Package 'Bullock'

July 18, 2014

Type Package

Title miscellaneous helper utilities

Version 1.14

Date 2014-05-25

Author John G. Bullock

Maintainer John G. Bullock <john.bullock@aya.yale.edu>

Imports gdata, stringr

Description functions that help me do miscellaneous tasks a little more more quickly. These range in complexity from a function that just removes NA values from a vector prior to summing it (sumNA) to a function that helps me to build LaTeX tables from regression output in the style that I like (latable).

License GPL (version 2 or later)

LazyLoad yes

R topics documented:

`alpha_cronbach` *Compute Cronbach's alpha for a battery of items.*

Description

This function is called by `reliability`. It generally should not be called by end users.

Usage

```
alpha_cronbach(S)
```

Arguments

`S`	Variance-covariance matrix of responses to a battery of measurements.

Author(s)

Joseph F. Lucke

`latable` *Print LaTeX table of regression results*

Description

Takes a list of regression models and returns a table of regression output formatted for LaTeX. There are two columns per regression: one for the coefficient estimates, another for standard errors.

Usage

```
latable(tables, substrings.to.remove = NULL, rows.to.remove=NULL, npmakebox = TRUE)
```

Arguments

`tables`	List of regression models. Supports models of class `glm`, `ivreg`, `lm`, `negbin`, `polr`, `vglm`, and `zeroinfl`.
`substrings.to.remove`	List of strings or regular expressions. If it is not a list, it will be coerced to a list with as.list(). Substrings in the row names that match any element in `substrings.to.remove` will be removed before the output is created.
`rows.to.remove`	Should be a list of strings or regular expressions. If it is not a list, it will be coerced to a list with as.list(). Rows that contain substrings matching any element in `rows.to.remove` will be removed from the output table before it is returned by the function. This is useful for creating "incomplete" regression tables that do not contain rows for some variables, e.g., control variables.
`npmakebox`	Improves formatting of the "Number of observations" row, mainly by ensuring that the Ns for each regression aren't decimal-aligned with the coefficient estimates. Requires the `numprint` package to be loaded in LaTeX.

Value

Returns a table of regression output formatted for LaTeX. The table is designed to be copied directly into LaTeX.

Note

The format of the tables produced by `latable` is inspired by "Estimates of relative survival rates, by cancer site," a table in Edward Tufte's essay on "The Cognitive Style of PowerPoint."

The current version works well for `lm` and `ivreg` models. It may be buggy when applied to models of other classes.

The current version produces buggy output if the name of the intercept row (typically "(Intercept)" or "Intercept" is modified by `substrings.to.remove` or `rows.to.remove`.

Author(s)

John G. Bullock

See Also

There are other packages that perform similar functions. See the `xtable` and `apsrtable` functions for alternatives.

`lNA` *Calculate length of vector after omitting NA values*

Description

Calculate length of vector after omitting NA values.

Usage

```
lNA(x)
```

Arguments

`x`

Author(s)

John G. Bullock

`lsos` *Improved version of* `ls`

Description

Pretty-printed version of `ls` that indicates the size of every object in an environment.

Usage

```
.ls.objects(pos = 1, pattern, order.by, decreasing = FALSE, head = FALSE, n=5)
lsos(..., n = 8)
```

Arguments

`pos`	position, on the search path, of the environment to search
`pattern`	regular expression. Only names matching `pattern` are returned.
`order.by`	object of `character` class. Valid arguments are `Type`, `Size`, `Rows`, and `Columns`. If argument is unspecified, information on objects will be returned in alphabetical order.
`decreasing`	logical value. Has no effect unless `order.by` is specified.
`head`	logical value. IF `TRUE`, information on only `n` objects will be returned.
`n`	number of objects for which to report information. Has no effect unless `head == TRUE`.
`...`	arguments that are passed to `.ls.objects`.

Details

`lsos` is a wrapper to `.ls.objects`. The main use of these functions is to see which objects are taking up the most memory.

Value

The returned object is a data frame.

Author(s)

Dirk Edelbuettel, JD Long

References

Function created by Dirk Edelbuettel and modified by JD Long. See http://stackoverflow.com/questions/1358003/ for details.

See Also

`ls`

meanNA *Calculate mean of vector after omitting NA values*

Description

Calculate mean of vector after omitting NA values.

Usage

```
meanNA(x)
```

Arguments

x

Author(s)

John G. Bullock

merge_fac *Merge factors*

Description

Fill in missing values in one factor with missing values from another.

Usage

```
merge_fac(fac.names, ...)
```

Arguments

fac.names	character vector of factor names
...	arguments passed to get()

Details

All factors should be of the same length. Missing values in the first factor named in fac.names are filled in with corresponding values from the second factor. Missing values in this merged factor are filled in with corresponding values from the third factor. And so on.

Value

Returned object is a factor.

Note

Merging factors in this way is trickier than just using a command like fac1[is.na(fac1)] <- fac2[is.na(fac1)] because fac1 and fac2 may have different factor levels. This commands takes care of the problem by merging the levels among different factors.

Author(s)

John G. Bullock

`modal_value` *Find modal value of a vector*

Description

Find modal value of a vector.

Usage

```
modal_value(x, na.rm=FALSE)
```

Arguments

`x`	a vector
`na.rm`	should NAs be removed from the vector before modal value is determined?

Author(s)

Unknown. Function copied from http://rwiki.sciviews.org/doku.php?id=tips:stats-basic:modalvalue.

`move.to.df` *Move a list of variables into a data frame.*

Description

Copy variables matching the pattern into a data frame, and perhaps delete the free-standing original variables.

Usage

```
move.to.df(pattern = NULL, move = TRUE)
```

Arguments

`pattern`	object of class `character`. Can specify a regular expression.
`move`	logical variable.

Details

IF `move == TRUE`, the variables in the environment will be deleted after they are moved into the data frame.

Value

Returned object is a data frame.

noNAmatrix *Remove rows with any NA from a matrix.*

Description

Performs listwise deletion on a matrix, removing all rows from which any data are missing.

Usage

```
noNAmatrix(x)
```

Arguments

x	matrix

Value

Returned value is a matrix.

Note

noNAmatrix is deprecrated. It remains in this package exclusively for backward compatibility. Those wishing to perform listwise deletion on a matrix should instead use na.omit(x).

See Also

na.omit

push *Perl-like stack utilities for R*

Description

Perl-like stack utilities for R: new_stack, push(), pop(), shift(), and unshift().

Usage

```
new_stack(value = NULL)
push(stack, value)
pop(stack)
shift(stack, value)
unshift(stack)
```

Arguments

stack	Object of class stack, created with new_stack.
value	For new_stack, the initial value of a stack object. For push and shift, something to be added to a stack object.

Value

new_stack returns an object of class stack. unshift and pop return the first and last values of stack, respectively.

Author(s)

Jeffrey A. Ryan, John G. Bullock

References

Adapted from Jeffrey A. Ryan's code at http://www.lemnica.com/esotericR/Introducing-Closures/.

See Also

See http://stackoverflow.com/questions/14488206 for related discussion, including a simpler implementation of push and pop by Matthew Plourde.

Examples

```
nb <- new_stack()
push(nb, 1:3)
nb$.Data     # [1] 1 2 3

pop(nb)      # from the back
unshift(nb)  # from the front
shift(nb, 3)
push(nb, 1)
nb$.Data     # [1] 3 2 1
```

---

qw *Perl-like qw() function for quoting a list of words*

Description

qw takes a string of words separated by spaces. It returns a vector in which each element is a word. The point of the function is to speed the creation of vectors of words.

Usage

```
qw(x)
```

Arguments

x	character string

Value

Character vector.

Author(s)

Florent Delmotte

References

Code taken from post by Florent Delmotte ("flodel") at http://stackoverflow.com/questions/520810/.

Examples

```
qw("You can type    text here
  with    linebreaks if you
  wish")
#  [1] "You"        "can"        "type"       "text"
#  [5] "here"       "with"       "linebreaks" "if"
#  [9] "you"        "wish"
```

---

reliability *Compute Cronbach's alpha for a battery of items.*

Description

Compute Cronbach's alpha for a battery of items, and show the reliability for all different batteries that might be created by removing one item from the original battery.

Usage

```
reliability (x, ...)
```

Arguments

x	Matrix of measurements, e.g., survey responses. Cannot have missing data.
...	Arguments to be passed to `alpha.cronbach()`. Currently serves no function.

Author(s)

Peter Ellis

---

rescale *Rescale a variable*

Description

Linear rescaling of numeric vectors. For example, a variable that ranges from 1 to 7 can be rescaled to range from 0 to 1.

Usage

```
rescale(x, newrange)
```

Arguments

x	numeric object
newrange	two-element numeric vector

Author(s)

Simon D. Jackman

Examples

```
vec <- 1:10
vecRescaled <- rescale(vec, c(2:5))
range(vecRescaled)  # 2 5
```

---

sdNA *Calculate standard deviation of vector after omitting NA values*

Description

Calculate standard deviation of vector after omitting NA values

Usage

```
sdNA(x, na.rm = TRUE)
```

Arguments

x	a numeric vector or an R object which is coercible to one by as.vector.
na.rm	logical. Should missing values be removed?

See Also

sd

---

split_fac *Create dummy variables for each level of a factor.*

Description

Create dummy variables for each level of a factor.

Usage

```
split_fac(
  fac,
  prefix = paste(deparse(substitute(NES.year.fac)), '.', sep = ''),
  env    = .GlobalEnv,
  ...)
```

Arguments

fac	factor variable
prefix	substring that begins the name of each created dummy variable
env	environment in which the dummy variables are created
...	arguments passed to assign()

Value

split_fac returns nothing. Instead, it creates, as a side effect, a set of logical variables – one for each level of fac.

Author(s)

John G. Bullock

Examples

```
fac <- factor(rep(1:3, each = 3))
split_fac(fac, prefix = 'fac')  # creates logical variables fac1, fac2, and fac3 in .GlobalEnv
```

---

sumNA *Calculate sum of vector after omitting NA values*

Description

Calculate sum of vector after omitting NA values.

Definition is `function(x) { return(sum(x, na.rm=TRUE)) }`.

Usage

```
sumNA(x)
```

Arguments

x	logical, integer, numeric, or complex vector

Value

The sum. If all elements of x are of type integer or logical, then the sum is an integer. Otherwise it is a length-one numeric or complex vector.

See Also

sum

table.sep *helper function for latable()*

Description

Interleaves columns between the columns of a table. Typically used to pretty-print tables.

Usage

```
table.sep(table, separator = "&", sig.digits = 2)
```

Arguments

table	object of class table
separator	object of class character
sig.digits	integer

varNA *Calculate variance of vector after omitting NA values*

Description

Calculate variance of vector after omitting NA values

Usage

```
varNA(x)
```

Arguments

x	numeric vector, matrix, or data frame

Details

The definition of varNA is function(x) {var(x, na.rm = TRUE)}.

See Also

var

`%IN%` *Value matching*

Description

`%IN%` returns a logical vector indicating whether there is a match for its left operand. It is like `%in%`, but it has one crucial difference: if there are `NA` values in the left operand, the corresponding values in the returned vector will also be `NA` (rather than `FALSE`, as with `%in%`.)

Usage

```
x %IN% table
```

Arguments

`x`	vector or `NULL`: the values to be matched.
`table`	vector or `NULL`: the values to be matched against.

Value

A logical vector of the same length as `x`. It indicates whether a match was found for each non-`NA` element of `x`. `NA` elements of `x` are matched by `NA` elements in the returned vector.

Note

The ordinary binary match operator, `%in%`, can be misleading because it seems more closely related to `==` than it is. The problem is that `==` will return `NA` in some (expected) cases, but `%in%` will never return `NA`. Instead, when using `%in%`, the returned vector will be `FALSE` for every `NA` value in the left operand.

Like `==`, `%IN%` will return `NA` when there are `NA` values in the left operand. See below for an example.

`%IN%` will always return `TRUE` values when `%in%` would do so, and vice versa. The two operators differ only in the sense that `%IN%` returns `FALSE` in some cases where `%in%` returns `NA`.

Author(s)

John G. Bullock

See Also

`%in%`

Examples

```
tmp <- c(1, 2, 3, NA)
tmp == 1      # TRUE FALSE FALSE NA
tmp %in% 1:2  # TRUE TRUE  FALSE FALSE
tmp %IN% 1:2  # TRUE TRUE  FALSE NA
```

Index