Package 'predicts'

May 6, 2025

Type Package

Title Spatial Prediction Tools

Description
Methods for spatial predictive modeling, especially for spatial distribution models. This includes algorithms for model fitting and prediction, as well as methods for model evaluation.

Version 0.1-19

Date 2025-05-06

Depends R (>= 3.5.0), methods, terra

Encoding UTF-8

Suggests disdat, rJava

Maintainer Robert J. Hijmans <r.hijmans@gmail.com>

License GPL (>= 3)

LazyLoad yes

URL https://rspatial.org/sdm/

BugReports https://github.com/rspatial/predicts/issues/

NeedsCompilation no

Author Robert J. Hijmans [cre, aut] (ORCID:
<https://orcid.org/0000-0001-5872-2872>),
Steven Phillips [ctb],
Márcia Barbosa [ctb],
Chris Brunsdon [ctb],
Barry Rowlingson [ctb]

Repository CRAN

Date/Publication 2025-05-06 13:10:02 UTC

Contents

`predicts-package` *Spatial prediction*

Description

This package implements functions for spatial predictions methods, especially spatial (species) distribution models, including an R link to the 'maxent' model.

Author(s)

Robert J. Hijmans

`backgroundSample` *Random points*

Description

Generate random points that can be used to extract background values ("random-absence"). The points are sampled (without replacement) from the cells that are not 'NA' in raster 'mask'.

If the coordinate reference system (of mask) is longitude/latitude, sampling is weighted by the size of the cells. That is, because cells close to the equator are larger than cells closer to the poles, equatorial cells have a higher probability of being selected.

Usage

```
backgroundSample(mask, n, p, ext=NULL, extf=1.1, excludep=TRUE,
            cellnumbers=FALSE, tryf=3, warn=2)
```

Arguments

mask	SpatRaster. If the object has cell values, cells with NA are excluded (of the first layer of the object if there are multiple layers)
n	integer. Number of points
p	Presence points (if provided, random points won't be in the same cells (as defined by mask)
ext	SpatExtent. Can be used to restrict sampling to a spatial extent
extf	numeric. Multiplyer to adjust the size of extent 'ext'. The default increases of 1.1 increases the extent a little (5% at each side of the extent)
excludep	logical. If TRUE, presence points are exluded from background
cellnumbers	logical. If TRUE, cell numbers for mask are returned rather than coordinates
tryf	numeric > 1. Multiplyer used for initial sample size from which the requested sample size is extracted after removing NA points (outside of mask)
warn	integer. 2 or higher gives most warnings. 0 or lower gives no warnings if sample size n is not reached

Value

matrix with coordinates, or, if cellnumbers=TRUE, a vector with cell numbers.

---

biovars *bioclimatic variables*

Description

Function to create 'bioclimatic variables' from monthly climate data.

Usage

```
## S4 method for signature 'SpatRaster,SpatRaster,SpatRaster'
bcvars(prec, tmin, tmax, filename="", ...)

## S4 method for signature 'numeric,numeric,numeric'
bcvars(prec, tmin, tmax)

## S4 method for signature 'matrix,matrix,matrix'
bcvars(prec, tmin, tmax)
```

Arguments

prec	numeric vector (12 values), matrix (12 columns), or SpatRaster with monthly (12 layers) precipitation data
tmin	same as prec
tmax	same as prec
filename	character. Output filename
...	additional arguments for writing files as in writeRaster

Details

Input data is normally monthly. I.e. there should be 12 values (layers) for each variable, but the function should also work for e.g. weekly data (with some changes in the meaning of the output variables. E.g. #8 would then not be for a quarter (3 months), but for a 3 week period).

Value

Same class as input, but 19 values/variables

bio1 = Mean annual temperature

bio2 = Mean diurnal range (mean of max temp - min temp)

bio3 = Isothermality (bio2/bio7) (* 100)

bio4 = Temperature seasonality (standard deviation *100)

bio5 = Max temperature of warmest month

bio6 = Min temperature of coldest month

bio7 = Temperature annual range (bio5-bio6)

bio8 = Mean temperature of the wettest quarter

bio9 = Mean temperature of driest quarter

bio10 = Mean temperature of warmest quarter

bio11 = Mean temperature of coldest quarter

bio12 = Total (annual) precipitation

bio13 = Precipitation of wettest month

bio14 = Precipitation of driest month

bio15 = Precipitation seasonality (coefficient of variation)

bio16 = Precipitation of wettest quarter

bio17 = Precipitation of driest quarter

bio18 = Precipitation of warmest quarter

Examples

```
tmin <- c(10,12,14,16,18,20,22,21,19,17,15,12)
tmax <- tmin + 5
prec <- c(0,2,10,30,80,160,80,20,40,60,20,0)
bcvars(prec, tmin, tmax)

tmn <- tmx <- prc <- rast(nrow=1, ncol=1, nlyr=12)
values(tmn) <- t(matrix(c(10,12,14,16,18,20,22,21,19,17,15,12)))
tmx <- tmn + 5
values(prc) <- t(matrix(c(0,2,10,30,80,160,80,20,40,60,20,0)))
b <- bcvars(prc, tmn, tmx)
as.matrix(b)
```

`cm_evaluate` *Model evaluation with a confusion matrix*

Description

Get model evaluation statistics from a confusion matrix. This is useful when predicting (multiple) classes.

Usage

```
cm_evaluate(cmat, stat="overall")
```

Arguments

`cmat`	confusion matrix. Normally created with table (see examples)
`stat`	character. Either "overall" (overall accuracy), "kappa", "class" for user and producer accuracy

Value

numeric

See Also

`pa_evaluate`

Examples

```
classes <- c("forest", "water", "urban", "agriculture")
set.seed(1)
observed <- sample(classes, 100, replace=TRUE)
predicted <- observed
i <- seq(1,100,2)
predicted[i] <- sample(classes, length(i), replace=TRUE)
conmat <- table(observed, predicted)
conmat

cm_evaluate(conmat, "kappa")
cm_evaluate(conmat, "class")
```

divide_polygons *Divide polygons into equal area parts*

Description

stripper divides polygons into horizontal or vertical strips of a specified relative size.

divider divides a SpatVector of polygons into n compact and approximately equal area parts. The results are not deterministic so you should use set.seed to be able to reproduce your results. If you get a warning about non-convergence, you can increase the number of iterations used with additional argument iter.max

Usage

```
divider(x, n, env=NULL, alpha=1, ...)
stripper(x, f=c(1/3, 2/3), vertical=TRUE)
```

Arguments

x	SpatVector of polygons
n	positive integer. The number of parts requested
env	SpatRaster with environmental data
alpha	numeric. One or two numbers that act as weights for the x and y coordinates
...	additional arguments such as iter.max passed on to kmeans
f	numeric vector of fractions. These must be > 0 and < 1, and in ascending order
vertical	logical. If TRUE the strips are vertical

Value

SpatVector

Author(s)

stripper was derived from a function by Barry Rowlingson

Examples

```
f <- system.file("ex/lux.shp", package="terra")
v <- aggregate(vect(f))
set.seed(33)
d1 <- divider(v, 10)
plot(d1)

d2 <- divider(v, 100)
boxplot(expanse(d2, "km"))

x <- stripper(v, seq(0.1, 0.9, 0.1))
```

```
round(expanse(x,"km"), 1)
plot(x, col=rainbow(12))
```

`envelope` *Fit a (climate) envelope model and make predictions*

Description

The envelope algorithm has been extensively used for species distribution modeling under the name "bioclim model". This is the classic 'climate-envelope-model' that started what was later called species distribution modeling and ecological niche modeling. Although it generally does not perform as good as some other methods (Elith et al. 2006) and is unsuited for predicting climate change effects (Hijmans and Graham, 2006). It may be useful in certain cases, among other reasons because the algorithm is easy to understand and thus useful in teaching species distribution modeling.

The algorithm computes the similarity of a location by comparing the values of environmental variables at any location to a percentile distribution of the values at known locations of occurrence ('training sites'). The closer to the 50th percentile (the median), the more suitable the location is. The tails of the distribution are not distinguished, that is, 10 percentile is treated as equivalent to 90 percentile.

In this R implementation, percentile scores are between 0 and 1, but predicted values larger than 0.5 are subtracted from 1. Then, the minimum percentile score across all the environmental variables is computed (i.e. this is like Liebig's law of the minimum, except that high values can also be limiting factors). The final value is subtracted from 1 and multiplied with 2 so that the results are between 0 and 1. The reason for this transformation is that the results become more like that of other distribution modeling methods and are thus easier to interpret. The value 1 will rarely be observed as it would require a location that has the median value of the training data for all the variables considered. The value 0 is very common as it is assigned to all cells with a value of an environmental variable that is outside the percentile distribution (the range of the training data) for at least one of the variables.

When using the predict function you can choose to ignore one of the tails of the distribution (for example, to make low rainfall a limiting factor, but not high rainfall).

Usage

```
envelope(x, ...)
```

Arguments

`x`	matrix or data.frame where each column is an environmental variable and each row an occurrence. Alternatively, a SpatRaster where each layer is an environmental variable, in which case you must also provide argument 'p': a SpatVector of the occurrence points, or a data.frame of their spatial coordinates. Only grid cells that overlap with occurrences are used.
`...`	Additional arguments

Value

An object of class 'envelope_model'

Author(s)

Robert J. Hijmans

References

Nix, H.A., 1986. A biogeographic analysis of Australian elapid snakes. In: Atlas of Elapid Snakes of Australia. (Ed.) R. Longmore, pp. 4-15. Australian Flora and Fauna Series Number 7. Australian Government Publishing Service: Canberra.

Booth, T.H., H.A. Nix, J.R. Busby and M.F. Hutchinson, 2014. BIOCLIM: the first species distribution modelling package, its early applications and relevance to most current MAXENT studies. Diversity and Distributions 20: 1-9

Elith, J., C.H. Graham, R.P. Anderson, M. Dudik, S. Ferrier, A. Guisan, R.J. Hijmans, F. Huettmann, J. Leathwick, A. Lehmann, J. Li, L.G. Lohmann, B. Loiselle, G. Manion, C. Moritz, M. Nakamura, Y. Nakazawa, J. McC. Overton, A.T. Peterson, S. Phillips, K. Richardson, R. Scachetti-Pereira, R. Schapire, J. Soberon, S. Williams, M. Wisz and N. Zimmerman, 2006. Novel methods improve prediction of species' distributions from occurrence data. Ecography 29: 129-151. doi:10.1111/j.2006.09067590.04596.x

Hijmans R.J., and C.H. Graham, 2006. Testing the ability of climate envelope models to predict the effect of climate change on species distributions. Global change biology 12: 2272-2281. doi:10.1111/j.13652486.2006.01256.x

See Also

predict, maxent

Examples

```
# file with presence points
fsp <- system.file("/ex/bradypus.csv", package="predicts")
occ <- read.csv(fsp)[,-1]

#predictors
f <- system.file("ex/bio.tif", package="predicts")
preds <- rast(f)[[c(1,7,9)]]

v <- extract(preds, occ)
bc <- envelope(v[,-1])

d <- preds[18324:18374]
predict(bc, d)

p1 <- predict(bc, preds)
p2 <- predict(bc, preds, tails=c("both", "low", "high"))
```

folds *Make folds for k-fold partitioning*

Description

k-fold partitioning of a data set for model testing purposes. Each record in a matrix (or similar data structure) is randomly assigned to a group. Group numbers are between 1 and k. The function assures that each fold has the same size (or as close to that as possible).

Usage

```
folds(x, k=5, by)
```

Arguments

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

Value

a vector with group assignments

Author(s)

Robert J. Hijmans

Examples

```
library(disdat)
train <- disPo("NSW")
## a single species
srsp1 <- subset(train, spid=="nsw01")
folds(srsp1, k = 5)

## all species
k = folds(train, k=5, by=train$spid)

## each group has the same number of records
##(except for adjustments if the number of records
## divided by k is not an integer)

table(k[train$spid=="nsw01"])
```

`hull_model` *hull model*

Description

The hull model predicts that a species is present at sites inside the a hull that contains the training points, and is absent outside that circle.

The hull can be "convex", "circle", or "rectangle"

Usage

```
## S4 method for signature 'SpatVector'
hullModel(p, type="convex", n=1)

## S4 method for signature 'data.frame'
hullModel(p, type="convex", crs="", n=1)

## S4 method for signature 'matrix'
hullModel(p, type="convex", crs="", n=1)
```

Arguments

`p`	point locations (presence). Two column matrix, data.frame or SpatVector
`type`	character. The type of hull. One of "convex", "circle", or "rectangle"
`crs`	character. The coordinate reference system
`n`	positive integer. The number of hulls to make

Value

hull_model

Examples

```
r <- rast(system.file("ex/logo.tif", package="terra"))
#presence data
pts <- matrix(c(17, 42, 85, 70, 19, 53, 26, 84, 84, 46, 48, 85, 4,
    95, 48, 54, 66, 74, 50, 48, 28, 73, 38, 56, 43, 29, 63, 22, 46, 45,
    7, 60, 46, 34, 14, 51, 70, 31, 39, 26), ncol=2)
train <- pts[1:12, ]
test <- pts[13:20, ]

ch <- hullModel(train, crs="+proj=longlat")
predict(ch, test)

plot(r)
plot(ch, border="red", lwd=2, add=TRUE)
points(train, col="red", pch=20, cex=2)
```

```
points(test, col="black", pch=20, cex=2)

pr <- predict(ch, r)
plot(pr)
points(test, col="black", pch=20, cex=2)
points(train, col="red", pch=20, cex=2)

# to get the polygons:
p <- geometry(ch)
p
```

MaxEnt *MaxEnt*

Description

Build a "maxent" (Maximum Entropy) species distribution model (see references below). The function uses environmental data for locations of known presence and for a large number of 'background' locations. Environmental data can be extracted from raster files. The result is a model object that can be used to predict the suitability of other locations, for example, to predict the entire range of a species.

Background points are sampled randomly from the cells that are not NA in the first predictor variable, unless background points are specified with argument a.

This function uses the MaxEnt species distribution model software by Phillips, Dudik and Schapire.

Usage

```
## S4 method for signature 'SpatRaster,SpatVector'
MaxEnt(x, p, a=NULL, removeDuplicates=TRUE, nbg=10000, ...)

## S4 method for signature 'data.frame,numeric'
MaxEnt(x, p, args=NULL, path, silent=FALSE, ...)

## S4 method for signature 'missing,missing'
MaxEnt(x, p, silent=FALSE, ...)
```

Arguments

x	Predictors. Either a SpatRaster to extract values from for the locations in y; or a data.frame, in which case each column should be a predictor variable and each row a presence or background record. Either can include categorical variables (see as.factor)
p	If x is a SpatRaster: occurence data. This can be a data.frame, matrix, or SpatVector. If p is a data.frame or matrix it represents a set of point locations; and it must have two columns with the first being the x-coordinate (longitude) and the second the y-coordinate (latitude).

If x is a data.frame, p should be a vector with a length equal to nrow(x) and contain 0 (background) and 1 (presence) values, to indicate which records (rows) in data.frame x are presence records, and which are background records

a — Background points. Only used if p is not a vector and not missing

nbg — Number of background points to use. These are sampled randomly from the cells that are not NA in the first predictor variable. Ignored if background points are specified with argument a

args — character. Additional argument that can be passed to MaxEnt. See the MaxEnt help for more information. The R MaxEnt function only uses the arguments relevant to model fitting. There is no point in using args='outputformat=raw' when *fitting* the model; but you can use arguments relevant for *prediction* when using the predict function. Some other arguments do not apply at all to the R implementation. An example is 'outputfiletype', because the 'predict' function has its own 'filename' argument for that

removeDuplicates — Boolean. If TRUE, duplicate presence points (that fall in the same grid cell) are removed

path — character. Optional argument to set where you want the MaxEnt output files to be stored. This allows you to permanently keep these files. If not supplied the MaxEnt files will be stored in a temporary file. These are the files that are shown in a browser when typing the model name or when you use "show(model)"

silent — Boolean. If TRUE a message is printed

... — Additional arguments

Value

An object of class 'MaxEnt_model'. Or a 'MaxEnt_model_replicates' object if you use 'replicates=' as part of the args argument.

If the function is run without any arguments a boolean value is returned (TRUE if MaxEnt.jar was found).

Author(s)

Steven Phillips and Robert J. Hijmans

References

Steven J. Phillips, Miroslav Dudik, Robert E. Schapire, 2004. A maximum entropy approach to species distribution modeling. Proceedings of the Twenty-First International Conference on Machine Learning. p. 655-662.

Steven J. Phillips, Robert P. Anderson, Robert E. Schapire, 2006. Maximum entropy modeling of species geographic distributions. Ecological Modelling 190:231-259.

Jane Elith, Steven J. Phillips, Trevor Hastie, Miroslav Dudik, Yung En Chee, Colin J. Yates, 2011. A statistical explanation of MaxEnt for ecologists. Diversity and Distributions 17:43-57. doi:10.1111/j.14724642.2010.00725.x

See Also

predict

Examples

```
# test the MaxEnt version
MaxEnt()


# get predictor variables
ff <- list.files("tif$", path=system.file("ex", package="predicts"), full.names=TRUE)
preds <- rast(ff)
plot(preds)

# file with presence points
occurence <- system.file("/ex/bradypus.csv", package="predicts")
occ <- read.csv(occurence)[,-1]

# witholding a 20% sample for testing
fold <- folds(occ, k=5)
occtest <- occ[fold == 1, ]
occtrain <- occ[fold != 1, ]

# fit model
me <- MaxEnt(preds, occtrain)

# see the MaxEnt results in a browser:
me

# use "args"
me2 <- MaxEnt(preds, occtrain, args=c("-J", "-P"))

# plot showing importance of each variable
plot(me)

# predict to entire dataset
r <- predict(me, preds)

# with some options:
r <- predict(me, preds, args=c("outputformat=raw"))

plot(r)
points(occ)

#testing
# background sample
bg <- backgroundSample(preds, 1000)

#simplest way to use 'evaluate'
e1 <- pa_evaluate(me, p=occtest, a=bg, x=preds)

# alternative 1
```

```
# extract values
pvtest <- data.frame(extract(preds, occtest))
avtest <- data.frame(extract(preds, bg))

e2 <- pa_evaluate(me, p=pvtest, a=avtest)

# alternative 2
# predict to testing points
testp <- predict(me, pvtest)
head(testp)
testa <- predict(me, avtest)

e3 <- pa_evaluate(p=testp, a=testa)
e3
threshold(e3)

plot(e3, 'ROC')
```

`mess` *Multivariate environmental similarity surfaces (MESS)*

Description

Compute multivariate environmental similarity surfaces (MESS), as described by Elith et al., 2010

Usage

```
## S4 method for signature 'SpatRaster'
mess(x, v, full=FALSE, filename="", ...)

## S4 method for signature 'data.frame'
mess(x, v, full=FALSE)
```

Arguments

`x`	SpatRaster or data.frame
`v`	matrix or data.frame containing the reference values; each column should correspond to one layer of the SpatRaster object. If `x` is a SpatRaster, it can also be a SpatVector with reference locations (points)
`full`	logical. If `FALSE` a SpatRaster with the MESS values is returned. If `TRUE`, a SpatRaster is returned with `n` layers corresponding to the layers of the input SpatRaster and an additional layer with the MESS values
`filename`	character. Output filename (optional)
`...`	additional arguments as for `writeRaster`

Details

v can be obtained for a set of points using `extract` .

Value

SpatRaster (or data.frame) with layers (columns) corresponding to the input layers and an additional layer with the mess values (if `full=TRUE` and `nlyr(x) > 1`) or a SpatRaster (data.frame) with the MESS values (if `full=FALSE`).

Author(s)

Jean-Pierre Rossi, Robert Hijmans, Paulo van Breugel

References

Elith J., M. Kearney M., and S. Phillips, 2010. The art of modelling range-shifting species. Methods in Ecology and Evolution 1:330-342. doi:10.1111/j.2041210X.2010.00036.x

Examples

```
set.seed(9)
r <- rast(ncol=10, nrow=10)
r1 <- setValues(r, (1:ncell(r))/10 + rnorm(ncell(r)))
r2 <- setValues(r, (1:ncell(r))/10 + rnorm(ncell(r)))
r3 <- setValues(r, (1:ncell(r))/10 + rnorm(ncell(r)))
s <- c(r1,r2,r3)
names(s) <- c('a', 'b', 'c')
xy <- cbind(rep(c(10,30,50), 3), rep(c(10,30,50), each=3))
refpt <- extract(s, xy)

ms <- mess(s, refpt, full=TRUE)
plot(ms)

## Not run:
filename <- paste0(system.file(package="predicts"), "/ex/bradypus.csv")
bradypus <- read.table(filename, header=TRUE, sep=',')
bradypus <- bradypus[,2:3]

predfile <- paste0(system.file(package="predicts"), "/ex/bio.tif")
predictors <- rast(predfile)
reference_points <- extract(predictors, bradypus, ID=FALSE)
mss <- mess(x=predictors, v=reference_points, full=TRUE)

breaks <- c(-500, -50, -25, -5, 0, 5, 25, 50, 100)
fcol <- colorRampPalette(c("blue", "beige", "red"))
plot(mss[[10]], breaks=breaks, col=fcol(9), plg=list(x="bottomleft"))

## End(Not run)
```

`partialResponse` *Get partial response data*

Description

Get partial response data.

Usage

```
partialResponse(model, data, var=NULL, rng=NULL, nsteps=25, plot=TRUE, nr, nc, ...)
partialResponse2(model, data, var1, var2, var2levels, rng=NULL, nsteps=25, ...)
```

Arguments

`model`	a model object
`data`	data.frame with data for all model variables
`var`	character or positive integer to identify the variable(s) of interest in `data`. If this is `NULL`, the partial response is computed for all variables
`var1`	character or positive integer to identify the variable of interest in `data`
`var2`	character. A second variable of interest
`var2levels`	character. The levels of the second variable to consider
`rng`	optional vector of two numbers to set the range or the variable
`nsteps`	positive integer. Number of steps to consider for the variable
`plot`	logical. If `TRUE`, the responses are plotted
`nc`	positive integer. Optional. The number of columns to divide the plotting device in (when plotting multiple variables)
`nr`	positive integer. Optional. The number of rows to divide the plotting device in (when plotting multiple variables)
`...`	model specific additional arguments passed to `predict`

Value

list (invisible if `plot=TRUE`)

Examples

```
fsp <- system.file("/ex/bradypus.csv", package="predicts")
occ <- read.csv(fsp)[,-1]
f <- system.file("ex/bio.tif", package="predicts")
preds <- rast(f)[[c(1,7,9)]]
v <- extract(preds, occ, ID=FALSE)

bc <- envelope(v)

pr <- partialResponse(bc, data=v, var=c("bio1", "bio12"), nsteps=30)
str(pr)
```

pa_evaluate *Presence/absence Model evaluation*

Description

Evaluation of models with presence/absence data. Given a vector of presence and a vector of absence values, confusion matrices are computed for a sequence of thresholds, and model evaluation statistics are computed for each confusion matrix / threshold.

Usage

```
pa_evaluate(p, a, model=NULL, x=NULL, tr, ...)
```

Arguments

p	either (1) predictions for presence points (model is NULL); or (2) predictor values for presence points (model is not NULL, x is NULL; or locations for presence points (model and x are not NULL)
a	as above for absence or background points
model	A fitted model used to make predictions
x	SpatRaster used to extract predictor values from
tr	Optional. a vector of threshold values to use for computing the confusion matrices
...	Additional arguments passed on to predict(model,...)

Value

pa_ModelEvaluation object

details

A pa_ModelEvaluation object has the the following slots

presence: presence values used

absence: absence values used

confusion: confusion matrix for each threshold

stats: statistics that are not threshold dependent

tr_stats: statistics that are threshold dependent

thresholds: optimal thresholds to classify values into presence and absence

stats has the following values

np: number of presence points

na: number of absence points

`auc`: Area under the receiver operator (ROC) curve

`pauc`: p-value for the AUC (for the Wilcoxon test W statistic

`cor`: Correlation coefficient

`pcor`: p-value for correlation coefficient

`prevalence`: Prevalence

`ODP`: Overall diagnostic power

`tr_stats` has the following values

`tresholds`: vector of thresholds used to compute confusion matrices

`CCR`: Correct classification rate

`TPR`: True positive rate

`TNR`: True negative rate

`FPR`: False positive rate

`FNR`: False negative rate

`PPP`: Positive predictive power

`NPP`: Negative predictive power

`MCR`: Misclassification rate

`OR`: Odds-ratio

`kappa`: Cohen's kappa

`thresholds` has the following values

`max_kappa`: the threshold at which kappa is highest

`max_spec_sens`: the threshold at which the sum of the sensitivity (true positive rate) and specificity (true negative rate) is highest

`no_omission`: the highest threshold at which there is no omission

`prevalence`: modeled prevalence is closest to observed prevalence

`equal_sens_spec`: equal sensitivity and specificity

References

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

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

See Also

`cm_evaluate`

Examples

```
set.seed(0)
# p has the predicted values for 50 known cases (locations)
# with presence of the phenomenon (species)
p <- rnorm(50, mean=0.6, sd=0.3)
# a has the predicted values for 50 background locations (or absence)
a <- rnorm(50, mean=0.4, sd=0.4)

e <- pa_evaluate(p=p, a=a)
e

e@stats

plot(e, "ROC")
plot(e, "TPR")
plot(e, "boxplot")
plot(e, "density")

str(e)
```

---

plot *Plot predictor values*

Description

Plot predictor values for occurrence (presence and absence) data in a model object.

Usage

```
## S4 method for signature 'envelope_model,missing'
plot(x, a = 1, b = 2, p = 0.9, ocol="gray", icol="red", bcol="blue", cex=c(0.6, 0.6), ...)

## S4 method for signature 'MaxEnt_model,ANY'
plot(x, y, ...)
```

Arguments

x	model object
a	name or position of the variable to plot in the x axis
b	name or position of the variable to plot in the y axis
p	percentile for coloring the points in the plot and delimiting the envelope rectangle
ocol	color of the points outside the envelope
icol	color of the points inside the envelope
bcol	color of the envelope border
cex	size of the points outside and inside the envelope

y	not used
...	additional arguments. Not used

`predict` *Spatial model predictions*

Description

Make predictions with models defined in the `predicts` package

Usage

```
## S4 method for signature 'MaxEnt_model'
predict(object, x, ext=NULL, args="", filename="", ...)

## S4 method for signature 'envelope_model'
predict(object, x, tails=NULL, ext=NULL, filename="", ...)

## S4 method for signature 'hull_model'
predict(object, x, ext=NULL, mask=FALSE, filename="", ...)
```

Arguments

object	model defined in this package (e.g. "envelope_model" and "maxent_model")
x	data to predict to. Either a data.frame or a SpatRaster
tails	character. You can use this to ignore the left or right tail of the percentile distribution for a variable. If supplied, tails should be a character vector with a length equal to the number of variables used in the model. Valid values are "both" (the default), "low" and "high". For example, if you have a variable x with an observed distribution between 10 and 20 and you are predicting the bioclim value for a value of 25, the default result would be zero (outside of all observed values); but if you use tail='low', the high (right) tail is ignored and the value returned will be 1.
args	Pass *prediction* arguments (options) to the maxent software. See `maxent`
ext	`NULL` or a `SpatExtent` to limit the prediction to a sub-region of `x`
mask	logical. If `TRUE` areas that are `NA` in `x` are set to `NA` in the output
filename	character. Output filename
...	additional arguments for writing files as in `writeRaster`

Value

SpatRaster or vector (if `x` is a data.frame).

See Also

`predict` function in the "terra" package for spatial predictions with glm, randomForest, etc.

pwd_sample *Pair-wise distance sampling*

Description

Select pairs of points from two sets (without replacement) that have a similar distance to their nearest point in another set of points.

For each point in "fixed", a point is selected from "sample" that has a similar distance (as defined by threshold) to its nearest point in "reference" (note that these are likely to be different points in reference). The select point is either the nearest point nearest=TRUE, or a randomly select point nearest=FALSE that is within the threshold distance. If no point within the threshold distance is found in sample, the point in fixed is dropped.

Hijmans (2012) proposed this sampling approach to remove 'spatial sorting bias' from evaluation data used in cross-validation of presence-only species distribution models. In that context, fixed are the testing-presence points, sample the testing-absence (or testing-background) points, and reference the training-presence points.

Usage

```
pwd_sample(fixed, sample, reference, tr=0.33, nearest=TRUE, n=1, lonlat=TRUE, warn=TRUE)
```

Arguments

fixed	two column matrix (x, y) or (longitude/latitude) or SpatialPoints object, for point locations for which a pair should be found in sample
sample	as above for point locations from which to sample to make a pair with a point from fixed
reference	as above for reference point locations to which distances are computed
n	How many pairs do you want for each point in fixed
tr	Numeric, normally below 1. The threshold distance for a pair of points (one of fixed and one of sample) to their respective nearest points in reference to be considered a valid pair. The absolute difference in distance between the candidate point pairs in fixed and reference (dfr) and the distance between candidate point pairs in sample and reference (dsr) must be smaller than tr * dfr. I.e. if the dfr = 100 km, and tr = 0.1, dsr must be between >90 and <110 km to be considered a valid pair.
nearest	Logical. If TRUE, the pair with the smallest difference in distance to their nearest reference point is selected. If FALSE, a random point from the valid pairs (with a difference in distance below the threshold defined by tr) is selected (generally leading to higher SSB
lonlat	Logical. Use TRUE if the coordinates are spherical (in degrees), and use FALSE if they are planar
warn	Logical. If TRUE a warning is given if nrow(fixed) < nrow(sample)

Value

A matrix of nrow(fixed) and ncol(n), that indicates, for each point (row) in `fixed` which point(s) in `sample` it is paired to; or `NA` if no suitable pair was available.

References

Hijmans, R.J., 2012. Cross-validation of species distribution models: removing spatial sorting bias and calibration with a null-model. Ecology 93: 679-688

Examples

```
ref <- matrix(c(-54.5,-38.5, 2.5, -9.5, -45.5, 1.5, 9.5, 4.5, -10.5, -10.5), ncol=2)
fix <- matrix(c(-56.5, -30.5, -6.5, 14.5, -25.5, -48.5, 14.5, -2.5, 14.5,
               -11.5, -17.5, -11.5), ncol=2)
r <- rast()
ext(r) <- c(-110, 110, -45, 45)
r[] <- 1
set.seed(0)
sam <- spatSample(r, 50, xy=TRUE, as.points=TRUE)

plot(sam, pch='x')
points(ref, col='red', pch=18, cex=2)
points(fix, col='blue', pch=20, cex=2)

i <- pwd_sample(fix, sam, ref, lonlat=TRUE)
i
sfix <- fix[!is.na(i), ]
ssam <- sam[i[!is.na(i)], ]
ssam

plot(sam, pch='x', cex=0)
points(ssam, pch='x')
points(ref, col='red', pch=18, cex=2)
points(sfix, col='blue', pch=20, cex=2)

# try to get 3 pairs for each point in 'fixed'
pwd_sample(fix, sam, ref, lonlat=TRUE, n=3)
```

---

pycnophy *Pycnophylactic interpolation.*

Description

Given a `SpatVector` of polygons and population data for each polygon, compute a population density estimate based on Tobler's pycnophylactic interpolation algorithm.

Usage

```
pycnophy(x, v, pop, r = 0.2, converge = 3, verbose=FALSE)
```

Arguments

x	SpatRaster to interpolate to
v	SpatVector of polygons
pop	Either a character (name in v) or a numeric vector of length nrow(v)
r	A relaxation parameter for the iterative step in the pycnophylactic algorithm. Prevents over-compensation in the smoothing step. In practice the default value works well
converge	A convergence parameter, informing the decision on when iterative improvements on the smooth surface have converged sufficiently - see details
verbose	If TRUE the function report the maximum change in any grid cell value for each iterative step

Details

This method uses an iterative approach, and for each iteration notes the maximum change in a pixel. When this value falls below a certain level (10^(-converge) times the largest initial grid cell value) the iteration stops.

Value

SpatRaster

Note

Pycnophylatic interpolation has the property that the sum of the estimated values associated with all of the pixels in any polygon equals the supplied population for that polygon. A further property is that all pixel values are greater than or equal to zero. The method is generally used to obtain pixel-based population estimates when total populations for a set of irregular polygons (eg. counties) are known.

Author(s)

Chris Brunsdon (adapted for terra objects by Robert Hijmans)

References

Tobler, W.R. (1979) *Smooth Pycnophylactic Interpolation for Geographical Regions*. Journal of the American Statistical Association, v74(367) pp. 519-530.

Examples

```
f <- system.file("ex/lux.shp", package="terra")
v <- vect(f)
r <- rast(v, resolution = 0.01)
p <- pycnophy(r, v, "POP", converge=3, verbose=FALSE)
plot(p); lines(v)
```

RMSE *Root Mean Square Error*

Description

Compute the Root Mean Square Error (RMSE)

Usage

```
RMSE(obs, prd, na.rm=FALSE)

RMSE_null(obs, prd, na.rm=FALSE)
```

Arguments

obs	observed values
prd	predicted values
na.rm	logical. If TRUE, NAs are removed

Value

numeric

SDM *Class "SDM"*

Description

Parent class for a number of models defined in the predicts package. This is a virtual Class, no objects may be direclty created from it.

threshold *Find a threshold*

Description

Find a threshold (cut-off) to transform model predictions (probabilities, distances, or similar values) to a binary score (presence or absence).

Usage

```
## S4 method for signature 'paModelEvaluation'
threshold(x)
```

Arguments

x	paModelEvaluation object (see pa_evaluate

Value

data.frame with the following columns:

kappa: the threshold at which kappa is highest ("max kappa")

spec_sens: the threshold at which the sum of the sensitivity (true positive rate) and specificity (true negative rate) is highest

no_omission: the highest threshold at which there is no omission

prevalence: modeled prevalence is closest to observed prevalence

equal_sens_spec: equal sensitivity and specificity

Author(s)

Robert J. Hijmans and Diego Nieto-Lugilde

See Also

pa_evaluate

Examples

```
## See ?maxent for an example with real data.
# this is a contrived example:
# p has the predicted values for 50 known cases (locations)
# with presence of the phenomenon (species)
p <- rnorm(50, mean=0.7, sd=0.3)
# b has the predicted values for 50 background locations (or absence)
a <- rnorm(50, mean=0.4, sd=0.4)
e <- pa_evaluate(p=p, a=a)

threshold(e)
```

---

varImportance *Get variable importance*

Description

Get variable importance. The importance is expressed as the deterioration of the evaluation statistic. The statistic is computed n times for model predictions after randomizing a predictor variable and subtracting the statistic for the non-randomized data. The larger the difference, the more important the variable is.

Usage

```
varImportance(model, y, x, n=10, stat, value="relative", ...)
```

Arguments

model	a model object
y	the response variable used to fit the model. If missing, it is attempted to extract it from model. If that fails, it is computed from x. In the latter case the model would be assumed to have no error
x	data.frame with the predictor variables used to fit the model. If missing, it is attemted to extract it from model
n	positive integer. Number of simulations
stat	character. For models with a continuous response variable this can be one of "RMSE" (the default), "AUC", or "cor". See RMSE or pa_evaluate. For models with a categorical response variable this can be one of "overall" (overall accuracy, the default) or "kappa", see cm_evaluate
value	character specifying how to express the output. One of , "relative" (), "difference" (), "absolute" (no adjustments)
...	model specific additional arguments passed to predict

Value

named numeric vector

Examples

```
set.seed(1)
d <- data.frame(y=1:10, x1=runif(10), x2=runif(10))
m <- lm(y~., data=d)

varImportance(m, d[,1], d[,2:3])
```

Index