Package 'ryouready'

July 2, 2014

Maintainer Mark Heckmann <heckmann@uni-bremen.de>

License GPL (>= 2)

Title Companion to the 'R your ready?' book

LazyData yes

Encoding UTF-8

Type Package

LazyLoad yes

Author Mark Heckmann

Description Data, functions etc. from the 'R you ready?' book

Version 0.3

Date 2013-12-15

Depends stringr, car

Collate 'ryouready-package.r' 'association_measures.r' 'utils.r' 'collapse_response_set.R' 'data-doc.R'

NeedsCompilation no

Repository CRAN

Date/Publication 2013-12-15 16:18:46

R topics documented:

`collapse_responseset.data.frame`
Collapse multiple response sets to single variable

Description

This functions allows to collapse several multiple response set varables into one variable. It can be applied either to a `dataframe` or within the `transform` function.

Usage

```
 ## S3 method for class 'data.frame'
collapse_responseset(x,
   vars = NULL, rec = NULL, ...)

 ## Default S3 method:
collapse_responseset(..., rec = NULL)

 collapse_responseset(x, ...)
```

Arguments

`x`	A dataframe.
`...`	Several vector of the same length (for default method).
`vars`	The names or indexes of the dataframe columns that contain the multi response set. By default all variables from dataframe are used.
`rec`	A vector of the same length as the number of variables specifying the new values for each column.

Value

A vector with the with the new values.

Author(s)

Mark Heckmann

Examples

```
d <- data.frame(t1=c(1,0,NA,0,0),
                t2=c(0,1,0,NA,0),
                t3=c(0,0,1,0,0) )

 # collapse all variables of a dataframe
 collapse_responseset(d)

 # collapse columns 1 to 3 (which is all in this case as well)
 collapse_responseset(d, vars=1:3)
 collapse_responseset(d, vars=c("t1", "t2", "t3"))

 # use letters instead fo numbers for recoding
 collapse_responseset(d, vars=1:3, rec=letters[1:3])

 # use with several vectors
 collapse_responseset(d$t1, d$t2, d$t3)

 # use inside of transform
 transform(d, new=collapse_responseset(t1, t2, t3))

 transform(d, new=collapse_responseset(t1, t2, t3, rec=letters[1:3]))
```

---

d.eta *Sample data set for eta function examples*

Description

Data set for eta examples.

---

d.ngo *NGO Dataset*

Description

Data set used by Kähler (2008).

References

Kähler, W.-M. (2008). *Statistische Datenanalyse: Verfahren verstehen und mit SPSS gekonnt einsetzen*. Wiesbaden: Vieweg.

d.superiority *Student self assessment data*

Description

The participipoants were asked to assess in h

eta *Eta coefficient for nominal/interval data.*

Description

Eta coefficient for nominal/interval data.

Usage

```
eta(x, y, breaks = NULL, na.rm = FALSE)
```

Arguments

x	Independent nominal variable (factor or numeric).
y	Dependent interval variable (numeric).
breaks	If x is interval data the breaks argument can be specified to classify the data. breaks is passed on to the function cut.
na.rm	Logical. Indicating if NA values are removed.

Value

Eta coefficient

Author(s)

Mark Heckmann

Examples

```
attach(d.eta)     # using d.eta dataset
eta(x1, y)

# removing missing data
eta(c(x1, 2), c(NA, y), na.rm=TRUE)   # NA added to y to show NA behaviour

# classify interval data x
eta(x, y, breaks=c(1, 4, 7,10))
# visualize classication
plot(x, y)
```

```
abline(v=c(1, 4, 7,10))

# setting number of breaks for classification
eta(x, y, breaks=7)
```

`foo` *A function to demonstrate how an R function is defined.*

Description

A function to demonstrate how an R function is defined.

Usage

```
foo(x, y)
```

Arguments

`x`	Numeric.
`y`	Numeric.

Value

The sum of `x` and `y`.

Author(s)

Mark Heckmann

`intervals` *Use standard mathematical interval notation in* `recode` *from* `car` *package*

Description

The `recode` function from the `car` package is an excellent function for recoding data. When defining open intervals though, the recoding definitions will quickly become hard to read. The `intervals` function allows to use standard mathematical interval notation, e.g. like `[1,4)`, to define (open) intervals. It will convert the intervals definition into a format required by the `recode` function from `car`. The standard intervals can simply be used additionally to the standard recoding definitions as required by `recode`.

Usage

```
intervals(rec, e = 10^-8)
```

Arguments

rec	recoding definition as required by the recode function from the car package, additionally allowing for standard mathematical interval notation. An interval notation consists of two brackets containing the interval values seperated by a comma. Open and closed intervals may be defined, e.g. (1,2), [1,2], (1,2], [1,2). The tags lo and hi for the highest and lowest value in the dataset may also be used, e.g. [lo,4], [0,hi).
e	Deviation from given interval values when an open interval is used (i.e. excluding the given value). The default deviation is 10e-8. This means that e.g. the interval (1,2) is converted into the definition 1+10e-8:2-10e-8 to be used in the recode function.

Value

A string with recoding definitions for intervals as required by recode from car.

Author(s)

Mark Heckmann

Examples

```
## Not run:
 library(car)

 # the standard way if we want to recode [1,2) to the value 3
 recode(c(1, 1.999, 2, 2.001), "1:2-1e-4=3")

 # the same using interval notation
 intervals("[1,2)=3")
 recode(c(1, 1.999, 2, 2.001), intervals("[1,2)=3"))

 # another example: the car way
 e <- 10^-8
 recode(1:9/3.01, "lo:1-e=0; 1:2-e=1; 2:3-e=2")
 # using intervals
 recode(1:9/3.01, intervals("[lo,1)=0; [1,2)=1; [2,3)=2"))

## End(Not run)
```

---

nom.lambda *Calculate Lambda for nominal data tables.*

Description

Calculate Lambda for nominal data tables.

Usage

```
nom.lambda(x)
```

Arguments

x	A table object.

Value

A named list with the three values:

lambda.cr	The row variable is used as independent, the column variable as dependent variable.
lambda.rc	The column variable is used as independent, the row variable as dependent variable.
lambda.symmetric	Symmetric Lambda (the mean of both above).

Note

The code for the calculation was supplied by Marc Schwartz (under GPL 2). Checked against SPSS results.

Author(s)

Marc Schwartz, Mark Heckmann

Examples

```
{

}
```

---

nom.uncertainty *Calculate the Uncertainty Coefficient (Theil's U)*

Description

Calculate the Uncertainty Coefficient (Theil's U)

Usage

```
nom.uncertainty(x)
```

Arguments

x	A table object.

Value

A named list with the three values:

`ucc.cr`	The row variable is used as independent, the column variable as dependent variable.
`uc.rc`	The column variable is used as independent, the row variable as dependent variable.
`uc.symmetric`	Symmetric uncertainty coefficient.

Note

The code for the calculation was supplied by Marc Schwartz (under GPL 2). Note: Asymmetric formulae denomiators corrected on May 4, 2007 thanks to Antti Arppe. Checked against SPSS results.

Author(s)

Marc Schwartz, Mark Heckmann

Examples

```
{

}
```

`ord.gamma` *Calculate Goodman-Kruskal gamma for ordinal data tables.*

Description

Calculate Goodman-Kruskal gamma for ordinal data tables.

Usage

```
ord.gamma(x)
```

Arguments

`x`	A table object.

Value

The gamma value.

Note

The code for the calculation was supplied by Marc Schwartz (under GPL 2). Checked against SPSS results.

Author(s)

Marc Schwartz, Mark Heckmann

Examples

```
{
 # TODO
}
```

`ord.somers.d` *Calculate Somers' d for ordinal data tables.*

Description

Calculate Somers' d for ordinal data tables.

Usage

```
ord.somers.d(x)
```

Arguments

`x`	A table object.

Value

Kendall's Tau-b value.

A named list with the three values:

`sd.cr`	The row variable is used as independent, the column variable as dependent variable.
`sd.rc`	The column variable is used as independent, the row variable as dependent variable.
`sd.symmetric`	Symmetric Somers' d.

Note

The code for the calculation was supplied by Marc Schwartz (under GPL 2)

Author(s)

Marc Schwartz, Mark Heckmann

Examples

```
{
 # TODO
}
```

`ord.tau` *Calculate Kendall's Tau statistics for ordinal data tables (Tau-b and Tau-c).*

Description

Calculate Kendall's Tau statistics for ordinal data tables (Tau-b and Tau-c).

Usage

```
ord.tau(x)
```

Arguments

`x`	A table object.

Value

A named list with the three values:

`tau.a`	Tau-a satistic (for quadratic tables only)
`tau.b`	Tau-b satistic
`tau.c`	Kendall-Stuart Tau-c satistic

Note

The code for the calculation was supplied by Marc Schwartz (under GPL 2)

Author(s)

Marc Schwartz, Mark Heckmann

Examples

```
{
 # TODO
}
```

rowMeans2 *Form row means taking into account a minimum number of values required*

Description

In the construction of psychometric scales the calculation of a value is sometimes only desired if a minimum number of items contain values. In SPSS it is possible to calculate a mean value only if a minimum number of values are supplied by using the syntax MEAN.MIN with MIN being a numeric value. The function rowMeans2 does the same.

Usage

```
rowMeans2(x, w, min = 0, na.rm = TRUE)
```

Arguments

x	A matrix of dataframe whose columns should be averaged.
w	A numerical vector of weights the same length as number of columns in x.
min	The minimum number of values required to calculate the mean value. Otherwise return NA.
na.rm	A logical value indicating whether NA values in x should be stripped before the computation proceeds.

Details

rowMeans2 is very similary to rowMeans. The differences are that rowMeans2 allows to indicate the minimum number of values that have to be supplied and to weight the columns.

Value

A vector of means.

Author(s)

Mark Heckmann

See Also

rowMeans

Examples

```
x <- replicate(3, runif(5))
 x[1:3, 1] <- NA       # add NAs to data
 x[1:2, 2] <- NA
 x[1, 3] <- NA
 x
 rowMeans2(x)          # the same as rowMeans, except that NAs are allowed
 rowMeans2(x, min=2)   # minimum two values to calculate mean
 rowMeans2(x, min=3)   # minimum three values to calculate mean

 # returns numeric(0) if x has zero rows
 d <- x[NULL, ]
 rowMeans2(d)

 # weights for each column
 rowMeans2(x, w=c(1,1,2))
```

Index