usage

```
data("d.everest")
```

Format

A data frame with 20 observations on the following 2 variables.

`y` numeric: guesstimates of the height

`g` factor with levels `low` `high`: anchoring group

Source

Klein RA, Ratliff KA, Vianello M et al. (2014). Investigating variation in replicability: A "many labs" replication project. Social Psychology. 2014; 45(3):142-152. https://doi.org/10.1027/1864-9335/a000178

Examples

```
data(d.everest)

(rr <- twosamples(log(y)~g, data=d.everest, var.equal=TRUE))
print(rr, show="classical")

pltwosamples(log(y)~g, data=d.everest)
```

d.negposChoice *Data of an 'anchoring' experiment in psychology*

Description

Is a choice influenced by the formulation of the options?

Here is the question: Confronted with a new contagious disease, the government has a choice between action A that would save 200 out of 600 people or action B which would save all 600 with probability 1/3. This was the 'positive' description. The negative one was that either (A) 400 would die or (B) all 600 would die with probability 2/3.

The dataset encompasses the results for Penn State (US) and Tilburg (NL) universities.

Usage

```
data("d.negposChoice")
```

Format

A data frame with 4 observations on the following 4 variables.

`uni` character: university

`negpos` character: formulation of the options

`A` number of students choosing option A

`B` number of students choosing option B

Source

Klein RA, Ratliff KA, Vianello M et al. (2014). Investigating variation in replicability: A "many labs" replication project. Social Psychology. 2014; 45(3):142-152. https://doi.org/10.1027/1864-9335/a000178

Examples

```
data(d.negposChoice)

d1 <- d.negposChoice[d.negposChoice$uni=="PSU",-1]
(r1 <- twosamples(table=d1[,-1]))
d2 <- d.negposChoice[d.negposChoice$uni=="Tilburg",-1]
r2 <- twosamples(table=d2[,-1])
```

drop1Wald *Drop Single Terms of a Model and Calculate Respective Wald Tests*

Description

drop1Wald calculates tests for single term deletions based on the covariance matrix of estimated coefficients instead of re-fitting a reduced model. This helps in cases where re-fitting is not feasible, inappropriate or costly.

Usage

```
drop1Wald(object, scope=NULL, scale = NULL, test = NULL, k = 2, ...)
```

Arguments

object	a fitted model.
scope	a formula giving the terms to be considered for dropping. If 'NULL', 'drop.scope(object)' is obtained
scale	an estimate of the residual mean square to be used in computing Cp. Ignored if '0' or 'NULL'.
test	see drop1
k	the penalty constant in AIC / Cp.
...	further arguments, ignored

Details

The test statistics and Cp and AIC values are calculated on the basis of the estimated coefficients and their (unscaled) covariance matrix as provided by the fit object. The function may be used for all model fitting objects that contain these two components as $coefficients and $cov.unscaled.

Value

An object of class 'anova' summarizing the differences in fit between the models.

Note

drop1Wald is used for models of class 'lm' or 'lmrob' for preparing a termtable.

Author(s)

Werner A. Stahel

See Also

drop1

Examples

```
data(d.blast)
r.blast <- lm(log10(tremor)~location+log10(distance)+log10(charge),
              data=d.blast)
drop1(r.blast)
drop1Wald(r.blast)

## Example from example(glm)
dd <- data.frame(treatment = gl(3,3), outcome = gl(3,1,9),
           counts = c(18,17,15,20,10,20,25,13,12))
r.glm <- glm(counts ~ outcome + treatment, data = dd, family = poisson())
drop1(r.glm, test="Chisq")
drop1Wald(r.glm)
```

`dropdata` *Drop Observations from a Data.frame*

Description

Allows for dropping observations (rows) determined by row names or factor levels from a data.frame or matrix.

Usage

```
dropdata(data, rowid = NULL, incol = "row.names", colid = NULL)
```

Arguments

`data`	a data.frame of matrix
`rowid`	vector of character strings identifying the rows to be dropped
`incol`	name or index of the column used to identify the observations (rows)
`colid`	vector of character strings identifying the columns to be dropped

Value

The data.frame or matrix without the dropped observations and/or variables. Attributes are passed on.

Note

Ordinary subsetting by `[...,...]` drops attributes.

Furthermore, the convenient way to drop rows or columns by giving negative indices to `[...,...]` cannot be used with names of rows or columns.

Author(s)

Werner A. Stahel, ETH Zurich

See Also

subset

Examples

```
dd <- data.frame(rbind(a=1:3,b=4:6,c=7:9,d=10:12))
dropdata(dd,"b")
dropdata(dd, col="X3")

d1 <- dropdata(dd,"d")
d2 <- dropdata(d1,"b")
naresid(attr(d2,"na.action"),as.matrix(d2))

dropdata(letters, 3:5)
```

---

dropNA *drop or replace NA values*

Description

dropNA returns the vector 'x', without elements that are NA or NaN or, if 'inf' is TRUE, equal to Inf or -Inf. replaceNA replaces these values by values from the second argument

Usage

```
dropNA(x, inf = TRUE)
replaceNA(x, na, inf = TRUE)
```

Arguments

x	vector from which the non-real values should be dropped or replaced
na	replacement or vector from which the replacing values are taken.
inf	logical: should 'Inf' and '-Inf' be considered "non-real"?

Value

For dropNA: Vector containing the 'real' values of 'x' only
For replaceNA: Vector with 'non-real' values replaced by the respective elements of na.

Note

The differences to 'na.omit(x)' are: 'Inf' and '-Inf' are also dropped, unless 'inf==FALSE'.\ no attribute 'na.action' is appended.

Author(s)

Werner A. Stahel

See Also

na.omit, sumNA, ifelse

Examples

```
dd <- c(1, NA, 0/0, 4, -1/0, 6)
dropNA(dd)
na.omit(dd)

replaceNA(dd, 99)
replaceNA(dd, 100+1:6)
```

---

formatNA *Print NA values by a Desired Code*

Description

Recodes the NA entries in output by a desired code like " ."

Usage

```
formatNA(x, na.print = " .", digits = getOption("digits"), ...)
```

Arguments

x	object to be printed, usually a numeric vector or data.frame
na.print	code to be used for NA values
digits	number of digits for formatting numeric values
...	other arguments to format

Details

The na.encode argument of print only applies to character objects. formatNA does the same for numeric arguments.

Value

Should mimik the value of format

Author(s)

Werner A. Stahel

See Also

format

Examples

```
formatNA(c(1,NA,3))

dd <- data.frame(X=c(1,NA,3), Y=c(4,5, NA), g=factor(c("a",NA,"b")))
(rr <- formatNA(dd, na.print="???"))
str(rr)
```

`getcoeftable` *Extract Components of a Fit*

Description

Retrieve the table of coefficients and standard errors, or the scale parameter, or the factors needed for standardizing coefficients from diverse model fitting results

Usage

```
getcoeftable(object)
getscalepar(object)
getcoeffactor(object, standardize = TRUE)
```

Arguments

`object`	an R object resulting from a model fitting function
`standardize`	ligical: should a scaling factor for the response variable be determined (calling `getscalepar`) and used?

Details

Object `regrModelClasses` contains the names of the classes for which the result should work. For other model classes, the function is not tested and may fail.

Value

For `getcoeftable`: Matrix containing at least the two columns containing the estimated coefficients (first column) and the standard errors (second column).

For `getscalepar`: scale parameter.

For `getcoeffactor`: vector of multiplicative factors, with attributes `scale`, `fitclass` and `family` or `dist` according to `object`.

Author(s)

Werner A. Stahel

Examples

```
rr <- lm(Fertility ~ . , data = swiss)
getcoeftable(rr) # identical to  coef(summary(rr))  or also summary(rr)$coefficients
getscalepar(rr)

if(requireNamespace("survival", quietly=TRUE)) {
data(ovarian) ## , package="survival"
rs <- survival::survreg(survival::Surv(futime, fustat) ~ ecog.ps + rx,
             data = ovarian, dist = "weibull")
getcoeftable(rs)
getcoeffactor(rs)
}
```

`inference` *Calculate Confidence Intervals and Relevance and Significance Values*

Description

Calculates confidence intervals and relevance and significance values given estimates, standard errors and, for relevance, additional quantities.

Usage

```
inference(estimate = NULL, se = NULL, n = NULL, df = NULL, stcoef=TRUE,
  rlv=TRUE, rlv.threshold=getOption("rlv.threshold"),
  testlevel = getOption("testlevel"), object=NULL, ...)
```

Arguments

`estimate`	vector of estimates or matrix containing estimates and their standard errors. The latter is needed if `se` is not given – ... or a model fit object
`se`	vector of standard errors of the estimates
`n`	number of observations
`df`	degrees of freedom of the residuals
`stcoef`	standardized coefficients, suitable for If `NULL`, these will be calculated from `object`.
`rlv`	logical: Should relevances be calculated?
`rlv.threshold`	Relevance threshold(s). May be a simple number for simple inference, or a vector containing the elements `stand`**:** threshold for (simple) standardized effects `rel`**:** for relative effects, `coef`**:** for standardized coefficients, `drop`**:** for drop effects, `pred`**:** for prediction intervals.

testlevel	1 - confidence level
object	model fit, needed for calculating stcoef.
...	furter arguments, passed to termtable and termeffects

Details

The coefficients divided by standard errors are assumed to be t-distributed with df degrees of freedom. For df==Inf, this is the standard normal distribution.

Value

A data.frame with the variables

estimate, se	coefficients and standard errors
ciLow, ciUp	lower and upper limit of the confidence interval
testst	t-test statistic
Sig0	significance value, i.e., test statistic divided by critical value, which in turn is the 1-testlevel/2-quantile of the t-distribution.
p.value	p value
p.symbol	the conventional symbol corresponding to the p value

If rlv is TRUE,

stcoef	standardized coefficient
st.Low, st.Up	confidence interval for stcoef
Rle	estimated relevance of coef
Rls	secured relevance, lower end of confidence interval for the relevance of coef
Rlp	potential relevance, upper end of confidence interval ...
Rls.symbol	symbols for the secured relevance

Author(s)

Werner A. Stahel

See Also

link{termtable}, link{termeffects}

Examples

```
data(d.blast)
rr <-
  lm(log10(tremor)~location+log10(distance)+log10(charge), data=d.blast)
inference(rr)
```

last *Last Elements of a Vector or of a Matrix*

Description

Selects or drops the last element or the last n elements of a vector or the last n rows or ncol columns of a matrix

Usage

```
last(data, n = NULL, ncol=NULL, drop=is.matrix(data))
```

Arguments

data	vector or matrix or data.frame from which to select or drop
n	if >0, last selects the last n elements (rows) form the result. if <0, the last abs(n) elements (rows) are dropped, and the first length(data)-abs(n) ones from the result
ncol	if data is a matrix or data.frame, the last ncol columns are selected (if ncol is positive) or dropped (if negative).
drop	if only one row or column of a matrix (or one column of a data.frame) is selected or left over, should the result be a vector or a row or column matrix (or one variable data.frame)

Value

The selected elements of the vector or matrix or data.frame

Note

This is a very simple function. It is defined mainly for selecting from the results of other functions without storing them.

Author(s)

Werner Stahel

Examples

```
  x <- runif(rpois(1,10))
  last(sort(x), 3)
  last(sort(x), -5)
##
  df <- data.frame(X=c(2,5,3,8), F=LETTERS[1:4], G=c(TRUE,FALSE,FALSE,TRUE))
  last(df,3,-2)
```

`logst` *Started Logarithmic Transformation*

Description

Transforms the data by a log10 transformation, modifying small and zero observations such that the transformation yields finite values.

Usage

```
logst(data, calib=data, threshold=NULL, mult = 1)
```

Arguments

`data`	a vector or matrix of data, which is to be transformed
`calib`	a vector or matrix of data used to calibrate the transformation(s), i.e., to determine the constant c needed
`threshold`	constant c that determines the transformation, possibly a vector with a value for each variable.
`mult`	a tuning constant affecting the transformation of small values, see Details

Details

Small values are determined by the threshold c. If not given by the argument `threshold`, then it is determined by the quartiles q_1 and q_3 of the non-zero data as those smaller than $c = q_1/(q_3/q_1)^{mult}$. The rationale is that for lognormal data, this constant identifies 2 percent of the data as small. Beyond this limit, the transformation continues linear with the derivative of the log curve at this point. See code for the formula.

The function chooses log10 rather than natural logs because they can be backtransformed relatively easily in the mind.

Value

the transformed data. The value c needed for the transformation is returned as `attr(.,"threshold")`.

Note

The names of the function alludes to Tudey's idea of "started logs".

Author(s)

Werner A. Stahel, ETH Zurich

Examples

```
dd <- c(seq(0,1,0.1),5*10^rnorm(100,0,0.2))
dd <- sort(dd)
r.dl <- logst(dd)
plot(dd, r.dl, type="l")
abline(v=attr(r.dl,"threshold"),lty=2)
```

---

ovarian *ovarian*

Description

copy of ovarian from package 'survival'. Will disappear

Usage

```
data("ovarian")
```

Format

A data frame with 26 observations on the following 6 variables.

`futime` a numeric vector

`fustat` a numeric vector

`age` a numeric vector

`resid.ds` a numeric vector

`rx` a numeric vector

`ecog.ps` a numeric vector

Details

This copy is here since the package was rejected because the checking procedure did not find it in the package

Examples

```
data(ovarian)
summary(ovarian)
```

plconfint *Plot Confidence Intervals*

Description

Plot confidence or relevance interval(s)

Usage

```
plconfint(x, pos = NULL, xlim = NULL, add = FALSE, bty = "L", col = 1,
  plpars=list(lwd=c(2,3,1,2,2), markheight=c(1,0.7,0.85), extend=NA,
    reflinecol="gray70"),
  xlab="", ...)

pltwosamples(x, ...)
## Default S3 method:
pltwosamples(x, y, overlap = TRUE, ...)
## S3 method for class 'formula'
pltwosamples(formula, data=NULL, ...)
```

Arguments

x	For `plconfint`: A vector of length >=3 or a matrix with this number of columns, containing • [,1]the estimate • [,2:3]the interval end points • [,4:5](or another number of additional columns) if desired, values for additional ticks on the intervals, typically indicating the end points of a shortened interal, see Details For `pltwosamples`: A formula or the data for the first sample – or a list or matrix or data.frame with two components/columns corresponding to the two samples
y	data for the second sample
pos	positions of the bars in vertical direction
xlim	limits for the horizontal axis. NAs will be replaced by the respective element of the range of the x values.
add	logical: should the plotted elements be added to an existing plot?
bty	type of 'box' around the plot, see `par`
col	color to be used for the confidence intervals, usually a vector of colors if used.
plpars	graphical options, see Details
xlab	label for horizontal axis
overlap	logical: should shortened intervals be shown to show significance of differences? see Details
formula, data	formula and data for the `formula` method
...	further arguments to the call of `plconfint`

Details

Columns 4 and 5 of x are typically used to indicate an "overlap interval", which allows for a graphical assessment of the significance of the test for zero difference(s), akin the "notches" in the box plots: The difference between a pair of groups is siginificant if their overlap intervals do not overlap. For equal standard errors of the groups, the standard error of the difference between two of them is larger by the factor `sqrt(2)`. Therefore, the intervals should be shortened by this factor, or multiplied by `1/sqrt(2)`, which is the default for `overlapfactor`. If only two groups are to be shown, the factor is adjusted to unequal standard errors, and accurate quantiles of a t distribution are used.

The graphical options are:

- lwdline widths for: [1] the interval, [2] middle mark, [3] end marks, [4] overlap interval marks, [5] vertical line marking the relevance threshold
- markheightdetermines the length of the middle mark, the end marks and the marks for the overlap interval as a multiplier of the default length
- extendextension of the vertical axis beyond the range
- reflinecolcolor to be used for the vertical lines at relevances 0 and 1

Value

none

Author(s)

Werner A. Stahel

See Also

plot.inference

Examples

```
## --- regression
data(swiss)
rr <- lm(Fertility ~ . , data = swiss)
rt <- termtable(rr)
plot(rt)

## --- termeffects
data(d.blast)
rlm <- lm(log10(tremor)~location+log10(distance)+log10(charge), data=d.blast)
rte <- termeffects(rlm)
plot(rte, single=TRUE)
```

`plot.inference` *Plot Inference Results*

Description

Plot confidence or relevance interval(s) for one or several items

Usage

```
## S3 method for class 'inference'
plot(x, pos = NULL, overlap = FALSE,
  reflines = c(0,1,-1), xlab = "relevance", ...)
## S3 method for class 'termeffects'
plot(x, pos = NULL, single=FALSE,
  overlap = TRUE, termeffects.gap = 0.2, ...)
```

Arguments

`x`	a vector or matrix of class `inference`.
`pos`	positions of the bars in vertical direction
`overlap`	logical: should shortened intervals be shown to show significance of differences? see Details
`reflines`	values for vertical reference lines
`xlab`	label for horizontal axis
`single`	logical: should terms with a single degree of freedom be plotted?
`termeffects.gap`	gap between blocks corresponding to terms
`...`	further arguments to the call of `plot.inference` (for`plot.termeffects`) and `plot`

Details

The overlap interval allows for a graphical assessment of the significance of the test for zero difference(s), akin the notches in the box plots: The difference between a pair of groups is siginificant if their overlap intervals do not overlap. For equal standard errors of the groups, the standard error of the difference between two of them is larger by the factor `sqrt(2)`. Therefore, the intervals should be shortened by this factor, or multiplied by `1/sqrt(2)`, which is the default for `overlapfactor`. If only two groups are to be shown, the factor is adjusted to unequal standard errors.

The graphical options are:

- lwdline widths for: [1] the interval, [2] middle mark, [3] end marks, [4] overlap interval marks, [5] vertical line marking the relevance threshold
- markheightdetermines the length of the middle mark, the end marks and the marks for the overlap interval as a multiplier of the default length
- extendextension of the vertical axis beyond the range
- framecolcolor to be used for the framing lines: axis and vertical lines at relevances 0 and 1

Value

none

Note

`plot.inference` displays `termtable` objects, too, since they inherit from class `inference`.

Author(s)

Werner A. Stahel

See Also

`plconfint`

Examples

```
## --- regression
data(swiss)
rr <- lm(Fertility ~ . , data = swiss)
rt <- termtable(rr)
plot(rt)

## --- termeffects
data(d.blast)
rlm <- lm(log10(tremor)~location+log10(distance)+log10(charge), data=d.blast)
rte <- termeffects(rlm)
plot(rte, single=TRUE)
```

---

print.inference *Print Tables with Inference Measures*

Description

Print methods for objects of class `"inference"`, `"termtable"`, `"termeffects"`, or `"printInference"`.

Usage

```
## S3 method for class 'inference'
print(x, show = getOption("show.inference"), print=TRUE,
  digits = getOption("digits.reduced"), transpose.ok = TRUE,
  legend = NULL, na.print = getOption("na.print"), ...)

## S3 method for class 'termtable'
print(x, show = getOption("show.inference"), ...)

## S3 method for class 'termeffects'
print(x, show = getOption("show.inference"),
```

```
  transpose.ok = TRUE, single = FALSE, print = TRUE, warn = TRUE, ...)

## S3 method for class 'printInference'
print(x, ...)
```

Arguments

x	object to be printed
show	determines items (columns) to be shown
digits	number of significant digits to be printed
transpose.ok	logical: May a single column be shown as a row?
single	logical: Should components with a single coefficient be printed?
legend	logical: should the legend(s) for the symbols characterizing p-values and relevances be printed? Defaults to `regroptions("show.symbolLegend")`.
na.print	string by which NAs are shown
print	logical: if FALSE, no printing will occur, used to edit the result before printing it.
warn	logical: Should the warning be issued if `termeffects` has nothing to print since there are no terms with more than one degree of freedom
...	further arguments, passed to `print.data.frame()`.

Details

The value, if assigned to `rr`, say, can be printed by using `print.printInference`, writing `print(rr)`, which is just what happens internally unless `print=FALSE` is used. This allows for editing the result before printing it, see Examples.

`printInference` objects can be a vector, a data.frame or a matrix, or a list of such items. Each item can have an attribute `head` of mode character that is printed by `cat` before the item, and analogous with a `tail` attribute.

Value

A kind of formatted version of `x`, with class `printInference`. For `print.inference`, it will be a character vector or a data.frame with attributes `head` and `tail` if applicable. For `print.termeffects`, it will be a list of such elements, with its own `head` and `tail`. It is invisibly returned.

Author(s)

Werner A. Stahel

See Also

`twosamples`, `termtable`, `termeffects`, `inference`.

Examples

```
data(d.blast)
r.blast <-
  lm(log10(tremor)~location+log10(distance)+log10(charge), data=d.blast)
rt <- termtable(r.blast)
## print() : first default, then "classical" :
rt
print(rt, show="classical")

class(te <- termeffects(r.blast)) #  "termeffects"
rr <- print(te, print=FALSE)
attr(rr, "head") <- sub("lm", "Linear Regression", attr(rr, "head"))
class(rr) # "printInference"
rr # <==>  print(rr)

str(rr)
```

`relevance.options` *Options for the relevnance Package*

Description

List of options used in the relevnance package to select items and formats for printing inference elements

Usage

```
relevance.options
rlv.symbols
p.symbols
```

Format

The format is: List of 22 $ digits.reduced : 3 $ testlevel : 0.05 $ rlv.threshold : stand rel prop corr coef drop pred 0.10 0.10 0.10 0.10 0.10 0.10 0.05 $ termtable : TRUE $ show.confint : TRUE $ show.doc : TRUE $ show.inference : "relevance" $ show.simple.relevance : "Rle" "Rlp" "Rls" "Rls.symbol" $ show.simple.test : "Sig0" "p.symbol" $ show.simple.classical : "statistic" "p.value" "p.symbol" $ show.term.relevance : "df" "R2.x" "coefRlp" "coefRls" ... $ show.term.test : "df" "ciLow" "ciUp" "R2.x" ... $ show.term.classical : "statistic" "df" "ciLow" "ciUp" ... $ show.termeff.relevance: "coef" "coefRls.symbol" $ show.termeff.test : "coef" "p.symbol" $ show.termeff.classical: "coef" "p.symbol" $ show.symbollegend : TRUE $ na.print : "." $ p.symbols : List, see below $ rlv.symbols : List, see below

rlv.symbols List $ symbol : " " "." "+" "++" "+++" $ cutpoint: -Inf 0 1 2 5 Inf

p.symbols List $ symbol : "***" "**" "*" "." " " $ cutpoint: 0 0.001 0.01 0.05 0.1 1

Examples

```
relevance.options
options(relevance.options) ## restores the package's default options
```

showd *Show a Part of a Data.frame*

Description

Shows a part of the data.frame which allows for grasping the nature of the data. The function is typically used to make sure that the data is what was desired and to grasp the nature of the variables in the phase of getting acquainted with the data.

Usage

```
showd(data, first = 3, nrow. = 4, ncol. = NULL, digits=getOption("digits"))
```

Arguments

`data`	a data.frame, a matrix, or a vector
`first`	the first `first` rows will be shown and ...
`nrow.`	a selection of `nrow.` rows will be shown in addition. They will be selected with equal row number differences. The last row is always included.
`ncol.`	number of columns (variables) to be shown. The first and last columns will also be included. If `ncol.` has more than one element, it is used to identify the columns directly.
`digits`	number of significant digits used in formatting numbers

Value

returns invisibly the character vector containing the formatted data

Author(s)

Werner A. Stahel, ETH Zurich

See Also

`head` and `tail`.

Examples

```
showd(iris)

data(d.blast)
names(d.blast)
## only show 3 columns, including the first and last
showd(d.blast, ncol=3)

showd(cbind(1:100))
```

sumNA *Count NAs*

Description

Count the missing or non-finite values for each column of a matrix or data.frame

Usage

```
sumNA(object, inf = TRUE)
```

Arguments

object	a vector, matrix, or data.frame
inf	if TRUE, Inf and NaN values are counted along with NAs

Value

numerical vector containing the missing value counts for each column

Note

This is a simple shortcut for `apply(is.na(object),2,sum)` or `apply(!is.finite(object),2,sum)`

Author(s)

Werner A. Stahel, ETH Zurich

See Also

is.na, is.finite, dropNA

Examples

```
t.d <- data.frame(V1=c(1,2,NA,4), V2=c(11,12,13,Inf), V3=c(21,NA,23,Inf))
sumNA(t.d)
```

termeffects *All Coefficients of a Model Fit*

Description

A list of all coefficients of a model fit, possibly with respective statistics

Usage

```
termeffects(object, se = 2, df = df.residual(object), rlv = TRUE,
  rlv.threshold = getOption("rlv.threshold"), ...)
```

Arguments

object	a model fit, produced, e.g., by a call to lm or regr.
se	logical: Should inference statistics be generated?
df	degrees of freedom for t-test
rlv	logical: Should relevances be calculated?
rlv.threshold	Relevance thresholds, see inference
...	further arguments, passed to inference

Value

a list with a component for each term in the model formula. Each component is a termtable for the coefficients corresponding to the term.

Author(s)

Werner A. Stahel

See Also

dummy.coef, inference, termtable

Examples

```
data(d.blast)
r.blast <-
  lm(log10(tremor)~location+log10(distance)+log10(charge), data=d.blast)
termeffects(r.blast)
```

termtable *Statistics for Linear Models, Including Relevance Statistics*

Description

Calculate a table of statistics for (multiple) regression mdels with a linear predictor

Usage

```
termtable(object, summary = summary(object), testtype = NULL,
  r2x = TRUE, rlv = TRUE, rlv.threshold = getOption("rlv.threshold"),
  testlevel = getOption("testlevel"))

relevance.modelclasses
```

Arguments

object	result of a model fitting function like lm
summary	result of summary(object). If NULL, the summary will be called.
testtype	type of test to be applied for dropping each term in turn. If NULL, it is selected according to the class of the object, see Details.
r2x	logical: should the collinearity measures "R2.x" (see below) for the terms be calculated?
rlv	logical: Should relevances be calculated?
rlv.threshold	Relevance thresholds, vector containing the elements rel: threshold for relative effects, coef: for standardized coefficients, drop: for drop effects, pred: for prediction intervals.
testlevel	1 - confidence level

Details

relevance.modelclasses collects the names of classes of model fitting results that can be handled by termtable.

If testtype is not specified, it is determined by the class of object and its attribute family as follows:
— For t for objects of class lm, lmrob and glm with families quasibinomial and quasipoisson,
— Chi-squaredfor other glms and survreg

Value

data.frame with columns

coef: coefficients for terms with a single degree of freedom

df: degrees of freedom

se: standard error of coef

statistic: value of the test statistic

p.value, p.symbol: p value and symbol for it

Sig0: significance value for the test of coef==0

ciLow, ciUp: confidence interval for coef

stcoef: standardized coefficient (standardized using the standard deviation of the 'error' term, sigma, instead of the response's standard deviation)

st.Low, st.Up: confidence interval for stcoef

R2.x: collinearity measure ($= 1 - 1/vif$, where vif is the variance inflation factor)

coefRle: estimated relevance of coef

coefRls: secured relevance, lower end of confidence interval for the relevance of coef

coefRlp: potential relevance, the upper end of the confidence interval.

dropRle, dropRls, dropRlp: analogous values for drop effect

predRle, predRls, predRlp: analogous values for prediction effect

In addition, it has attributes

testtype: as determined by the argument testtype or the class and attributes of object.

fitclass: class and attributes of object.

family, dist: more specifications if applicable

Author(s)

Werner A. Stahel

References

Werner A. Stahel (2020). Measuring Significance and Relevance instead of p-values. Submitted

See Also

getcoeftable; for printing options, print.inference

Examples

```
data(swiss)
rr <- lm(Fertility ~ . , data = swiss)
rt <- termtable(rr)
rt
```

`twosamples` *Relevance and Significance for One or Two Samples*

Description

Inference for a difference between two independent samples or for a single sample: Collect quantities for inference, including Relevance and Significance measures

Usage

```
twosamples(x, ...)
onesample(x, ...)

## Default S3 method:
twosamples(x, y = NULL, paired = FALSE, table = NULL,
  hypothesis = 0,var.equal = TRUE,
  testlevel=getOption("testlevel"), log = NULL, standardize = NULL,
  rlv.threshold=getOption("rlv.threshold"), ...)
## S3 method for class 'formula'
twosamples(x, data = NULL, subset, na.action, log = NULL, ...)
## S3 method for class 'table'
twosamples(x, ...)
```

Arguments

`x`	a formula or the data for the first or the single sample
`y`	data for the second sample
`table`	A `table` summarizing the data in case of binary (binomial) data. If given, `x` and `y` are ignored.
`paired`	logical: In case `x` and `y` are given. are their values paired?
`hypothesis`	the null effect to be tested, and anchor for the relevance
`var.equal`	logical: In case of two samples, should the variances be assumed equal? Only applies for quantitative data.
`testlevel`	level for the test, also determining the confidence level
`log`	logical...: Is the target variable on log scale? – or character: either "log" or "log10" (or "logst"). If so, no standardization is applied to it. By default, the function examines the formula to check whether the left hand side of the formula contains a log transformation.
`standardize`	logical: Should the effect be standardized (for quantiative data)?
`rlv.threshold`	Relevance threshold, or a vector of thresholds from which the element `stand` is taken for quantitative data and the element `prop`, for binary data.
	For the `formula` method:
`formula`	formula of the form `y~x` giving the target `y` and condition `x` variables. For a one-sample situation, use `y~1`.

`data` data from which the variables are obtained

`subset, na.action`
subset and na.action to be applied to `data`

`...` further arguments, ignored

Details

Argument `log`: If `log10` (or `logst` from package `plgraphics`) is used, rescaling is done (by `log(10)`) to obtain the correct relevance. Therefore, `log` needs to be set appropriately in this case.

Value

an object of `class` `'inference'`, a vector with elements

`effect`: for quantitative data: estimated difference between expectations of the two samples, or mean in case of a single sample.
For binary data: log odds (for one sample or paired samples) or log odds ratio (for two samples)

`ciLow, ciUp`: confidence interval for the effect

`Rle, Rls, Rlp`: relevance measures: estimated, secured, potential

`Sig0`: significance measure for test or 0 effect

`Sigth`: significance measure for test of `effect` == relevance threshold

`p.value`: p value for test against 0

In addition to the columns/components, it has `attributes`

`method`: problem and inference method

`effectname`: label for the effect

`hypothesis`: the null effect

`n`: number(s) of observations

`means`: in the case of 2 independent samples: their means

`statistic`: test statistic

`V`: single observation variance

`df`: degrees of freedom for the t distribution

`data`: if paired, vector of differences; if single sample, vector of data; if two independent samples, list containing the two samples

`rlv.threshold`: relevance threshold

Note

`onesample` and `twosamples` are identical. `twosamples.table(x,...)` just calls `twosamples.default(table=x, ...)`.

Author(s)

Werner A. Stahel

References

see those in relevance-package.

See Also

t.test, binom.test, fisher.test, mcnemar.test

Examples

```
data(sleep)
t.test(sleep[sleep$group == 1, "extra"], sleep[sleep$group == 2, "extra"])
twosamples(sleep[sleep$group == 1, "extra"], sleep[sleep$group == 2, "extra"])

## Two-sample test, wilcox.test example,  Hollander & Wolfe (1973), 69f.
## Permeability constants of the human chorioamnion (a placental membrane)
## at term and between 12 to 26 weeks gestational age
d.permeabililty <-
  data.frame(perm = c(0.80, 0.83, 1.89, 1.04, 1.45, 1.38, 1.91, 1.64, 0.73, 1.46,
                      1.15, 0.88, 0.90, 0.74, 1.21), atterm = rep(1:0, c(10,5))
             )
t.test(perm~atterm, data=d.permeabililty)
twosamples(perm~atterm, data=d.permeabililty)

## one sample
onesample(sleep[sleep$group == 2, "extra"])

## plot two samples
pltwosamples(extra ~ group, data=sleep)
```

Index