Package 'questionr'

January 31, 2022

Maintainer Julien Barnier <julien.barnier@cnrs.fr>

Version 0.7.7

Date 2022-01-31

License GPL (>= 2)

Encoding UTF-8

Title Functions to Make Surveys Processing Easier

Description Set of functions to make the processing and analysis of surveys easier : interactive shiny apps and addins for data recoding, contingency tables, dataset metadata handling, and several convenience functions.

Depends R (>= 3.5.0)

Imports shiny (>= 1.0.5), miniUI, rstudioapi, highr, styler, classInt, htmltools, graphics, stats, utils, labelled (>= 2.6.0)

Suggests testthat, roxygen2, dplyr, ggplot2, tidyr, janitor, forcats, knitr, rmarkdown, survey, Hmisc

SystemRequirements xclip (Linux)

VignetteBuilder knitr

URL https://juba.github.io/questionr/

BugReports https://github.com/juba/questionr/issues

RoxygenNote 7.1.2

NeedsCompilation no

Author Julien Barnier [aut, cre],
François Briatte [aut],
Joseph Larmarange [aut]

Repository CRAN

Date/Publication 2022-01-31 16:30:08 UTC

R topics documented:

---

addNAstr *Transform missing values of a factor to an extra level*

Description

This function modifies a factor by turning NA into an extra level (so that NA values are counted in tables, for instance). This version of addNA extends the same function provided in R by allowing to specify a string name for the extra level (see examples).

Usage

```
addNAstr(x, value = "NA", ...)
```

Arguments

x	a vector of data, usually taking a small number of distinct values.
value	string to use for the extra level name. If NULL, the extra level is created as NA, and the result is the same as the one of the addNA function.
...	arguments passed to addNA.

Value

an object of class "factor", original missing values being coded as an extra level named NA if as.string=FALSE, "NA" if as.string=TRUE, as specified by as.string if as.string is a string.

Source

Adapted from James (https://stackoverflow.com/a/5817181) by Joseph Larmarange <joseph@larmarange.net>

See Also

addNA (base).

Examples

```
f <- as.factor(c("a","b",NA,"a","b"))
f
addNAstr(f)
addNAstr(f, value="missing")
addNAstr(f, value=NULL)
```

children *A fertility survey - "children" table*

Description

Some fictive results from a fecondity survey.

Format

a data frame containing one record for each child of the surveyed women in the fertility survey.

chisq.residuals *Return the chi-squared residuals of a two-way frequency table.*

Description

Return the raw, standardized or Pearson's residuals (the default) of a chi-squared test on a two-way frequency table.

Usage

```
chisq.residuals(tab, digits = 2, std = FALSE, raw = FALSE)
```

Arguments

tab	frequency table
digits	number of digits to display
std	if TRUE, returns the standardized residuals. Otherwise, returns the Pearson residuals. Incompatible with raw.
raw	if TRUE, returns the raw (observed -expected) residuals. Otherwise, returns the Pearson residuals. Incompatible with std.

Details

This function is just a wrapper around the chisq.test base R function. See this function's help page for details on the computation.

See Also

chisq.test

Examples

```
## Sample table
data(Titanic)
tab <- apply(Titanic, c(1,4), sum)
## Pearson residuals
chisq.residuals(tab)
## Standardized residuals
chisq.residuals(tab, std = TRUE)
## Raw residuals
chisq.residuals(tab, raw = TRUE)
```

`clipcopy` *Transform an object into HTML and copy it for export*

Description

This function transforms its argument to HTML with knitr::kable and then copy it to the clipboard or to a file for later use in an external application.

Usage

```
clipcopy(obj, ...)

## Default S3 method:
clipcopy(
  obj,
  append = FALSE,
  file = FALSE,
  filename = "temp.html",
  clipboard.size = 4096,
  ...
)

## S3 method for class 'proptab'
clipcopy(obj, percent = NULL, digits = NULL, justify = "right", ...)
```

Arguments

`obj`	object to be copied
`...`	arguments passed to `knitr::kable`
`append`	if TRUE, append to the file instead of replacing it
`file`	if TRUE, export to a file instead of the clipboard
`filename`	name of the file to export to
`clipboard.size`	under Windows, size of the clipboard in kB
`percent`	whether to add a percent sign in each cell
`digits`	number of digits to display
`justify`	justification

Details

Under Linux, this function requires that `xclip` is installed on the system to copy to the clipboard.

Value

NULL

NULL

See Also

kable, format.proptab

clipcopy, format.proptab

Examples

```
data(iris)
tab <- table(cut(iris$Sepal.Length,8),cut(iris$Sepal.Width,4))
## Not run: copie(tab)
ptab <- rprop(tab, percent=TRUE)
## Not run: clipcopy(ptab)
```

---

cprop *Column percentages of a two-way frequency table.*

Description

Return the column percentages of a two-way frequency table with formatting and printing options.

Usage

```
cprop(tab, ...)

## S3 method for class 'table'
cprop(
  tab,
  digits = 1,
  total = TRUE,
  percent = FALSE,
  drop = TRUE,
  n = FALSE,
  ...
)

## S3 method for class 'data.frame'
cprop(
  tab,
  digits = 1,
```

```
  total = TRUE,
  percent = FALSE,
  drop = TRUE,
  n = FALSE,
  ...
)

## S3 method for class 'matrix'
cprop(
  tab,
  digits = 1,
  total = TRUE,
  percent = FALSE,
  drop = TRUE,
  n = FALSE,
  ...
)

## S3 method for class 'tabyl'
cprop(tab, digits = 1, total = TRUE, percent = FALSE, n = FALSE, ...)
```

Arguments

tab	frequency table
...	parameters passed to other methods.
digits	number of digits to display
total	if TRUE, add a row with the sum of percentages and a column with global percentages
percent	if TRUE, add a percent sign after the values when printing
drop	if TRUE, lines or columns with a sum of zero, which would generate NaN percentages, are dropped.
n	if TRUE, display number of observations per column.

Value

The result is an object of class table and proptab.

See Also

rprop, prop, table, prop.table

Examples

```
## Sample table
data(Titanic)
tab <- apply(Titanic, c(4,1), sum)
## Column percentages
cprop(tab)
```

```
## Column percentages with custom display
cprop(tab, digits=2, percent=TRUE, total=FALSE)
```

`cramer.v` *Compute Cramer's V of a two-way frequency table*

Description

This function computes Cramer's V for a two-way frequency table

Usage

```
cramer.v(tab)
```

Arguments

`tab`	table on which to compute the statistic

Examples

```
data(Titanic)
tab <- apply(Titanic, c(4,1), sum)
#' print(tab)
cramer.v(tab)
```

`cross.multi.table` *Two-way frequency table between a multiple choices question and a factor*

Description

This function allows to generate a two-way frequency table from a multiple choices question and a factor. The question's answers must be stored in a series of binary variables.

Usage

```
cross.multi.table(
  df,
  crossvar,
  weights = NULL,
  digits = 1,
  freq = FALSE,
  tfreq = "col",
  n = FALSE,
  na.rm = TRUE,
  ...
)
```

Arguments

df	data frame with the binary variables
crossvar	factor to cross the multiple choices question with
weights	optional weighting vector
digits	number of digits to keep in the output
freq	display percentages
tfreq	type of percentages to compute ("row" or "col")
n	if TRUE, and freq is TRUE, display number of observations per row or column
na.rm	Remove any NA values in crossvar
...	arguments passed to multi.table

Details

See the multi.table help page for details on handling of the multiple choices question and corresponding binary variables.

If freq is set to TRUE, the resulting table gives the columns percentages based on the contingency table of crossvar in the respondants population.

Value

Object of class table.

See Also

multi.table, multi.split, table

Examples

```
## Sample data frame
set.seed(1337)
sex <- sample(c("Man","Woman"),100,replace=TRUE)
jazz <- sample(c(0,1),100,replace=TRUE)
rock <- sample(c(TRUE, FALSE),100,replace=TRUE)
electronic <- sample(c("Y","N"),100,replace=TRUE)
weights <- runif(100)*2
df <- data.frame(sex,jazz,rock,electronic,weights)
## Two-way frequency table on 'music' variables by sex
cross.multi.table(df[,c("jazz", "rock","electronic")], df$sex, true.codes=list("Y"))
## Column percentages based on respondants
cross.multi.table(df[,c("jazz", "rock","electronic")], df$sex, true.codes=list("Y"), freq=TRUE)
## Row percentages based on respondants
cross.multi.table(df[,c("jazz", "rock","electronic")],
                  df$sex, true.codes=list("Y"), freq=TRUE, tfreq="row", n=TRUE)
```

describe *Describe the variables of a data.frame*

Description

This function describes the variables of a vector or a dataset that might include labels imported with **haven** packages.

Usage

```
describe(x, ...)

## S3 method for class 'factor'
describe(x, n = 10, show.length = TRUE, freq.n.max = 10, ...)

## S3 method for class 'numeric'
describe(x, n = 10, show.length = TRUE, freq.n.max = 10, ...)

## S3 method for class 'character'
describe(x, n = 10, show.length = TRUE, freq.n.max = 10, ...)

## Default S3 method:
describe(x, n = 10, show.length = TRUE, freq.n.max = 10, ...)

## S3 method for class 'haven_labelled'
describe(x, n = 10, show.length = TRUE, freq.n.max = 10, ...)

## S3 method for class 'data.frame'
describe(x, ..., n = 10, freq.n.max = 0)

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

Arguments

x	object to describe
...	further arguments passed to or from other methods, see details
n	number of first values to display
show.length	display length of the vector?
freq.n.max	display a frequency table if the number of unique values is less than this value, 0 to hide

Details

When describing a data.frame, you can provide variable names as character strings. Using the "*" or "|" wildcards in a variable name will search for it using a regex match. The search will also take into account variable labels, if any. See examples.

Value

an object of class `description`.

Author(s)

Joseph Larmarange <joseph@larmarange.net>

See Also

`lookfor`

Examples

```
data(hdv2003)
describe(hdv2003$sexe)
describe(hdv2003$age)
describe(hdv2003)
describe(hdv2003, "cuisine", "heures.tv")
describe(hdv2003, "trav*")
describe(hdv2003, "trav|lecture")
describe(hdv2003, "trav", "lecture")

data(fertility)
describe(women$residency)
describe(women)
describe(women, "id")
```

---

duplicated2 *Determine all duplicate elements*

Description

The native duplicated function determines which elements of a vector or data frame are duplicates of elements already observed in the vector or the data frame provided. Therefore, only the second occurence (or third or nth) of an element is considered as a duplicate. `duplicated2` is similar but will also mark the first occurence as a duplicate (see examples).

Usage

```
duplicated2(x)
```

Arguments

x	a vector, a data frame or a matrix

Value

A logical vector indicated wich elements are duplicated in `x`.

Source

http://forums.cirad.fr/logiciel-R/viewtopic.php?p=2968

See Also

duplicated

Examples

```
df <- data.frame(x=c("a","b","c","b","d","c"),y=c(1,2,3,2,4,3))
df
duplicated(df)
duplicated2(df)
```

---

enfants *A fertility survey - "enfants" table*

Description

Some fictive results from a fecondity survey.

Format

a data frame containing one record for each child of the surveyed women in the fecondite survey.

---

escape_regex *Escape regex special chars Code directly taken from Hmisc::escapeRegex*

Description

Escape regex special chars Code directly taken from Hmisc::escapeRegex

Usage

```
escape_regex(s)
```

Arguments

s	string to escape regex special chars from

fecondite *A fertility survey*

Description

Some fictive results from a fecondity survey, with French labels.

Format

3 data frames with labelled data (as if data would have been imported from SPSS with **haven**):

- `menages` contains some information from the households selected for the survey;
- `femmes` contains the questionnaire administered to all 15-49 years old women living in the selected households;
- `enfants` contains one record for each child of the surveyed women.

Data can be linked using the variables `id_menage` and `id_femme`.

See Also

fertility for an English version of this dataset.

Examples

```
data(fecondite)
describe(menages)
describe(femmes)
describe(enfants)
```

femmes *A fertility survey - "femmes" table*

Description

Some fictive results from a fecondity survey.

Format

a data frame containing the questionnaire administered to all 15-49 years old women living in the selected households for the fecondite survey.

fertility *A fertility survey*

Description

Some fictive results from a fecondity survey, with English labels.

Format

3 data frames with labelled data (as if data would have been imported from SPSS with **haven**):

- `households` contains some information from the households selected for the survey;
- `women` contains the questionnaire administered to all 15-49 years old women living in the selected households;
- `children` contains one record for each child of the surveyed women.

Data can be linked using the variables `id_household` and `id_woman`.

See Also

fecondite for an French version of this dataset.

Examples

```
data(fertility)
describe(households)
describe(women)
describe(children)
```

first_non_null *Return first non-null of two values*

Description

Return first non-null of two values

Usage

```
x %||% y
```

Arguments

`x`	first object
`y`	second object

format.proptab *S3 format method for proptab objects.*

Description

Format an object of class proptab for printing depending on its attributes.

Usage

```
## S3 method for class 'proptab'
format(x, digits = NULL, percent = NULL, justify = "right", ...)
```

Arguments

x	object of class proptab
digits	number of digits to display
percent	if not NULL, add a percent sign after each value
justify	justification of character vectors. Passed to format.default
...	other arguments to pass to format.default

Details

This function is designed for internal use only.

See Also

format.default, print.proptab

freq *Generate frequency tables.*

Description

Generate and format frequency tables from a variable or a table, with percentages and formatting options.

Usage

```
freq(
  x,
  digits = 1,
  cum = FALSE,
  total = FALSE,
  exclude = NULL,
  sort = "",
  valid = !(NA %in% exclude),
  levels = c("prefixed", "labels", "values"),
  na.last = TRUE
)
```

Arguments

x	either a vector to be tabulated, or a table object
digits	number of digits to keep for the percentages
cum	if TRUE, display cumulative percentages
total	if TRUE, add a final row with totals
exclude	vector of values to exclude from the tabulation (if x is a vector)
sort	if specified, allow to sort the table by increasing ("inc") or decreasing ("dec") frequencies
valid	if TRUE, display valid percentages
levels	the desired levels for the factor in case of labelled vector (**labelled** package must be installed): "labels" for value labels, "values" for values or "prefixed" for labels prefixed with values
na.last	if TRUE, NA values are always be last table row

Value

The result is an object of class data.frame.

See Also

table, prop, cprop, rprop

Examples

```
# factor
data(hdv2003)
freq(hdv2003$qualif)
freq(hdv2003$qualif, cum = TRUE, total = TRUE)
freq(hdv2003$qualif, cum = TRUE, total = TRUE, sort ="dec")

# labelled data
data(fecondite)
freq(femmes$region)
```

```
freq(femmes$region, levels = "l")
freq(femmes$region, levels = "v")
```

---

freq.na *Generate frequency table of missing values.*

Description

Generate a frequency table of missing values as raw counts and percentages.

Usage

```
freq.na(data, ...)
```

Arguments

data	either a vector or a data frame object
...	if x is a data frame, the names of the variables to examine or keywords to search for such variables. See lookfor for more details.

Value

The result is an object of class data.frame.

See Also

table, is.na

Examples

```
data(hdv2003)
## Examine a single vector.
freq.na(hdv2003$qualif)
## Examine a data frame.
freq.na(hdv2003)
## Examine several variables.
freq.na(hdv2003, "nivetud", "trav.satisf")
## To see only variables with the most number of missing values
head(freq.na(hdv2003))
```

ggsurvey *Easy ggplot2 with survey objects*

Description

A function to facilitate ggplot2 graphs using a survey object. It will initiate a ggplot and map survey weights to the corresponding aesthetic.

Usage

```
ggsurvey(design = NULL, mapping = NULL, ...)
```

Arguments

design	A survey design object, usually created with survey::svydesign()
mapping	Default list of aesthetic mappings to use for plot, to be created with ggplot2::aes().
...	Other arguments passed on to methods. Not currently used.

Details

Graphs will be correct as long as only weights are required to compute the graph. However, statistic or geometry requiring correct variance computation (like ggplot2::geom_smooth()) will be statistically incorrect.

Examples

```
if (require(survey) & require(ggplot2)) {
  data(api)
  dstrat <- svydesign(
    id = ~1, strata = ~stype,
    weights = ~pw, data = apistrat,
    fpc = ~fpc
  )
  ggsurvey(dstrat) +
    aes(x = cnum, y = dnum) +
    geom_count()

  d <- as.data.frame(Titanic)
  dw <- svydesign(ids = ~1, weights = ~Freq, data = d)
  ggsurvey(dw) +
    aes(x = Class, fill = Survived) +
    geom_bar(position = "fill")
}
```

happy *Data related to happiness from the General Social Survey, 1972-2006.*

Description

This data extract is taken from Hadley Wickham's `productplots` package. The original description follows, with minor edits.

The data is a small sample of variables related to happiness from the General Social Survey (GSS). The GSS is a yearly cross-sectional survey of Americans, run from 1972. We combine data for 25 years to yield 51,020 observations, and of the over 5,000 variables, we select nine related to happiness:

Format

A data frame with 51020 rows and 10 variables

Details

- age. age in years: 18–89.
- degree. highest education: lt high school, high school, junior college, bachelor, graduate.
- finrela. relative financial status: far above, above average, average, below average, far below.
- happy. happiness: very happy, pretty happy, not too happy.
- health. health: excellent, good, fair, poor.
- marital. marital status: married, never married, divorced, widowed, separated.
- sex. sex: female, male.
- wtsall. probability weight. 0.43–6.43.

References

Smith, Tom W., Peter V. Marsden, Michael Hout, Jibum Kim. *General Social Surveys, 1972-2006*. [machine-readable data file]. Principal Investigator, Tom W. Smith; Co-Principal Investigators, Peter V. Marsden and Michael Hout, NORC ed. Chicago: National Opinion Research Center, producer, 2005; Storrs, CT: The Roper Center for Public Opinion Research, University of Connecticut, distributor. 1 data file (57,061 logical records) and 1 codebook (3,422 pp).

hdv2003 *Histoire de vie 2003*

Description

Sample from 2000 people and 20 variables taken from the *Histoire de Vie* survey, produced in France in 2003 by INSEE.

Format

A data frame with 2000 rows and 20 variables

Source

https://www.insee.fr/fr/statistiques/2532244

households *A fertility survey - "households" table*

Description

Some fictive results from a fecondity survey.

Format

a data frame containing some information from the households selected for the fertility survey.

icut *Interactive conversion from numeric to factor*

Description

This function launches a shiny app in a web browser in order to do interactive conversion of a numeric variable into a categorical one.

Usage

```
icut(obj = NULL, var_name = NULL)
```

Arguments

obj	vector to recode or data frame to operate on
var_name	if obj is a data frame, name of the column to be recoded, as a character string (possibly without quotes)

Value

The function launches a shiny app in the system web browser. The recoding code is returned in the console when the app is closed with the "Done" button.

Examples

```
## Not run:
data(hdv2003)
icut(hdv2003, "age")
irec(hdv2003, heures.tv)

## End(Not run)
```

---

iorder *Interactive reordering of factor levels*

Description

This function launches a shiny app in a web browser in order to do interactive reordering of the levels of a categorical variable (character or factor).

Usage

```
iorder(obj = NULL, var_name = NULL)
```

Arguments

obj	vector to recode or data frame to operate on
var_name	if obj is a data frame, name of the column to be recoded, as a character string possibly without quotes)

Details

The generated convert the variable into a factor, as only those allow for levels ordering.

Value

The function launches a shiny app in the system web browser. The reordering code is returned in he console when the app is closed with the "Done" button.

Examples

```
## Not run:
data(hdv2003)
iorder(hdv2003, "qualif")

## End(Not run)
```

irec *Interactive recoding*

Description

This function launches a shiny app in a web browser in order to do interactive recoding of a categorical variable (character or factor).

Usage

```
irec(obj = NULL, var_name = NULL)
```

Arguments

obj	vector to recode or data frame to operate on
var_name	if obj is a data frame, name of the column to be recoded, as a character string possibly without quotes)

Value

The function launches a shiny app in the system web browser. The recoding code is returned in the onsole when the app is closed with the "Done" button.

Examples

```
## Not run:
data(hdv2003)
irec()
v <- sample(c("Red", "Green", "Blue"), 50, replace = TRUE)
irec(v)
irec(hdv2003, "qualif")
irec(hdv2003, sexe) ## this also works

## End(Not run)
```

ltabs *Cross tabulation with labelled variables*

Description

This function is a wrapper around xtabs, adding automatically value labels for labelled vectors if **labelled** package eis installed.

Usage

```
ltabs(
  formula,
  data,
  levels = c("prefixed", "labels", "values"),
  variable_label = TRUE,
  ...
)
```

Arguments

formula	a formula object (see xtabs)
data	a data frame
levels	the desired levels in case of labelled vector: "labels" for value labels, "values" for values or "prefixed" for labels prefixed with values
variable_label	display variable label if available?
...	additional arguments passed to xtabs

See Also

xtabs.

Examples

```
data(fecondite)
ltabs(~radio, femmes)
ltabs(~radio+tv, femmes)
ltabs(~radio+tv, femmes, "l")
ltabs(~radio+tv, femmes, "v")
ltabs(~radio+tv+journal, femmes)
ltabs(~radio+tv, femmes, variable_label = FALSE)
```

---

menages *A fertility survey - "menages" table*

Description

Some fictive results from a fecondity survey.

Format

a data frame containing some information from the households selected for the fecondite survey.

`multi.split` *Split a multiple choices variable in a series of binary variables*

Description

Split a multiple choices variable in a series of binary variables

Usage

```
multi.split(var, split.char = "/", mnames = NULL)
```

Arguments

`var`	variable to split
`split.char`	character to split at
`mnames`	names to give to the produced variabels. If NULL, the name are computed from the original variable name and the answers.

Details

This function takes as input a multiple choices variable where choices are recorded as a string and separated with a fixed character. For example, if the question is about the favourite colors, answers could be "red/blue", "red/green/yellow", etc. This function splits the variable into as many variables as the number of different choices. Each of these variables as a 1 or 0 value corresponding to the choice of this answer. They are returned as a data frame.

Value

Returns a data frame.

See Also

`multi.table`

Examples

```
v <- c("red/blue","green","red/green","blue/red")
multi.split(v)
## One-way frequency table of the result
multi.table(multi.split(v))
```

multi.table *One-way frequency table for multiple choices question*

Description

This function allows to generate a frequency table from a multiple choices question. The question's answers must be stored in a series of binary variables.

Usage

```
multi.table(df, true.codes = NULL, weights = NULL, digits = 1, freq = TRUE)
```

Arguments

df	data frame with the binary variables
true.codes	optional list of values considered as 'true' for the tabulation
weights	optional weighting vector
digits	number of digits to keep in the output
freq	add a percentage column

Details

The function is applied to a series of binary variables, each one corresponding to a choice of the question. For example, if the question is about seen movies among a movies list, each binary variable would correspond to a movie of the list and be true or false depending of the choice of the answer.

By default, only '1' and 'TRUE' as considered as 'true' values fro the binary variables, and counted in the frequency table. It is possible to specify other values to be counted with the true.codes argument. Note than '1' and 'TRUE' are always considered as true values even if true.codes is provided.

If freq is set to TRUE, a percentage column is added to the resulting table. This percentage is computed by dividing the number of TRUE answers for each value by the total number of (potentially weighted) observations. Thus, these percentages sum can be greater than 100.

Value

Object of class table.

See Also

cross.multi.table, multi.split, table

Examples

```
## Sample data frame
set.seed(1337)
sex <- sample(c("Man","Woman"),100,replace=TRUE)
jazz <- sample(c(0,1),100,replace=TRUE)
rock <- sample(c(TRUE, FALSE),100,replace=TRUE)
electronic <- sample(c("Y","N"),100,replace=TRUE)
weights <- runif(100)*2
df <- data.frame(sex,jazz,rock,electronic,weights)
## Frequency table on 'music' variables
multi.table(df[,c("jazz", "rock","electronic")], true.codes=list("Y"))
## Weighted frequency table on 'music' variables
multi.table(df[,c("jazz", "rock","electronic")], true.codes=list("Y"), weights=df$weights)
## No percentages
multi.table(df[,c("jazz", "rock","electronic")], true.codes=list("Y"), freq=FALSE)
```

`na.rm` *Remove observations with missing values*

Description

`na.rm` is similar to na.omit but allows to specify a list of variables to take into account.

Usage

```
na.rm(x, v = NULL)
```

Arguments

`x`	a data frame
`v`	a list of variables

Details

If `v` is not specified, the result of `na.rm` will be the same as na.omit. If a list of variables is specified through `v`, only observations with a missing value (`NA`) for one of the specified variables will be removed from `x`. See examples.

Author(s)

Joseph Larmarange <joseph@larmarange.net>

See Also

na.omit

Examples

```
df <- data.frame(x = c(1, 2, 3), y = c(0, 10, NA), z= c("a",NA,"b"))
df
na.omit(df)
na.rm(df)
na.rm(df, c("x","y"))
na.rm(df, "z")
```

`odds.ratio` *Odds Ratio*

Description

S3 method for odds ratio

Usage

```
odds.ratio(x, ...)

## S3 method for class 'glm'
odds.ratio(x, level = 0.95, ...)

## S3 method for class 'multinom'
odds.ratio(x, level = 0.95, ...)

## S3 method for class 'factor'
odds.ratio(x, fac, level = 0.95, ...)

## S3 method for class 'table'
odds.ratio(x, level = 0.95, ...)

## S3 method for class 'matrix'
odds.ratio(x, level = 0.95, ...)

## S3 method for class 'numeric'
odds.ratio(x, y, level = 0.95, ...)

## S3 method for class 'odds.ratio'
print(x, signif.stars = TRUE, ...)
```

Arguments

`x`	object from whom odds ratio will be computed
`...`	further arguments passed to or from other methods
`level`	the confidence level required
`fac`	a second factor object
`y`	a second numeric object
`signif.stars`	logical; if `TRUE`, p-values are encoded visually as 'significance stars'

Details

For models calculated with `glm`, `x` should have been calculated with `family=binomial`. p-value are the same as `summary(x)$coefficients[,4]`. Odds ratio could also be obtained with `exp(coef(x))` and confidence intervals with `exp(confint(x))`.

For models calculated with `multinom` (nnet), p-value are calculated according to https://stats.oarc.ucla.edu/r/dae/multinomial-logistic-regression/.

For 2x2 `table`, `factor` or `matrix`, `odds.ratio` uses `fisher.test` to compute the odds ratio.

Value

Returns a data.frame of class `odds.ratio` with odds ratios, their confidence interval and p-values.

If `x` and `y` are proportions, `odds.ratio` simply returns the value of the odds ratio, with no confidence interval.

Author(s)

Joseph Larmarange <joseph@larmarange.net>

See Also

`glm` in the stats package.

`multinom` in the nnet package.

`fisher.test` in the stats package.

`printCoefmat` in the stats package.

Examples

```
data(hdv2003)
reg <- glm(cinema ~ sexe + age, data=hdv2003, family=binomial)
odds.ratio(reg)
odds.ratio(hdv2003$sport, hdv2003$cuisine)
odds.ratio(table(hdv2003$sport, hdv2003$cuisine))
M <- matrix(c(759, 360, 518, 363), ncol = 2)
odds.ratio(M)
odds.ratio(0.26, 0.42)
```

---

print.proptab *S3 print method for proptab objects.*

Description

Print an object of class proptab.

Usage

```
## S3 method for class 'proptab'
print(x, digits = NULL, percent = NULL, justify = "right", ...)
```

Arguments

x	object of class proptab
digits	number of digits to display
percent	if not NULL, add a percent sign after each value
justify	justification of character vectors. Passed to `format.default`
...	other arguments to pass to `format.default`

See Also

`format.proptab`

---

prop *Global percentages of a two-way frequency table.*

Description

Return the percentages of a two-way frequency table with formatting and printing options.

Usage

```
prop(tab, ...)

prop_table(
  tab,
  digits = 1,
  total = TRUE,
  percent = FALSE,
  drop = TRUE,
  n = FALSE,
  ...
)

## S3 method for class 'data.frame'
prop(
  tab,
  digits = 1,
  total = TRUE,
  percent = FALSE,
  drop = TRUE,
  n = FALSE,
  ...
)

## S3 method for class 'matrix'
prop(
```

```
  tab,
  digits = 1,
  total = TRUE,
  percent = FALSE,
  drop = TRUE,
  n = FALSE,
  ...
)

## S3 method for class 'tabyl'
prop(tab, digits = 1, total = TRUE, percent = FALSE, n = FALSE, ...)
```

Arguments

tab	frequency table
...	parameters passed to other methods
digits	number of digits to display
total	if TRUE, add a column with the sum of percentages and a row with global percentages
percent	if TRUE, add a percent sign after the values when printing
drop	if TRUE, lines or columns with a sum of zero, which would generate NaN percentages, are dropped.
n	if TRUE, display number of observations per row and per column.

Value

The result is an object of class table and proptab.

See Also

rprop, cprop, table, prop.table

Examples

```
## Sample table
data(Titanic)
tab <- apply(Titanic, c(1,4), sum)
## Percentages
prop(tab)
## Percentages with custom display
prop(tab, digits=2, percent=TRUE, total=FALSE, n=TRUE)
```

qload *Load one or more packages, installing them first if necessary*

Description

This function quickly loads one or more packages, installing them quietly if necessary.

Usage

```
qload(..., load = TRUE, silent = TRUE)
```

Arguments

...	the packages to load/install. Packages are loaded with `library` and installed first with `install.packages` if necessary.
load	load the packages. Set to `FALSE` to just install any missing packages. Defaults to `TRUE`.
silent	keep output as silent as possible. Defaults to `TRUE`.

Details

The function probably requires R 3.0.0 or above to make use of the `quiet` argument when calling `install.packages`. It is not clear what the argument previously achieved in older versions of R.

Value

The result is a list of packages cited in the scripts.

Author(s)

François Briatte <f.briatte@gmail.com>

See Also

`qscan`, `install.packages`, `library`

Examples

```
qload("questionr")
qload("questionr", silent = FALSE)
```

qscan *Scan R scripts and load/install all detected packages*

Description

This function scans one or more R scripts and tries to quick-load/install the packages mentioned by `library` or `require` functions.

Usage

```
qscan(..., load = TRUE, detail = TRUE)
```

Arguments

`...`	the scripts to scan. Defaults to all R scripts in the current working directory.
`load`	quick-load/install the cited packages (see details). Defaults to `TRUE`.
`detail`	show the list of packages found in each script. Defaults to `TRUE`.

Details

The function calls the `qload` function to quick-load/install the packages.

Value

The result is a list of packages cited in the scripts.

Author(s)

François Briatte <f.briatte@gmail.com>

See Also

`qload`, `library`

Examples

```
## Scan the working directory.
## Not run: qscan()
```

quant.cut *Transform a quantitative variable into a qualitative variable*

Description

This function transforms a quantitative variable into a qualitative one by breaking it into classes with the same frequencies.

Usage

```
quant.cut(var, nbclass, include.lowest = TRUE, right = FALSE, dig.lab = 5, ...)
```

Arguments

var	variable to transform
nbclass	number of classes
include.lowest	argument passed to the cut function
right	argument passed to the cut function
dig.lab	argument passed to the cut function
...	arguments passed to the cut function

Details

This is just a simple wrapper around the cut and quantile functions.

Value

The result is a factor.

See Also

cut, quantile

Examples

```
data(iris)
sepal.width3cl <- quant.cut(iris$Sepal.Width,3)
table(sepal.width3cl)
```

recode.na *Recode values of a variable to missing values, using exact or regular expression matching.*

Description

This function recodes selected values of a quantitative or qualitative variable by matching its levels to exact or regular expression matches.

Usage

```
recode.na(x, ..., verbose = FALSE, regex = TRUE, as.numeric = FALSE)
```

Arguments

x	variable to recode. The variable is coerced to a factor if necessary.
...	levels to recode as missing in the variable. The values are coerced to character strings, meaning that you can pass numeric values to the function.
verbose	print a table of missing levels before recoding them as missing. Defaults to FALSE.
regex	use regular expressions to match values that include the "*" or "|" wildcards. Defaults to TRUE.
as.numeric	coerce the recoded variable to numeric. The function recommends the option when the recode returns only numeric values. Defaults to FALSE.

Value

The result is a factor with properly encoded missing values. If the recoded variable contains only numeric values, it is converted to an object of class numeric.

Author(s)

François Briatte <f.briatte@gmail.com>

See Also

regex

Examples

```
data(hdv2003)
## With exact string matches.
hdv2003$nivetud = recode.na(hdv2003$nivetud, "Inconnu")
## With regular expressions.
hdv2003$relig = recode.na(hdv2003$relig, "[A|a]ppartenance", "Rejet|NSP")
## Showing missing values.
hdv2003$clso = recode.na(hdv2003$clso, "Ne sait pas", verbose = TRUE)
```

```
## Test results with freq.
freq(recode.na(hdv2003$trav.satisf, "Equilibre"))
## Truncate a count variable (recommends numeric conversion).
freq(recode.na(hdv2003$freres.soeurs, 5:22))
```

`rename.variable` *Rename a data frame column*

Description

Rename a data frame column

Usage

```
rename.variable(df, old, new)
```

Arguments

`df`	data frame
`old`	old name
`new`	new name

Value

A data frame with the column named "old" renamed as "new"

Examples

```
data(iris)
str(iris)
iris <- rename.variable(iris, "Species", "especes")
str(iris)
```

`rm.unused.levels` *Remove unused levels*

Description

This function removes unused levels of a factor or in a data.frame. See examples.

Usage

```
rm.unused.levels(x, v = NULL)
```

Arguments

x	a factor or a data frame
v	a list of variables (optional, if x is a data frame)

Details

If x is a data frame, only factor variables of x will be impacted. If a list of variables is provided through v, only the unused levels of the specified variables will be removed.

Author(s)

Joseph Larmarange <joseph@larmarange.net>

Examples

```
df <- data.frame(v1=c("a","b","a","b"),v2=c("x","x","y","y"))
df$v1 <- factor(df$v1,c("a","b","c"))
df$v2 <- factor(df$v2,c("x","y","z"))
df
str(df)
str(rm.unused.levels(df))
str(rm.unused.levels(df,"v1"))
```

---

rp2012 *2012 French Census - French cities of more than 2000 inhabitants*

Description

Sample from the 2012 national french census. It contains results for every french city of more than 2000 inhabitants, and a small subset of variables, both in population counts and proportions.

Format

A data frame with 5170 rows and 60 variables

Source

https://www.insee.fr/fr/information/2008354

rp2018 *2018 French Census - French cities of more than 2000 inhabitants*

Description

Sample from the 2018 national french census. It contains results for every french city of more than 2000 inhabitants, and a small subset of variables, both in population counts and proportions.

Format

A data frame with 5417 rows and 62 variables

Source

https://www.insee.fr/fr/information/5369871

rprop *Row percentages of a two-way frequency table.*

Description

Return the row percentages of a two-way frequency table with formatting and printing options.

Usage

```
rprop(tab, ...)

## S3 method for class 'table'
rprop(
  tab,
  digits = 1,
  total = TRUE,
  percent = FALSE,
  drop = TRUE,
  n = FALSE,
  ...
)

## S3 method for class 'data.frame'
rprop(
  tab,
  digits = 1,
  total = TRUE,
  percent = FALSE,
  drop = TRUE,
```

```
  n = FALSE,
  ...
)

## S3 method for class 'matrix'
rprop(
  tab,
  digits = 1,
  total = TRUE,
  percent = FALSE,
  drop = TRUE,
  n = FALSE,
  ...
)

## S3 method for class 'tabyl'
rprop(tab, digits = 1, total = TRUE, percent = FALSE, n = FALSE, ...)
```

Arguments

tab	frequency table
...	parameters passed to other methods.
digits	number of digits to display
total	if TRUE, add a column with the sum of percentages and a row with global percentages
percent	if TRUE, add a percent sign after the values when printing
drop	if TRUE, lines or columns with a sum of zero, which would generate NaN percentages, are dropped.
n	if TRUE, display number of observations per row.

Value

The result is an object of class table and proptab.

See Also

cprop, prop, table, prop.table

Examples

```
## Sample table
data(Titanic)
tab <- apply(Titanic, c(1,4), sum)
## Column percentages
rprop(tab)
## Column percentages with custom display
rprop(tab, digits=2, percent=TRUE, total=FALSE)
```

tabs *Weighted Crossresult*

Description

Generate table with multiple weighted crossresult (full sample is first column). kable(), which is found in library(knitr), is recommended for use with RMarkdown.

Usage

```
tabs(
  df,
  x,
  y,
  type = "percent",
  percent = FALSE,
  weight = NULL,
  normwt = FALSE,
  na.rm = TRUE,
  na.show = FALSE,
  exclude = NULL,
  digits = 1
)
```

Arguments

df	A data.frame that contains x and (optionally) y and weight.
x	variable name (found in df). tabs(my.data, x = 'q1').
y	one (or more) variable names. tabs(my.data, x = 'q1', y = c('sex', 'job')).
type	'percent' (default ranges 0-100), 'proportion', or 'counts' (type of table returned).
percent	if TRUE, add a percent sign after the values when printing
weight	variable name for weight (found in df).
normwt	if TRUE, normalize weights so that the total weighted count is the same as the unweighted one
na.rm	if TRUE, remove NA values before computation
na.show	if TRUE, show NA count in table output
exclude	values to remove from x and y. To exclude NA, use na.rm argument.
digits	Number of digits to display; ?format.proptab for formatting details.

Details

tabs calls wtd.table on ‘x‘ and, as applicable, each variable named by ‘y‘.

Author(s)

Pete Mohanty

Examples

```
data(hdv2003)
tabs(hdv2003, x = "relig", y = c("qualif", "trav.imp"), weight = "poids")
result <- tabs(hdv2003, x = "relig", y = c("qualif", "trav.imp"), type = "counts")
format(result, digits = 3)
# library(knitr)
# xt <- tabs(hdv2003, x = "relig", y = c("qualif", "trav.imp"), weight = "poids")
# kable(format(xt))                        # to use with RMarkdown...
```

---

women *A fertility survey - "women" table*

Description

Some fictive results from a fecondity survey.

Format

a data frame containing the questionnaire administered to all 15-49 years old women living in the selected households for the fertility survey.

---

wtd.mean *Weighted mean and variance of a vector*

Description

Compute the weighted mean or weighted variance of a vector. Exact copies of Hmisc functions.

Usage

```
wtd.mean(x, weights = NULL, na.rm = TRUE)
```

Arguments

x	Numeric data vector
weights	Numeric weights vector. Must be the same length as x
na.rm	if TRUE, delete NA values.

Details

If weights is NULL, then an uniform weighting is applied.

Author(s)

These functions are exact copies of the `wtd.mean` and `wtd.var` function from the wtd.stats package. They have been created by Frank Harrell, Department of Biostatistics, Vanderbilt University School of Medicine, <f.harrell@vanderbilt.edu>.

See Also

`mean`,`var`, `wtd.table` and the `survey` package.

Examples

```
data(hdv2003)
mean(hdv2003$age)
wtd.mean(hdv2003$age, weights=hdv2003$poids)
```

`wtd.table` *Weighted one-way and two-way frequency tables.*

Description

Generate weighted frequency tables, both for one-way and two-way tables.

Usage

```
wtd.table(
  x,
  y = NULL,
  weights = NULL,
  digits = 3,
  normwt = FALSE,
  useNA = c("no", "ifany", "always"),
  na.rm = TRUE,
  na.show = FALSE,
  exclude = NULL
)
```

Arguments

`x`	a vector
`y`	another optional vector for a two-way frequency table. Must be the same length as `x`
`weights`	vector of weights, must be the same length as `x`
`digits`	Number of significant digits.
`normwt`	if TRUE, normalize weights so that the total weighted count is the same as the unweighted one
`useNA`	wether to include NA values in the table

na.rm	(deprecated) if TRUE, remove NA values before computation
na.show	(deprecated) if TRUE, show NA count in table output
exclude	values to remove from x and y. To exclude NA, use na.rm argument.

Details

If weights is not provided, an uniform weghting is used.

If some weights are missing ('NA'), they are converted to zero. In case of missing weights with 'normwt=TRUE', the observations with missing weights are still counted in the unweighted count. You have to filter them out before using this function if you don't want them to be taken into account when using 'normwt'.

Value

If y is not provided, returns a weighted one-way frequency table of x. Otherwise, returns a weighted two-way frequency table of x and y

See Also

wtd.table, table, and the survey package.

Examples

```
data(hdv2003)
wtd.table(hdv2003$sexe, weights=hdv2003$poids)
wtd.table(hdv2003$sexe, weights=hdv2003$poids, normwt=TRUE)
table(hdv2003$sexe, hdv2003$hard.rock)
wtd.table(hdv2003$sexe, hdv2003$hard.rock, weights=hdv2003$poids)
```

Index