Package ‘dismo’

September 10, 2014

Type Package

Title Species distribution modeling

Version 1.0-5

Date 2014-9-9

Depends R (>= 2.13.0), methods, raster (>= 2.2-30), sp (>= 1.0-13)

Suggests rJava (>= 0.5-0), rgdal, rgeos, gbm, splines, XML, ROCR,deldir, gstat, randomForest, kernlab, maptools

Author Robert J. Hijmans, Steven Phillips, John Leathwick and Jane Elith

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

Description Functions for species distribution modeling, that is, predicting entire geographic distributions form occurrences at a number of sites.

License GPL (>= 3)

LazyLoad yes

NeedsCompilation yes

Repository CRAN

Date/Publication 2014-09-10 09:33:32

R topics documented:

`dismo-package` *Species distribution modeling*

Description

This package implements a few species distribution models, including an R link to the 'maxent' model, and native implementations of Bioclim and Domain. It also provides a number of functions that can assist in using Boosted Regresssion Trees.

A good place to start is the vignette, which you can access by typing `vignette('sdm', 'dismo')`

In addition there are a number of functions, such sampling background points, k-fold sampling, and for model evaluation (AUC) that are useful for these and for other species distribution modeling methods available in R (e.g. GLM, GAM, and RandomForest).

Author(s)

Robert J. Hijmans, Steven Phillips, John Leathwick and Jane Elith

---

`acaule` *Solanum acaule data*

Description

Distribution data for Solanum acaule (a plant species that occurs in the high Andes of Peru and Bolivia). Downloaded from GBIF with the `gbif` function. For use in the 'species distribution modeling' vignette.

Usage

```
data(acaule)
```

References

`http://www.gbif.org`

`Anguilla data` *Anguilla australis distribution data*

Description

A number of sites with presence or absence of the short-finned eel (Anguilla australis) in New Zealand, and environmental data at these sites; and gridded data of the environmental variables for the study area.

type	variable name	values
Reach	LocSed	weighted average of proportional cover of bed sediment
Segment	SegSumT	Summer air temperature (degrees C)
	SegTSeas	Winter air temperature (degrees C), normalised with respect to SegJanT
	SegLowFlow	segment low flow (m3/sec), fourth root transformed
Downstream	DSDist	distance to coast (km)
	DSDam	presence of known downstream obstructions, mostly dams
	DSMaxSlope	maximum downstream slope (degrees)
Upstream / catchment	USAvgT	average temperature in catchment (deg C) compared to segment, normalised with r
	USRainDays	days/month with rain greater than 25 mm
	USSlope	average slope in the upstream catchment (degrees)
	USNative	area with indigenous forest (proportion)
	Fishing method	fishing method in five classes: electric, net, spot, trap & mixture

Usage

```
data(Anguilla_train)
data(Anguilla_test)
data(Anguilla_grids)
```

Author(s)

John R. Leathwick and Jane Elith

References

Elith, J., J.R. Leathwick and T. Hastie, 2009. A working guide to boosted regression trees. Journal of Animal Ecology 77: 802-81

`bioclim` *Bioclim*

Description

The Bioclim algorithm has been extensively used for species distribution modeling. Bioclim is the classic 'climate-envelope-model'. Although it generally does not perform as good as some other modeling methods (Elith et al. 2006) and is unsuited for predicting climate change effects (Hijmans and Graham, 2006). It is still used, however, among other reasons because the algorithm is easy to understand and thus useful in teaching species distribution modeling.

The BIOCLIM 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.

In the predict function, you can choose to ignore one of the tails of the distribution (e.g, to make low rainfall a limiting factor, but not high rainfall),

Usage

```
bioclim(x, p, ...)
```

Arguments

`x`	Raster* object or matrix
`p`	two column matrix or SpatialPoints* object
`...`	Additional arguments

Value

An object of class 'Bioclim' (inherits from `DistModel-class`)

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. http://dx.doi.org/10.1111/j.2006.0906-7590.04596.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. http://dx.doi.org/10.1111/j.1365-2486.2006.01256.x

See Also

predict, maxent, domain, mahal

Examples

```
logo <- stack(system.file("external/rlogo.grd", package="raster"))
#presence data
pts <- matrix(c(48.243420, 48.243420, 47.985820, 52.880230, 49.531423, 46.182616, 54.168232,
  69.624263, 83.792291, 85.337894, 74.261072, 83.792291, 95.126713, 84.565092, 66.275456, 41.803408,
  25.832176, 3.936132, 18.876962, 17.331359,7.048974, 13.648543, 26.093446, 28.544714, 39.104026,
  44.572240, 51.171810, 56.262906, 46.269272, 38.161230, 30.618865, 21.945145, 34.390047, 59.656971,
   69.839163, 73.233228, 63.239594, 45.892154, 43.252326, 28.356155) , ncol=2)
bc <- bioclim(logo, pts)

#or
v <- extract(logo, pts)
bc <- bioclim(v)
p1 <- predict(logo, bc)
p2 <- predict(logo, bc, tails=c('both', 'low', 'high'))

#or
#sp <- SpatialPoints(pts)
#bc <- bioclim(logo, pts)
```

---

biovars *bioclimatic variables*

Description

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

Usage

```
biovars(prec, tmin, tmax, ...)
```

Arguments

prec	vector, matrix, or RasterStack/Brick of precipitation data
tmin	vector, matrix, or RasterStack/Brick of minimum temperature data
tmax	vector, matrix, or RasterStack/Brick of maximum temperature data
...	Additional arguments

Details

Input data is normaly 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 quater (3 months), but for a 3 week period).

Value

Depending on the class of the input data, an object of class 'vector', 'matrix' or 'RasterBrick' with 19 variables (columns, layers)

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

Author(s)

Robert J. Hijmans

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)
biovars(prec, tmin, tmax)

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

boxplot *Box plot of model evaluation data*

Description

Make a box plot of model evaluation data, i.e., the model predictions for known presence and absence points.

Details

Arguments:

x Object of class ModelEvaluation ... Additional arguments that can be passed to boxplot

Author(s)

Robert J. Hijmans

See Also

evaluate

calc.deviance *Calculate deviance*

Description

Function to calculate deviance given two vectors of observed and predicted values. Requires a family argument which is set to binomial by default

Usage

```
calc.deviance(obs, pred, weights = rep(1,length(obs)),
                family="binomial", calc.mean = TRUE)
```

Arguments

obs	a vector with observed values
pred	a vector with predicted values that correspond the the values in obs
weights	a vector of weight values
family	One of "binomial", "bernoulli", "poisson", "laplace", or "gaussian"
calc.mean	Logical. If TRUE, the mean deviance is returned

Author(s)

John R. Leathwick and Jane Elith

`Circles` *Circles range*

Description

The circles model predicts that a species is present at sites within a certain distance from a training point, and absent further away.

Usage

```
circles(p, ...)
```

Arguments

`p`	point locations (presence). Two column matrix, data.frame or SpatialPoints* object
`...`	Additional arguments. See Details

Details

The following additional arguments can be supplied to the circles function:

`d`	The radius of each circle in meters. A single number of a vector with elements corresponding to rows in 'p'. If m
`n`	How many vertices in the circle? Default is 360
`lonlat`	Are these longitude/latidue data? Default value is FALSE
`r`	Radius of the earth. Only relevant for longitude/latitude data. Default is 6378137 m

Value

An object of class 'CirclesRange' (inherits from `DistModel-class`)

Author(s)

Robert J. Hijmans

See Also

`predict`, `geoDist`, `convHull`, `maxent`, `domain`,`mahal`, `convexHull`

Examples

```
r <- raster(system.file("external/rlogo.grd", package="raster"))
#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, ]
```

```
cc <- circles(train, lonlat=FALSE)
predict(cc, test)

plot(r)
plot(cc@polygons, border='red', lwd=2, add=TRUE)
points(train, col='red', pch=20, cex=2)
points(test, col='black', pch=20, cex=2)

pr = predict(cc, r, progress='')
plot(r, legend=FALSE)
plot(pr, add=TRUE, col='blue')
points(test, col='black', pch=20, cex=2)
points(train, col='red', pch=20, cex=2)
```

`Convex Hull` *Convex hull model*

Description

The Convex hull model predicts that a species is present at sites inside the convex hull of a set of training points, and absent outside that hull. I.e. this is the spatial convex hull, not an environmental hull.

Usage

```
convHull(p, ...)
```

Arguments

`p`	point locations (presence). Two column matrix, data.frame or SpatialPoints* object
`...`	Additional arguments. See details

Details

You can supply an argument n (>= 1) to get n convex hulls around subset of the points. You can also set n=1:x, to get a set of overlapping polygons consisting of 1 to x parts. I.e. the first polygon has 1 part, the second has 2 parts, and x has x parts.

Value

An object of class 'ConvexHull' (inherits from `DistModel-class`)

Author(s)

Robert J. Hijmans

See Also

predict, geoDist, maxent, domain, mahal

Examples

```
r <- raster(system.file("external/rlogo.grd", package="raster"))
#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 <- convHull(train)
predict(ch, test)

plot(r)
plot(ch@polygons, 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, progress='')
plot(pr)
points(test, col='black', pch=20, cex=2)
points(train, col='red', pch=20, cex=2)
```

---

dcEvaluate *Evaluate by distance class*

Description

Evaluate a model for intervals of distances to the nearest point in a reference dataset for presence data and a sample of the absence data selected to have a low spatial sorting bias (obtained with pwdSample).

Usage

```
dcEvaluate(p, a, reference, lonlat=TRUE, binsize=15, predp, preda, model,
                 predictors, fun=predict)
```

Arguments

p	two column matrix (x, y) or (longitude/latitude) or SpatialPoints object, for point locations
a	two column matrix (x, y) or (longitude/latitude) or SpatialPoints object, for point locations
reference	as above for reference point locations to which distances are computed
lonlat	Logical. Use TRUE if the coordinates are spherical (in degrees), and use FALSE if they are planar

`binsize`	postive integer. How many presence points in each distance bin?
`predp`	p
`preda`	a
`model`	m
`predictors`	pr
`fun`	function

Value

list with Evaluation objects

Author(s)

Robert J. Hijmans

See Also

`pwdSample`, `ssb`

`density` *density*

Description

Create a density plots of presence and absence data

Value

A density plot. Presence data are in red, and absence data (if available) are in blue.

Methods

```
density(x, ...)
```

`x`	Object of class 'ModelEvaluation' or of a class that inherits from 'DistModel, (such as 'MaxEnt', 'Bioclim')
`...`	Additional arguments that can be passed to plot

Author(s)

Robert J. Hijmans

See Also

`evaluate`

DistModel *Class "DistModel"*

Description

Parent class for a number of distribution models defined in the dismo package (those created by `bioclim`, `domain`, `maxent` and `mahal`). This is a virtual Class, no objects may be direclty created from it.

Slots

`presence`: presence data used

`absence`: absence or background data used

`hasabsence`: Logical indicating whether there is any absence data

Author(s)

Robert J. Hijmans

---

domain *Domain*

Description

The Domain algorithm (Carpenter et al. 1993) that has been extensively used for species distribution modeling. It is included here for that reason but please note that it generally does not perform very well in model comparison (Elith et al. 2006, Hijmans and Graham, 2006). The Domain algorithm computes the Gower distance between environmental variables at any location and those at any of the known locations of occurrence ('training sites'). For each variable the minimum distance between a site and any of the training points is taken. To integrate over environmental variables, the maximum distance to any of the variables is used. This distance is subtracted from one, and (in this R implementation) values below zero are truncated so that the scores are between 0 (low) and 1 (high).

Usage

```
domain(x, p, ...)
```

Arguments

`x`	Raster* object or matrix
`p`	two column matrix or SpatialPoints* object
`...`	Additional arguments

Value

An object of class 'Domain' (inherits from DistModel-class)

Author(s)

Robert J. Hijmans

References

Carpenter G., A.N. Gillison and J. Winter, 1993. Domain: a flexible modelling procedure for mapping potential distributions of plants and animals. Biodiversity Conservation 2:667-680.

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. http://dx.doi.org/10.1111/j.2006.0906-7590.04596.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. http://dx.doi.org/10.1111/j.1365-2486.2006.01256.x

See Also

predict, maxent, bioclim, mahal

Examples

```
logo <- stack(system.file("external/rlogo.grd", package="raster"))
#presence data
pts <- matrix(c(48.243420, 48.243420, 47.985820, 52.880230, 49.531423, 46.182616, 54.168232,
  69.624263, 83.792291, 85.337894, 74.261072, 83.792291, 95.126713, 84.565092, 66.275456,
  41.803408, 25.832176, 3.936132, 18.876962, 17.331359,7.048974, 13.648543, 26.093446,
  28.544714, 39.104026, 44.572240, 51.171810, 56.262906, 46.269272, 38.161230, 30.618865,
  21.945145, 34.390047, 59.656971, 69.839163, 73.233228, 63.239594, 45.892154, 43.252326,
  28.356155), ncol=2)
d <- domain(logo, pts)
p <- predict(d, logo)
```

---

ecocrop *Ecocrop model*

Description

Very simple mechanistic model for plants.

Usage

```
ecocrop(crop, tmin, tavg, prec, rainfed=TRUE, ...)
getCrop(name)
data(ECOcrops)
```

Arguments

crop	An object of class 'ECOCROP', or the name of a crop as in getCrop
tmin	Vector of monthly minimum temperature (degrees C)
tavg	Vector of monthly average temperature (degrees C)
prec	Vector of monthly precipitation (mm)
rainfed	Logical. If FALSE, the crop is assumed to be irrigated
...	Additinal arguments
name	Name of a crop (character). If missing a data.frame with all crop names is returned

Value

Object of class ECOCROP

Author(s)

Robert J. Hijmans

Examples

```
ecocrop('potato', 5:16, 15:26, runif(12)*100)
getCrop('Acacia brachystachya Benth.')
crop <- getCrop('Hot pepper')
ecocrop(crop, 5:16, 15:26, rainfed=FALSE)
```

---

ecolim *Ecolim model*

Description

Simple generic limiting factor based model, in the tradition of the PLANTGRO model (Hackett, 1991)

Usage

```
## S4 method for signature 'matrix,matrix'
ecolim(x, y, extrapolate=TRUE, ...)
```

Arguments

x	numeric matrix with driver variables (each column has values for the variables). Values have to be in ascending order
y	numeric matrix with responses (between 0 and 1), one column for each column in x
extrapolate	logical. Should the model extrapolate beyond the extremes of x? If TRUE the value of y at the closest data extreme in x is used, else NA is returned for such records
...	Additional arguments. None implemented

Author(s)

Robert J. Hijmans

References

Hackett, C., 1991. PLANTGRO, a software package for coarse prediction of plant growth. CSIRO, Melbourne, Australia

Examples

```
# get predictor variables
fnames <- list.files(path=paste(system.file(package="dismo"), '/ex', sep=''),
              pattern='grd', full.names=TRUE )
env <- stack(fnames)

bio1 <- c(200,250,400,450)
bio12 <- c(0,1000, 3000, 4000)
r1 <- c(0, 1, 1, 0)
r2 <- c(0, 0, 1, 1)
x <- cbind(bio1, bio12)
y <- cbind(r1, r2)

e <- ecolim(x, y)
plot(e, lwd=2, col='red')
p <- predict(e, env)
plot(p)

# no extrapolation:
ef <- ecolim(x, y, extrapolate=FALSE)
pf <- predict(ef, env)
plot(pf)


occurence <- paste(system.file(package="dismo"), '/ex/bradypus.csv', sep='')
occ <- read.table(occurence, header=TRUE, sep=',')[,-1]
fold <- kfold(occ, k=5)
occtest <- occ[fold == 1, ]
occtrain <- occ[fold != 1, ]
bg <- randomPoints(env, 1000)
```

```
## Not run:
# An approach to optimize the values based on
# some known presences and (here random) absences
# for the same species as in the maxent example

# intial parameters
v <- c(200, 250, 400, 450, 0, 1000, 3000, 4000)

# function to be minimized
f <- function(p) {
x[] <- p
# numbers must go up
if ( any(x[-1,] < x[-nrow(x), ]) ) return(Inf)
e <- ecolim(x, y)
# we are minimizing, hence 1-AUC
1-evaluate(e, p=occtrain, a=bg, x=env)@auc
}

# patience...
set.seed(0)
z <- optim(v, f)

x[] <- z$par
eco <- ecolim(x, y)
evaluate(eco, p=occtest, a=bg, x=env)

set.seed(0)
pwd <- pwdSample(occtest,bg,occtrain)
ptest <- occtest[!is.na(pwd),]
atest <- bg[na.omit(pwd),]
evaluate(eco, p=ptest, a=atest, x=env)

p2 <- predict(eco, env)
plot(p2)

## End(Not run)
```

`evaluate` *Model evaluation*

Description

Cross-validation of models with presence/absence data. Given a vector of presence and a vector of absence values (or a model and presence and absence points and predictors), confusion matrices are computed (for varying thresholds), and model evaluation statistics are computed for each confusion matrix / threshold. See the description of class `ModelEvaluation-class` for more info.

Usage

```
evaluate(p, a, model, x, tr, ...)
```

Arguments

p	presence points (x and y coordinates or SpatialPoints* object). Or, if x is missing, values at presence points Or, a matrix with values to compute predictions for
a	absence points (x and y coordinates or SpatialPoints* object). Or, if x is missing, values at presence points. Or, a matrix with values to compute predictions for
model	any fitted model, including objects inheriting from 'DistModel'; not used when x is missing
x	Optional. Predictor variables (object of class Raster*). If present, p and a are interpreted as (spatial) points
tr	Optional. a vector of threshold values to use for computing the confusion matrices
...	Additional arguments for the predict function

Value

An object of ModelEvaluation-class

Author(s)

Robert J. Hijmans

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

See Also

threshold

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 <- evaluate(p=p, a=a)

threshold(e)
```

```
plot(e, 'ROC')
plot(e, 'TPR')
boxplot(e)
density(e)

str(e)
```

`evaluateROCR` *Model testing with the ROCR package*

Description

Preparing data for model testing with the ROCR package.

Usage

```
evaluateROCR(model, p, a, x)
```

Arguments

`model`	any fitted model, including objects inherting from 'DistModel'
`p`	presence points (x and y coordinates or SpatialPoints* object). Or, if `x` is missing, values at presence points Or, a matrix with values to compute predictions for
`a`	absence points (x and y coordinates or SpatialPoints* object). Or, if `x` is missing, values at presence points. Or, a matrix with values to compute predictions for
`x`	optional. predictor variables, if present, `p` and `a` are considered

Value

An object of class "prediction" (defined in the ROCR package)

Author(s)

Robert J. Hijmans

`Evaluation plots` *Plot model evaluation data*

Description

Make a ROC curve, or a plot of a threshold dependent measure against threshold values

Methods

usage: `plot(x, y, ...)`

x	Object of class ModelEvaluation
y	Character. Either 'ROC' or a threshold dependent measure such as 'kappa', 'TPR'
...	Additional arguments that can be passed to plot

Author(s)

Robert J. Hijmans

See Also

`ModelEvaluation-class`, `density`, `pairs`, `plot`

Examples

```
# p = the predicted value for 50 known cases (locations) with presence of the phenomenon (species)
p = rnorm(50, mean=0.7, sd=0.3)
# b = the predicted value for 50 known cases (locations) with absence of the phenomenon (species)
a = rnorm(50, mean=0.4, sd=0.4)
e = evaluate(p=p, a=a)
plot(e, 'ROC')
plot(e, 'kappa')
plot(e, 'FPR')
plot(e, 'prevalence')
```

---

gbif *Data from GBIF*

Description

This function downloads species occurence records from the Global Biodiversity Information Facility (GBIF) data portal. You can download either a single species (if you append a '*' to the species name) or a subspecies of comparable level. You can download the data for an entire genus by using `species='*'`. Before using this function, please first check the GBIF data use agreement.

Usage

```
gbif(genus, species="", concept=FALSE, ext=NULL, args=NULL, geo=TRUE, sp=FALSE,
   removeZeros=FALSE, download=TRUE, getAlt=TRUE, returnConcept=FALSE,
   ntries=5, nrecs=1000, start=1, end=NULL, feedback=3)
```

Arguments

genus	character. genus name
species	character. species name. Use '*' to download the entire genus. Append '*' to the species name to get all naming variants (e.g. with and witout species author name) and sub-taxa

concept	logical or integer. If TRUE records for species that are synonyms (according to the GBIF species concept key) are also downloaded. If concept is an integer (or character representation thereof, it is interprted as a GBIF taxonomic concept number, and the query will use that concept (adn arguments genus and species are ignored)
ext	Extent object to limit the geographic extent of the records. An extent can be created using functions like drawExtent and extent
args	character. Additional arguments to refine the query. See examples for the format, and the (web service documentation) (under 3. SEARCH FOR RECORDS) for more details. This is intended for use with arguments like 'originisocountrycode' or 'startdate' that narrow down the search. Other arguments such as 'format' may cause the function to fail
geo	logical. If TRUE, only records that have a georeference (longitude and latitude values) will be downloaded
sp	logical. If TRUE, geo will be set to TRUE and a SpatialPointsDataFrame will be returned
removeZeros	logical. If TRUE, all records that have a latiude OR longitude of zero will be removed if geo==TRUE, or set to NA if geo==FALSE. If FALSE, only records that have a latiude AND longitude that are zero will be removed or set to NA
download	logical. If TRUE, records will be downloaded, else only the number of records will be shown
getAlt	logical. If TRUE, elevation data (4 character variables) will be processed into a single new numerical variable
returnConcept	logical. If TRUE, only the GBIF taxonomic concept number is returned
ntries	integer. How many times should the function attempt to download the data, if an invalid response is returned (perhaps because the GBIF server is very busy)
nrecs	integer. How many records to download in a single request (max is 1000)?
start	integer. Record number from which to start requesting data
end	integer. Last record to request
feedback	integer. Lower values give less feedback (0-no messages; 3-all messages)

Value

data frame

Author(s)

Robert J. Hijmans

References

http://data.gbif.org/occurrences/

Examples

```
## Not run:
# note the differences:

gbif('solanum', download=F)
gbif('solanum', '*', download=F)
gbif('solanum', 'acaule', download=F)
gbif('solanum', 'acaule f. acaule', download=F)
gbif('solanum', 'acaule*', download=F)

gbif('Batrachoseps', '*' , geo=F, down=F)
gbif('Batrachoseps', '*' , geo=T, down=F)
gbif('Batrachoseps', 'luciae', geo=T, down=F)
g <- gbif('Batrachoseps', 'luciae', geo=T)
plot(g$lon, g$lat)

gs <- gbif('Batrachoseps', 'luciae', sp=T)
plot(gs)

# using additional 'args'
gbif('solanum', 'nigrum', download=F)
gbif('solanum', 'nigrum', download=F, args='originisocountrycode=NL')
gbif('solanum', 'nigrum', download=F, args=c('originisocountrycode=NL', 'originisocountrycode=BE'))

# taxonomic concepts
g0 <- gbif('solanum', 'brevicaule')
unique(g0$species)

g1 <- gbif('solanum', 'brevicaule*')
unique(g1$species)

g2 <- gbif('solanum', 'brevicaule', concept=TRUE)
unique(g2$species)

gcon <- gbif('solanum', 'brevicaule', returnConcept=TRUE)
g3 <- gbif(concept=gcon)

## End(Not run)
```

`gbm.fixed` *gbm fixed*

Description

Calculates a gradient boosting (gbm) object with a fixed number of trees. The optimal number of trees can be identified using gbm.step or some other procedure. Mostly used as a utility function, e.g., when being called by gbm.simplify. It takes as input a dataset and arguments selecting x and y variables, learning rate and tree complexity.

Usage

```
gbm.fixed(data, gbm.x, gbm.y, tree.complexity = 1, site.weights = rep(1, nrow(data)),
 verbose = TRUE, learning.rate = 0.001, n.trees = 2000, bag.fraction = 0.5,
 family = "bernoulli", keep.data = FALSE, var.monotone = rep(0, length(gbm.x)))
```

Arguments

`data`	data.frame
`gbm.x`	indices of the predictors in the input dataframe
`gbm.y`	index of the response in the input dataframe
`tree.complexity`	the tree depth - sometimes referred to as interaction depth
`site.weights`	by default set equal
`verbose`	to control reporting
`learning.rate`	controls speed of the gradient descent
`n.trees`	default number of trees
`bag.fraction`	varies random sample size for each new tree
`family`	can be any of "bernoulli", "poisson", "gaussian", or "laplace"
`keep.data`	Logical. If `TRUE`, original data is kept
`var.monotone`	constrain to positive (1) or negative monontone (-1)

Value

object of class gbm

Author(s)

John R. Leathwick and Jane Elith

References

Elith, J., J.R. Leathwick and T. Hastie, 2009. A working guide to boosted regression trees. Journal of Animal Ecology 77: 802-81

`gbm.holdout` *gbm holdout*

Description

Calculates a gradient boosting (gbm) object in which model complexity is determined using a training set with predictions made to a withheld set. An initial set of trees is fitted, and then trees are progressively added testing performance along the way, using gbm.perf until the optimal number of trees is identified.

As any structured ordering of the data should be avoided, a copy of the data set is BY DEFAULT randomly reordered each time the function is run.

Usage

```
gbm.holdout(data, gbm.x, gbm.y, learning.rate = 0.001, tree.complexity = 1,
 family = "bernoulli", n.trees = 200, add.trees = n.trees, max.trees = 20000,
 verbose = TRUE, train.fraction = 0.8, permute = TRUE, prev.stratify = TRUE,
 var.monotone = rep(0, length(gbm.x)), site.weights = rep(1, nrow(data)),
 refit = TRUE, keep.data = TRUE)
```

Arguments

`data`	data.frame
`gbm.x`	indices of the predictors in the input dataframe
`gbm.y`	index of the response in the input dataframe
`learning.rate`	typically varied between 0.1 and 0.001
`tree.complexity`	sometimes called interaction depth
`family`	"bernoulli","poisson", etc. as for gbm
`n.trees`	initial number of trees
`add.trees`	number of trees to add at each increment
`max.trees`	maximum number of trees to fit
`verbose`	controls degree of screen reporting
`train.fraction`	proportion of data to use for training
`permute`	reorder data to start with
`prev.stratify`	stratify selection for presence/absence data
`var.monotone`	allows constraining of response to monotone
`site.weights`	set equal to 1 by default
`refit`	refit the model with the full data but id'd no of trees
`keep.data`	keep copy of the data

Value

A gbm object

Author(s)

John R. Leathwick and Jane Elith

References

Elith, J., J.R. Leathwick and T. Hastie, 2009. A working guide to boosted regression trees. Journal of Animal Ecology 77: 802-81

gbm.interactions — *gbm interactions*

Description

Tests whether interactions have been detected and modelled, and reports the relative strength of these. Results can be visualised with gbm.perspec

The function assesses the magnitude of 2nd order interaction effects in gbm models fitted with interaction depths greater than 1. This is achieved by:

1. forming predictions on the linear scale for each predictor pair;

2. fitting a linear model that relates these predictions to the predictor pair, with the the predictors fitted as factors;

3. calculating the mean value of the residuals, the magnitude of which increases with the strength of any interaction effect;

4. results are stored in an array;

5. finally, the n most important interactions are identified, where n is 25

Usage

```
gbm.interactions(gbm.object, use.weights=FALSE, mask.object)
```

Arguments

`gbm.object`	A gbm object
`use.weights`	Logical. If `TRUE`, weights are used for samples
`mask.object`	a gbm object describing sample intensity

Value

object of class gbm

gbm.perspec — *gbm perspective plot*

Description

Takes a gbm boosted regression tree object produced by gbm.step and plots a perspective plot showing predicted values for two predictors as specified by number using x and y. Values for all other variables are set at their mean by default but values can be specified by giving a list consisting of the variable name and its desired value, e.g., c(name1 = 12.2, name2 = 57.6)

Usage

```
gbm.perspec(gbm.object, x = 1, y = 2, pred.means = NULL, x.label = NULL, x.range = NULL,
 y.label = NULL, z.label = "fitted value", y.range = NULL, z.range = NULL,
 leg.coords = NULL, ticktype = "detailed", theta = 55, phi = 40, smooth = "none",
 mask = FALSE, perspective = TRUE, ...)
```

Arguments

`gbm.object`	object of class gbm
`x`	the first variable to be plotted
`y`	the second variable to be plotted
`pred.means`	allows specification of values for other variables
`x.label`	allows manual specification of the x label
`x.range`	manual range specification for the x variable
`y.label`	and y label
`z.label`	default z label
`y.range`	and the y
`z.range`	allows control of the vertical axis
`leg.coords`	can specify coords (x, y) for legend
`ticktype`	specifiy detailed types - otherwise "simple"
`theta`	rotation
`phi`	and elevation
`smooth`	controls smoothing of the predicted surface
`mask`	controls masking using a sample intensity model
`perspective`	controls whether a contour or perspective plot is drawn
`...`	allows the passing of additional arguments to plotting routine useful options include shade, ltheta, lphi for controlling illumination and cex for controlling text size - cex.axis and cex.lab have no effect

Author(s)

John R. Leathwick and Jane Elith

References

Elith, J., J.R. Leathwick and T. Hastie, 2009. A working guide to boosted regression trees. Journal of Animal Ecology 77: 802-81

`gbm.plot` *gbm plot*

Description

Function to plot gbm response variables, with the option of adding a smooth representation of the response if requested additional options in this version allow for plotting on a common scale. Note hat fitted functions are centered by subtracting their mean.

Usage

```
gbm.plot(gbm.object, variable.no=0, smooth=FALSE, rug=TRUE, n.plots=length(pred.names),
  common.scale=TRUE, write.title=TRUE, y.label="fitted function", x.label=NULL,
  show.contrib=TRUE, plot.layout=c(3, 4), ...)
```

Arguments

`gbm.object`	a gbm object - could be one from gbm.step
`variable.no`	the var to plot - if zero then plots all
`smooth`	Logical. If `TRUE`, a smoothed version of the fitted function is added
`rug`	Logical. If `TRUE`, a rug of deciles is plotted
`n.plots`	plot the first n most important preds
`common.scale`	Logical. If `TRUE`, a common scale is used on the y axis
`write.title`	Logical. If `TRUE`, the plot gets a title
`y.label`	the default y-axis label
`x.label`	the default x-axis label
`show.contrib`	Logical. If `TRUE`, the contribution is shown on the x axis
`plot.layout`	define the default layout for graphs on the page
`...`	other arguments to pass to the plotting useful options include cex.axis, cex.lab, etc.

Author(s)

John R. Leathwick and Jane Elith

References

Elith, J., J.R. Leathwick and T. Hastie, 2009. A working guide to boosted regression trees. Journal of Animal Ecology 77: 802-81

`gbm.plot.fits` *gbm plot fitted values*

Description

Plots the fitted values from a gbm object returned by any of the model fitting options. This can give a more reliable guide to the shape of the fitted surface than can be obtained from the individual functions, particularly when predictor variables are correlated and/or samples are unevenly distributed in environmental space. Allows masking out of absences to enable focus on sites with high predicted values.

Usage

```
gbm.plot.fits(gbm.object, v=0, mask.presence=FALSE, use.factor=FALSE )
```

Arguments

`gbm.object`	a gbm object
`v`	variable numbers to be plotted (if 0 then all are plotted)
`mask.presence`	Logical. If `TRUE`, the function only plots fitted values for presences
`use.factor`	Logical. If `TRUE`, forces to use quicker printing box and whisker plot

Author(s)

John R. Leathwick and Jane Elith

References

Elith, J., J.R. Leathwick and T. Hastie, 2009. A working guide to boosted regression trees. Journal of Animal Ecology 77: 802-81

`gbm.simplify` *gbm simplify*

Description

The function takes an inital cross-validated model as produced by gbm.step and then assesses the potential to remove predictors using k-fold cross validation. This done for each fold, removing the lowest contributing predictor, and repeating this process for a set number of steps. After the removal of each predictor, the change in predictive deviance is computed relative to that obtained when using all predictors. The function returns a list containing the mean change in deviance and its standard error as a function of the number of variables removed. Having completed the cross validation, it then identifies the sequence of variable to remove when using the full data set, testing this up to the number of steps used in the cross-validation phase of the analysis with results reported to the screen.

The function returns a table containing the order in which variables are to be removed and some vectors, each of which specifies the predictor column numbers in the original dataframe - the latter can be used as an argument to gbm.step e.g., gbm.step(data = data, gbm.x = simplify.object$pred.list[[4]]... would implement a new analysis with the original predictor set, minus its four lowest contributing predictors.

Usage

```
gbm.simplify(gbm.object, n.folds = 10, n.drops = "auto", alpha = 1, prev.stratify = TRUE,
   eval.data = NULL, plot = TRUE)
```

Arguments

`gbm.object`	a gbm object describing sample intensity
`n.folds`	number of times to repeat the analysis
`n.drops`	can be automatic or an integer specifying the number of drops to check
`alpha`	controls stopping when n.drops = "auto"
`prev.stratify`	use prevalence stratification in selecting evaluation data
`eval.data`	an independent evaluation data set - leave here for now
`plot`	plot results

Value

A list with these elements: deviance.summary, deviance.matrix, drop.count, final.drops, pred.list, and gbm.call = gbm.call))

Author(s)

John R. Leathwick and Jane Elith

References

Elith, J., J.R. Leathwick and T. Hastie, 2009. A working guide to boosted regression trees. Journal of Animal Ecology 77: 802-81

`gbm.step` *gbm step*

Description

Function to assess the optimal number of boosting trees using k-fold cross validation. This is an implementation of the cross-validation procedure described on page 215 of Hastie et al (2001).

The data is divided into 10 subsets, with stratification by prevalence if required for presence/absence data. The function then fits a gbm model of increasing complexity along the sequence from `n.trees` to `n.trees + (n.steps * step.size)`, calculating the residual deviance at each step along the way. After each fold processed, the function calculates the average holdout residual deviance and its standard error and then identifies the optimal number of trees as that at which the holdout deviance is minimised. It fits a model with this number of trees, returning it as a gbm model along with additional information from the cross-validation selection process.

Usage

```
gbm.step(data, gbm.x, gbm.y, offset = NULL, fold.vector = NULL, tree.complexity = 1,
 learning.rate = 0.01, bag.fraction = 0.75, site.weights = rep(1, nrow(data)),
 var.monotone = rep(0, length(gbm.x)), n.folds = 10, prev.stratify = TRUE,
 family = "bernoulli", n.trees = 50, step.size = n.trees, max.trees = 10000,
 tolerance.method = "auto", tolerance = 0.001, plot.main = TRUE, plot.folds = FALSE,
 verbose = TRUE, silent = FALSE, keep.fold.models = FALSE, keep.fold.vector = FALSE,
 keep.fold.fit = FALSE, ...)
```

Arguments

`data`	input data.frame
`gbm.x`	indices or names of predictor variables in `data`
`gbm.y`	index or name of response variable in `data`
`offset`	offset
`fold.vector`	a fold vector to be read in for cross validation with offsets
`tree.complexity`	sets the complexity of individual trees
`learning.rate`	sets the weight applied to inidivudal trees
`bag.fraction`	sets the proportion of observations used in selecting variables
`site.weights`	allows varying weighting for sites
`var.monotone`	restricts responses to individual predictors to monotone
`n.folds`	number of folds
`prev.stratify`	prevalence stratify the folds - only for presence/absence data
`family`	family - bernoulli (=binomial), poisson, laplace or gaussian
`n.trees`	number of initial trees to fit
`step.size`	numbers of trees to add at each cycle
`max.trees`	max number of trees to fit before stopping
`tolerance.method`	method to use in deciding to stop - "fixed" or "auto"
`tolerance`	tolerance value to use - if method == fixed is absolute, if auto is multiplier * total mean deviance

`plot.main`	Logical. plot hold-out deviance curve
`plot.folds`	Logical. plot the individual folds as well
`verbose`	Logical. control amount of screen reporting
`silent`	Logical. to allow running with no output for simplifying model)
`keep.fold.models`	Logical. keep the fold models from cross valiation
`keep.fold.vector`	Logical. allows the vector defining fold membership to be kept
`keep.fold.fit`	Logical. allows the predicted values for observations from cross-validation to be kept
`...`	Logical. allows for any additional plotting parameters

Value

object of S3 class gbm

Note

This and other boosted regression trees (BRT) functions in the dismo package do not work if you use only one predictor. There is an easy work around: make a dummy variable with a constant value and then fit a model with two predictors, the one of interest and the dummy variable, which will be ignored by the model fitting as it has no useful information.

Author(s)

John R. Leathwick and Jane Elith

References

Hastie, T., R. Tibshirani, and J.H. Friedman, 2001. The Elements of Statistical Learning: Data Mining, Inference, and Prediction. Springer-Verlag, New York Elith, J., J.R. Leathwick and T. Hastie, 2009. A working guide to boosted regression trees. Journal of Animal Ecology 77: 802-81

Examples

```
data(Anguilla_train)
# reduce data set to speed things up a bit
Anguilla_train = Anguilla_train[1:200,]
angaus.tc5.lr01 <- gbm.step(data=Anguilla_train, gbm.x = 3:14, gbm.y = 2, family = "bernoulli",
       tree.complexity = 5, learning.rate = 0.01, bag.fraction = 0.5)
```

`geocode` *Georeferencing with Google*

Description

A wrapper around the Google geocoding web-service. It returns 0 to n matches. It is important to be as precise as possible, e.g. always include the country in the locality description.

The purpose of using this function should be to display the locations on a map in a browser. You should check the Google terms of use http://code.google.com/apis/maps/terms.html to see if your usage of this function (and the underlying Google API) is permitted.

Usage

```
geocode(x, oneRecord=FALSE, extent=NULL, progress='', ...)
```

Arguments

`x`	A vector of locality descriptions
`oneRecord`	Logical. If `TRUE` a single record for each item in x is returned. If the API returned multiple records, the values of this record are computed by averaging the coordinates and taking the union of all bounding boxes
`extent`	An Extent object, or an object that can be coerced to one, to bias the search towards that region
`progress`	Character. Valid values are "" (no progress indicator), "text" or "window"
`...`	additional arguments (currently none implemeted)

Value

`data.frame` with the following fields:

`originalPlace`	the locality description as provided (in argument `x`)
`interpretedPlace`	the locality as interpreted by the Google API
`lon`	longitude
`lat`	latitude
`lonmin`	minimum longitude of the bounding box
`lonmax`	maximum longitude of the bounding box
`latmin`	minimum latitude of the bounding box
`latmax`	maximum latitude of the bounding box
`uncertainty`	distance from `c(lon, lat)` to the farthest corner of the bounding box

Note

It is important to compare fields `originalPlace` and `interpretedPlace` as the Google interpretation of a (perhaps vague) locality description can be very speculative

Author(s)

Robert J. Hijmans

Examples

```
## Not run:
geocode(c('1600 Pennsylvania Ave NW, Washington DC', 'Luca, Italy', 'Kampala'))
geocode(c('San Jose', 'San Jose, Mexico'))
geocode(c('San Jose', 'San Jose, Mexico'), oneRecord=TRUE)

## End(Not run)
```

Geographic Distance *Geographic distance model*

Description

The geographic distance model predicts that the likelyhood of presence is highest near places where a species has been observed. It can be used as a null-model to calibrate cross-validation scores with.

The predicted values are the inverse distance to the nearest known presence point. Distances smaller than or equal to zero are set to 1 (highest score).

Usage

```
geoDist(p, ...)
```

Arguments

`p`	point locations (presence). Two column matrix, data.frame or SpatialPoints* object
`...`	Additional arguments. You must supply a lonlat= argument (logical), unless p is a SpatialPoints* object and has a valid CRS (coordinate reference system). You can also supply an additional argument 'a' for absence points (currently ignored). Argument 'a' should be of the same class as argument 'p'

Value

An object of class 'GeographicDistance' (inherits from `DistModel-class`)

Author(s)

Robert J. Hijmans

See Also

`predict`, `convHull`, `maxent`, `domain`, `mahal`, `voronoiHull`, `geoIDW`

Examples

```
r <- raster(system.file("external/rlogo.grd", package="raster"))
#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)
colnames(pts) <- c('x', 'y')

train <- pts[1:12, ]
test <- pts[13:20, ]

gd <- geoDist(train, lonlat=FALSE)
p <- predict(gd, r)

## Not run:
plot(p)
points(test, col='black', pch=20, cex=2)
points(train, col='red', pch=20, cex=2)

## End(Not run)
```

---

gmap *Get a Google map*

Description

Retrieve a 'Google Map' that can be used as a background for plotting points and other spatial data.

The purpose of using this function should be to display the map in a browser. You should check the Google terms of use `http://code.google.com/apis/maps/terms.html` to see if your usage of this function (the Google API that creates the maps) is permitted.

The projection of the returned Raster object is "Mercator" (unless you use `lonlat=TRUE`), and other spatial data may need to be transformed before it can be plotted on top of the Google map. You can use the `Mercator` function to transform points from longitude/latitude to Mercator. For SpatialLines and SpatialPolygons objects, use `spTransform` in the `rgdal` package.

This function uses the Google static maps web-service, and is based on functions by Markus Loecher for the `RgoogleMaps` package.

Usage

```
gmap(x, exp=1, type='terrain', filename='', style=NULL, scale=1, zoom=NULL,
      size=c(640, 640), rgb=FALSE, lonlat=FALSE, ...)

Mercator(p, inverse = FALSE)
```

Arguments

`x`	a textual locality description, or an Extent object (with longitude/latitude coordinates), or an object that can be coerced to one (such as a Raster* or Spatial* object), in any (known) coordinate system

`exp`	numeric. An expansion factor to enlarge (by multiplication) the extent specified by `x`
`type`	character. Choose from 'roadmap', 'satellite', 'hybrid', 'terrain'
`filename`	character. Filename (optional). You can open the resulting file in a GIS program
`style`	character. Additional style arguments. See `http://code.google.com/apis/maps/documentation/staticmaps/#StyledMapFeatures`. Note that certain style features do not work in combination with (the default) `type='terrain'`
`scale`	1 or 2. Using 2 doubles the number of pixels returned (and thus gives you better image quality if you need a large image)
`zoom`	integer between 0 (the whole world) to 21 (very small area), centered on the center of the extent
`size`	vector of two integers indicating the number of columns and rows that is requested (what is returned depends on other factors as well). Maximum values are `c(640, 640)`, so you can only select a smaller area than the default. Note that the number of pixels returned can be doubled by using `scale=2`
`rgb`	logical. If `TRUE`, a RasterBrick is returned with three layers (red, green, blue). This can be plotted with `plotRGB`
`lonlat`	logical. If `TRUE` the Raster object returned has a longitude/latitude CRS instead of Mercator
`...`	additional parameters
`p`	Points. A two-column matrix, or a SpatialPoints object
`inverse`	Should the inverse projection be done (from Mercator to longitude/latitude?)

Details

If argument `x` is a textual locality description, the `geocode` function is used to retrieve the extent that should be mapped.

Change the type to 'roadmap' if the map returned says "sorry we have no imagery here"; or use a larger extent.

The returned RasterLayer has a Mercator projection. To plot points (or lines or polygons) on top of it, these need to be transformed first.

A matrix of longitude/latitude data can be transformed with the Mercator function used in the example below. 'Spatial*' objects can be transformed with `spTransform` p <- spTransform(x, "+proj=merc +a=6378137 +b=6378137 +lat_ts=0.0 +lon_0=0.0 +x_0=0.0 +y_0=0 +k=1.0 +units=m +nadgrids=@null +no_defs")

Value

RasterLayer

Author(s)

Robert Hijmans and Sebastien Rochette, based on code by Markus Loecher, Sense Networks <markus at sensenetworks.com> in the RgoogleMaps package

Examples

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

# get a map using names
g = gmap('Australia')
plot(g, inter=TRUE)
gs = gmap('Sydney, New South Wales, Australia', type='satellite')
plot(gs, inter=TRUE)
gs = gmap('Sydney, Australia', type='satellite', exp=3)
plot(gs, inter=TRUE)
gs = gmap('Sydney, Australia', type='hybrid', zoom=10, scale=2)
plot(gs, inter=TRUE)

# from a maxtrix with lon/lat points
x = runif(30)*10 + 40
y = runif(30)*10 - 20
xy = cbind(x, y)
g = gmap(xy, type='hybrid')
plot(g, inter=TRUE)
points(Mercator(xy) , col='red', pch=20)

# or from an Extent object
e = extent( -121.9531 , -120.3897 , 35.36 , 36.61956 )
# you can also get an Extent object by clicking on the map twice after using:
# drawExtent()
r = gmap(e)
plot(r, interpolate=TRUE)

# transform points to Mercator for plotting on top of map:
pt <- matrix(c(-121, 36), ncol=2)
ptm <- Mercator(pt)
points(ptm, cex=3, pch=20, col='blue')
Mercator(ptm, inverse=TRUE)

# transform Spatial objects to Mercator for plotting on top of map
# here for points, but particularly relevant for lines and polygons
pt <- data.frame(pt)
coordinates(pt) <- ~X1 + X2
proj4string(pt) <-"+proj=longlat +datum=WGS84 +ellps=WGS84"
ptm2 <- spTransform(pt, CRS("+proj=merc +a=6378137 +b=6378137 +lat_ts=0.0
      +lon_0=0.0 +x_0=0.0 +y_0=0 +k=1.0 +units=m +nadgrids=@null +no_defs"))
points(ptm, col='red', pch='x', cex=3)

# styles:
gmap("Brooklyn", style="feature:road.local|element:geometry|hue:0x00ff00|saturation:100
  &style=feature:landscape|element:geometry|lightness:-100", type='roadmap')

## End(Not run)
```

`gridSample` *Stratified regular sample on a grid*

Description

Sample points from xy, using a grid (raster) as stratification. Up to n points are sampled from each stratum (cell). For "chessboard" sampling (i.e. sampling from half the cells), use the argument `chess='black'`, or `chess='white'`.

Usage

```
gridSample(xy, r, n=1, chess='')
```

Arguments

`xy`	A two column matrix or data.frame with x and y coordinates (or longitude and latitude), or a SpatialPoints* object
`r`	Raster* object
`n`	Maximum number of samples per cell
`chess`	Character. ", 'black', or 'white'. If 'black' or 'white', "chess-board" sampling is used. I.e. only the 'white' fields, or only the 'black' fields are sampled. Cell number 1 (the upper left corner of `r`) is white.

Value

A two column matrix with x and y coordinates (or longitude and latitude)

Author(s)

Robert J. Hijmans

See Also

`pwdSample`

Examples

```
x <- rnorm(1000, 10, 5)
y <- rnorm(1000, 50, 5)
xy <- cbind(x,y)
res <- 5
r <- raster(extent(range(xy[,1]), range(xy[,2])) + res)
res(r) <- res

samp <- gridSample(xy, r, n=1)
plot(xy, cex=0.1)
points(samp, pch='x', col='red')
```

`InvDistW` *Inverse-distance weighted model*

Description

Inverse-distance weighted predictions for presence/absence data. Computed with the gstat function from the gstat package.

Usage

```
geoIDW(p, a, ...)
```

Arguments

`p`	Presence points. Two column matrix, data.frame, or a SpatialPoints* object
`a`	Absence points. Must be of the same class as p
`...`	Addtional arguments. None implemented

Value

An object of class InvDistWeightModel (inherits from `DistModel-class`)

Author(s)

Robert J. Hijmans

Examples

```
r <- raster(system.file("external/rlogo.grd", package="raster"))
# presence points
p <- 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)

# absence points
a <- matrix(c(30, 23, 5, 5, 31, 33, 91, 63, 60, 88, 93, 97, 65, 68, 85, 97, 35, 32, 29, 55,
      3, 8, 19, 71, 49, 36, 69, 41, 20, 28, 18, 9, 5, 9, 25, 71, 8, 32, 46, 60), ncol=2)

idw <- geoIDW(p, a)
prd <- predict(r, idw)

## Not run:
plot(prd)
points(p)
points(a, pch='x')

## End(Not run)
```

`kfold` *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.

Usage

```
kfold(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)
#data(NSWtrain)
## a single species
#srsp1 <- subset(NSWtrain, spid=='srsp1')
#kfold(srsp1, k = 5)

## all species
#k = kfold(NSWtrain, k=5, by=NSWtrain$spid)

#k[NSWtrain$spid=='srsp1']
## each group has the same number of records
##(except for adjustments if the number of records divided by k is not an integer)
#table(k[NSWtrain$spid=='srsp1'])
#k[NSWtrain$spid=='ousp5']
```

lookup	*lookup*

Description

Look up the elevation or the names of countries of adminstrative subdivisions for point locations (longitude/latitude)

Usage

```
country(lonlat, radius=0)
adm(lonlat, radius=0, maxrows=1)
alt(lonlat)
```

Arguments

lonlat	matrix with longitude and latitude values
radius	radius around each point to consider
maxrows	maximum number of rows (records) to be returned

Value

Character or numeric

Author(s)

Robert J. Hijmans

mahal	*Mahalanobis model*

Description

Distribution model based on the Mahalanobis distance. The predictions are (1-distance). I.e. the highest possible value is 1, and there will likely be large negative numbers.

Usage

```
mahal(x, p, ...)
```

Arguments

x	Raster* object or matrix
p	two column matrix or SpatialPoints* object
...	Additional arguments. Currently not used

Value

An object of class Mahalanobis (inherits from DistModel-class)

Author(s)

Robert J. Hijmans

See Also

predict, maxent, bioclim, domain

Examples

```
logo <- stack(system.file("external/rlogo.grd", package="raster"))

#presence data
pts <- matrix(c(48.243420, 48.243420, 47.985820, 52.880230, 49.531423, 46.182616,
  54.168232, 69.624263, 83.792291, 85.337894, 74.261072, 83.792291, 95.126713,
  84.565092, 66.275456, 41.803408, 25.832176, 3.936132, 18.876962, 17.331359,
  7.048974, 13.648543, 26.093446, 28.544714, 39.104026, 44.572240, 51.171810,
  56.262906, 46.269272, 38.161230, 30.618865, 21.945145, 34.390047, 59.656971,
  69.839163, 73.233228, 63.239594, 45.892154, 43.252326, 28.356155), ncol=2)

# fit model
m <- mahal(logo, pts)

# make a prediction
predict(m, logo[1])

x <- predict(m, logo)

# or x <- predict(logo, m) via raster::predict

# plot(x > 0)
```

---

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.

This function uses the MaxEnt species distribution model software, which is a java program that you can download here. Put the file 'maxent.jar' in the 'java' folder of this package. That is the folder returned by `system.file("java", package="dismo")`. You need MaxEnt version 3.3.3b or higher. Please note that this program (maxent.jar) cannot be redistributed or used for commercial or for-profit purposes.

Usage

```
maxent(x, p, ...)
```

Arguments

x	Predictors. Raster* object or SpatialGridDataFrame, containing grids with predictor variables. These will be used to extract values from for the point locations. `x` can also be a data.frame, in which case each column should be a predictor variable and each row a presence or background record.
p	Occurrence data. This can be a data.frame, matrix, SpatialPoints* object, or a vector. 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). Coordinates can also be specified with a SpatialPoints* object (defined in the sp package). Background points are sampled randomly from the cells that are not NA in the first predictor variable, unless background points are specified with an additional argument `a` (see Details). 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.
...	Additional arguments. See Details

Details

Additional arguments:

a	Background points. Only used if 'p' is not missing, and not a vector.
factors	Which (if any) variables should be considered as categorical? Either by (layer)name or by index. On
args	Additional argument that can be passed to MaxEnt. See the MaxEnt help for more information. The
removeDuplicates	Boolean. If TRUE, duplicate presence points (that fall in the same grid cell) are removed.
path	Optional. Where do you want the MaxEnt output files to be stored. This allows you to permanently k

Value

An object of class 'MaxEnt' (inherits from DistModel-class). Or a 'MaxEntReplicates' object if you use 'replicates=' as part of the `args` argument.

Note

If you want to give MaxEnt (the Java virtual machine that runs it) more memory, you can do that by running something like this (for 1 GB) **before** you load the dismo library.

```
options(java.parameters = "-Xmx1g" )
```

Some people have reported problems when using this function on a Mac (Apple) computer. Specifically, the following error message occurs:

```
Error in .jcall(mxe, "S", "fit", c("autorun", "-e", afn, "-o", dirout, : java.lang.InternalError: Can'
Java was started on the first thread. Make sure StartOnFirstThread is not specified in your applicatio
```

This is a known problem with certain Java applications on Macs. There are two work-arounds that we are aware of: 1) run `Sys.setenv(NOAWT=TRUE)` before running `library rJava` (this is what dismo does if rJava is not loaded).

2) use the JGR interface (a Java based R GUI). You can install JGR from here: http://www.rforge.net/JGR/

Author(s)

Steven Phillips and Robert J. Hijmans

References

http://www.cs.princeton.edu/~schapire/maxent/

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. http://dx.doi.org/10.1111/j.1472-4642.2010.00725.x

See Also

predict

Examples

```
# only run if the maxent.jar file is available, in the right folder
jar <- paste(system.file(package="dismo"), "/java/maxent.jar", sep='')

# checking if maxent can be run (normally not part of your script)
if (file.exists(jar) & require(rJava)) {

# get predictor variables
fnames <- list.files(path=paste(system.file(package="dismo"), '/ex', sep=''),
              pattern='grd', full.names=TRUE )
predictors <- stack(fnames)
#plot(predictors)

# file with presence points
occurence <- paste(system.file(package="dismo"), '/ex/bradypus.csv', sep='')
occ <- read.table(occurence, header=TRUE, sep=',')[,-1]

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

# fit model, biome is a categorical variable
```

```
me <- maxent(predictors, occtrain, factors='biome')

# see the maxent results in a browser:
# me

# use "args"
# me2 <- maxent(predictors, occtrain, factors='biome', args=c("-J", "-P"))

# plot showing importance of each variable
plot(me)

# response curves
# response(me)

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

# with some options:
# r <- predict(me, predictors, args=c("outputformat=raw"), progress='text',
#      filename='maxent_prediction.grd')

plot(r)
points(occ)

#testing
# background data
bg <- randomPoints(predictors, 1000)

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

# alternative 1
# extract values
pvtest <- data.frame(extract(predictors, occtest))
avtest <- data.frame(extract(predictors, bg))

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

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

e3 <- 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

```
mess(x, v, full=FALSE, filename='', ...)
```

Arguments

`x`	Raster* object
`v`	matrix or data.frame containing the reference values. Each column should correspond to one layer of the Raster* object
`full`	logical. If `FALSE` a RasterLayer with the MESS values is returned. If `TRUE`, a RasterBrick is returned with `n` layers corresponding to the layers of the input Raster object 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

A RasterBrick with layers corresponding to the input layers and an additional layer with the mess values (if `full=TRUE` and `nlayers(x) > 1`) or a RasterLayer with the MESS values (if `full=FALSE`).

Author(s)

Jean-Pierre Rossi <jean-pierre.rossi@supagro.inra.fr>, 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.

Examples

```
set.seed(9)
r <- raster(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 <- stack(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 <- paste(system.file(package="dismo"), '/ex/bradypus.csv', sep='')
bradypus <- read.table(filename, header=TRUE, sep=',')
bradypus <- bradypus[,2:3]
files <- list.files(path=paste(system.file(package="dismo"),'/ex', sep=''),
   pattern='grd', full.names=TRUE )
predictors <- stack(files)
predictors <- dropLayer(x=predictors,i=9)
reference_points <- extract(predictors, bradypus)
mss <- mess(x=predictors, v=reference_points, full=TRUE)
plot(mss)

## End(Not run)
```

`ModelEvaluation` *Class "ModelEvaluation"*

Description

Class to store results of model cross-validation with presence/absence (0/1) data

Slots

`presence`: presence data used

`absence`: absence data used

`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

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

`confusion`: confusion matrices

`prevalence`: Prevalence

`ODP`: Overall diagnostic power

`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

Author(s)

Robert J. Hijmans

References

Fielding, A. H. & 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

`evaluate`

`nicheEquivalency` *Niche equivalency*

Description

Compute niche equivalency for two species following Warren et al. (2009). The statistic ranges from 0 to 1.

Usage

```
nicheEquivalency(sp1, sp2, predictors, n=99, model=maxent, verbose=TRUE, ...)
```

Arguments

`sp1`	coordinates for species 1 (matrix with (x, y) or (lon, lat), or SpatialPoints)
`sp2`	coordinates for species 2 (matrix with (x, y) or (lon, lat), or SpatialPoints)
`predictors`	Raster object with environmental variables
`n`	integer. Number of randomizations
`model`	function. modeling algorithm to me used
`verbose`	logical. If `TRUE` some progress indicators are printed
`...`	additional arguments (none)

Value

numeric

Author(s)

Brian Anacker. Based on a similar function in by Christoph Heibl in the phyloclim package

References

Warren, D.L., R.E. Glor, M. Turelli, 2008. Environmental niche equivalency versus conservatism: quantitative approaches to niche evolution. Evolution 62:2868-2883.

`nicheOverlap` *Niche overlap*

Description

Compute niche overlap from predictions of species distributions with the 'I' or 'D' similarity statistic of Warren et al. (2009). The statistic ranges from 0 (no overlap) to 1 (the distributions are identical).

Usage

```
nicheOverlap(x, y, stat='I', mask=TRUE, checkNegatives=TRUE)
```

Arguments

`x`	RasterLayer with non-negative values (predictions of the probability that a site is suitable for a species)
`y`	RasterLayer with non-negative values, as above
`stat`	character either 'I' or 'D' to get the statistic with that name
`mask`	logical. If `TRUE` the function removes cells from `x` that are `NA` in `y` and vice-versa. If you are sure that such cases do not occur you can set this to `FALSE` to speed up computations
`checkNegatives`	logical. If `TRUE` the function checks of any of the values in `x` and `y` are negative. If you are sure that such cases do not occur you can set this to `FALSE` to speed up computations

Value

numeric

Author(s)

Based on SDMTools::Istat by Jeremy VanDerWal

References

Warren, D.L., R.E. Glor, M. Turelli, and D. Funk. 2009. Environmental niche equivalency versus conservatism: quantitative approaches to niche evolution. Evolution 62:2868-2883; Erratum: Evolution 65: 1215

Examples

```
r1 <- raster(nr=18, nc=36)
r2 <- raster(nr=18, nc=36)
set.seed(0)
r1[] <- runif(ncell(r1))
r2[] <- runif(ncell(r1))
nicheOverlap(r1, r2)
```

---

pairs *Pair plots*

Description

Pair plots of presence and absence (background) data.

Methods

```
pairs(x, v=NULL, pa='pa', hist=TRUE, cor=TRUE)
```

`x`	Object of class DistModel or derived from that class (such as 'MaxEnt', 'Bioclim')
`v`	numeric, to select a subset of pairs, e.g. v=1:3 to plot only the first three variables
`pa`	Character. Either 'pa', 'p', or 'a' to show presence and absence, presence, or absence data respectively
`hist`	logical. If `TRUE` a histogram of the values is shown on the diagonal
`cor`	logical. If `TRUE` the correlation coefficient is shown in the upper panels

Author(s)

Robert J. Hijmans

See Also

`density`, `plot`

plot — *Plot predictor values*

Description

Plot predictor values for occurence (presence and absence) data in a DistModel (or derived) object.

Methods

usage: `plot(x, y, ...)`

`x`	Object of class DistModel or from a class that inherits from it
`y`	missing
`...`	Additional arguments that can be passed to plot

Author(s)

Robert J. Hijmans

See Also

`density`, `pairs`, `plot`

pointValues — *point values*

Description

Extract values from a Raster* object for point locations. This function adds a few options that can be useful in the context of species distribution modeling to `extract` function in the raster package.

Usage

```
pointValues(x, p, a, uniquecells = TRUE, na.rm = TRUE)
```

Arguments

`x`	A Raster* object
`p`	Points. Two-colum matrix or data.frame; or a SpatialPoints* object
`a`	Additional points (absences). Two-colum matrix or data.frame; or a SpatialPoints* object
`uniquecells`	Logical. If `TRUE`, each cell can be included only once (i.e. 'duplicate' points are removed)
`na.rm`	Logical. If `TRUE`, cell values of NA are not returned

Value

matrix

Author(s)

Robert J. Hijmans

See Also

`extract`

`predict` *Distribution model predictions*

Description

Make a RasterLayer with a prediction based on a model object of class the inherits from 'Dist-Model', including: Bioclim, Domain, MaxEnt, Mahalanobis, and GeographicDistance. Predictions with model objects that do not inherit from DistModel can be made using the similar `predict` function in the 'raster' package.

Provide a Raster* object with the independent variables. The names of the layers in the Raster* object should include those expected by the model.

Value

A RasterLayer or, (if `x` is a matrix), a vector.

Methods

```
predict(object, x, ext=NULL, filename='', progress='text', ...)
```

`object`	A fitted model of class Bioclim, Domain, MaxEnt, ConvexHull, or Mahalanobis (classes that inherit from Dist
`x`	A Raster* object or a data.frame
`ext`	An extent object to limit the prediction to a sub-region of 'x'. Or an object that can be coerced to an Extent ob
`filename`	Output filename for a new raster; if NA the result is not written to a file but returned with the RasterLayer obje
`progress`	Character. Valid values are "" (no progress bar), "text" and "windows" (on that platform only)
`...`	Additional model specific arguments. And additional arguments for file writing as for writeRaster

For maxent models, there is an additional argument `'args'` used to pass arguments (options) to the maxent software. See the help page for `maxent` for more information.

For bioclim models, there is an additional argument `'tails'` which you can use 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 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.

For geoDist models, there is an additional argument `fun` that allows you to use your own (inverse) distance function, and argument `scale=1` that allows you to scale the values (distances smaller than this value become one, and the others are divided by this value before computing the inverse distance).

Author(s)

Robert J. Hijmans

See Also

For spatial predictions with GLM, GAM, BRT, randomForest, etc., see predict in the Raster package.

To fit a model that can be used with this predict method, see `maxent`, `bioclim`, `mahal`, `domain`, `geoDist`, `convHull`

Extent object: `extent`

Examples

```
logo <- stack(system.file("external/rlogo.grd", package="raster"))
pts <- matrix(c(48, 48, 48, 53, 50, 46, 54, 70, 84, 85, 74, 84, 95, 85, 66,
   42, 26, 4, 19, 17, 7, 14, 26, 29, 39, 45, 51, 56, 46, 38, 31, 22, 34, 60,
   70, 73, 63, 46, 43, 28), ncol=2)
b <- bioclim(logo, pts)
# prediction for a sub-region
e <- extent(30,90,20,60)
p <- predict(b, logo, progress='text', ext=e)
plot(p)
```

---

prepareData *Prepare data for model fitting*

Description

Simple helper function to prepare data for model fitting

Usage

```
prepareData(x, p, b, factors, xy=FALSE)
```

Arguments

`x`	Raster* object
`p`	presence points
`b`	background (or absence) points
`factors`	vectors indicating which variables are factors (using layer names or numbers)
`xy`	logical. If `TRUE`, the first two columns of the returned data.frame will be the coordinates of p and b (labeled 'x' and 'y')

Value

data.frame with `nlayers(x)+1` columns and `nrow(p) + nrow(b)` rows. The first column, 'pb' indicates whether a record represents presence '1' or background '0' values. The other columns have the values from the Raster* object.

Author(s)

Robert J. Hijmans

`pwdSample` *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' (`ssb`) 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

```
pwdSample(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.

Author(s)

Robert J. Hijmans

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

See Also

gridSample

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 <- raster()
extent(r) <- c(-110, 110, -45, 45)
r[] <- 1
set.seed(0)
sam <- randomPoints(r, n=50)

par(mfrow=c(1,2))
plot(sam, pch='x')
points(ref, col='red', pch=18, cex=2)
points(fix, col='blue', pch=20, cex=2)

i <- pwdSample(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'
pwdSample(fix, sam, ref, lonlat=TRUE, n=3)
```

Random null model *Random null model*

Description

Null model based on randomization of locations as suggested by Raes and ter Steege (2007).

Usage

```
nullRandom(x, model, n=25, rep=25, pa=FALSE)
```

Arguments

x	data.frame with environmental predictor values for collecting localities
model	Model function that creates a model of class 'DistModel'
n	Sample size
rep	Number of repetitions
pa	Boolean. Prensence-only or presence/background model (e.g. Maxent)

Value

List with n object of class ModelEvaluation-class

Author(s)

Robert J. Hijmans

References

Raes, N. & H. ter Steege, 2007. A null-model for significance testing of presence-only species distribution models. Ecography 30:727-736.

See Also

geoDist

Examples

```
predictors <- stack(list.files(path=paste(system.file(package="dismo"), '/ex', sep=''),
               pattern='grd', full.names=TRUE ))
occurence <- paste(system.file(package="dismo"), '/ex/bradypus.csv', sep='')
occ <- read.table(occurence, header=TRUE, sep=',')[,-1]

x <- extract(predictors, occ)
nr <- nullRandom(x, bioclim, n=25, rep=25, pa=FALSE)
mean(sapply(nr, function(x)x@auc))
```

`randomPoints` *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

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

Arguments

mask	Raster* object. If the object has cell values, cells with NA are excluded (or the first layer of the object if there are multiple layers), unless weights=TRUE
n	integer. Number of points
p	Presence points (if provided, random points won't be in the same cells (as defined by mask)
ext	Extent object. 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
prob	logical. If TRUE the values in mask are interpreted as probability weights (and the values should be positive numbers (or NA)). NOTE: this currently only works for rasters of a relatively modest size (that can be loaded into RAM)
cellnumbers	logical. If TRUE, cell numbers for]codemask 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

`lonlatCorrection`
: logical. If `TRUE` then correct for the fact that longitude/latitude is not a planar coordinate system

Value

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

Author(s)

Robert J. Hijmans

`response` *response plots*

Description

Generate 'response plots', i.e. single variable response curves for a model

Usage

```
response(x, ...)
```

Arguments

`x`	Model object that inherits from 'DistModel', e.g. 'MaxEnt'. Also works for some other models (e.g. GLM)
`...`	Additional arguments. See Details

Details

`var`	Variable to be plotted (if NULL, all variables will be plotted)
`at`	Function to indicate at what level the other variables should be. E.g. median (the default), mean, min, max. Note
`range`	'pa' (default) or 'p'. Show responses for the range of the presence data (p) or presence and absence (background)
`expand`	percentage to expand the range of values with. Default is 10
`rug`	Logical. If `TRUE` (the default) a 'rug' of deciles is plotted on the horizontal axes)
`data`	data.frame with data to use, normally this is the data used to fit the model and does not needs to be supplied as th
`fun`	predict function. The default is "predict"; but you may change this to e.g., function(x, y, ...) predict(x, y, type='re
`...`	Additional graphical parameters

Value

Used for the side-effect of a plot

Author(s)

Robert J. Hijmans

See Also

`density`, `plot`, `pairs`

`ssb` *Spatial sorting bias*

Description

Determine "spatial sorting bias", or the difference between two point data sets in the average distance to the nearest point in a reference dataset.

Usage

```
ssb(p, a, reference, lonlat=TRUE, avg=TRUE)
```

Arguments

`p`	two column matrix (x, y) or (longitude/latitude) or SpatialPoints object, for point locations
`a`	two column matrix (x, y) or (longitude/latitude) or SpatialPoints object, for point locations
`reference`	as above for reference point locations to which distances are computed
`lonlat`	Logical. Use `TRUE` if the coordinates are spherical (in degrees), and use `FALSE` if they are planar
`avg`	Logical. If `TRUE` the distances are averaged

Value

matrix with two values. 'dp': the average distance from a point in `p` to the nearest point in `reference` and 'da': the average distance from a point in `a` to the nearest point in `reference`. Distance is in meters if `lonlat=TRUE`, and in mapunits (typically also meters) if `lonlat=FALSE`

Author(s)

Robert J. Hijmans

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.

See Also

pwdSample

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)
p <- 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 <- raster()
extent(r) <- c(-110, 110, -45, 45)
r[] <- 1
set.seed(0)
a <- randomPoints(r, n=50)
b <- ssb(p, a, ref)

# distances in km
b / 1000

# an index of spatial sorting bias (1 is no bias, near 0 is extreme bias)
b[1] / b[2]
```

---

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 'ModelEvaluation'
threshold(x, stat='', sensitivity=0.9, ...)
```

Arguments

x	A ModelEvaluation object (see evaluate
stat	character. To select a particular threshold (see section 'value' for possible values)
sensitivity	numeric between 0 and 1. For the fixed sensitivity threshold
...	Additional arguments. None implemented

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

sensitivty: fixed (specified) sensitivity

Author(s)

Robert J. Hijmans and Diego Nieto-Lugilde

See Also

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 <- evaluate(p=p, a=a)

threshold(e)
```

`voronoi` *Voronoi polygons*

Description

Voronoi polygons

Usage

```
voronoi(xy)
```

Arguments

xy	SpatialPoints* or two column matrix

Value

SpatialPolygonsDataFrame

Author(s)

Adapted from code by Carson Farmer: http://www.carsonfarmer.com/?p=455

Examples

```
# points
p <- 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)

v <- voronoi(p)
v
```

`Voronoi Hull` *Voronoi hull model*

Description

Voronoi polygons for presence/absence data

Usage

```
voronoiHull(p, a, ...)
```

Arguments

`p`	Presence points. Two column matrix, data.frame, or a SpatialPoints* object
`a`	Absence points. Must be of the same class as p
`...`	Addtional arguments

Value

A VoronoiHull object (inherits from `DistModel-class`)

Note

This function is only correct when using a planar coordinate reference system (not longitude/latitude).

Author(s)

Robert J. Hijmans. Adapted from code by Carson Farmer: http://www.carsonfarmer.com/?p=455

See Also

`convexHull`

Examples

```
r <- raster(system.file("external/rlogo.grd", package="raster"))
# presence points
p <- 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)

# absence points
a <- matrix(c(30, 23, 5, 5, 31, 33, 91, 63, 60, 88, 93, 97, 65, 68, 85, 97, 35, 32, 29, 55,
      3, 8, 19, 71, 49, 36, 69, 41, 20, 28, 18, 9, 5, 9, 25, 71, 8, 32, 46, 60), ncol=2)

v <- voronoiHull(p, a)

x <- predict(r, v)

## Not run:
plot(x)
points(p, col='black', pch=20, cex=2)
points(a, col='red', pch=20, cex=2)

## End(Not run)
```

Index