Package 'modEvA'

December 13, 2021

Type Package

Title Model Evaluation and Analysis

Version 3.0

Date 2021-12-13

Author Barbosa A.M., Brown J.A., Jimenez-Valverde A., Real R.

Maintainer A. Marcia Barbosa <ana.marcia.barbosa@gmail.com>

Imports graphics, grDevices, stats, methods

Description Analyses species distribution models and evaluates their performance. It includes functions for performing variation partitioning, calculating several measures of model discrimination and calibration, optimizing prediction thresholds based on a number of criteria, performing multivariate environmental similarity surface (MESS) analysis, and displaying various analytical plots.

LazyLoad yes

LazyData yes

License GPL-3

URL http://modeva.r-forge.r-project.org/

R topics documented:

modEvA-package 2
arrangePlots 3
AUC 5
confusionLabel 8
Dsquared 10
evaluate 11
evenness 13
getBins 14
getModEqn 16
HLfit 18
MESS 21
MillerCalib 24
mod2obspred 26

modEvAmethods 27
multModEv 28
OA 30
optiPair 31
optiThresh 34
plotGLM 36
predDensity 38
predPlot 39
prevalence 41
range01 43
rotif.mods 44
RsqGLM 45
standard01 46
threshMeasures 48
varPart 51

Index **55**

`modEvA-package` *Model Evaluation and Analysis*

Description

The modEvA package can analyse species distribution models and evaluate their performance. It includes functions for performing variation partitioning; calculating several measures of model discrimination, classification, explanatory power, and calibration; optimizing prediction thresholds based on a number of criteria; performing multivariate environmental similarity surface (MESS) analysis; and displaying various analytical plots.

Details

Package:	modEvA
Type:	Package
Version:	3.0
Date:	2021-12-13
License:	GPL-3

Author(s)

Barbosa A.M., Brown J.A., Jimenez-Valverde A., Real R.

A. Marcia Barbosa <ana.marcia.barbosa@gmail.com>

References

Barbosa A.M., Real R., Munoz A.R. & Brown J.A. (2013) New measures for assessing model equilibrium and prediction mismatch in species distribution models. Diversity and Distributions 19: 1333-1338 (DOI: 10.1111/ddi.12100)

See Also

PresenceAbsence, **ROCR**, **verification**

Examples

```
# load sample models:
data(rotif.mods)

# choose a particular model to play with:
mod <- rotif.mods$models[[1]]

# plot this model:
plotGLM(model = mod)

# calculate the area under the ROC curve for the model:
AUC(model = mod)

# calculate some threshold-based measures for this model:
threshMeasures(model = mod, thresh = 0.5)
threshMeasures(model = mod, thresh = "preval")

# calculate optimal thresholds based on several criteria:
optiThresh(model = mod, measures = c("CCR", "Sensitivity", "kappa", "TSS"),
ylim = c(0, 1))

# calculate the optimal threshold balancing two evaluation measures:
optiPair(model = mod, measures = c("Sensitivity", "Specificity"))

# calculate the explained deviance, Hosmer-Lemeshow goodness-of-fit,
# Miller's calibration stats, and (pseudo) R-squared values for the model:
Dsquared(model = mod)
HLfit(model = mod, bin.method = "quantiles")
MillerCalib(model = mod)
RsqGLM(model = mod)

# calculate a bunch of evaluation measures for a set of models:
multModEv(models = rotif.mods$models[1:4], thresh = "preval",
bin.method = "quantiles")
```

---

arrangePlots *Arrange plots*

Description

Get an appropriate row/column combination (for `par(mfrow)`) for arranging a given number of plots within a plotting window.

Usage

```
arrangePlots(n.plots, landscape = FALSE)
```

Arguments

`n.plots`	number of plots to be placed in the graphics device.
`landscape`	logical, whether the plotting window should be landscape/horizontal (number of columns larger than the number of rows) or not. The value does not make a difference if the number of plots makes for a square plotting window.

Details

This function is used internally by `optiThresh`, but can also be useful outside it.

Value

An integer vector of the form c(nr, nc) indicating, respectively, the number of rows and of columns of plots to set in the graphics device.

Author(s)

A. Marcia Barbosa

See Also

`plot`, `layout`

Examples

```
arrangePlots(10)

arrangePlots(10, landscape = TRUE)


# a more practical example:

data(iris)

names(iris)

# say you want to plot all columns in a nicely arranged plotting window:

par(mfrow = arrangePlots(ncol(iris)))

for (i in 1:ncol(iris)) {
  plot(1:nrow(iris), iris[, i])
```

```
}
```

AUC *Area Under the Curve*

Description

This function calculates the Area Under the Curve of the receiver operating characteristic (ROC) plot, or alternatively the precision-recall (PR) plot, for either a model object or two matching vectors of observed binary (1 for occurrence vs. 0 for non-occurrence) and predicted (continuous, e.g. occurrence probability) values, respectively.

Usage

```
AUC(model = NULL, obs = NULL, pred = NULL, simplif = FALSE,
interval = 0.01, FPR.limits = c(0, 1), curve = "ROC",
method = "rank", plot = TRUE, diag = TRUE, diag.col = "grey",
diag.lty = 1, curve.col = "black", curve.lty = 1, curve.lwd = 2,
plot.values = TRUE, plot.digits = 3, plot.preds = FALSE,
grid = FALSE, xlab = "auto", ylab = "auto", ticks = FALSE, ...)
```

Arguments

model	a model object of class "glm", "gam", "gbm", "randomForest" or "bart". Alternatively, you can use the 'obs' and 'pred' arguments instead of 'model'.
obs	alternatively to 'model', a vector of observed presences (1) and absences (0) or another binary response variable. This argument is ignored if 'model' is provided.
pred	alternatively to 'model' and together with 'obs', a vector with the corresponding predicted values of presence probability, habitat suitability, environmental favourability or alike. Must be of the same length and in the same order as 'obs'. This argument is ignored if 'model' is provided.
simplif	logical, whether to use a faster version that returns only the AUC value (and the plot if 'plot = TRUE').
FPR.limits	(NOT YET IMPLEMENTED) numerical vector of length 2 indicating the limits of false positive rate between which to calculate a partial AUC. The default is c(0, 1), for considering the whole AUC.
curve	character indicating whether to compute the "ROC" (receiver operating charateristic) or the "PR" (precision-recall) curve.
interval	interval of threshold values at which to calculate the true and false positive rates. Defaults to 0.01 for relatively quick while still relatively accurate computation. Note that, if method = "rank" (the default if curve = "ROC"), this does not affect the obtained AUC value (although it can affect the size of the plotted curve, especially when prevalence is low), as the AUC is calculated with the Mann-Whitney-Wilcoxon statistic and is therefore threshold-independent. If method

	!= "rank" (or, by extension, if curve = "PR" – see 'method' argument), setting 'interval' to smaller values will provide more accurate AUC values. The size of the 'interval' also affects the resulting 'meanPrecision', as this is averaged across all threshold values.
`method`	character indicating with which method to calculate the AUC value. Available options are "rank" (the default and most accurate, but implemented only if curve = "ROC") and "trapezoid" (the default if curve = "PR"). The latter is computed more accurately if 'interval' is decreased (see 'interval' argument).
`plot`	logical, whether or not to plot the curve. Defaults to TRUE.
`diag`	logical, whether or not to add the reference diagonal (if plot = TRUE). Defaults to TRUE.
`diag.col`	line colour for the reference diagonal (if diag = TRUE).
`diag.lty`	line type for the reference diagonal (if diag = TRUE).
`curve.col`	line colour for the curve.
`curve.lty`	line type for the curve.
`curve.lwd`	line width for the curve.
`plot.values`	logical, whether or not to show in the plot the values associated to the curve (e.g., the AUC). Defaults to TRUE.
`plot.digits`	integer number indicating the number of digits to which the values in the plot should be `round`ed. Defaults to 3. This argument is ignored if 'plot' or 'plot.values' are set to FALSE.
`plot.preds`	logical value indicating whether the proportions of 'pred' values for each threshold should be plotted as proportionally sized blue circles. Can also be provided as a character vector specifying if the circles should be plotted on the "curve" (the default) and/or at the "bottom" of the plot. The default is FALSE for no circles, but it may be interesting to try it, especially if your curve has long straight lines or does not cover the full length of the plot.
`grid`	logical, whether or not to add a grid to the plot, marking the analysed thresholds. Defaults to FALSE.
`xlab`	label for the x axis. By default, a label is automatically generated according to the specified 'curve'.
`ylab`	label for the y axis. By default, a label is automatically generated according to the specified 'curve'.
`ticks`	logical, whether or not to add blue tick marks at the bottom of the plot to mark the thresholds at which there were values from which to to draw the curve. Defaults to FALSE.
`...`	further arguments to be passed to the `plot` function.

Details

In the case of the "ROC" curve (the default), the AUC is a measure of the overall discrimination power of the predictions, or the probability that an occurrence site has a higher predicted value than a non-occurrence site. It can thus be calculated with the Wilcoxon rank sum statistic, as is done with the default method="rank". There's also an option to compute, instead of the ROC curve, the

precision-recall ("PR") curve, which is more robust to imbalanced data, e.g. species rarity (Sofaer et al. 2019), as it doesn't value true negatives.

If 'curve' is set to "PR", or if 'method' is manually set to "trapezoid", the AUC value will be more accurate if 'interval' is decreased (see 'method' and 'interval' arguments above). The plotted curve will also be more accurate with smaller 'interval' values, especially for imbalanced datasets (which can cause an apparent disagreement between the look of the curve and the actual value of the AUC).

Mind that the AUC has been widely criticized (e.g. Lobo et al. 2008, Jimenez-Valverde et al. 2013), but is still among the most widely used metrics in model evaluation. It is highly correlated with species prevalence (as are all model discrimination and classification metrics), so prevalence is also output by the AUC function (if `simplif = FALSE`, the default) for reference.

Although there are functions to calculate the AUC in other R packages (e.g. **ROCR**, **PresenceAbsence**, **verification**, **Epi**, **PRROC**, **PerfMeas**, **precrec**), the `AUC` function is more compatible with the remaining functions in **modEvA**, and it can be applied not only to a set of observed vs. predicted values, but also directly to a model object of class "glm", "gam", "gbm", "randomForest" or "bart".

Value

If `simplif = TRUE`, the function returns only the AUC value (a numeric value between 0 and 1). Otherwise (the default), it returns a `list` with the following components:

`thresholds`	a data frame of the true and false positives, the sensitivity, specificity and recall of the predictions, and the number of predicted values at each analysed threshold.
`N`	the total number of obervations.
`prevalence`	the proportion of presences (i.e., ones) in the data (which correlates with the AUC of the "ROC" plot).
`AUC`	the value of the AUC).
`AUCratio`	the ratio of the obtained AUC value to the null expectation (0.5).
`meanPrecision`	the arithmetic mean of precision (proportion of predicted presences actually observed as presences) across all threshold values (defined by 'interval'). It is close to the AUC of the precision-recall (PR) curve.

Author(s)

A. Marcia Barbosa

References

Lobo, J.M., Jimenez-Valverde, A. & Real, R. (2008). AUC: a misleading measure of the performance of predictive distribution models. Global Ecology and Biogeography 17: 145-151

Jimenez-Valverde, A., Acevedo, P., Barbosa, A.M., Lobo, J.M. & Real, R. (2013). Discrimination capacity in species distribution models depends on the representativeness of the environmental domain. Global Ecology and Biogeography 22: 508-516

Sofaer, H.R., Hoeting, J.A. & Jarnevich, C.S. (2019). The area under the precision-recall curve as a performance metric for rare binary events. Methods in Ecology and Evolution, 10: 565-577

See Also

threshMeasures

Examples

```
# load sample models:
data(rotif.mods)

# choose a particular model to play with:
mod <- rotif.mods$models[[1]]


AUC(model = mod, simplif = TRUE)

AUC(model = mod, curve = "PR")

AUC(model = mod, interval = 0.1, grid = TRUE)

AUC(model = mod, plot.preds = TRUE)

AUC(model = mod, ticks = TRUE)

AUC(model = mod, plot.preds = c("curve", "bottom"))

# you can also use AUC with vectors of observed and predicted values
# instead of with a model object:

presabs <- mod$y
prediction <- mod$fitted.values

AUC(obs = presabs, pred = prediction)

AUC(obs = presabs, pred = prediction, plot.preds = TRUE)
```

---

confusionLabel *Label predictions according to their confusion matrix category*

Description

This function labels the predictions of a binary response model according to their confusion matrix categories, i.e., it classifies each prediction into a false positive, false negative, true positive or true negative, given a user-defined threshold value.

Usage

```
confusionLabel(model = NULL, obs = NULL, pred = NULL, thresh, verbosity = 2)
```

Arguments

model	a model object of class "glm", "gam", "gbm", "randomForest" or "bart".
obs	a vector of observed presences (1) and absences (0) or another binary response variable. Not necessary (and ignored) if 'model' is provided.
pred	a vector with the corresponding predicted values of presence probability, habitat suitability, environmental favourability or alike. Not necessary (and ignored) if 'model' is provided.
thresh	numeric value of the threshold to separate predicted presences from predicted absences; can be "preval", to use the prevalence of 'obs' (or of the response variable in 'model') as the threshold, or any real number between 0 and 1. See Details in threshMeasures for an informed choice.
verbosity	integer specifying the amount of messages to display. Defaults to the maximum implemented; lower numbers (down to 0) decrease the number of messages.

Value

This function returns a character vector of the same length as 'obs' and 'pred', or of the same number of rows as the data in 'model', containing the confusion matrix label for each value.

Author(s)

A. Marcia Barbosa

See Also

threshMeasures

Examples

```
# load sample models:
data(rotif.mods)

# choose a particular model to play with:
mod <- rotif.mods$models[[1]]

confusionLabel(model = mod, thresh = 0.5)


# you can also use confusionLabel with vectors of observed and predicted values
# instead of with a model object:

presabs <- mod$y
prediction <- mod$fitted.values

confusionLabel(obs = presabs, pred = prediction, thresh = 0.5)
```

Dsquared *Proportion of deviance explained by a GLM*

Description

This function calculates the (adjusted) amount of deviance accounted for by a generalized linear model.

Usage

```
Dsquared(model = NULL, obs = NULL, pred = NULL, family = NULL,
adjust = FALSE, npar = NULL)
```

Arguments

model	a model object of class "glm".
obs	a numeric vector of the observed data. This argument is ignored if model is provided.
pred	a numeric vector of the values predicted by a GLM of the observed data. This argument is ignored if model is provided. Must be of the same length and in the same order as obs.
family	a character vector (i.e. in quotes) of length 1 specifying the family of the GLM. This argument is ignored if model is provided; otherwise (i.e. if 'obs' and 'pred' are provided rather than a model object), only families 'binomial' (logit link) and 'poisson' (log link) are currently implemented.
adjust	logical, whether or not to adjust the D-squared value for the number of observations and parameters in the model (see Details). The default is FALSE; TRUE requires either providing the model object, or specifying the number of parameters in the model that produced the pred values.
npar	an integer vector indicating the number of parameters in the model. This argument is ignored if model is provided or if adjust = FALSE.

Details

Linear models come with an R-squared value that measures the proportion of variation that the model accounts for. The R-squared is provided with summary(model) in R. For generalized linear models (GLMs), the equivalent is the amount of deviance accounted for (D-squared; Guisan & Zimmermann 2000), but this value is not normally provided with the model summary. The Dsquared function calculates it. There is also an option to calculate the adjusted D-squared, which takes into account the number of observations and the number of predictors, thus allowing direct comparison among different models (Weisberg 1980, Guisan & Zimmermann 2000).

Value

This function returns a numeric value indicating the (adjusted) proportion of deviance accounted for by the model.

Author(s)

A. Marcia Barbosa

References

Guisan, A. & Zimmermann, N.E. (2000) Predictive habitat distribution models in ecology. Ecological Modelling 135: 147-186

Weisberg, S. (1980) Applied Linear Regression. Wiley, New York

See Also

glm, plotGLM

Examples

```
# load sample models:
data(rotif.mods)

# choose a particular model to play with:
mod <- rotif.mods$models[[1]]

Dsquared(model = mod)

Dsquared(model = mod, adjust = TRUE)


# you can also use Dsquared with vectors of observed and predicted values
# instead of with a model object:

presabs <- mod$y
prediction <- mod$fitted.values
parameters <- attributes(logLik(mod))$df

Dsquared(obs = presabs, pred = prediction, family = "binomial")

Dsquared(obs = presabs, pred = prediction, family = "binomial",
adjust = TRUE, npar = parameters)
```

`evaluate` *Evaluate a GLM based on the elements of a confusion matrix.*

Description

This function evaluates the classification performance of a model based on the values of a confusion matrix obtained at a particular threshold.

Usage

```
evaluate(a, b, c, d, N = NULL, measure = "CCR")
```

Arguments

a	number of correctly predicted presences
b	number of absences incorrectly predicted as presences
c	number of presences incorrectly predicted as absences
d	number of correctly predicted absences
N	total number of cases. If NULL (the dafault) it is calculated automatically by adding up a, b, c and d.)
measure	a character vector of length 1 indicating the the evaluation measure to use. Type 'modEvAmethods("threshMeasures")' for available options.

Details

A number of measures can be used to evaluate continuous model predictions against observed binary occurrence data (Fielding & Bell 1997; Liu et al. 2011; Barbosa et al. 2013). The 'evaluate' function can calculate a few threshold-based classification measures from the values of a confusion matrix obtained at a particular threshold. The 'evaluate' function is used internally by threshMeasures. It can also be accessed directly by the user, but it is usually more practical to use 'threshMeasures', which calculates the confusion matrix automatically.

Value

The value of the specified evaluation measure.

Note

Some measures (e.g. NMI, odds ratio) don't work with zeros in (some parts of) the confusion matrix. Also, TSS and NMI are not symmetrical, i.e. "obs" vs "pred" different from "pred" vs "obs".

Author(s)

A. Marcia Barbosa

References

Barbosa A.M., Real R., Munoz A.R. & Brown J.A. (2013) New measures for assessing model equilibrium and prediction mismatch in species distribution models. Diversity and Distributions, 19: 1333-1338

Fielding A.H. & Bell J.F. (1997) A review of methods for the assessment of prediction errors in conservation presence/absence models. Environmental Conservation 24: 38-49

Liu C., White M., & Newell G. (2011) Measuring and comparing the accuracy of species distribution models with presence-absence data. Ecography, 34, 232-243.

See Also

threshMeasures

Examples

```
evaluate(23, 44, 21, 34)

evaluate(23, 44, 21, 34, measure = "TSS")
```

---

evenness *Evenness in a binary vector.*

Description

For building and evaluating species distribution models, the porportion of presences (prevalence) of a species and the balance between the number of presences and absences may be issues to take into account (e.g. Jimenez-Valverde & Lobo 2006, Barbosa et al. 2013). The evenness function calculates the presence-absence balance in a binary (e.g., presence/absence) vector.

Usage

```
evenness(obs)
```

Arguments

obs	a vector of binary observations (e.g. 1 or 0, male or female, disease or no disease, etc.)

Value

A number ranging between 0 when all values are the same, and 1 when there are the same number of cases with each value in obs.

Author(s)

A. Marcia Barbosa

References

Barbosa A.M., Real R., Munoz A.R. & Brown J.A. (2013) New measures for assessing model equilibrium and prediction mismatch in species distribution models. Diversity and Distributions, 19: 1333-1338

Jimenez-Valverde A. & Lobo J.M. (2006) The ghost of unbalanced species distribution data in geographical model predictions. Diversity and Distributions, 12: 521-524.

See Also

prevalence

Examples

```
(x <- rep(c(0, 1), each = 5))
(y <- c(rep(0, 3), rep(1, 7)))
(z <- c(rep(0, 7), rep(1, 3)))

prevalence(x)
evenness(x)

prevalence(y)
evenness(y)

prevalence(z)
evenness(z)
```

---

getBins *Get bins of continuous values.*

Description

Get continuous predicted values into bins according to specific criteria.

Usage

```
getBins(model = NULL, obs = NULL, pred = NULL, id = NULL,
bin.method, n.bins = 10, fixed.bin.size = FALSE, min.bin.size = 15,
min.prob.interval = 0.1, quantile.type = 7, simplif = FALSE,
verbosity = 2)
```

Arguments

model	a model object of class "glm", "gam", "gbm", "randomForest" or "bart".
obs	a vector of 1-0 values of a modelled binary variable. This argument is ignored if model is provided.
pred	a vector of the corresponding predicted values. This argument is ignored if model is provided.
id	optional vector of row identifiers; must be of the same length and in the same order of obs and pred (or of the cases used to build model)
bin.method	the method with which to divide the values into bins. Type modEvAmethods("getBins") for available options and see Details for more information on these methods.
n.bins	the number of bins in which to divide the data.
fixed.bin.size	logical, whether all bins should have (approximally) the same size.
min.bin.size	integer value defining the minimum number of observations to include in each bin. The default is 15, the minimum required for accurate comparisons within bins (Jovani & Tella 2006, Jimenez-Valverde et al. 2013).

`min.prob.interval`
: minimum range of probability values in each bin. The default is 0.1.

`quantile.type`
: argument to pass to `quantile` specifying the algorithm to use if bin.method = "quantiles". The default is 7 (the `quantile` default in R), but check out other types, e.g. 3 (used by SAS), 6 (used by Minitab and SPSS) or 5 (appropriate for deciles, which correspond to the default n.bins = 10).

`simplif`
: logical, whether to calculate a faster, simplified version (used internally in other functions). The default is FALSE.

`verbosity`
: integer specifying the amount of messages or warnings to display. Defaults to the maximum implemented; lower numbers (down to 0) decrease the number of messages.

Details

Mind that different `bin.methods` can lead to visibly different results regarding the bins and any operations that depend on them (such as `HLfit`). Currently available `bin.methods` are:

- `round.prob`: probability values are rounded to the number of digits of `min.prob.interval` - e.g., if min.prob.interval = 0.1 (the default), values under 0.05 get into bin 1 (rounded probability = 0), values between 0.05 and 0.15 get into bin 2 (rounded probability = 0.1), etc. until values with probability over 0.95, which get into bin 11. Arguments n.bins, fixed.bin.size and min.bin.size are ignored by this bin.method.

- `prob.bins`: probability values are grouped into bins of the given probability intervals - e.g., if min.prob.interval = 0.1 (the default), bin 1 gets the values between 0 and 0.1, bin 2 gets the values between 0.1 and 0.2, etc. until bin 10 which gets the values between 0.9 and 1. Arguments n.bins, fixed.bin.size and min.bin.size are ignored by this bin.method.

- `size.bins`: probability values are grouped into bins of (approximately) equal size, defined by argument min.bin.size. Arguments n.bins and min.prob.interval are ignored by this bin.method.

- `n.bins`: probability values are divided into the number of bins given by argument n.bins, and their sizes may or may not be forced to be (approximately) equal, depending on argument fixed.bin.size (which is FALSE by default). Arguments min.bin.size and min.prob.interval are ignored by this bin.method.

- `quantiles`: probability values are divided using R function `quantile`, with probability cutpoints defined by the given n.bins (i.e., deciles by default), and with the quantile algorithm defined by argument quantile.type. Arguments fixed.bin.size, min.bin.size and min.prob.interval are ignored by this bin.method.

Value

The output of `getBins` is a list with the following components:

`prob.bin`
: the first and last value of each bin

`bins.table`
: a data frame with the sample size, number of presences, number of absences, prevalence, mean and median probability, and the difference between predicted and observed values (mean probability - observed prevalence) in each bin.

`N`
: the total number of observations in the analysis.

`n.bins`
: the total number of bins obtained.

Note

This function is still under development and may fail for some datasets and binning methods (e.g., ties may sometimes preclude binning under some bin.methods). Fixes and further binning methods are in preparation. Feedback is welcome.

Author(s)

A. Marcia Barbosa

References

Jimenez-Valverde A., Acevedo P., Barbosa A.M., Lobo J.M. & Real R. (2013) Discrimination capacity in species distribution models depends on the representativeness of the environmental domain. Global Ecology and Biogeography 22: 508-516.

Jovani R. & Tella J.L. (2006) Parasite prevalence and sample size: misconceptions and solutions. Trends in Parasitology 22: 214-218.

See Also

HLfit

Examples

```
# load sample models:

data(rotif.mods)


# choose a particular model to play with:

mod <- rotif.mods$models[[1]]


# try getBins using different binning methods:

getBins(model = mod, bin.method = "quantiles")

getBins(model = mod, bin.method = "n.bins")

getBins(model = mod, bin.method = "n.bins", fixed.bin.size = TRUE)
```

---

getModEqn *Get model equation*

Description

This function retrieves the equation of a model, to print or apply elsewhere.

Usage

```
getModEqn(model, type = "Y", digits = NULL, prefix = NULL,
suffix = NULL)
```

Arguments

model	a model object of class 'lm' or glm'.
type	the type of equation to get; can be either "Y" (the default, for the linear model equation), "P" (for probabiity) or "F" (for favourability).
digits	the number of digits to which to round the coefficient estimates in the equation.
prefix	the prefix to add to each variable name in the equation.
suffix	the suffix to add to each variable name in the equation.

Details

The summary of a model in R gives you a table of the coefficient estimates and other parameters. Sometimes it may be useful to have a string of text with the model's equation, so that you can present it in an article (e.g. Real et al. 2005) or apply it in a (raster map) calculation, either in R (although here you can usually use the 'predict' function for this) or in a GIS software (e.g. Barbosa et al. 2010). The getModEqn function gets this equation for linear or generalized linear models.

By default it prints the "Y" linear equation, but for generalized linear models you can also set type = "P" (for the equation of probability) or type = "F" (for favourability, which corrects the intercept to eliminate the effect of prevalence - see Real et al. 2006).

If the variables to which you want to apply the model have a prefix or suffix (e.g. prefix = "raster.stack$" for the R raster package, or prefix = "mydata$" for a data frame, or suffix = "@1" in Quantum GIS, or suffix = "@mapset" in GRASS), you can get these in the equation too, using the prefix and/or the suffix argument.

Value

A charachter string of model the equation.

Author(s)

A. Marcia Barbosa

References

Barbosa A.M., Real R. & Vargas J.M. (2010) Use of coarse-resolution models of species' distributions to guide local conservation inferences. Conservation Biology 24: 1378-87

Real R., Barbosa A.M., Martinez-Solano I. & Garcia-Paris, M. (2005) Distinguishing the distributions of two cryptic frogs (Anura: Discoglossidae) using molecular data and environmental modeling. Canadian Journal of Zoology 83: 536-545

Real R., Barbosa A.M. & Vargas J.M. (2006) Obtaining environmental favourability functions from logistic regression. Environmental and Ecological Statistics 13: 237-245

Examples

```
# load sample models:
data(rotif.mods)

# choose a particular model to play with:
mod <- rotif.mods$models[[1]]

getModEqn(mod)

getModEqn(mod, type = "P", digits = 3, suffix = "@mapset")

getModEqn(mod, type = "F", digits = 2)
```

HLfit *Hosmer-Lemeshow goodness of fit*

Description

This function calculates a model's calibration performance (reliability) with the Hosmer & Lemeshow goodness-of-fit statistic, which compares predicted probability to observed occurrence frequency at each portion of the probability range.

Usage

```
HLfit(model = NULL, obs = NULL, pred = NULL, bin.method,
n.bins = 10, fixed.bin.size = FALSE, min.bin.size = 15,
min.prob.interval = 0.1, quantile.type = 7, simplif = FALSE,
verbosity = 2, alpha = 0.05, plot = TRUE, plot.values = TRUE,
plot.bin.size = TRUE, xlab = "Predicted probability",
ylab = "Observed prevalence", ...)
```

Arguments

model	a model object of class "glm", "gam", "gbm", "randomForest" or "bart".
obs	a vector of observed presences (1) and absences (0) or another binary response variable. This argument is ignored if model is provided.
pred	a vector with the corresponding predicted probabilities as given e.g. by logistic regression. A warning is emitted if it includes values outside the [0, 1] interval. This argument is ignored if model is provided.
bin.method	argument to pass to getBins specifying the method for grouping the records into bins within which to compare predicted probability to observed prevalence; type modEvAmethods("getBins") for available options, and see Details for more information.
n.bins	argument to pass to getBins specifying the number of bins to use if bin.method = n.bins or bin.method = quantiles. The default is 10.

fixed.bin.size	argument to pass to getBins, a logical value indicating whether to force bins to have (approximately) the same size. The default is FALSE.
min.bin.size	argument to pass to getBins specifying the minimum number of records in each bin. The default is 15, the minimum required for accurate comparisons within bins (Jovani & Tella 2006, Jimenez-Valverde et al. 2013).
min.prob.interval	argument to pass to getBins specifying the minimum interval (range) of probability values within each bin. The default is 0.1.
quantile.type	argument to pass to quantile specifying the algorithm to use if bin.method = "quantiles". The default is 7 (the quantile default in R), but check out other types, e.g. 3 (used by SAS), 6 (used by Minitab and SPSS) or 5 (appropriate for deciles, which correspond to the default n.bins = 10).
simplif	logical, wheter to perform a faster simplified version returning only the basic statistics. The default is FALSE.
verbosity	integer specifying the amount of messages or warnings to display. Defaults to the maximum implemented; lower numbers (down to 0) decrease the number of messages.
alpha	alpha value for confidence intervals if plot = TRUE.
plot	logical, whether to produce a plot of the results. The default is TRUE.
plot.values	logical, whether to report measure values in the plot. The default is TRUE.
plot.bin.size	logical, whether to report bin sizes in the plot. The default is TRUE.
xlab	label for the x axis.
ylab	label for the y axis.
...	further arguments to pass to the plot function.

Details

Most of the commonly used measures for evaluating model performance focus on the discrimination or the classification capacity, i.e., how well the model is capable of distinguishing or classifying presences and absences (often after the model's continuous predictions of presence probability or alike are converted to binary predictions of presence or absence). However, there is another important facet of model evaluation: calibration or reliability, i.e., the relationship between predicted probability and observed occurrence frequency (Pearce & Ferrier 2000; Jimenez-Valverde et al. 2013). The HLfit function measures model reliability with the Hosmer & Lemeshow goodness-of-fit statistic (Hosmer & Lemeshow 1980).

Note that this statistic has strong limitations and caveats (see e.g. http://www.statisticalhorizons.com/hosmer-lemeshow, Allison 2014), mainly due to the need to group the values into bins within which to compare probability and prevalence, and the strong influence of the binning method on the results. The 'HLfit' function can use several binning methods, which are implemented and roughly explained in the getBins function and can be accessed by typing 'modEvAmethods("getBins")'. You should try 'HLfit' with different binning methods to see how if the results are robust.

Value

HLfit returns a list with the following components:

bins.table	a data frame of the obtained bins and the values resulting from the hosmer-Lemeshow goodness-of-fit analysis.
chi.sq	the value of the Chi-squared test.
DF	the number of degrees of freedom.
p.value	the p-value of the Hosmer-Lemeshow test. Note that this is one of those tests for which higher p-values are better.
RMSE	the root mean squared error.

Note

The 4 lines of code from "observed" to "p.value" were adapted from the 'hosmerlem' function available at http://www.stat.sc.edu/~hitchcock/diseaseoutbreakRexample704.txt. The plotting code was loosely based on the `calibration.plot` function in package **PresenceAbsence**. `HLfit` still needs some code simplification, and may fail for some datasets and binning methods. Fixes are being applied. Feedback is welcome.

Author(s)

A. Marcia Barbosa

References

Allison P.D. (2014) Measures of Fit for Logistic Regression. SAS Global Forum, Paper 1485

Hosmer D.W. & Lemeshow S. (1980) A goodness-of-fit test for the multiple logistic regression model. Communications in Statistics, A10: 1043-1069

Jimenez-Valverde A., Acevedo P., Barbosa A.M., Lobo J.M. & Real R. (2013) Discrimination capacity in species distribution models depends on the representativeness of the environmental domain. Global Ecology and Biogeography 22: 508-516

Jovani R. & Tella J.L. (2006) Parasite prevalence and sample size: misconceptions and solutions. Trends in Parasitology 22: 214-218

Pearce J. & Ferrier S. (2000) Evaluating the Predictive Performance of Habitat Models Developed using Logistic Regression. Ecological Modeling, 133: 225-245

See Also

`getBins`, `MillerCalib`

Examples

```
# load sample models:

data(rotif.mods)


# choose a particular model to play with:

mod <- rotif.mods$models[[1]]
```

```
# try HLfit using different binning methods:

HLfit(model = mod, bin.method = "round.prob",
main = "HL GOF with round.prob (n=10)")

HLfit(model = mod, bin.method = "prob.bins",
main = "HL GOF with prob.bins (n=10)")

HLfit(model = mod, bin.method = "size.bins",
main = "HL GOF with size.bins (min size=15)")

HLfit(model = mod, bin.method = "size.bins", min.bin.size = 30,
main = "HL GOF with size.bins min size 30")

HLfit(model = mod, bin.method = "n.bins",
main = "HL GOF with 10 bins")

HLfit(model = mod, bin.method = "n.bins", fixed.bin.size = TRUE,
main = "HL GOF with 10 bins of fixed size")

HLfit(model = mod, bin.method = "n.bins", n.bins = 20,
main = "HL GOF with 20 bins")

HLfit(model = mod, bin.method = "quantiles",
main = "HL GOF with quantile bins (n=10)")

HLfit(model = mod, bin.method = "quantiles", n.bins = 20,
main = "HL GOF with quantile bins (n=20)")
```

MESS *Multivariate Environmental Similarity Surfaces based on a data frame*

Description

This function performs the MESS analysis of Elith et al. (2010) to determine the extent of the environmental differences between model training and model projection (extrapolation) data. It is applicable to variables in a matrix or data frame.

Usage

```
MESS(V, P, id.col = NULL)
```

Arguments

V	a matrix or data frame containing the variables (one in each column) in the training dataset.
P	a matrix or data frame containing the same variables in the area to which the model(s) will be projected. Variables (columns) must be in the same order as in V, and `colnames(P)` must exist.

`id.col` optionally, the index number of a column containing the row identifiers in P. If provided, this column will be excluded from MESS calculations but included in the output.

Details

When model predictions are projected into regions, times or spatial resolutions not analysed in the training data, it may be important to measure the similarity between the new environments and those in the training sample (Elith et al. 2010), as models are not so reliable when predicting outside their domain (Barbosa et al. 2009). The Multivariate Environmental Similarity Surfaces (MESS) analysis measures the similarity in the analysed variables between any given locality in the projection dataset and the localities in the reference (training) dataset (Elith et al. 2010).

MESS analysis is implemented in the MAXENT software (Phillips et al. 2006) and in the **dismo** R package, but there it requires input variables in raster format. This implies not only the use of complex spatial data structures, but also that the units of analysis are rectangular pixels, whereas we often need to model distribution data recorded on less regular units (e.g. provinces, river basins), or on equal-area cells that are not necessarily rectangular (e.g. UTM cells, equal-area hexagons or other geometric shapes). The MESS function computes this analysis for variables in a data frame, where localities (in rows) may be of any size or shape.

Value

The function returns a data frame with the same column names as `P`, plus a column named `TOTAL`, quantifying the similarity between each point in the projection dataset and those in the reference dataset. Negative values indicate localities that are environmentally dissimilar from the reference region. The last column, `MoD`, indicates which of the column names of P corresponds to the most dissimilar variable, i.e., the limiting factor or the variable that drives the MESS in that locality (Elith et al. 2010).

Note

Newer and apparently more complete methods for analysing environmental dissimilarities have been developed, such as extrapolation detection (ExDet; Mesgaran et al. 2014) and Mobility-Oriented Parity analysis (MOP; Owens et al. 2013).

Author(s)

Alberto Jimenez-Valverde, A. Marcia Barbosa

References

Barbosa A.M., Real R. & Vargas J.M. (2009) Transferability of environmental favourability models in geographic space: the case of the Iberian desman (Galemys pyrenaicus) in Portugal and Spain. Ecological Modelling 220: 747-754

Elith J., Kearney M. & Phillips S. (2010) The art of modelling range-shifting species. Methods in Ecology and Evolution 1: 330-342

Mesgaran M.B., Cousens R.D. & Webber B.L. (2014) Here be dragons: a tool for quantifying novelty due to covariate range and correlation change when projecting species distribution models. Diversity and Distributions, 20: 1147-1159

Owens H.L., Campbell L.P., Dornak L.L., Saupe E.E., Barve N., Soberon J., Ingenloff K., Lira-Noriega A., Hensz C.M., Myers C.E. & Peterson A.T. (2013) Constraints on interpretation of ecological niche models by limited environmental ranges on calibration areas. Ecological Modelling, 263: 10-18

Phillips S.J., Anderson R.P. & Schapire R.E. (2006) Maximum entropy modeling of species geographic distributions. Ecological Modelling 190: 231-259

See Also

OA; mess in package **dismo**; ecospat.climan in package **ecospat**; kuenm_mop and kuenm_mmop in package **kuenm**

Examples

```
## Not run:
# load package 'fuzzySim' and its sample data:
require(fuzzySim)
data(rotif.env)


# add a column specifying the hemisphere:

unique(rotif.env$CONTINENT)

rotif.env$HEMISPHERE <- "Eastern"

rotif.env$HEMISPHERE[rotif.env$CONTINENT %in%
c("NORTHERN_AMERICA", "SOUTHERN_AMERICA")] <- "Western"

head(rotif.env)


# perform a MESS analysis
# suppose you'll extrapolate models from the Western hemisphere (Americas)
# to the Eastern hemisphere (rest of the world):

names(rotif.env)  # variables are in columns 5:17

west <- subset(rotif.env, HEMISPHERE == "Western", select = 5:17)
east <- subset(rotif.env, HEMISPHERE == "Eastern", select = 5:17)
east.with.ID <- subset(rotif.env, HEMISPHERE == "Eastern",
select = c(1, 5:17))

head(east)
head(east.with.ID)  # ID is in column 1

mess <- MESS(V = west, P = east)
mess.with.ID <- MESS(V = west, P = east.with.ID, id.col = 1)

head(mess)
head(mess.with.ID)
```

```
range(mess[ , "TOTAL"])

## End(Not run)
```

`MillerCalib` *Miller's calibration satistics for logistic regression models*

Description

This function calculates Miller's (1991) calibration statistics for a generalized linear model with binomial distribution and logistic link, namely the intercept and slope of the regression of the response variable on the logit of predicted probabilities. Optionally and by default, it also plots the corresponding regression line over the reference diagonal.

Usage

```
MillerCalib(model = NULL, obs = NULL, pred = NULL, plot = TRUE,
line.col = "black", diag = TRUE, diag.col = "grey",
plot.values = TRUE, digits = 2, xlab = "", ylab = "",
main = "Miller calibration", ...)
```

Arguments

`model`	a binary-response model object of class "glm", "gam", "gbm", "randomForest" or "bart".
`obs`	a vector of observed presences (1) and absences (0) or another binary response variable. Not necessary (and ignored) if 'model' is provided.
`pred`	a vector with the corresponding predicted values of presence probability. Must be of the same length and in the same order as 'obs'. A warning is emitted if it includes values outside the [0, 1] interval. Not necessary (and ignored) if 'model' is provided.
`plot`	logical, whether or not to produce a plot of the Miller regression line. Defaults to TRUE.
`line.col`	colour for the Miller regression line (if plot = TRUE).
`diag`	logical, whether or not to add the reference diagonal (if plot = TRUE). Defaults to TRUE.
`diag.col`	line colour for the reference diagonal.
`plot.values`	logical, whether or not to report the values of the intercept and slope on the plot. Defaults to TRUE.
`digits`	integer number indicating the number of digits to which the values in the plot should be rounded. Dafaults to 2. This argument is ignored if 'plot' or 'plot.values' are set to FALSE.
`xlab`	label for the x axis.
`ylab`	label for the y axis.
`main`	title for the plot.
`...`	additional arguments to pass to `plot`.

Details

Calibration or reliability measures how a model's predicted probabilities relate to observed species prevalence or proportion of presences in the modelled data (Pearce & Ferrier 2000; Wintle et al. 2005; Franklin 2010). If predictions are perfectly calibrated, the slope will equal 1 and the intercept will equal 0, so the model's calibation line will perfectly overlap with the reference diagonal. Note that Miller's statistics assess the model globally: a model is well calibrated if the average of all predicted probabilities equals the proportion of presences in the modelled data. Good calibration is always attained on the same data used for building the model (Miller 1991); Miller's calibration statistics are mainly useful when extrapolating a model outside those training data.

Value

This function returns a list of two integer values:

`intercept`	the calibration intercept.
`slope`	the calibration slope.

If `plot = TRUE`, a plot will be produced with the model calibration line, optionally (if diag = TRUE) over the reference diagonal, and optionally (if plot.values = TRUE) with the intercept and slope values printed on it.

Author(s)

A. Marcia Barbosa

References

Franklin, J. (2010) Mapping Species Distributions: Spatial Inference and Prediction. Cambridge University Press, Cambridge.

Miller M.E., Hui S.L. & Tierney W.M. (1991) Validation techniques for logistic regression models. Statistics in Medicine, 10: 1213-1226

Pearce J. & Ferrier S. (2000) Evaluating the predictive performance of habitat models developed using logistic regression. Ecological Modelling, 133: 225-245

Wintle B.A., Elith J. & Potts J.M. (2005) Fauna habitat modelling and mapping: A review and case study in the Lower Hunter Central Coast region of NSW. Austral Ecology, 30: 719-738

See Also

`HLfit`, `Dsquared`, `RsqGLM`

Examples

```
# load sample models:
data(rotif.mods)

# choose a particular model to play with:
mod <- rotif.mods$models[[1]]

MillerCalib(model = mod)
```

```
MillerCalib(model = mod, plot.values = FALSE)
MillerCalib(model = mod, main = "Model calibration line")


# you can also use MillerCalib with vectors of observed and predicted values
# instead of a model object:

MillerCalib(obs = mod$y, pred = mod$fitted.values)
```

`mod2obspred` *Observed and predicted values from a model object.*

Description

This function takes a model object and returns the observed and (optionally) the fitted values in that model.

Usage

```
mod2obspred(model, obs.only = FALSE)
```

Arguments

`model`	a model object of class "glm", "gam", "gbm", "randomForest" or "bart" from which the response variable and fitted values can be extracted.
`obs.only`	logical, whether only 'obs' should be obtained (saves computing time when 'pred' not needed – used e.g. by `prevalence`). Defaults to FALSE.

Value

A data frame with one column containing the observed and (if obs.only=FALSE, the default) another column containing the predicted values from 'model'.

Author(s)

A. Marcia Barbosa

See Also

`prevalence`

Examples

```
data(rotif.mods)
mod <- rotif.mods$models[[1]]
obspred <- mod2obspred(mod)
head(obspred)
```

`modEvAmethods` *Methods implemented in modEvA functions*

Description

This function allows retrieving the methods available for some of the functions in modEvA, such as `threshMeasures`, `optiThresh`, `multModEv` and `getBins`.

Usage

```
modEvAmethods(fun)
```

Arguments

`fun`	a character vector of length 1 specifying the name (in quotes) of the function for which to obtain the available methods.

Value

a character vector of the available methods for the specified function.

Author(s)

A. Marcia Barbosa

See Also

`threshMeasures`, `optiThresh`, `getBins`, `multModEv`

Examples

```
modEvAmethods("threshMeasures")

modEvAmethods("multModEv")

modEvAmethods("optiThresh")

modEvAmethods("getBins")
```

multModEv *Multiple model evaluation*

Description

If you have a list of GLM model objects (created, e.g., with the multGLM function of the 'fuzzySim' R-Forge package), or a data frame with presence-absence data and the corresponding predicted values for a set of species, you can use the multModEv function to get a set of evaluation measures for all models simultaneously, as long as they all have the same sample size.

Usage

```
multModEv(models = NULL, obs.data = NULL, pred.data = NULL,
measures = modEvAmethods("multModEv"), standardize = FALSE,
thresh = NULL, bin.method = NULL, verbosity = 0, ...)
```

Arguments

models	a list of model object(s) of class "glm", all applied to the same data set. Evaluation is based on the cases included in the models.
obs.data	a data frame with observed (training or test) binary data. This argument is ignored if 'models' is provided.
pred.data	a data frame with the corresponding predicted (training or test) values, with both rows and columns in the same order as in 'obs.data'. This argument is ignored if 'models' is provided. Note that, for calibration measures (based on HLfit or MillerCalib), the results are only valid if the input predictions represent probability.
measures	character vector of the evaluation measures to calculate. The default is all implemented measures, which you can check by typing 'modEvAmethods("multModEv")'. But beware: calibration measures (i.e., HL and Miller) are only valid if your predicted values reflect actual presence probability (not favourability, habitat suitability or others); you should exclude them otherwise.
standardize	logical, whether to standardize measures that vary between -1 and 1 to the 0-1 scale (see standard01). The default is FALSE.
thresh	argument to pass to threshMeasures if any of 'measures' is calculated by that function. The default is NULL, but a valid method must be specified if any of 'measures' is threshold-based - i.e., any of those in 'modEvAmethods("threshMeasures")'.
bin.method	the method with which to divide the data into groups or bins, for calibration or reliability measures such as HLfit. The default is NULL, but a valid method must be specified if 'measures' includes "HL" or "HL.p". Type modEvAmethods("getBins") for available options), and see HLfit and getBins for more information.
verbosity	integer specifying the amount of messages or warnings to display. Defaults to 0, but can also be 1 or 2 for more messages from the functions within.

... optional arguments to pass to HLfit (if "HL" or "HL.p" are included in 'measures'), namely n.bins, fixed.bin.size, min.bin.size, min.prob.interval or quantile.type.

Value

A data frame with the value of each evaluation measure for each model.

Author(s)

A. Marcia Barbosa

See Also

threshMeasures

Examples

```
data(rotif.mods)

eval1 <- multModEv(models = rotif.mods$models[1:6], thresh = 0.5,
bin.method = "n.bins", fixed.bin.size = TRUE)

head(eval1)


eval2 <- multModEv(models = rotif.mods$models[1:6],
thresh = "preval", measures = c("AUC", "AUCPR", "CCR",
"Sensitivity", "TSS"))

head(eval2)


# you can also calculate evaluation measures for a set of
# observed vs predicted data, rather than from model objects:

obses <- sapply(rotif.mods$models, `[[`, "y")
preds <- sapply(rotif.mods$models, `[[`, "fitted.values")

eval3 <- multModEv(obs.data = obses[ , 1:4],
pred.data = preds[ , 1:4], thresh = "preval",
bin.method = "prob.bins")

head(eval3)
```

OA *Overlap Analysis*

Description

This function analyses the range of values of the given environmental variables at the sites where a species has been recorded present.

Usage

```
OA(data, sp.cols, var.cols)
```

Arguments

`data`	a data frame with your species' occurrence data and the predictor variables.
`sp.cols`	index number of the column containing the occurrence data of the species to be modelled. Currently only one species can be analysed at a time.
`var.cols`	index numbers of the columns containing the predictor variables to be used.

Details

Overlap Analysis is one of the simplest forms of modelling species' distributions. It assesses the ranges of values of the given environmental variables at the sites where a species has been recorded present, and predicts where that species should be able to occur based on those presence data (e.g. Brito et al. 1999, Arntzen & Teixeira 2006).

OA can also be useful when extrapolating models outside their original scope (geographical area, time period or spatial resolution), as it can identify which localities are within the model's domain - i.e., within the analysed ranges of values of the variables, outside which the model may not be reliable (e.g. Barbosa et al. 2009). In this case, the response is not a species' presence, but rather the sites that have been included in the model. See also the `MESS` function for a comparison between modelled and extrapolation environments.

Input data for the `OA` function are a vector or column with ones and zeros (presences vs. absences of a species if we want to model its occurrence, or modelled vs. non-modelled sites if we want to know which non-modelled sites are within the modelled range), and a matrix or data frame with the corresponding values of the environmental variables to consider (one variable in each column, values in rows).

Value

A binary vector whith 1 where the values of all predictors lie within the ranges observed for the presence records, and 0 otherwise.

Author(s)

A. Marcia Barbosa

References

Arntzen J.W, Teixeira J. (2006) History and new developments in the mapping and modelling of the distribution of the golden-striped salamander, Chioglossa lusitanica. Zeitschrift fur Feldherpetologie, Supplement: 1-14.

Barbosa, A.M., Real, R. & Vargas, J.M. (2009) Transferability of environmental favourability models in geographic space: the case of the Iberian desman (Galemys pyrenaicus) in Portugal and Spain. Ecological Modelling 220: 747-754.

Brito J.C., Crespo E.G., Paulo O.S. (1999) Modelling wildlife distributions: Logistic Multiple Regression vs Overlap Analysis. Ecography 22: 251-260.

See Also

MESS

Examples

```
## Not run:
# load package 'fuzzySim' and its sample data:
require(fuzzySim)
data(rotif.env)

names(rotif.env)

OA(rotif.env, sp.cols = 18, var.cols = 5:17)

## End(Not run)
```

`optiPair` *Optimize the classification threshold for a pair of related model evaluation measures.*

Description

This function can optimize a model's classification threshold based on a pair of model evaluation measures that balance each other, such as sensitivity-specificity, precision-recall (i.e., positive predictive power vs. sensitivity), or omission-commission, or underprediction-overprediction (Fielding & Bell 1997; Liu et al. 2011; Barbosa et al. 2013). The function plots both measures of the given pair against all thresholds with a given interval, and calculates the optimal sum, difference and mean of the two measures.

Usage

```
optiPair(model = NULL, obs = NULL, pred = NULL,
measures = c("Sensitivity", "Specificity"), interval = 0.01,
plot = TRUE, plot.sum = FALSE, plot.diff = FALSE, ylim = NULL,
na.rm = TRUE, exclude.zeros = TRUE, ...)
```

Arguments

model	a model object of class "glm", "gam", "gbm", "randomForest" or "bart".
obs	a vector of observed presences (1) and absences (0) or another binary response variable. This argument is ignored if 'model' is provided.
pred	a vector with the corresponding predicted values of presence probability, habitat suitability, environmental favourability or alike. This argument is ignored if 'model' is provided.
measures	a character vector of length 2 indicating the pair of measures whose curves to plot and whose combined threshold to optimize. Available measures can be obtained with 'modEvAmethods("threshMeasures")', but note that this function expects you to use two measures that counter-balance one another, such as c("Sensitivity", "Specificity") [the default], c("Omission", "Commission"), or c("Precision", "Recall").
interval	the interval of thresholds at which to calculate the measures. The default is 0.01.
plot	logical indicating whether or not to plot the pair of measures.
plot.sum	logical, whether to plot the sum (+) of both measures in the pair. Defaults to FALSE.
plot.diff	logical, whether to plot the difference (-) between both measures in the pair. Defaults to FALSE.
ylim	a character vector of length 2 indicating the lower and upper limits for the y axis. The default is NULL for an automatic definition of 'ylim' based on the values of the measures and their sum and/or difference if any of these are set to TRUE.
na.rm	logical, whether NA values should be removed from the calculation of minimum/maximum/mean values to get the optimized measures. Defaults to TRUE.
exclude.zeros	logical, whether non-finite and zero values should be removed from the calculation of minimum/maximum/mean values to get the optimized measures. Defaults to TRUE.
...	additional arguments to be passed to the plot function.

Value

The output is a list with the following components:

measures.values	a data frame with the values of the chosen pair of measures, as well as their difference, sum and mean, at each threshold.
MinDiff	numeric value, the minimum difference between both measures.
ThreshDiff	numeric value, the threshold that minimizes the difference between both measures.
MaxSum	numeric value, the maximum sum of both measures.
ThreshSum	numeric value, the threshold that maximizes the sum of both measures.
MaxMean	numeric value, the maximum mean of both measures.
ThreshMean	numeric value, the threshold that maximizes the mean of both measures.

If plot=TRUE (the default), a plot is also produced with the value of each of 'measures' at each threshold, and horizontal and vertical lines marking, respectively, the threshold and value at which the difference between the two 'measures' is minimal.

Author(s)

A. Marcia Barbosa

References

Barbosa, A.M., Real, R., Munoz, A.-R. & Brown, J.A. (2013) New measures for assessing model equilibrium and prediction mismatch in species distribution models. Diversity and Distributions 19: 1333-1338

Fielding A.H. & Bell J.F. (1997) A review of methods for the assessment of prediction errors in conservation presence/absence models. Environmental Conservation 24: 38-49

Liu C., White M., & Newell G. (2011) Measuring and comparing the accuracy of species distribution models with presence-absence data. Ecography, 34, 232-243.

See Also

optiThresh, threshMeasures

Examples

```
# load sample models:
data(rotif.mods)


# choose a particular model to play with:
mod <- rotif.mods$models[[1]]

optiPair(model = mod)

optiPair(model = mod, measures = c("Precision", "Recall"))

optiPair(model = mod, measures = c("UPR", "OPR"))

optiPair(model = mod, measures = c("CCR", "F1score"))


# you can also use 'optiPair' with vectors of observed
# and predicted values, instead of a model object:

optiPair(obs = mod$y, pred = mod$fitted.values)
```

optiThresh *Optimize threshold for model evaluation.*

Description

The 'optiThresh' function calculates optimal thresholds for a number of model evaluation measures (see threshMeasures). Optimization is given for each measure, and/or for all measures according to particular criteria (e.g. Jimenez-Valverde & Lobo 2007; Liu et al. 2005; Nenzen & Araujo 2011). Results are given numerically and in plots.

Usage

```
optiThresh(model = NULL, obs = NULL, pred = NULL, interval = 0.01,
measures = modEvAmethods("threshMeasures"),
optimize = modEvAmethods("optiThresh"), simplif = FALSE,
plot = TRUE, sep.plots = FALSE, xlab = "Threshold", ...)
```

Arguments

model	a model object of class "glm", "gam", "gbm", "randomForest" or "bart".
obs	a vector of observed presences (1) and absences (0) or another binary response variable. This argument is ignored if 'model' is provided.
pred	a vector with the corresponding predicted values of presence probability, habitat suitability, environmental favourability or alike. This argument is ignored if 'model' is provided.
interval	numeric value between 0 and 1 indicating the interval between the thresholds at which to calculate the evaluation measures. Defaults to 0.01.
measures	character vector indicating the names of the model evaluation measures for which to calculate optimal thresholds. The default is using all measures available in 'modEvAmethods("threshMeasures")'.
optimize	character vector indicating the threshold optimization criteria to use; "each" calculates the optimal threshold for each model evaluation measure, while the remaining options optimize all measures according to the specified criterion. The default is using all criteria available in 'modEvAmethods("optiThresh")'.
simplif	logical, whether to calculate a faster simplified version. Used internally in other functions.
plot	logical, whether to plot the values of each evaluation measure at all thresholds.
sep.plots	logical. If TRUE, each plot is presented separately (you need to be recording R plot history to be able to browse through them all); if FALSE(the default), all plots are presented together in the same plotting window.
xlab	character vector indicating the label of the x axis.
...	additional arguments to pass to plot.

Value

This function returns a list with the following components:

`all.thresholds` a data frame with the values of all analysed measures at all analysed thresholds.

`optimals.each` if "each" is among the threshold criteria specified in 'optimize', optimals.each is output as a data frame with the value of each measure at its optimal threshold, as well as the type of optimal for that measure (which may be the maximum for measures of goodness such as "Sensitivity", or the minimum for measures of badness such as "Omission").

`optimals.criteria` a data frame with the values of measure at the threshold that maximizes each of the criteria specified in 'optimize' (except for "each", see above).

Note

"Sensitivity" is the same as "Recall", and "PPP" (positive predictive power) is the same as "Precision". "F1score"" is the harmonic mean of precision and recall.

Note

Some measures cannot be calculated for thresholds at which there are zeros in the confusion matrix, hence the eventual 'NaN' or 'Inf' in results. Also, optimization may be deceiving for some measures; use 'plot = TRUE' and inspect the plot(s).

Author(s)

A. Marcia Barbosa

References

Jimenez-Valverde A. & Lobo J.M. (2007) Threshold criteria for conversion of probability of species presence to either-or presence-absence. Acta Oecologica 31: 361-369.

Liu C., Berry P.M., Dawson T.P. & Pearson R.G. (2005) Selecting thresholds of occurrence in the prediction of species distributions. Ecography 28: 385-393.

Nenzen H.K. & Araujo M.B. (2011) Choice of threshold alters projections of species range shifts under climate change. Ecological Modelling 222: 3346-3354.

See Also

`threshMeasures`, `optiPair`

Examples

```
# load sample models:
data(rotif.mods)

# choose a particular model to play with:
mod <- rotif.mods$models[[1]]
```

```
## Not run:
optiThresh(model = mod)

## End(Not run)


# change some of the parameters:

optiThresh(model = mod, pch = 20,
measures = c("CCR", "Sensitivity", "kappa", "TSS"), ylim = c(0, 1))


# you can also use optiThresh with vectors of observed and predicted
# values instead of with a model object:

## Not run:
optiThresh(obs = mod$y, pred = mod$fitted.values, pch = ".")

## End(Not run)
```

---

plotGLM *Plot a generalized linear model*

Description

This function plots the observed (presence/absence) data and the predicted (probability) values of a Generalized Linear Model against the y regression equation (logit) values. Only logistic regression (binomial response, logit link) is currently implemented.

Usage

```
plotGLM(model = NULL, obs = NULL, pred = NULL, link = "logit",
plot.values = TRUE, plot.digits = 3, xlab = "Logit (Y)",
ylab = "Predicted probability", main = "Model plot", ...)
```

Arguments

model	a model object of class "glm".
obs	a vector of presence/absence or other binary (1-0) observed data. Not necessary (and ignored) if 'model' is provided.
pred	a vector of the values predicted by a GLM of the binary observed data. Not necessary (and ignored) if 'model' is provided.
link	the link function of the GLM; only 'logit' (the default) is implemented.
plot.values	logical, whether to include in the plot diagnostic values such as explained deviance (calculated with the Dsquared function) and pseudo-R-squared measures (calculated with the RsqGLM function). Defaults to TRUE.
plot.digits	integer number indicating the number of digits to which the values in the plot should be rounded (if 'plot.values = TRUE'). Defaults to 3.

xlab	character string specifying the label for the x axis.
ylab	character string specifying the label for the y axis.
main	character string specifying the title for the plot.
...	additional arguments to pass to plot.

Value

This function outputs a plot of model predictions against observations.

Author(s)

A. Marcia Barbosa

References

Guisan A. & Zimmermann N.E. (2000) Predictive habitat distribution models in ecology. Ecological Modelling 135: 147-186

Weisberg S. (1980) Applied Linear Regression. Wiley, New York

See Also

predPlot, predDensity

Examples

```
# load sample models:
data(rotif.mods)

# choose a particular model to play with:
mod <- rotif.mods$models[[1]]

plotGLM(model = mod)

plotGLM(model = mod, plot.values = FALSE)


# you can also use 'plotGLM' with vectors of observed and
# predicted values instead of with a model object:

plotGLM(obs = mod$y, pred = mod$fitted.values)
```

predDensity *Plot the density of predicted values for presences and absences.*

Description

This function produces a histogram and/or a kernel density plot of predicted values for a binomial GLM, by default separately for the observed presences and absences, given a model object or a vector of predicted values and (optionally) a vector of the corresponding observed values.

Usage

```
predDensity(model = NULL, obs = NULL, pred = NULL, separate = TRUE,
type = c("both"), legend.pos = "topright", main = "Density of predicted values")
```

Arguments

model	a binary-response model object of class "glm", "gam", "gbm", "randomForest" or "bart".
obs	numeric vector of the observed (response) variable, consisting of zeros and ones. This argument is ignored if 'model' is provided.
pred	numeric vector of probability values predicted for 'obs'. This argument is ignored if 'model' is provided. Must be of the same length and in the same order as 'obs'.
separate	logical value indicating whether prediction densities should be computed separately for observed presences (ones) and absences (zeros). Defaults to TRUE, but is changed to FALSE if either 'model' or 'obs' not provided.
type	character vector specifying whether to produce a "histogram", a "density" plot, or "both" (the default). Partial argument matching is used.
legend.pos	character specifying the position for the legend; NA or "n" for no legend. Position can be "topright" (the default), "topleft, "bottomright"", "bottomleft", "top", "bottom", "left", "right", or "center". Partial argument matching is used.
main	main title for the plot.

Details

For more details, please refer to the documentation of the functions mentioned under "See Also".

Value

This function outputs and plots the object(s) specified in 'type' – by default, a density object and a histogram.

Author(s)

A. Marcia Barbosa

See Also

hist, density, predPlot

Examples

```
# load sample models:
data(rotif.mods)

# choose a particular model to play with:
mod <- rotif.mods$models[[1]]

predDensity(model = mod)

predDensity(model = mod, type = "histogram")

predDensity(model = mod, type = "density")

# you can also use 'predDensity' with vectors of
# observed and predicted values, instead of a model object:

presabs <- mod$y
prediction <- mod$fitted.values

predDensity(obs = presabs, pred = prediction)

predDensity(pred = prediction)
```

---

predPlot *Plot predicted values for presences and absences, classified according to a prediction threshold.*

Description

This function plots predicted values separated into observed presences and absences and coloured according to whether they are above or below a given prediction threshold. The plot imitates (with permission from the author) one of the graphical outputs of the 'summary' of models built with the **embarcadero** package (Carlson, 2020), but it can be applied to any 'glm' object or any set of observed and predicted values, and it allows specifying a user-defined threshold.

Usage

```
predPlot(model = NULL, obs = NULL, pred = NULL, thresh = "preval",
main = "Classified predicted values", legend.pos = "n", pch = 1, col = c("black", "grey"))
```

Arguments

model	a binary-response model object of class "glm", "gam", "gbm", "randomForest" or "bart".

obs	(instead of a 'model' object) numeric vector of the observed (response) variable, consisting of zeros (absences) and ones (presences). This argument is ignored if 'model' is provided.
pred	(instead of a 'model' object) numeric vector of the values predicted by a model of the 'obs'erved data. This argument is ignored if 'model' is provided. Must be of the same length and in the same order as 'obs'.
thresh	threshold value to separate predicted presences from predicted absences in 'pred'; can be "preval" (the default), to use the prevalence (i.e. proportion of presences) in 'obs', or any real number between 0 and 1. This value will be used to draw a vertical line on the plot and to colour the points (predicted values) according to whether they fall below or above the threshold. See Details in help(threshMeasures) for an informed choice.
main	Main title for the plot.
legend.pos	character value specifying the position for the legend on the plot. Can be "bottomleft", "bottom", "bottomright", "topleft", "left", "top", "topright", "right", "center", or NA or "n" for no legend (the default). Partial argument matching is used.
pch	plotting character for the presences and absences (see points).
col	vector of length 2 indicating the colours with which to plot predicted presences and absences (points above and below the threshold), respectively.

Value

This function outputs a plot as per 'Description'.

Note

Points are jittered randomly along the y axis to minimize visual overlap. So, each run of 'predPlot' (unless you use set.seed first) will produce a different arrangement of points for the same data, although their x-axis values are faithful.

Author(s)

A. Marcia Barbosa

References

Carlson C.J. (2020) embarcadero: Species distribution modelling with Bayesian additive regression trees in R. Methods in Ecology and Evolution, 11: 850-858.

See Also

predDensity, plotGLM

Examples

```
# load sample models:
data(rotif.mods)

# choose a particular model to play with:
mod <- rotif.mods$models[[1]]

predPlot(model = mod)

predPlot(model = mod, thresh = 0.5)


# you can first select a threshold optimized according to a particular metric:

## Not run:
threshold <- optiThresh(mod, measures = "TSS", optimize = "each")
threshold <- threshold$optimals.each[ , "threshold"]
threshold
predPlot(model = mod, thresh = threshold)

## End(Not run)

# you can also use 'predPlot' with vectors of observed and predicted values
# instead of a model object:

presabs <- mod$y
prediction <- mod$fitted.values

predPlot(obs = presabs, pred = prediction)

predPlot(obs = presabs, pred = prediction, thresh = 0.5)
```

`prevalence` *Prevalence*

Description

For building and evaluating species distribution models, the porportion of presences of the species may be an issue to take into account (e.g. Jimenez-Valverde & Lobo 2006, Barbosa et al. 2013). The `prevalence` function calculates this measure.

Usage

```
prevalence(obs, model = NULL, event = 1, na.rm = TRUE)
```

Arguments

`obs`	a vector or a factor of binary observations (e.g. 1 vs. 0, male vs. female, disease vs. no disease, etc.). This argument is ignored if 'model' is provided.

model	alternatively to 'obs', a binary-response model object of class "glm", "gam", "gbm", "randomForest" or "bart" from which the response variable can be extracted.
event	the value whose prevalence we want to calculate (e.g. 1, "present", etc.). This argument is ignored if 'model' is provided.
na.rm	logical, whether NA values should be excluded from the calculation. The default is TRUE.

Value

Numeric value of the prevalence of event in the obs vector.

Author(s)

A. Marcia Barbosa

References

Barbosa A.M., Real R., Munoz A.R. & Brown J.A. (2013) New measures for assessing model equilibrium and prediction mismatch in species distribution models. Diversity and Distributions, in press

Jimenez-Valverde A. & Lobo J.M. (2006) The ghost of unbalanced species distribution data in geographical model predictions. Diversity and Distributions, 12: 521-524.

See Also

evenness

Examples

```
# calculate prevalence from binary vectors:

(x <- rep(c(0, 1), each = 5))

(y <- c(rep(0, 3), rep(1, 7)))

(z <- c(rep(0, 7), rep(1, 3)))

prevalence(x)

prevalence(y)

prevalence(z)


(w <- c(rep("yes", 3), rep("nope", 7)))

prevalence(w, event = "yes")
```

```
# calculate prevalence from a model object:

data(rotif.mods)

prevalence(mod = rotif.mods$models[[1]])
```

---

range01 *Shrink or stretch a vector to make it range between 0 and 1*

Description

This function re-scales a numeric vector so that it ranges between 0 and 1. So, the lowest value becomes 0, the highest becomes 1, and the ones in the middle retain their rank and relative diference.

Usage

```
range01(x, na.rm = TRUE)
```

Arguments

x	a numeric vector.
na.rm	logical, whether to remove NA values.

Details

This function was borrowed from http://stackoverflow.com/questions/5468280/scale-a-series-between-two-points-in-r/5468527#5468527 and adapted to handle also missing values.

Value

A numeric vector of the same length as the input, now with the values ranging from 0 to 1.

Author(s)

A. Marcia Barbosa

See Also

standard01

Examples

```
range01(0:10)

range01(-12.3 : 21.7)
```

`rotif.mods` *Rotifer distribution models*

Description

A set of generalized linear models of rotifer species distributions on TDWG level 4 regions of the world (Fontaneto et al. 2012), together with their predicted values. Mind that these models are provided just as sample data and have limited application, due to limitations in the underlying distribution records. See Details for more information.

Usage

```
data(rotif.mods)
```

Format

A list of 2 elements:

$ predictions: a data.frame with 291 observations of 60 variables, namely the presence probability (P) and environmental favourability (F) for each of 30 species of rotifers, obtained from the rotif.env dataset in the 'fuzzySim' R-Forge package

$ models: a list of the 30 generalized linear model (`glm`) objects which generated those predictions.

Details

These models were obtained with the 'multGLM' function and the `rotif.env` dataset from R-Forge package 'fuzzySim' using the following code:

require(fuzzySim)

data(rotif.env)

rotif.mods <- multGLM(data = rotif.env, sp.cols = 18:47, var.cols = 5:17, step = FALSE, trim = TRUE)

See package 'fuzzySim' (currently available on R-Forge at http://fuzzysim.r-forge.r-project.org) for more information on the source data that were used to build these models.

References

Fontaneto D., Barbosa A.M., Segers H. & Pautasso M. (2012) The 'rotiferologist' effect and other global correlates of species richness in monogonont rotifers. Ecography, 35: 174-182.

Examples

```
data(rotif.mods)
head(rotif.mods$predictions)
rotif.mods$models[[1]]
```

RsqGLM *R-squared measures for GLMs*

Description

This function calculates some (pseudo) R-squared statistics for binomial Generalized Linear Models.

Usage

```
RsqGLM(model = NULL, obs = NULL, pred = NULL, use = "pairwise.complete.obs",
plot = TRUE, ...)
```

Arguments

model	a model object of class "glm".
obs	a vector of observed presences (1) and absences (0) or another binary response variable. Not necessary (and ignored) if model is provided.
pred	a vector with the corresponding predicted values of presence probability. Must be of the same length and in the same order as obs. Not necessary (and ignored) if model is provided.
use	argument to be passed to cor for handling mising values.
plot	logical value indicating whether or not to display a barplot of the calculated measures.
...	additional arguments to pass to the plot function (see Examples).

Details

Implemented measures include the R-squareds of McFadden (1974), Cox-Snell (1989), Nagelkerke (1991, which corresponds to the corrected Cox-Snell, eliminating its upper bound), and Tjur (2009). See Allison (2014) for a brief review of these measures.

Value

The function returns a named list of the calculated R-squared values.

Note

Tjur's R-squared can only be calculated for models with binomial response variable; otherwise, NA will be returned.

Author(s)

A. Marcia Barbosa

References

Allison P. (2014) Measures of fit for logistic regression. SAS Global Forum, Paper 1485-2014

Cox, D.R. & Snell E.J. (1989) The Analysis of Binary Data, 2nd ed. Chapman and Hall, London

McFadden, D. (1974) Conditional logit analysis of qualitative choice behavior. In: Zarembka P. (ed.) Frontiers in Economics. Academic Press, New York

Nagelkerke, N.J.D. (1991) A note on a general definition of the coefficient of determination. Biometrika, 78: 691-692

Tjur T. (2009) Coefficients of determination in logistic regression models - a new proposal: the coefficient of discrimination. The American Statistician, 63: 366-372.

See Also

Dsquared, AUC, threshMeasures, HLfit

Examples

```
# load sample models:
data(rotif.mods)

# choose a particular model to play with:
mod <- rotif.mods$models[[1]]

RsqGLM(model = mod)


# you can also use RsqGLM with vectors of observed and predicted values
# instead of a model object:

RsqGLM(obs = mod$y, pred = mod$fitted.values)


# plotting arguments can be modified:

RsqGLM(obs = mod$y, pred = mod$fitted.values, col = "seagreen", border = NA,
ylim = c(0, 1), las = 2, main = "Pseudo-R-squared values")
```

---

standard01 *Standardize to 0-1 (or vice-versa)*

Description

This function converts the score of a measure that ranges from -1 to 1 (e.g. a kappa or TSS value obtained for a model) into its (linearly) corresponding value in 0-to-1 scale, so that it can be compared directly with measures that range between 0 and 1 (such as CCR or AUC). It can also perform the conversion in the opposite direction.

Usage

```
standard01(score, direction = c("-1+1to01", "01to-1+1"))
```

Arguments

score	numeric value indicating the score of the measure of interest.
direction	character value indicating the direction in which to perform the standardization. The default, "-1+1to01", can be switched to "01to-1+1".

Details

While most of the threshold-based measures of model evaluation range theoretically from 0 to 1, some of them (such as Cohen's kappa and the true skill statistic, TSS) may range from -1 to 1 (Allouche et al. 2006). Thus, the values of different measures may not be directly comparable (Barbosa 2015). We do not usually get negative values of TSS or kappa (nor values under 0.5 for CCR or AUC, for example) because that only happens when model predictions perform worse than random guesses; still, such values are mathematically possible, and can occur e.g. when extrapolating models to regions where where the species-environment relationships differ. This standardization is included as an option in the threshMeasures function.

Value

The numeric value of 'score' when re-scaled to the 0-to-1 (or to the -1 to +1) scale.

Note

Note that this is not the same as re-scaling a vector so that it ranges between 0 and 1, which is done by range01.

Author(s)

A. Marcia Barbosa

References

Allouche O., Tsoar A. & Kadmon R. (2006) Assessing the accuracy of species distribution models: prevalence, kappa and the true skill statistic (TSS). Journal of Applied Ecology 43: 1223-1232

Barbosa, A.M. (2015) Re-scaling of model evaluation measures to allow direct comparison of their values. The Journal of Brief Ideas, 18 Feb 2015, DOI: 10.5281/zenodo.15487

See Also

threshMeasures, range01

Examples

```
standard01(0.6)

standard01(0.6, direction = "-1+1to01")

standard01(0.6, direction = "01to-1+1")
```

`threshMeasures` *Threshold-based measures of model evaluation*

Description

This function calculates a number of measures for evaluating the classification accuracy of a species distribution (or ecological niche, or bioclimatic envelope...) model against observed presence-absence data (Fielding & Bell 1997; Liu et al. 2011; Barbosa et al. 2013), upon the choice of a threshold value above which the model is considered to predict that the species should be present.

Usage

```
threshMeasures(model = NULL, obs = NULL, pred = NULL, thresh,
measures = modEvAmethods("threshMeasures"), simplif = FALSE,
plot = TRUE, plot.ordered = FALSE, standardize = TRUE,
verbosity = 2, ylim = c(0, 1), ...)
```

Arguments

`model`	a binary-response model object of class "glm", "gam", "gbm", "randomForest" or "bart".
`obs`	a vector of observed presences (1) and absences (0) or another binary response variable. Not necessary (and ignored) if 'model' is provided.
`pred`	a vector with the corresponding predicted values of presence probability, habitat suitability, environmental favourability or alike. Not necessary (and ignored) if 'model' is provided.
`thresh`	numeric value of the threshold to separate predicted presences from predicted absences in 'pred'; can be "preval", to use the prevalence of 'obs' as the threshold, or any real number between 0 and 1. See Details for an informed choice.
`measures`	character vector of the evaluation measures to use. By default, all measures available in 'modEvAmethods("threshMeasures")' are calculated.
`simplif`	logical, whether to calculate a faster, simplified version. Used internally by other functions in the package. Defaults to FALSE.
`plot`	logical, whether to produce a barplot of the calculated measures. Defaults to TRUE.
`plot.ordered`	logical, whether to plot the measures in decreasing order rather than in input order. Defaults to FALSE.

standardize	logical, whether to change measures that may range between -1 and +1 (namely kappa and TSS) to their corresponding value in the 0-to-1 scale (skappa and sTSS), so that they can compare directly to other measures (see standard01). The default is TRUE, but a message is displayed to inform the user about it.
verbosity	integer specifying the amount of messages to display. Defaults to the maximum implemented; lower numbers (down to 0) decrease the number of messages.
ylim	limits for the y axis.
...	additional arguments to be passed to the plot function.

Details

The threshold value can be chosen according to a number of criteria (see e.g. Liu et al. 2005, Jimenez-Valverde & Lobo 2007, Nenzen & Araujo 2011). You can set 'thresh' to "preval" (species' prevalence or proportion of presences **in the data input to this function**), or calculate optimal threshold values according to different criteria with the optiThresh or the optiPair function. If you are using "environmental favourability" as input 'pred' data (Real et al. 2006; see 'Fav' function in R package 'fuzzySim'), then the 0.5 threshold equates to using training prevalence in logistic regression (GLM with binomial error distribution and logit link function).

While most of these threshold-based measures range from 0 to 1, some of them (such as kappa and TSS) may range from -1 to 1 (Allouche et al. 2006), so their raw scores are not directly comparable. 'threshMeasures' includes an option (used by default) to standardize these measures to 0-1 (Barbosa 2015) using the standard01 function, so that you obtain the standardized versions skappa and sTSS.

This function can also be used to calculate the agreement between different presence-absence (or other types of binary) data, as e.g. Barbosa et al. (2012) did for comparing mammal distribution data from atlas and range maps. Notice, however, that some of these measures, such as TSS or NMI, are not symmetrical (obs vs. pred is different from pred vs. obs).

Value

If 'simplif = TRUE', the output is a numeric matrix with the name and value of each measure. If 'simplif = FALSE' (the default), the ouptut is a bar plot of the calculated measures and a list with the following components:

N	the number of observations (records) in the analysis.
Prevalence	the prevalence (proportion of presences) in 'obs'.
Threshold	the threshold value used to calculate the 'measures'.
ConfusionMatrix	the confusion matrix obtained with the used threshold.
ThreshMeasures	a numeric matrix with the name and value of each measure.

Note

"Sensitivity" is the same as "Recall", and "PPP" (positive predictive power) is the same as "Precision". "F1score"" is the harmonic mean of precision and recall.

Note

"Sensitivity" is the same as "Recall", and "PPP" (positive predictive power) is the same as "Precision". Some of these measures (like NMI, UPR, OPR, PPP, NPP) cannot be calculated for thresholds at which there are zeros in the confusion matrix, so they can yield NaN values.

Author(s)

A. Marcia Barbosa

References

Allouche O., Tsoar A. & Kadmon R. (2006) Assessing the accuracy of species distribution models: prevalence, kappa and the true skill statistic (TSS). Journal of Applied Ecology 43: 1223-1232.

Barbosa, A.M. (2015) Re-scaling of model evaluation measures to allow direct comparison of their values. The Journal of Brief Ideas, 18 Feb 2015, DOI: 10.5281/zenodo.15487

Barbosa A.M., Estrada A., Marquez A.L., Purvis A. & Orme C.D.L. (2012) Atlas versus range maps: robustness of chorological relationships to distribution data types in European mammals. Journal of Biogeography 39: 1391-1400

Barbosa A.M., Real R., Munoz A.R. & Brown J.A. (2013) New measures for assessing model equilibrium and prediction mismatch in species distribution models. Diversity and Distributions 19: 1333-1338

Fielding A.H. & Bell J.F. (1997) A review of methods for the assessment of prediction errors in conservation presence/absence models. Environmental Conservation 24: 38-49

Jimenez-Valverde A. & Lobo J.M. (2007) Threshold criteria for conversion of probability of species presence to either-or presence-absence. Acta Oecologica 31: 361-369

Liu C., Berry P.M., Dawson T.P. & Pearson R.G. (2005) Selecting thresholds of occurrence in the prediction of species distributions. Ecography 28: 385-393.

Liu C., White M. & Newell G. (2011) Measuring and comparing the accuracy of species distribution models with presence-absence data. Ecography 34: 232-243.

Nenzen H.K. & Araujo M.B. (2011) Choice of threshold alters projections of species range shifts under climate change. Ecological Modelling 222: 3346-3354

Real R., Barbosa A.M. & Vargas J.M. (2006) Obtaining environmental favourability functions from logistic regression. Environmental and Ecological Statistics 13: 237-245.

See Also

optiThresh, optiPair, AUC

Examples

```
# load sample models:
data(rotif.mods)

# choose a particular model to play with:
mod <- rotif.mods$models[[1]]
```

```
threshMeasures(model = mod, simplif = TRUE, thresh = 0.5)

threshMeasures(model = mod, thresh = "preval")

threshMeasures(model = mod, plot.ordered = TRUE, thresh = "preval")

threshMeasures(model = mod, measures = c("CCR", "TSS", "kappa"),
thresh = "preval")

threshMeasures(model = mod, plot.ordered = TRUE, thresh = "preval")


# you can also use threshMeasures with vectors of observed and
# predicted values instead of with a model object:

threshMeasures(obs = mod$y, pred = mod$fitted.values, thresh = "preval")
```

`varPart` *Variation partitioning*

Description

This function performs variation partitioning (Borcard et al. 1992) among two factors (e.g. Ribas et al. 2006) or three factors (e.g. Real et al. 2003) for either multiple linear regression models (LM) or generalized linear models (GLM).

Usage

```
varPart(A, B, C = NA, AB, AC = NA, BC = NA, ABC = NA,
model.type = NULL, A.name = "Factor A", B.name = "Factor B",
C.name = "Factor C", plot = TRUE, plot.digits = 3, cex.names = 1.5,
cex.values = 1.2, main = "", cex.main = 2, plot.unexpl = TRUE,
coloured = FALSE)
```

Arguments

`A`	numeric value of the R-squared of the regression of the response variable on the variables related to factor 'A'
`B`	numeric value of the R-squared of the regression of the response variable on the variables related to factor 'B'
`C`	(optionally, if there are 3 factors) numeric value of the R-squared of the regression of the response on the variables related to factor 'C'
`AB`	numeric value of the R-squared of the regression of the response on the variables of factors 'A' and 'B' simultaneously
`AC`	(if there are 3 factors) numeric value of the R-squared of the regression of the response on the variables of factors 'A' and 'C' simultaneously

BC	(if there are 3 factors) numeric value of the R-squared of the regression of the response on the variables of factors 'B' and 'C' simultaneously
ABC	(if there are 3 factors) numeric value of the R-squared of the regression of the response on the variables of factors 'A', 'B' and 'C' simultaneously
model.type	deprecated argument, kept here for back-compatibility
A.name	character string indicating the name of factor 'A'
B.name	character string indicating the name of factor 'B'
C.name	character string indicating the name of factor 'C' (if there are 3 factors)
plot	logical, whether to plot the variation partitioning diagram. The default is TRUE.
plot.digits	integer value of the number of digits to which to round the values in the plot. The default is 3.
cex.names	numeric value indicating character expansion factor to define the size of the names of the factors displayed in the plot.
cex.values	numeric value indicating character expansion factor to define the size of the values displayed in the plot.
main	optional character string indicating the main title for the plot. The default is empty.
cex.main	numeric value indicating character expansion factor to define the font size of the plot title (if provided).
plot.unexpl	logical value indicating whether the amount of unexplained variation should be included in the plot. The default is TRUE.
coloured	logical value indicating whether or not to colour the circles in the plot. The default is FALSE for back-compatibility.

Details

If you have linear models, input data for 'varPart' are the coefficients of determination (R-squared values) of the linear regressions of the target variable on all the variables in the model, on the variables related to each particular factor, and (when there are 3 factors) on the variables related to each pair of factors. The outputs are the amounts of variance explained exclusively by each factor, the amounts explained exclusively by the overlapping effects of each pair of factors, and the amount explained by the overlap of the 3 factors if this is the case (e.g. Real et al. 2003). The amount of variation not explained by the complete model is also provided.

If you have generalized linear models (GLMs) such as logistic regression (see glm), you have no true R-squared values; inputs can then be the squared coefficients of correlation between the model predictions given by each factor (or pair of factors) and the predictions of the complete model (e.g. Munoz & Real 2006), or the R-squared values of the corresponding logit (y) functions (Real et al. 2013), or an adjusted R-squared (De Araujo et al. 2013). In these cases, the "total variation" (AB or ABC, depending on whether you have two or three factors) is 1 (correlation of the predictions of the complete model with themselves), and output values are not the total amounts of variance (of the target variable) explained by factors and overlaps, but rather their proportional contribution to the total variation explained by the model.

Value

The output consists of a data frame indicating the proportion of variance accounted for by each of the factors, and (if 'plot = TRUE') a Venn diagram of the contributions of each factor.

Note

These results derive from arithmetic operations between your input values, and they always sum up to 1; if your input is incorrect, the results will be incorrect as well, even if they sum up to 1.

This function had a bug up to modEvA version 0.8: a badly placed line break prevented the ABC overlap from being calculated correctly. Thanks to Jurica Levatic for pointing this out and helping to solve it!

Oswald van Ginkel also suggested a fix to some plotting awkwardness when using only two factors, and a nice option for colouring the plot. Many thanks!

Author(s)

A. Marcia Barbosa

References

Borcard D., Legendre P., Drapeau P. (1992) Partialling out the spatial component of ecological variation. Ecology 73: 1045-1055

De Araujo C.B., Marcondes-Machado L.O. & Costa G.C. (2013) The importance of biotic interactions in species distribution models: a test of the Eltonian noise hypothesis using parrots. Journal of Biogeography, early view (DOI: 10.1111/jbi.12234)

Munoz A.-R. & Real R. (2006) Assessing the potential range expansion of the exotic monk parakeet in Spain. Diversity and Distributions 12: 656-665.

Real R., Barbosa A.M., Porras D., Kin M.S., Marquez A.L., Guerrero J.C., Palomo L.J., Justo E.R. & Vargas J.M. (2003) Relative importance of environment, human activity and spatial situation in determining the distribution of terrestrial mammal diversity in Argentina. Journal of Biogeography 30: 939-947.

Real R., Romero D., Olivero J., Estrada A. & Marquez A.L. (2013) Estimating how inflated or obscured effects of climate affect forecasted species distribution. PLoS ONE 8: e53646.

Ribas A., Barbosa A.M., Casanova J.C., Real R., Feliu C. & Vargas J.M. (2006) Geographical patterns of the species richness of helminth parasites of moles (Talpa spp.) in Spain: separating the effect of sampling effort from those of other conditioning factors. Vie et Milieu 56: 1-8.

Examples

```
# if you have a linear model (LM), use (non-adjusted) R-squared values
# for each factor and for their combinations as inputs:

# with 2 factors:

varPart(A = 0.456, B = 0.315, AB = 0.852, A.name = "Spatial",
B.name = "Environmental", main = "Small whale")
```

```
varPart(A = 0.456, B = 0.315, AB = 0.852, A.name = "Spatial",
B.name = "Environmental", main = "Small whale", col = TRUE)


# with 3 factors:

varPart(A = 0.456, B = 0.315, C = 0.281, AB = 0.051, BC = 0.444,
AC = 0.569, ABC = 0.624, A.name = "Spatial", B.name = "Human",
C.name = "Environmental", main = "Small whale")

varPart(A = 0.456, B = 0.315, C = 0.281, AB = 0.051, BC = 0.444,
AC = 0.569, ABC = 0.624, A.name = "Spatial", B.name = "Human",
C.name = "Environmental", main = "Small whale", col = TRUE)


# if you have a generalized linear model (GLM),
# you can use squared correlation coefficients of the
# predictions of each factor with those of the complete model:

varPart(A = (-0.005)^2, B = 0.698^2, C = 0.922^2, AB = 0.696^2,
BC = 0.994^2, AC = 0.953^2, ABC = 1, A.name = "Topographic",
B.name = "Climatic", C.name = "Geographic", main = "Big bird")

# but "Unexplained variation" can be deceiving in these cases
# (see Details); try also adding 'plot.unexpl = FALSE'
```

Index

∗Topic **datasets**
rotif.mods, 44
∗Topic **package**
modEvA-package, 2

arrangePlots, 3
AUC, 5, *46*, *50*

confusionLabel, 8
cor, *45*

density, *38, 39*
Dsquared, 10, *25*, *36*, *46*

evaluate, 11
evenness, 13, *42*

getBins, 14, *18–20*, *27, 28*
getModEqn, 16
glm, *11*, *36*, *44*, *52*

hist, *38, 39*
HLfit, *15, 16*, 18, *25*, *28, 29*, *46*

jitter, *40*

layout, *4*

MESS, 21, *30, 31*
MillerCalib, *20*, 24, *28*
mod2obspred, 26
modEvA *(*modEvA-package*)*, 2
modEvA-package, 2
modEvAmethods, 27
multModEv, *27*, 28

OA, *23*, 30
optiPair, 31, *35*, *49, 50*
optiThresh, *27*, *33*, 34, *49, 50*

plot, *4*, *6*, *19*, *24*, *32*, *34*, *37*, *45*, *49*
plotGLM, *11*, 36, *40*
points, *40*
predDensity, *37*, 38, *40*
predPlot, *37*, *39*, 39
prevalence, *13*, *26*, *40*, 41

quantile, *15*, *19*

range01, 43, *47*
rotif.mods, 44
round, *6*, *36*, *52*
RsqGLM, *25*, *36*, 45

set.seed, *40*
standard01, *28*, *43*, 46, *49*

threshMeasures, *8, 9*, *12, 13*, *27–29*, *33–35*, *46, 47*, 48

varPart, 51