Package 'boris'

August 11, 2021

Title Boris' miscellanea

Description Miscellaneous functions of boris. It contains general utils, model, plot and scientific utils. Probably in future to be splitted into multiple packages.

Version 0.22

Maintainer Gianni Boris Pezzatti <boris.pezzatti@wsl.ch>

Author Gianni Boris Pezzatti

License GPL-3

Imports stringr, foreach, zoo, dismo, PRROC, dplyr, tibble, tidyr

Suggests testthat, roxygen2

RoxygenNote 7.1.1

NeedsCompilation no

R topics documented:

`accuracy_glm_cross` *Accuracy function to evaluate a gml model for training and test dataset*

Description

Evaluate according to multiple criteria a gml model for training and test dataset

Usage

```
accuracy_glm_cross(m, abundance, test, depvar_name, abundance_name)
```

Arguments

`m`	glm model
`abundance`	abundances, should be passed in same order as presences and same length
`test`	dataframe holding test data
`depvar_name`	name of te dependent variable in the test dataframe
`abundance_name`	name of te abundance variable in the test dataframe

Value

a vector of named arguments (n=number data, np=numer of presences,)

`accuracy_glm_simple` *Accuracy function to evaluate a glm model*

Description

Evaluate according to multiple criteria a glm model

Usage

```
accuracy_glm_simple(m, p, a, abundance = NULL)
```

Arguments

`m`	glm model
`p`	presences
`a`	absences
`abundance`	abundances, should be passed in same order as presences and same length

Value

a vector of named arguments (n=number data, np=numer of presences,)

`accuracy_me_cross` *Accuracy function to evaluate a maxent model for training and test dataset*

Description

Evaluate according to multiple criteria a maxent model (`maxent`) for training and test dataset

Usage

```
accuracy_me_cross(me, abundance, test = NULL, depvar_name, abundance_name)
```

Arguments

`me`	maxent model
`abundance`	abundances, should be passed in same order as presences and same length
`test`	dataframe holding test data
`depvar_name`	name of te dependent variable in the test dataframe
`abundance_name`	name of te abundance variable in the test dataframe

Value

a vector of named arguments (n=number data, np=numer of presences,)

---

accuracy_me_simple *Accuracy function to evaluate a maxent model*

Description

Evaluate according to multiple criteria a maxent model (maxent)

Usage

```
accuracy_me_simple(me, p, a, abundance = NULL)
```

Arguments

me	maxent model
p	presences
a	absences
abundance	abundances, should be passed in same order as presences and same length

Value

a vector of named arguments (n=number data, np=numer of presences, auc=usual auc, auc.bg=auc on background, auc.me=auc for maxent, aicc.me=aicc for maxent)

---

accuracy_simple *Accuracy function to evaluate a presence/absence/background model*

Description

Evaluate according to multiple criteria

Usage

```
accuracy_simple(p, a, abundance = NULL)
```

Arguments

p	presences
a	absences
abundance	abundances, should be passed in same order as presences and same length

Value

a vector of named arguments (n=number data, np=numer of presences, auc=auc, auc.bg=auc on the background)

bkr *Background proportion*

Description

Background proportion

Usage

```
bkr(d, thr, depvar)
```

Arguments

d	dataframe
thr	threshold value
depvar	column holding the model output

Value

background proportion

bkr_for_tpr *Background portion for a given true positive rate*

Description

Background portion for a given true positive rate

Usage

```
bkr_for_tpr(d, tp.rate, depvar, occurrence)
```

Arguments

d	dataframe
tp.rate	true positive rate
depvar	column holding the model output
occurrence	column holding presence [0/1]

Value

background proportion

`boris` *boris*

`calcArea` *Area under a curve*

Description

Calculates the trapezoid area (boxes+traingles) under the curve y=f(x)

Usage

```
calcArea(x, y)
```

Arguments

`x`	vector holding the x values
`y`	vector holding the y values

Value

area under the curve

See Also

`calcAreaLim`

`calcAreaLim` *Area under a curve*

Description

Calculates the trapezoid area (boxes+traingles) under the curve y=f(x) up to a given x limit (xupper), when given

Usage

```
calcAreaLim(x, y, xupper = NULL)
```

Arguments

`x`	vector holding the x values
`y`	vector holding the y values
`xupper`	x value

Details

If xupper is not one of the x values, the corresponding y value is calculated using the approx function

Value

area under the curve

See Also

calcArea

---

cor2df *Variable pairs correlated above a threshold*

Description

This function returns a dataframe with the variable pairs above a given correlation threshold

Usage

```
cor2df(cor.matrix, threshold = 0.6)
```

Arguments

cor.matrix	correlation matrix
threshold	correlation threshold

Details

It is based on the cor function, but instead of a correlation matrix it returns a dataframe with the pairwise combinations above a threshold.

Value

a dataframe holding the variable pairs with a correlation higher than the specified threshold

See Also

cordf

`cor2df_fire` *Correlations above a threshold, showing aicc's of a logistic according to fire presence*

Description

This function returns a dataframe with the variable pairs above a given correlation threshold, and the aic value of a logistic model with fire occurrence

Usage

```
cor2df_fire(
  data,
  vars = NULL,
  fire,
  threshold,
  use = "everything",
  method = c("pearson", "kendall", "spearman")
)
```

Arguments

`data`	dataframe with the data
`vars`	vector of column names or column numbers holding the variables to analyse. If not specified all the columns will be used.
`fire`	column name of number holding fire presence [0/1]
`threshold`	correlation threshold
`use`	an optional character string giving a method for computing covariances in the presence of missing values. This must be (an abbreviation of) one of the strings "everything", "all.obs", "complete.obs", "na.or.complete", or "pairwise.complete.obs"
`method`	a character string indicating which correlation coefficient is to be computed. One of "pearson" (default), "kendall", or "spearman", can be abbreviated

Details

It is based on the `cor` function, but instead of a correlation matrix it returns a dataframe with the pairwise combinations above a threshold.

Value

a list with

- cors: dataframe holding the variable pairs with a correlation higher than the specified threshold, and the relative aicc's'
- aiccs: dataframe holding allthe aicc's' for all variables

See Also

cordf

`cordf` *Correlations above a threshold*

Description

This function returns a dataframe with the variable pairs above a given correlation threshold

Usage

```
cordf(
  data,
  vars = NULL,
  threshold = 0.6,
  use = "everything",
  method = c("pearson", "kendall", "spearman")
)
```

Arguments

`data`	dataframe with the data
`vars`	vector of column names or column numbers holding the variables to analyse. If not specified all the columns will be used.
`threshold`	correlation threshold
`use`	an optional character string giving a method for computing covariances in the presence of missing values. This must be (an abbreviation of) one of the strings "everything", "all.obs", "complete.obs", "na.or.complete", or "pairwise.complete.obs"
`method`	a character string indicating which correlation coefficient is to be computed. One of "pearson" (default), "kendall", or "spearman", can be abbreviated

Details

It is based on the `cor` function, but instead of a correlation matrix it returns a dataframe with the pairwise combinations above a threshold.

Value

a dataframe holding the variable pairs with a correlation higher than the specified threshold

See Also

cor2df

dep_vars *Dependent variable*

Description

Extract the name of the dependent variable from formula

Usage

```
dep_vars(formula)
```

Arguments

formula	formula to inspect, either as formula object or string

Value

name of the dependent variable

See Also

all.vars from base package to get all variables

evaltext *Concatenate and evaluate string expressions in a specified environment*

Description

This function allows to write in a shorter form the evaluation of a vector of characters.

Usage

```
evaltext(
  ...,
  envir = parent.frame(),
  enclos = if (is.list(envir) || is.pairlist(envir)) parent.frame() else baseenv(),
  sep = ""
)
```

Arguments

...	character strings holding the code to be evaluated
envir	the environment in which expr is to be evaluated. May also be NULL, a list, a data frame, a pairlist or an integer as specified to sys.call.
enclos	Relevant when envir is a (pair)list or a data frame. Specifies the enclosure, i.e., where R looks for objects not found in envir. This can be NULL (interpreted as the base package environment, baseenv()) or an environment.
sep	separator character to be used as in the paste function

Value

The result of evaluating the object: for an expression vector this is the result of evaluating the last element

---

filename *Filename without extension*

Description

Strips the extension form the filename.

Usage

```
filename(file)
```

Arguments

file	name of the file

Value

file name without extension

---

fill_1_na *Fill 1-value gaps in a vector*

Description

Fill gaps of single values with linearization (mean of the adjacent values) or repetition of previous/next value.

Usage

```
fill_1_na(x, method = c("linearize", "previous", "next"))
```

Arguments

x	numeric vector
method	how to fill in the gaps (default by linearization, otherwise by previous/next value duplication)

Value

numeric vector with filled 1-value gaps

---

fill_na *Fill gaps in a vector*

Description

Fill gaps with linearization (mean of the adjacent values) or repetition of previous/next value, by using the na.approx function.

Usage

```
fill_na(x, method = c("linearize", "previous", "next"), maxgap = 2)
```

Arguments

x	numeric vector
method	how to fill in the gaps (default by linearization, otherwise by previous/next value duplication)
maxgap	maximum number of consecutive NAs to fill. Any longer gaps will be left unchanged.

Value

numeric vector with filled 1-value gaps

---

formulae *Formulae from variable combinations*

Description

Build the formulae (as strings) from variable names

Usage

```
formulae(
  formula,
  dep = NULL,
  vars = NULL,
  nullmodelterm = "1",
  minsize = 1,
  maxsize = NULL
)
```

Arguments

`formula`	formula with all the terms (beyond optimal model)
`dep`	name of the dependent variable. If the `formula` is specified, this argument is not considered.
`vars`	character vector with the names of the independent variables (wihtout nullmodel term). If the `formula` is specified, this argument is not considered.
`nullmodelterm`	to specify in case of an always required fixed term (should not be included in the vars)
`minsize`	minimum size of the formula (number of independent variables)
`maxsize`	maximum size of the formula (number of independent variables). NULL means unrestricted.

Value

character vector hold the strings of the generated formulae.

`formulae_cleaned` *Formulae from variable combinations without correlated variables*

Description

Build the formulae (as strings) from variable names

Usage

```
formulae_cleaned(
  formula,
  dep = NULL,
  vars = NULL,
  nullmodelterm = "1",
  minsize = 1,
  maxsize = NULL,
  data,
```

```
  threshold = 0.6,
  use = "everything",
  method = c("pearson", "kendall", "spearman")
)
```

Arguments

formula	formula with all the terms (beyond optimal model)
dep	name of the dependent variable. If the formula is specified, this argument is not considered.
vars	character vector with the names of the independent variables (wihtout nullmodel term). If the formula is specified, this argument is not considered.
nullmodelterm	to specify in case of an always required fixed term (should not be included in the vars)
minsize	minimum size of the formula (number of independent variables)
maxsize	maximum size of the formula (number of independent variables). NULL means unrestricted.
data	dataframe holding the dataset with the column names corresponding to vars
threshold	correlation threshold
use	an optional character string giving a method for computing covariances in the presence of missing values. This must be (an abbreviation of) one of the strings "everything", "all.obs", "complete.obs", "na.or.complete", or "pairwise.complete.obs"
method	a character string indicating which correlation coefficient is to be computed. One of "pearson" (default), "kendall", or "spearman", can be abbreviated

Value

list of

- formulae: character vector hold the strings of the generated formulae
- vars: list of variables names combinations

---

fpr *False positive rate (1- specificity)*

Description

False positive rate (1- specificity)

Usage

```
fpr(d, thr, depvar, occurrence)
```

Arguments

d	dataframe
thr	threshold value
depvar	column holding the model output
occurrence	column holding presence [0/1]

Value

false positive rate (1- specificity)

---

fpr_for_tpr *False positive rate for a given true positive rate*

Description

False positive rate for a given true positive rate

Usage

```
fpr_for_tpr(d, tp.rate, depvar, occurrence)
```

Arguments

d	dataframe
tp.rate	true positive rate
depvar	column holding the model output
occurrence	column holding presence [0/1]

Value

false positive rate

---

getArgs *Extract and Load command line arguments into session*

Description

Loads the command line arguments supplied when this R session was invoked into the session environment.

Usage

```
getArgs()
```

Value

Nothing

glm_pseudoabsence *GLM with pseudoabsences*

Description

Wrapper around `glm` to use a pseudoabsence approach, substituting absences (zeros's) with the prevalence (mean occurrence).

Usage

```
glm_pseudoabsence(formula, family = gaussian, data, ...)
```

Arguments

`formula`	formula with dependent and independent variables
`family`	a description of the error distribution and link function to be used in the model. This can be a character string naming a family function, a family function or the result of a call to a family function. (See `family` for details of family functions.)
`data`	dataframe holding the data
`...`	additional parameters to pass to glm

Value

glm model

See Also

`glm`

ind_vars *Independent variable(s)*

Description

Extract the name of the independent variable(s) from formula

Usage

```
ind_vars(formula, simplify = F)
```

Arguments

`formula`	formula to inspect, either as formula object or string
`simplify`	if terms such as I(x^2) should be retained as variable or not

Value

name of the independent variable(s)

See Also

`all.vars` from base package to get all variables

---

kfold_seq *Sequential k-fold partitioning*

Description

Modified version of the `kfold` function, that returns subsequent (not random) folds (consistent among runs)

Usage

```
kfold_seq(x, k = 5, by = NULL)
```

Arguments

`x`	a vector, matrix, data.frame, or Spatial object
`k`	number of groups
`by`	Optional argument. A vector or factor with sub-groups (e.g. species). Its length should be the same as the number of records in x

Value

a vector with group assignments

See Also

`kfold`

maxent_formula *Maxent with formula*

Description

Wrapper for Maxent to use a formula instead of data and presences

Usage

```
maxent_formula(formula, data, ...)
```

Arguments

formula	formula with dependent and independent variables
data	dataframe holding the data
...	additional parameters to pass to maxent

Value

Maxent model

See Also

maxent

me_constants *Constants of a maxent model*

Description

Extract the constants of a maxent model (maxent)

Usage

```
me_constants(m)
```

Arguments

m	either a maxent model or a character vector containing the lines of a maxent lambda file

Value

dataframe holding the constants (what, value)

me_lambdas *Extract lambda file values*

Description

Fill a data frame with the lambda file values of a maxent model (maxent)

Usage

```
me_lambdas(m)
```

Arguments

m	either a maxent model or a character vector containing the lines of a maxent lambda file

Value

dataframe with the lambda values (what, lambda, min, max)

me_parNum *Count the number of Maxent parameters (with lambda!=0)*

Description

Counts the number of parameters (with lambda!=0) of a maxent model (maxent)

Usage

```
me_parNum(m)
```

Arguments

m	maxent model

Value

number of parameters

me_predict *Predict new values of a maxent model*

Description

Calculates new values of a maxent model (maxent)

Usage

```
me_predict(m, data)
```

Arguments

m	either a maxent model or a character vector containing the lines of a maxent lambda file
data	data frame holding the data

Value

predictions

mirror_na *Fill gaps in a dataframe with data from another dataframe*

Description

Function replacing NA values in a dataframe with sequentially corresponding data from another dataframe of the same length and with same column names.

Usage

```
mirror_na(to, from, colnames, case.sensitive = T)
```

Arguments

to	dataframe holding the NA values to replace
from	dataframe holding the values to replace the NA's
colnames	character vector with the names of the columns
case.sensitive	logcial indicating if column names are considered according to case or not

Value

dataframe with replaces NA's

models *Models from formulae*

Description

Build a list of models according to different formulae, a prefix and a suffix

Usage

```
models(prefix, formulae, suffix, envir = parent.frame(1))
```

Arguments

prefix	string representation of the model prefix (e.g. "glm(")
formulae	list of formulae as string expressions
suffix	string representation of the model suffix (e.g. ", family="binomial")")
envir	environment in which to evaluate the model expression. By default evaluates in the environment that calls this function

Value

a list of models

orderfactor *Change levels order*

Description

This function changes the oder of the levels of a factor

Usage

```
orderfactor(x, neworder, ordered = is.ordered(x), ...)
```

Arguments

x	factor
neworder	numeric or character vector specifiyng the new order of the levels
ordered	logical specifying if the factor will be ordered or not (defaults to input factor class)
...	other parameters to be passed to factor function (labels, exclude)

Value

factor with levels order changed according to specifications

`read_fwf_fixedheader` *Reads a fixed width formatted data with the header in the same format*

Description

The base function `read.fwf` can read fixed width formatted data, however when including an header, this needs to have another format (e.g. tab-separated, as specified by the sep argument). This function allows to read data with the header specifically in the same fixed width format as the data.

Usage

```
read_fwf_fixedheader(file, widths, ...)
```

Arguments

`file`	name of the file.
`widths`	integer vector, giving the widths of the fixed-width fields (of one line).
`...`	further arguments to be passed to `read.fwf`.

Value

A data.frame as produced by `read.fwf` which is called internally.

`replace_na` *Replace NA's*

Description

Substitutes NA values in the given vector, dataframe, matrix or list

Usage

```
replace_na(data, value)
```

Arguments

`data`	vector, dataframe, list or matrix
`value`	replacement value

Value

a data structure with the given value instead of NA's

`resample_meteo_h2d` *Resample a data.frame with meteorological data with hourly interval to a daily interval, allowing the specification at which time to cut the day (e.g. can be summarized foro noon to noon).*

Description

Resample a data.frame with meteorological data with hourly interval to a daily interval, allowing the specification at which time to cut the day (e.g. can be summarized foro noon to noon).

Usage

```
resample_meteo_h2d(
  h,
  time_h = 24,
  timevar,
  varnames,
  aggregation = c("sample", "sum", "mean", "max", "min"),
  na.rm = F,
  add_suffix = F
)
```

Arguments

`h`	a data.frame holding the hourly data
`time_h`	hour value at which to cut the hourly data.frame to build summaries (e.g. 12 for noon to noon). Defaults to 24.
`timevar`	name of the column holding the date-time information (in POSIX numeric format).
`varnames`	character vector holding the names of the columns holding the variables to be processed.
`aggregation`	character vector holding the types of aggregations to perform on the selected variables.
`na.rm`	boolean to specify if aggregation function should consider or skip NA's.
`add_suffix`	boolean to specify if the vairable names sohould be completed with the specification of the aggregation.

Value

a new data.frame with daily timestep and the selected variables and aggregations .

See Also

`kfold`

rescale01 *Rescale a vector of numbers between 0 and 1*

Description

This function rescales the values of a numeric vector between 0 and 1

Usage

```
rescale01(x, na.rm = FALSE)
```

Arguments

x	numeric vector to rescale
na.rm	logical indicating whether missing values should be removed

Value

numeric vector with rescaled values

tpr *True positive rate (sensitivity)*

Description

True positive rate (sensitivity)

Usage

```
tpr(d, thr, depvar, occurrence = presence)
```

Arguments

d	dataframe
thr	threshold value
depvar	column holding the model output
occurrence	column holding presence [0/1]

Value

true positive rate (sensitivity)

vpd *Vapour pressure deficit*

Description

Calculate vapour pressure deficit from temperature and humidity

Usage

```
vpd(T, H)
```

Arguments

T	numeric vector with air temperature values [C]
H	numeric vector with air humidity values [%]

Value

vapur pressure deficit

without_na *Remove NA's*

Description

Return the given dataframe without the rows where one of the independent variables (extracted from formula) are NA

Usage

```
without_na(data, selection)
```

Arguments

data	dataframe with the data
selection	column names or formula with dependent and independent variables

Details

This is useful to link abundances to the model datasets, since the built models internally exclude those rows

Value

A dataframe without NA's in the columns holded by independent variables of the formula

`yearplots` *Multiple plots of daily data over years*

Description

Plots summaries of daily data in a yearly plot

Usage

```
yearplots(
  data,
  vars,
  year = year,
  doy = doy,
  what = c("data", "mean", "na"),
  rows = floor(length(vars)/cols + 1),
  cols = 4
)
```

Arguments

`data`	dataframe with data
`vars`	character vector with variable (columns) names holding the data to summarise
`year`	name of the column holding the year
`doy`	name of the column holding the doy [1-366]
`what`	what to plot: all values, means over the years or presence of NA's
`rows`	number of rows for the multiple plots. Defaults accordind to the number of
`cols`	number of columns for the multiple plots. Defaults to 4

Value

plot

Index