Package 'SSDM'

April 14, 2025

Type Package

Title Stacked Species Distribution Modelling

Version 0.2.10

Maintainer Sylvain Schmitt <sylvain.m.schmitt@gmail.com>

URL https://github.com/sylvainschmitt/SSDM

BugReports https://github.com/sylvainschmitt/SSDM/issues

Description Allows to map species richness and endemism based on stacked species distribution models (SSDM). Individuals SDMs can be created using a single or multiple algorithms (ensemble SDMs). For each species, an SDM can yield a habitat suitability map, a binary map, a between-algorithm variance map, and can assess variable importance, algorithm accuracy, and between-algorithm correlation. Methods to stack individual SDMs include summing individual probabilities and thresholding then summing. Thresholding can be based on a specific evaluation metric or by drawing repeatedly from a Bernoulli distribution. The SSDM package also provides a user-friendly interface.

License GPL (>= 3) | file LICENSE

LazyData TRUE

Imports sf (>= 1.0-14), raster (>= 2.9-5), methods (>= 3.2.2), mgcv (>= 1.8.7), earth (>= 4.4.3), rpart (>= 4.1.10), gbm (>= 2.1.1), randomForest (>= 4.6.10), dismo (>= 1.0.12), nnet (>= 7.3.10), e1071 (>= 1.6.7), ggplot2 (>= 3.1.1), reshape2 (>= 1.4.3), scales (>= 1.0.0), shiny (>= 0.12.2), shinydashboard (>= 0.5.1), spThin (>= 0.1.0), poibin (>= 1.3.0), foreach (>= 1.4.4), doParallel (>= 1.0.14), iterators (>= 1.0.10), itertools (>= 0.1-3), parallel (>= 3.5.2), leaflet (>= 2.2.0), magrittr (>= 2.0.3), sdm (>= 1.1.8)

Depends R (>= 3.2.2)

Collate 'SDM.R' 'Algorithm.SDM.R' 'Ensemble.SDM.R' 'Env.R' 'Occurrences.R' 'PA.select.R' 'SSDM.R' 'Stacked.SDM.R' 'accuracy.R' 'checkargs.R' 'data.values.R' 'ensemble.R' 'optim.thresh.R' 'evaluate.R' 'modelling.R' 'ensemble_modelling.R' 'evaluate.axes.R' 'get_PA.R'

'get_model.R' 'gui.R' 'load_model.R' 'load_occ.R' 'load_var.R'
'mapDiversity.R' 'plot.model.R' 'project.R' 'save.model.R'
'stack_modelling.R' 'stacking.R' 'update.stack.R'
'utils-pipe.R' 'zzz.R'

Suggests testthat, knitr, rmarkdown, shinyFiles

RoxygenNote 7.3.1

VignetteBuilder knitr, rmarkdown

Encoding UTF-8

NeedsCompilation no

Author Sylvain Schmitt [aut, cre],
Robin Pouteau [aut],
Dimitri Justeau [aut],
Florian de Boissieu [aut],
Lukas Baumbach [aut],
Philippe Birnbaum [aut]

Repository CRAN

Date/Publication 2025-04-14 10:20:05 UTC

Contents

Algorithm.SDM-class . . . 3
ensemble . . . 3
Ensemble.SDM-class . . . 6
ensemble_modelling . . . 7
Env . . . 13
evaluate . . . 14
gui . . . 16
load.model . . . 17
load_occ . . . 17
load_var . . . 19
mapDiversity . . . 20
modelling . . . 21
Occurrences . . . 27
plot.model . . . 28
project . . . 29
save.model . . . 31
SSDM . . . 32
Stacked.SDM-class . . . 33
stacking . . . 34
stack_modelling . . . 37
update,Stacked.SDM-method . . . 44

Index **47**

Algorithm.SDM-class *An S4 class to represent an SDM based on a single algorithm*

Description

This is an S4 class to represent an SDM based on a single algorithm (including generalized linear model, general additive model, multivariate adpative splines, generalized boosted regression model, classification tree analysis, random forest, maximum entropy, artificial neural network, and support vector machines). This S4 class is obtained with `modelling`.

Slots

`name` character. Name of the SDM (by default Species.SDM).

`projection` raster. Habitat suitability map produced by the SDM.

`binary` raster. Presence/Absence binary map produced by the SDM.

`evaluation` data frame. Evaluation of the SDM (available metrics include AUC, Kappa, sensitivity, specificity and proportion of correctly predicted occurrences) and identification of the optimal threshold to convert the habitat suitability map into a binary presence/absence map.

`variable.importance` data frame. Relative importance of each variable in the SDM.

`data` data frame. Data used to build the SDM.

`parameters` data frame. Parameters used to build the SDM.

See Also

Ensemble.SDM an S4 class for ensemble SDMs, and Stacked.SDM an S4 class for SSDMs.

ensemble *Methods to assemble multiple algorithms in an ensemble SDM*

Description

This is a method to assemble several algorithms in an ensemble SDM. The function takes as inputs several S4 Algorithm.SDM class objects returned by the `modelling` function. The function returns an S4 Ensemble.SDM class object containing the habitat suitability map, the binary map, and the uncertainty map (based on the between-algorithm variance) and the associated evaluation tables (model evaluation, algorithm evaluation, algorithm correlation matrix and variable importance).

Usage

```
ensemble(
  x,
  ...,
  name = NULL,
  ensemble.metric = c("AUC"),
  ensemble.thresh = c(0.75),
  weight = TRUE,
  thresh = 1001,
  uncertainty = TRUE,
  SDM.projections = FALSE,
  cores = 0,
  verbose = TRUE,
  GUI = FALSE
)

## S4 method for signature 'Algorithm.SDM'
ensemble(
  x,
  ...,
  name = NULL,
  ensemble.metric = c("AUC"),
  ensemble.thresh = c(0.75),
  weight = TRUE,
  thresh = 1001,
  uncertainty = TRUE,
  SDM.projections = FALSE,
  cores = 0,
  verbose = TRUE,
  GUI = FALSE
)

## S4 method for signature 'Algorithm.SDM'
sum(
  x,
  ...,
  name = NULL,
  ensemble.metric = c("AUC"),
  ensemble.thresh = c(0.75),
  weight = TRUE,
  thresh = 1001,
  format = TRUE,
  verbose = TRUE,
  na.rm = TRUE
)
```

Arguments

x, ...	SDMs. SDMs to be assembled.
name	character. Optional name given to the final Ensemble.SDM produced (by default 'Ensemble.SDM').
ensemble.metric	character. Metric(s) used to select the best SDMs that will be included in the ensemble SDM (see details below).
ensemble.thresh	numeric. Threshold(s) associated with the metric(s) used to compute the selection.
weight	logical. If TRUE, SDMs are weighted using the ensemble metric or, alternatively, the mean of the selection metrics.
thresh	numeric. A integer value specifying the number of equal interval threshold values between 0 and 1.
uncertainty	logical. If TRUE, generates an uncertainty map and an algorithm correlation matrix.
SDM.projections	logical. If FALSE (default), the Algorithm.SDMs inside the 'sdms' slot will not contain projections (for memory saving purposes).
cores	integer. Specify the number of CPU cores used to do the computing. You can use detectCores) to automatically
verbose	logical. If set to true, allows the function to print text in the console.
GUI, format, na.rm	logical. Do not take those arguments into account (parameters for the user interface and sum function).

Details

ensemble.metric (metric(s) used to select the best SDMs that will be included in the ensemble SDM) can be chosen from among:

AUC Area under the receiver operating characteristic (ROC) curve

Kappa Kappa from the confusion matrix

sensitivity Sensitivity from the confusion matrix

specificity Specificity from the confusion matrix

prop.correct Proportion of correctly predicted occurrences from the confusion matrix

calibration Calibration metric (Naimi & Araujo 2016)

Value

an S4 Ensemble.SDM class object viewable with the plot.model function.

See Also

ensemble_modelling to build an ensemble SDM from multiple algorithms.

Examples

```
## Not run:
# Loading data
data(Env)
data(Occurrences)
Occurrences <- subset(Occurrences, Occurrences$SPECIES == 'elliptica')

# ensemble SDM building
CTA <- modelling('CTA', Occurrences, Env, Xcol = 'LONGITUDE', Ycol = 'LATITUDE')
SVM <- modelling('SVM', Occurrences, Env, Xcol = 'LONGITUDE', Ycol = 'LATITUDE')
ESDM <- ensemble(CTA, SVM, ensemble.thresh = c(0.6))

# Results plotting
plot(ESDM)

## End(Not run)
```

Ensemble.SDM-class *An S4 class to represent an ensemble SDM*

Description

This is an S4 class to represent an ensemble SDM from multiple algorithms (including generalized linear model, general additive model, multivariate adaptive splines, generalized boosted regression model, classification tree analysis, random forest, maximum entropy, artificial neural network, and support vector machines). This S4 class is returned by `ensemble_modelling` or `ensemble`.

Slots

`uncertainty` raster. Between-algorithm variance map.

`algorithm.correlation` data frame. Between-algorithm correlation matrix.

`algorithm.evaluation` data frame. Evaluation of the ensemble SDM (available

`sdms` list. Individual SDMs used to create the ESDM. metrics include AUC, Kappa, sensitivity, specificity and proportion of correctly predicted occurrences) and identification of the optimal threshold to convert the habitat suitability map into a binary presence/absence map.

See Also

Algorithm.SDM an S4 class to represent an SDM based on a single algorithm, and Stacked.SDM an S4 class for SSDMs.

`ensemble_modelling` *Build an ensemble SDM that assembles multiple algorithms*

Description

Build an ensemble SDM that assembles multiple algorithms for a single species. The function takes as inputs an occurrence data frame made of presence/absence or presence-only records and a raster object for data extraction and projection. The function returns an S4 Ensemble.SDM class object containing the habitat suitability map, the binary map, the between-algorithm variance map and the associated evaluation tables (model evaluation, algorithm evaluation, algorithm correlation matrix and variable importance).

Usage

```
ensemble_modelling(
  algorithms,
  Occurrences,
  Env,
  Xcol = "Longitude",
  Ycol = "Latitude",
  Pcol = NULL,
  rep = 10,
  name = NULL,
  save = FALSE,
  path = getwd(),
  cores = 0,
  parmode = "replicates",
  PA = NULL,
  cv = "holdout",
  cv.param = c(0.7, 1),
  final.fit.data = "all",
  bin.thresh = "SES",
  metric = NULL,
  thresh = 1001,
  axes.metric = "Pearson",
  uncertainty = TRUE,
  tmp = FALSE,
  SDM.projections = FALSE,
  ensemble.metric = c("AUC"),
  ensemble.thresh = c(0.75),
  weight = TRUE,
  verbose = TRUE,
  GUI = FALSE,
  ...
)
```

Arguments

algorithms	character. A character vector specifying the algorithm name(s) to be run (see details below).
Occurrences	data frame. Occurrences table (can be processed first by load_occ).
Env	raster object. RasterStack object of environmental variables (can be processed first by load_var).
Xcol	character. Name of the column in the occurrence table containing Latitude or X coordinates.
Ycol	character. Name of the column in the occurrence table containing Longitude or Y coordinates.
Pcol	character. Name of the column in the occurrence table specifying whether a line is a presence or an absence. A value of 1 is presence and value of 0 is absence. If NULL presence-only dataset is assumed.
rep	integer. Number of repetitions for each algorithm.
name	character. Optional name given to the final Ensemble.SDM produced (by default 'Ensemble.SDM').
save	logical. If TRUE, the ensemble SDM is automatically saved.
path	character. If save is If TRUE, the path to the directory in which the ensemble SDM will be saved.
cores	integer. Specify the number of CPU cores used to do the computing. You can use detectCores) to automatically
parmode	character. Parallelization mode: along 'algorithms' or 'replicates'. Defaults to 'replicates'.
PA	list(nb, strat) defining the pseudo-absence selection strategy used in case of presence-only dataset. If PA is NULL, recommended PA selection strategy is used depending on the algorithm (see details below).
cv	character. Method of cross-validation used to evaluate the ensemble SDM (see details below).
cv.param	numeric. Parameters associated to the method of cross-validation used to evaluate the ensemble SDM (see details below).
final.fit.data	strategy used for fitting the final/evaluated Algorithm.SDMs: 'holdout'= use same train and test data as in (last) evaluation, 'all'= train model with all data (i.e. no test data) or numeric (0-1)= sample a custom training fraction (left out fraction is set aside as test data)
bin.thresh	character. Classification threshold (threshold) used to binarize model predictions into presence/absence and compute the confusion matrix (see details below).
metric	(deprecated) character. Classification threshold (SDMTools::optim.thresh) used to binarize model predictions into presence/absence and compute the confusion matrix (see details below). This argument is only kept for backwards compatibility, if possible please use bin.thresh instead.
thresh	(deprecated) integer. Number of equally spaced thresholds in the interval 0-1 (SDMTools::optim.thresh). Only needed when metric is set.

`axes.metric` Metric used to evaluate variable relative importance (see details below).

`uncertainty` logical. If `TRUE`, generates an uncertainty map and an algorithm correlation matrix.

`tmp` logical or character. If `FALSE`, no temporary rasters are written (this could quickly fill up your working memory, if many replicates are modelled). If `TRUE`, temporary rasters are written to the „tmp“ directory of your R environment. If `character`, temporary rasters are written to a custom path. But beware: if you close R, temporary files will be deleted. To avoid any loss you can save your ensemble SDM with `save.model`. Depending on number, resolution and extent of models, temporary files can take a lot of disk space. Temporary files are written to the R environment temporary folder.

`SDM.projections` logical. If FALSE (default), the Algorithm.SDMs inside the 'sdms' slot will not contain projections (for memory saving purposes).

`ensemble.metric` character. Metric(s) used to select the best SDMs that will be included in the ensemble SDM (see details below).

`ensemble.thresh` numeric. Threshold(s) associated with the metric(s) used to compute the selection.

`weight` logical. If `TRUE`, SDMs are weighted using the ensemble metric or, alternatively, the mean of the selection metrics.

`verbose` logical. If `TRUE`, allows the function to print text in the console.

`GUI` logical. Do not take this argument into account (parameter for the user interface).

`...` additional parameters for the algorithm modelling function (see details below).

Details

algorithms 'all' calls all the following algorithms. Algorithms include Generalized linear model (**GLM**), Generalized additive model (**GAM**), Multivariate adaptive regression splines (**MARS**), Generalized boosted regressions model (**GBM**), Classification tree analysis (**CTA**), Random forest (**RF**), Maximum entropy (**MAXENT**), Artificial neural network (**ANN**), and Support vector machines (**SVM**). Each algorithm has its own parameters settable with the **...** (see each algorithm section below to set their parameters).

"PA" list with two values: **nb** number of pseudo-absences selected, and **strat** strategy used to select pseudo-absences: either random selection or disk selection. We set default recommendation from Barbet-Massin et al. (2012) (see reference).

cv **Cross-validation** method used to split the occurrence dataset used for evaluation: **holdout** data are partitioned into a training set and an evaluation set using a fraction (*cv.param[1]*) and the operation can be repeated (*cv.param[2]*) times, **k-fold** data are partitioned into k (*cv.param[1]*) folds being k-1 times in the training set and once the evaluation set and the operation can be repeated (*cv.param[2]*) times, **LOO** (Leave One Out) each point is successively taken as evaluation data.

metric Choice of the metric used to compute the binary map threshold and the confusion matrix (by default SES as recommended by Liu et al. (2005), see reference below): **Kappa** maximizes the Kappa, **CCR** maximizes the proportion of correctly predicted observations, **TSS** (True Skill Statistic) maximizes the sum of sensitivity and specificity, **SES** uses the sensitivity-specificity equality, **LW** uses the lowest occurrence prediction probability, **ROC** minimizes the distance between the ROC plot (receiving operating characteristic curve) and the upper left corner (1,1).

axes.metric Metric used to evaluate the variable relative importance (difference between a full model and one with each variable successively omitted): **Pearson** (computes a simple Pearson's correlation *r* between predictions of the full model and the one without a variable, and returns the score *1-r*: the highest the value, the more influence the variable has on the model), **AUC**, **Kappa**, **sensitivity**, **specificity**, and **prop.correct** (proportion of correctly predicted occurrences).

ensemble.metric Ensemble metric(s) used to select SDMs: **AUC**, **Kappa**, **sensitivity**, **specificity**, and **prop.correct** (proportion of correctly predicted occurrences).

"..." See algorithm in detail section

Value

an S4 Ensemble.SDM class object viewable with the `plot.model` function.

Generalized linear model (GLM)

Uses the `glm` function from the package 'stats'. You can set parameters by supplying `glm.args=list(arg1=val1,arg2=val2` (see `glm` for all settable arguments). The following parameters have defaults:

test character. Test used to evaluate the SDM, default 'AIC'.

control list (created with `glm.control`). Contains parameters for controlling the fitting process. Default is `glm.control(epsilon = 1e-08, maxit = 500)`. 'epsilon' is a numeric and defines the positive convergence tolerance (eps). The iterations converge when $|dev - dev\_old|/(|dev| + 0.1) < eps$. 'maxit' is an integer giving the maximal number of IWLS (Iterative Weighted Last Squares) iterations.

Generalized additive model (GAM)

Uses the `gam` function from the package 'mgcv'. You can set parameters by supplying `gam.args=list(arg1=val1,arg2=val` (see `gam` for all settable arguments). The following parameters have defaults:

test character. Test used to evaluate the model, default 'AIC'.

control list (created with `gam.control`). Contains parameters for controlling the fitting process. Default is `gam.control(epsilon = 1e-08, maxit = 500)`. 'epsilon' is a numeric used for judging the conversion of the GLM IRLS (Iteratively Reweighted Least Squares) loop. 'maxit' is an integer giving the maximum number of IRLS iterations to perform.

Multivariate adaptive regression splines (MARS)

Uses the `earth` function from the package 'earth'. You can set parameters by supplying `mars.args=list(arg1=val1,arg2=` (see `earth` for all settable arguments). The following parameters have defaults:

degree integer. Maximum degree of interaction (Friedman's mi) ; 1 meaning build an additive model (i.e., no interaction terms). By default, set to 2.

Generalized boosted regressions model (GBM)

Uses the gbm function from the package 'gbm'. You can set parameters by supplying gbm.args=list(arg1=val1,arg2=val2 (see gbm for all settable arguments). The following parameters have defaults:

distribution character. Automatically detected from the format of the presence column in the occurrence dataset.

n.trees integer. The total number of trees to fit. This is equivalent to the number of iterations and the number of basis functions in the additive expansion. By default, set to 2500.

n.minobsinnode integer. minimum number of observations in the trees terminal nodes. Note that this is the actual number of observations, not the total weight. By default, set to 1.

cv.folds integer. Number of cross-validation folds to perform. If cv.folds>1 then gbm - in addition to the usual fit - will perform a cross-validation. By default, set to 3.

shrinkage numeric. A shrinkage parameter applied to each tree in the expansion (also known as learning rate or step-size reduction). By default, set to 0.001.

bag.fraction numeric. Fraction of the training set observations randomly selected to propose the next tree in the expansion.

train.fraction numeric. Training fraction used to fit the first gbm. The remainder is used to compute out-of-sample estimates of the loss function. By default, set to 1 (since evaluation/holdout is done with SSDM::evaluate.

n.cores integer. Number of cores to use for parallel computation of the CV folds. By default, set to 1. If you intend to use this, please set ncores=0 to avoid conflicts.

Classification tree analysis (CTA)

Uses the rpart function from the package 'rpart'. You can set parameters by supplying cta.args=list(arg1=val1,arg2=va (see rpart for all settable arguments). The following parameters have defaults:

control list (created with rpart.control). Contains parameters for controlling the rpart fit. The default is rpart.control(minbucket=1, xval=3). 'mibucket' is an integer giving the minimum number of observations in any terminal node. 'xval' is an integer defining the number of cross-validations.

Random Forest (RF)

Uses the randomForest function from the package 'randomForest'. You can set parameters by supplying cta.args=list(arg1=val1,arg2=val2) (see randomForest all settable arguments). The following parameters have defaults:

ntree integer. Number of trees to grow. This should not be set to a too small number, to ensure that every input row gets predicted at least a few times. By default, set to 2500.

nodesize integer. Minimum size of terminal nodes. Setting this number larger causes smaller trees to be grown (and thus take less time). By default, set to 1.

Maximum Entropy (MAXENT)

Uses the `maxent` function from the package 'dismo'. Make sure that you have correctly installed the maxent.jar file in the folder ~\R\library\version\dismo\java available at https://biodiversityinformatics.amnh.org/open_source/maxent/. As with the other algorithms, you can set parameters by supplying `maxent.args=list(arg1=val1,arg2=val2)`. Mind that arguments are passed from dismo to the MAXENT software again as an argument list (see `maxent` for more details). No specific defaults are set with this method.

Artificial Neural Network (ANN)

Uses the `nnet` function from the package 'nnet'. You can set parameters by supplying `ann.args=list(arg1=val1,arg2=val` (see `nnet` for all settable arguments). The following parameters have defaults:

size integer. Number of units in the hidden layer. By default, set to 6.

maxit integer. Maximum number of iterations, default 500.

Support vector machines (SVM)

Uses the `svm` function from the package 'e1071'. You can set parameters by supplying `svm.args=list(arg1=val1,arg2=va` (see `svm` for all settable arguments). The following parameters have defaults:

type character. Regression/classification type SVM should be used with. By default, set to "eps-regression".

epsilon float. Epsilon parameter in the insensitive loss function, default 1e-08.

cross integer. If an integer value k>0 is specified, a k-fold cross-validation on the training data is performed to assess the quality of the model: the accuracy rate for classification and the Mean Squared Error for regression. By default, set to 3.

kernel character. The kernel used in training and predicting. By default, set to "radial".

gamma numeric. Parameter needed for all kernels, default `1/(length(data) -1)`.

Warning

Depending on the raster object resolution the process can be more or less time and memory consuming.

References

M. Barbet-Massin, F. Jiguet, C. H. Albert, & W. Thuiller (2012) "Selecting pseudo-absences for species distribution models: how, where and how many?" *Methods Ecology and Evolution* 3:327-338 http://onlinelibrary.wiley.com/doi/10.1111/j.2041-210X.2011.00172.x/full

C. Liu, P. M. Berry, T. P. Dawson, R. & G. Pearson (2005) "Selecting thresholds of occurrence in the prediction of species distributions." *Ecography* 28:85-393 https://www.researchgate.net/publication/230246974_Selecting_Thresholds_of_Occurrence_in_the_Prediction_of_Species_Distributions

See Also

`modelling` to build SDMs with a single algorithm, `stack_modelling` to build SSDMs.

Examples

```
## Not run:
# Loading data
data(Env)
data(Occurrences)
Occurrences <- subset(Occurrences, Occurrences$SPECIES == 'elliptica')

# ensemble SDM building
ESDM <- ensemble_modelling(c('CTA', 'MARS'), Occurrences, Env, rep = 1,
                           Xcol = 'LONGITUDE', Ycol = 'LATITUDE',
                           ensemble.thresh = c(0.6))

# Results plotting
plot(ESDM)

## End(Not run)
```

Env *A stack of three environmental variables*

Description

A stack of three 30 arcsec-resolution rasters covering the north part of the main island of New Caledonia 'Grande Terre'. CRAINFALL and TEMPERATURE rasters are climatic variables from the WorldClim database, and SUBSTRATE raster is from the IRD Atlas of New Caledonia (2012) (see reference below).

Usage

```
Env
```

Format

A stack of three rasters:

RAINFALL Annual mean rainfall (mm)

TEMPERATURE Annual mean temperature (x10 degree Celsius)

SUBSTRATE Substrate type (categorical variable)

References

R.J. Hijmans, C.H. & Graham (2006) "The ability of climate envelope models to predict the effect of climate change on species distributions." *Global Change Biology* 12:2272-2281 https://onlinelibrary.wiley.com/doi/full/10.1111/j.1365-2486.2006.01256.x

E. Fritsch (2012) "Les sols. Atlas de la Nouvelle-Caledonie (ed. by J. Bonvallot, J.-C. Gay and E. Habert)" *IRD-Congres de la Nouvelle-Caledonie, Marseille.* 73-76

evaluate *Evaluate*

Description

Evaluation of SDM or ESDM habitat suitability predictions or evalaution of SSDM floristic composition with Pottier et al, 2013 method (see reference below)

Usage

```
evaluate(obj, ...)

## S4 method for signature 'Algorithm.SDM'
evaluate(
  obj,
  cv,
  cv.param,
  final.fit.data = "all",
  bin.thresh = "SES",
  metric = NULL,
  thresh = 1001,
  Env,
  ...
)

## S4 method for signature 'MAXENT.SDM'
evaluate(
  obj,
  cv,
  cv.param,
  final.fit.data = "all",
  bin.thresh = "SES",
  metric = NULL,
  thresh = 1001,
  Env,
  ...
)

## S4 method for signature 'Stacked.SDM'
evaluate(obj, ...)
```

Arguments

obj	Stacked.SDM. SSDM to evaluate
...	arguments for internal use (get_model), such as argument lists to be passed to the source functions (e.g. glm.args=list(test="AIC",singular.ok=FALSE))

cv	character. Method of cross-validation used to evaluate the SDM (see details below).
cv.param	numeric. Parameters associated to the method of cross-validation used to evaluate the SDM (see details below).
final.fit.data	strategy used for fitting the final model to be returned: 'holdout'= use same train and test data as in (last) evaluation, 'all'= train model with all data (i.e. no test data) or numeric (0-1)= sample a custom training fraction (left out fraction is set aside as test data)
bin.thresh	character. Classification threshold (threshold) used to binarize model predictions into presence/absence and compute the confusion matrix (including related scores such as TPR, TNR, omission rate, Kappa, etc.).
metric	(deprecated) character. Classification threshold (SDMTools::optim.thresh) used to binarize model predictions into presence/absence and compute the confusion matrix (including related scores such as TPR, TNR, omission rate, Kappa, etc.).
thresh	(deprecated) integer. Number of equally spaced thresholds in the interval 0-1 (SDMTools::optim.thresh).
Env	raster object. Stacked raster object of environmental variables (can be processed first by load_var).

Value

SDM/ESDM/SSDM evaluation in a data.frame

References

Pottier, J., Dubuis, A., Pellissier, L., Maiorano, L., Rossier, L., Randin, C. F., Guisan, A. (2013). The .accuracy of plant assemblage prediction from species distribution models varies along environmental gradients. Global Ecology and Biogeography, 22(1), 52-63. https://doi.org/10.1111/j.1466-8238.2012.00790.x

Examples

```
## Not run:
# Loading data
data(Env)
data(Occurrences)
# SSDM building
SSDM <- stack_modelling(c('CTA', 'SVM'), Occurrences, Env, rep = 1,
                        Xcol = 'LONGITUDE', Ycol = 'LATITUDE',
                        Spcol = 'SPECIES')

# Evaluation
evaluate(SSDM)


## End(Not run)
```

gui *SSDM package Graphic User Interface*

Description

User interface of the SSDM package.

Usage

```
gui(
  port = getOption("shiny.port"),
  host = getOption("shiny.host", "127.0.0.1"),
  working.directory = getwd()
)
```

Arguments

port	char. The TCP port that the application should listen on (see runApp for more details).
host	char. The IPv4 address that the application should listen on (see runApp for more details).
working.directory	char. Directory in which the application will run.

Details

If your environmental variables have an important size, you should give enough memory to the interface with the (maxmem parameter). Note that only one instance of gui can be run at a time.

Value

Open a window with a shiny app to use the SSDM package with an user-friendly interface.

Examples

```
## Not run:
gui()

## End(Not run)
```

load.model *Load ensemble SDMs and SSDMs.*

Description

Load S4 Ensemble.SDM and Stacked.SDM objects saved with their respective save function.

Usage

```
load_esdm(name, path = getwd())

load_stack(name = "Stack", path = getwd(), GUI = FALSE)
```

Arguments

name	character. Name of the folder containing the model to be loaded.
path	character. Path to the directory containing the model to be loaded, by default the path to the current directory.
GUI	logical. Do not take this argument into account (parameter for the user interface).

Value

The corresponding SDM object.

See Also

save.model

load_occ *Load occurrence data*

Description

Load occurrence data from CSV file to perform modelling, ensemble_modelling or stack_modelling.

Usage

```
load_occ(
  path = getwd(),
  Env,
  file = NULL,
  ...,
  Xcol = "Longitude",
  Ycol = "Latitude",
```

```
  Spcol = NULL,
  GeoRes = TRUE,
  reso = max(res(Env@layers[[1]])),
  verbose = TRUE,
  GUI = FALSE
)
```

Arguments

path	character. Path to the directory that contains the occurrence table.
Env	raster stack. Environmental variables in the form of a raster stack used to perform spatial thinning (can be the result of the load_var function).
file	character. File containing the occurrence table, if NULL (default) the .csv file located in the path will be loaded.
...	additional parameters given to read.csv.
Xcol	character. Name of the Latitude or X coordinate variable.
Ycol	character. Name of the Longitude or Y coordinate variable.
Spcol	character. Name of the column containing species names or IDs.
GeoRes	logical. If TRUE, performs geographical thinning on occurrences to limit geographical biases in the SDMs.
reso	numeric. Resolution used to perform the geographical thinning, default is the resolution of Env.
verbose	logical. If TRUE, allows the function to print text in the console.
GUI	logical. Parameter reserved for graphical interface.

Value

A data frame containing the occurrence dataset (spatially thinned or not).

See Also

load_var to load environmental variables.

Examples

```
## Not run:
load_occ(path = system.file('extdata',  package = 'SSDM'), Env,
         Xcol = 'LONGITUDE', Ycol = 'LATITUDE',
         file = 'Occurrences.csv', sep = ',')

## End(Not run)
```

load_var *Load environmental variables*

Description

Function to load environmental variables in the form of rasters to perform modelling, ensemble_modelling or stack_modelling.

Usage

```
load_var(
  path = getwd(),
  files = NULL,
 format = c(".grd", ".tif", ".asc", ".sdat", ".rst", ".nc", ".envi", ".bil", ".img"),
  categorical = NULL,
  Norm = FALSE,
  tmp = TRUE,
  verbose = TRUE,
  GUI = FALSE
)
```

Arguments

path	character. Path to the directory that contains the environmental variables files.
files	character. Files containing the environmental variables If NULL (default) all files present in the path in the selected format will be loaded.
format	character. Format of environmental variables files (including .grd, .tif, .asc, .sdat, .rst, .nc, .tif, .envi, .bil, .img).
categorical	character. Specify environmental variables that are categorical.
Norm	logical. Default FALSE. If set to TRUE, normalizes environmental variables into a range between 0 and 1.
tmp	logical. If set to TRUE, rasters are read in temporary file avoiding to overload the random access memory. But beware: if you close R, temporary files will be deleted.
verbose	logical. If set to TRUE, allows the function to print text in the console.
GUI	logical. Do not take that argument into account (parameter for the user interface).

Value

A stack containing the environmental rasters (normalized or not).

See Also

load_occ to load occurrences.

Examples

```
## Not run:
load_var(system.file('extdata',  package = 'SSDM'))

## End(Not run)
```

`mapDiversity` *Map Diversity*

Description

Methods for Stacked.SDM or SSDM to map diversity and communities composition.

Usage

```
mapDiversity(obj, ...)

## S4 method for signature 'Stacked.SDM'
mapDiversity(obj, method, rep.B = 1000, verbose = TRUE, Env = NULL, ...)
```

Arguments

`obj`	Stacked.SDM. SSDM to map diversity with.
`...`	other arguments pass to the method.
`method`	character. Define the method used to create the local species richness map (see details below).
`rep.B`	integer. If the method used to create the local species richness is the random Bernoulli (**Bernoulli**), rep.B parameter defines the number of repetitions used to create binary maps for each species.
`verbose`	logical. If set to true, allows the function to print text in the console.
`Env`	raster object. Stacked raster object of environmental variables (can be processed first by `load_var`). Needed only for stacking method using probability ranking from richness (**PRR**).

Details

Methods: Choice of the method used to compute the local species richness map (see Calabrese et al. (2014) and D'Amen et al (2015) for more informations, see reference below):

pSSDM sum probabilities of habitat suitability maps

Bernoulli draw repeatedly from a Bernoulli distribution

bSSDM sum the binary map obtained with the thresholding (depending on the metric of the ESDM).

MaximumLikelihood adjust species richness of the model by linear regression

PRR.MEM model richness with a macroecological model (MEM) and adjust each ESDM binary map by ranking habitat suitability and keeping as much as predicted richness of the MEM

PRR.pSSDM model richness with a pSSDM and adjust each ESDM binary map by ranking habitat suitability and keeping as much as predicted richness of the pSSDM

Value

a list with a diversity map and eventually ESDMs for stacking method using probability ranking from richness (**PPR**).

References

M. D'Amen, A. Dubuis, R. F. Fernandes, J. Pottier, L. Pelissier, & A Guisan (2015) "Using species richness and functional traits prediction to constrain assemblage predicitions from stacked species distribution models" *Journal of Biogeography* 42(7):1255-1266 http://doc.rero.ch/record/235561/files/pel_usr.pdf

J.M. Calabrese, G. Certain, C. Kraan, & C.F. Dormann (2014) "Stacking species distribution models and adjusting bias by linking them to macroecological models." *Global Ecology and Biogeography* 23:99-112 https://onlinelibrary.wiley.com/doi/full/10.1111/geb.12102

See Also

stacking to build SSDMs.

Examples

```
## Not run:
# Loading data
data(Env)
data(Occurrences)
# SSDM building
SSDM <- stack_modelling(c('CTA', 'SVM'), Occurrences, Env, rep = 1,
                        Xcol = 'LONGITUDE', Ycol = 'LATITUDE',
                        Spcol = 'SPECIES')

# Diversity mapping
mapDiversity(SSDM, mathod = 'pSSDM')


## End(Not run)
```

---

modelling *Build an SDM using a single algorithm*

Description

This is a function to build an SDM with one algorithm for a single species. The function takes as inputs an occurrence data frame made of presence/absence or presence-only records and a raster object for data extraction and projection. The function returns an S4 Algorithm.SDM class object containing the habitat suitability map, the binary map and the evaluation table.

Usage

```
modelling(
  algorithm,
  Occurrences,
  Env,
  Xcol = "Longitude",
  Ycol = "Latitude",
  Pcol = NULL,
  name = NULL,
  PA = NULL,
  cv = "holdout",
  cv.param = c(0.7, 2),
  final.fit.data = "all",
  bin.thresh = "SES",
  metric = NULL,
  thresh = 1001,
  axes.metric = "Pearson",
  select = FALSE,
  select.metric = c("AUC"),
  select.thresh = c(0.75),
  verbose = TRUE,
  GUI = FALSE,
  ...
)
```

Arguments

algorithm	character. Choice of the algorithm to be run (see details below).
Occurrences	data frame. Occurrence table (can be processed first by load_occ).
Env	raster object. Raster object of environmental variable (can be processed first by load_var).
Xcol	character. Name of the column in the occurrence table containing Latitude or X coordinates.
Ycol	character. Name of the column in the occurrence table containing Longitude or Y coordinates.
Pcol	character. Name of the column in the occurrence table specifying whether a line is a presence or an absence. A value of 1 is presence and value of 0 is absence. If NULL presence-only dataset is assumed.
name	character. Optional name given to the final SDM produced (by default 'Algorithm.SDM').

PA	list(nb, strat) defining the pseudo-absence selection strategy used in case of presence-only dataset. If PA is NULL, recommended PA selection strategy is used depending on the algorithms (see details below).
cv	character. Method of cross-validation used to evaluate the SDM (see details below).
cv.param	numeric. Parameters associated to the method of cross-validation used to evaluate the SDM (see details below).
final.fit.data	strategy used for fitting the final model to be returned: 'holdout'= use same train and test data as in (last) evaluation, 'all'= train model with all data (i.e. no test data) or numeric (0-1)= sample a custom training fraction (left out fraction is set aside as test data)
bin.thresh	character. Classification threshold (threshold) used to binarize model predictions into presence/absence and compute the confusion matrix (see details below).
metric	(deprecated) character. Classification threshold (SDMTools::optim.thresh) used to binarize model predictions into presence/absence and compute the confusion matrix (see details below). This argument is only kept for backwards compatibility, if possible please use bin.thresh instead.
thresh	(deprecated) integer. Number of equally spaced thresholds in the interval 0-1 (SDMTools::optim.thresh). Only needed when metric is set.
axes.metric	Metric used to evaluate variable relative importance (see details below).
select	logical. If set to true, models are evaluated before being projected, and not kept if they don't meet selection criteria (see details below).
select.metric	character. Metric(s) used to pre-select SDMs that reach a sufficient quality (see details below).
select.thresh	numeric. Threshold(s) associated with the metric(s) used to compute the selection.
verbose	logical. If set to true, allows the function to print text in the console.
GUI	logical. Don't take that argument into account (parameter for the user interface).
...	additional parameters, e.g. argument lists for the source algorithm modelling functions (see details below).

Details

algorithm 'all' allows to call directly all available algorithms. Currently, available algorithms include Generalized linear model (**GLM**), Generalized additive model (**GAM**), Multivariate adaptive regression splines (**MARS**), Generalized boosted regressions model (**GBM**), Classification tree analysis (**CTA**), Random forest (**RF**), Maximum entropy (**MAXENT**), Artificial neural network (**ANN**), and Support vector machines (**SVM**). Each algorithm has its own parameters settable with the **...** by supplying argument lists (see each algorithm section below to set their parameters).

'PA' list with two values: **nb** number of pseudo-absences selected, and **strat** strategy used to select pseudo-absences: either random selection or disk selection. We set default recommendation from Barbet-Massin et al. (2012) (see reference).

cv **Cross-validation** method used to split the occurrence dataset used for evaluation: **holdout** data are partitioned into a training set and an evaluation set using a fraction (*cv.param[1]*) and the operation can be repeated (*cv.param[2]*) times, **k-fold** data are partitioned into k (*cv.param[1]*) folds being k-1 times in the training set and once the evaluation set and the operation can be repeated (*cv.param[2]*) times, **LOO** (Leave One Out) each point is successively taken as evaluation data.

bin.thresh Choice of the metric used to binarize model predictions and compute the confusion matrix (by default SES as recommended by Liu et al. (2005), see reference below): **Kappa** maximizes the Kappa, **NOM** highest threshold without omission, **TSS** (True Skill Statistic) maximizes the sum of sensitivity and specificity, **SES** uses the sensitivity-specificity equality, **EP** threshold where modeled prevalence is closest to observed prevalence.

metric (deprecated) Choice of the metric used to compute the binary map threshold and the confusion matrix (by default SES as recommended by Liu et al. (2005), see reference below): **Kappa** maximizes the Kappa, **CCR** maximizes the proportion of correctly predicted observations, **TSS** (True Skill Statistic) maximizes the sum of sensitivity and specificity, **SES** uses the sensitivity-specificity equality, **LW** uses the lowest occurrence prediction probability, **ROC** minimizes the distance between the ROC plot (receiving operating curve) and the upper left corner (1,1).

axes.metric Choice of the metric used to evaluate the variable relative importance (difference between a full model and one with each variable successively omitted): **Pearson** (computes a simple Pearson's correlation *r* between predictions of the full model and the one without a variable, and returns the score *1-r*: the highest the value, the more influence the variable has on the model), **AUC**, **Kappa**, **sensitivity**, **specificity**, and **prop.correct** (proportion of correctly predicted occurrences).

select.metric Selection metric(s) used to select SDMs: **AUC**, **Kappa**, **sensitivity**, **specificity**, and **prop.correct** (proportion of correctly predicted occurrences), **calibration** (calibration statistic as used by Naimi & Araujo 2016).

'...' See algorithm in detail section

Value

an S4 Algorithm.SDM Class object viewable with the `plot.model` method.

Generalized linear model (GLM)

Uses the `glm` function from the package 'stats'. You can set parameters by supplying `glm.args=list(arg1=val1,arg2=val2` (see `glm` for all settable arguments). The following parameters have defaults:

test character. Test used to evaluate the SDM, default 'AIC'.

control list (created with `glm.control`). Contains parameters for controlling the fitting process. Default is `glm.control(epsilon = 1e-08, maxit = 500)`. 'epsilon' is a numeric and defines the positive convergence tolerance (eps). 'maxit' is an integer giving the maximal number of IWLS (Iterative Weighted Last Squares) iterations.

Generalized additive model (GAM)

Uses the `gam` function from the package 'mgcv'. You can set parameters by supplying `gam.args=list(arg1=val1,arg2=val` (see `gam` for all settable arguments). The following parameters have defaults:

test character. Test used to evaluate the model, default 'AIC'.

control list (created with gam.control). Contains parameters for controlling the fitting process. Default is gam.control(epsilon = 1e-08, maxit = 500). 'epsilon' is a numeric used for judging the conversion of the GLM IRLS (Iteratively Reweighted Least Squares) loop. 'maxit' is an integer giving the maximum number of IRLS iterations to perform.

Multivariate adaptive regression splines (MARS)

Uses the earth function from the package 'earth'. You can set parameters by supplying mars.args=list(arg1=val1,arg2= (see earth for all settable arguments). The following parameters have defaults:

degree integer. Maximum degree of interaction (Friedman's mi) ; 1 meaning build an additive model (i.e., no interaction terms). By default, set to 2.

Generalized boosted regressions model (GBM)

Uses the gbm function from the package 'gbm'. You can set parameters by supplying gbm.args=list(arg1=val1,arg2=val2 (see gbm for all settable arguments). The following parameters have defaults:

distribution character. Automatically detected from the format of the presence column in the occurrence dataset.

n.trees integer. The total number of trees to fit. This is equivalent to the number of iterations and the number of basis functions in the additive expansion. By default, set to 2500.

n.minobsinnode integer. minimum number of observations in the trees terminal nodes. Note that this is the actual number of observations, not the total weight. By default, set to 1.

cv.folds integer. Number of cross-validation folds to perform. If cv.folds>1 then gbm - in addition to the usual fit - will perform a cross-validation. By default, set to 3.

shrinkage numeric. A shrinkage parameter applied to each tree in the expansion (also known as learning rate or step-size reduction). By default, set to 0.001.

bag.fraction numeric. Fraction of the training set observations randomly selected to propose the next tree in the expansion.

train.fraction numeric. Training fraction used to fit the first gbm. The remainder is used to compute out-of-sample estimates of the loss function. By default, set to 1 (since evaluation/holdout is done with SSDM::evaluate.

n.cores integer. Number of cores to use for parallel computation of the CV folds. By default, set to 1. If you intend to use this, please set ncores=0 to avoid conflicts.

Classification tree analysis (CTA)

Uses the rpart function from the package 'rpart'. You can set parameters by supplying cta.args=list(arg1=val1,arg2=va (see rpart for all settable arguments). The following parameters have defaults:

control list (created with rpart.control). Contains parameters for controlling the rpart fit. The default is rpart.control(minbucket=1, xval=3). 'mibucket' is an integer giving the minimum number of observations in any terminal node. 'xval' is an integer defining the number of cross-validations.

Random Forest (RF)

Uses the `randomForest` function from the package 'randomForest'. You can set parameters by supplying `cta.args=list(arg1=val1,arg2=val2)` (see `randomForest` all settable arguments). The following parameters have defaults:

ntree integer. Number of trees to grow. This should not be set to a too small number, to ensure that every input row gets predicted at least a few times. By default, set to 2500.

nodesize integer. Minimum size of terminal nodes. Setting this number larger causes smaller trees to be grown (and thus take less time). By default, set to 1.

Maximum Entropy (MAXENT)

Uses the `maxent` function from the package 'dismo'. Make sure that you have correctly installed the maxent.jar file in the folder ~\R\library\version\dismo\java available at `https://biodiversityinformatics.amnh.org/open_source/maxent/`. As with the other algorithms, you can set parameters by supplying `maxent.args=list(arg1=val1,arg2=val2)`. Mind that arguments are passed from dismo to the MAXENT software again as an argument list (see `maxent` for more details). No specific defaults are set with this method.

Artificial Neural Network (ANN)

Uses the `nnet` function from the package 'nnet'. You can set parameters by supplying `ann.args=list(arg1=val1,arg2=val`
(see `nnet` for all settable arguments). The following parameters have defaults:

size integer. Number of units in the hidden layer. By default, set to 6.

maxit integer. Maximum number of iterations, default 500.

Support vector machines (SVM)

Uses the `svm` function from the package 'e1071'. You can set parameters by supplying `svm.args=list(arg1=val1,arg2=va`
(see `svm` for all settable arguments). The following parameters have defaults:

type character. Regression/classification type SVM should be used with. By default, set to "eps-regression".

epsilon float. Epsilon parameter in the insensitive loss function, default 1e-08.

cross integer. If an integer value k>0 is specified, a k-fold cross-validation on the training data is performed to assess the quality of the model: the accuracy rate for classification and the Mean Squared Error for regression. By default, set to 3.

kernel character. The kernel used in training and predicting. By default, set to "radial".

gamma numeric. Parameter needed for all kernels, default `1/(length(data) -1)`.

Warning

Depending on the raster object resolution the process can be more or less time and memory consuming.

References

M. Barbet-Massin, F. Jiguet, C. H. Albert, & W. Thuiller (2012) 'Selecting pseudo-absences for species distribution models: how, where and how many?' *Methods Ecology and Evolution* 3:327-338 https://besjournals.onlinelibrary.wiley.com/doi/full/10.1111/j.2041-210X.2011.00172.x

C. Liu, P. M. Berry, T. P. Dawson, R. & G. Pearson (2005) 'Selecting thresholds of occurrence in the prediction of species distributions.' *Ecography* 28:85-393 https://www.researchgate.net/publication/230246974_Selecting_Thresholds_of_Occurrence_in_the_Prediction_of_Species_Distributions

See Also

ensemble_modelling to build ensemble SDMs, stack_modelling to build SSDMs.

Examples

```
# Loading data
data(Env)
data(Occurrences)
Occurrences <- subset(Occurrences, Occurrences$SPECIES == 'elliptica')

# SDM building
SDM <- modelling('GLM', Occurrences, Env, Xcol = 'LONGITUDE', Ycol = 'LATITUDE')

# Results plotting
## Not run:
plot(SDM)

## End(Not run)
```

Occurrences *Plant occurrences data frame*

Description

A dataset containing plant occurrences of five Cryptocarya species native to New Caledonia. Occurrence data come from the Noumea Herbarium (NOU) and NC-PIPPN network (see Ibanez et al (2014) in reference below).

Usage

```
Occurrences
```

Format

A data frame with 57 rows and 3 variables:

SPECIES Species of the occurrence

LONGITUDE Longitude of the occurrence

LATITUDE Latitude of the occurrence

References

T. Ibanez, J. Munzinger, G. Dagostini, V. Hequet, F. Rigault, T. Jaffre, & P. Birnbaum (2014) "Structural and floristic characteristics of mixed rainforest in New Caledonia: new data from the New Caledonian Plant Inventory and Permanent Plot Network (NC-PIPPN)." *Applied Vegetation Science* 17:386-397

http://www.researchgate.net/profile/Jerome_Munzinger/publication/258499017_Structural_and_floristic_diversity_of_mixed_tropical_rain_forest_in_New_Caledonia_new_data_from_the_New_Caledonian_Plant_Inventory_and_Permanent_Plot_Network_%28NC-PIPPN%29/links/0deec52b8b1996488e000000.pdf

---

`plot.model` *Plot SDMs, ensemble SDMs, and SSDMs*

Description

Allows to plot S4 Algorithm.SDM, Ensemble.SDM and Stacked.SDM class objects.

Usage

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

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

Arguments

x	Object to be plotted (S4 Algorithm.SDM, Ensemble.SDM or Stacked.SDM object).
y, ...	Plot-based parameter not used.

Value

Open a window with a shiny app rendering all the results (habitat suitability map, binary map, evaluation table, variable importance and/or between-algorithm variance map, and/or algorithm evaluation, and/or algorithm correlation matrix and/or local species richness map) in a user-friendly interface.

`project` *Project model into environment*

Description

This is a collection of methods to project SDMs, ESDMs or SSDMs into the supplied environment. The function is used internally to calculate the input for the projection slot of .SDM classes but can also be used to project existing .SDM objects (see Details).

Usage

```
project(obj, Env, ...)

## S4 method for signature 'Algorithm.SDM'
project(obj, Env, output.format = "model", ...)

## S4 method for signature 'MAXENT.SDM'
project(obj, Env, output.format = "model", ...)

## S4 method for signature 'Ensemble.SDM'
project(
  obj,
  Env,
  uncertainty = TRUE,
  output.format = "model",
  SDM.projections = FALSE,
  cores = 0,
  minimal.memory = FALSE,
  tmp = FALSE,
  ...
)

## S4 method for signature 'Stacked.SDM'
project(
  obj,
  Env,
  method = NULL,
  uncertainty = TRUE,
  output.format = "model",
  SDM.projections = FALSE,
  cores = 0,
  minimal.memory = FALSE,
  tmp = FALSE,
  ...
)
```

Arguments

obj	Object of class Algorithm.SDM, Ensemble.SDM or Stacked.SDM. Model(s) to be projected.
Env	Raster stack. Updated environmental rasters to be used for projection.
...	arguments for internal use (get_model), such as argument lists to be passed to the source functions (e.g. glm.args=list(test="AIC",singular.ok=FALSE)). See modelling, algorithm section for more details.
output.format	character. If 'model' (default), the original .SDM object will be returned with updated projection slots. If 'rasters', the projected rasters will be returned as a list of rasters.
uncertainty	logical. If set to TRUE, generates an uncertainty map. If output.format is 'model' an algorithm correlation matrix is additionally returned.
SDM.projections	logical. If FALSE (default), the projections of the Algorithm.SDMs will not be returned (only applies to Ensemble.SDMs and Stack.SDMs).
cores	integer. Specify the number of CPU cores used to do the computing. You can use detectCores) to automatically use all the available CPU cores.
minimal.memory	logical. Only relevant if cores >1. If TRUE, only one model will be sent to each worker at a time, reducing used working memory.
tmp	logical or character. If FALSE, no temporary rasters are written. If TRUE, temporary rasters are written to the „tmp" directory of your R environment. If character, temporary rasters are written to a custom path. Very useful to reduce working memory consumption (use together with minimal.memory=TRUE for maximal effect). But beware: Depending on number, resolution and extent of models, temporary files can take a lot of disk space.
method	character. Define the method used to create the local species richness map (for details see stack_modelling). If NULL (default), the method used for building the SSDM is used.

Details

The function uses any S4 .SDM class object and a raster stack of environmental layers of the variables the model was trained with.

Value

Either returns the original .SDM object with updated projection slots (default) or if output.format = 'rasters' only returns the projections as Raster* objects or a list thereof.

save.model *Save ensemble SDMs and SSDMs*

Description

Allows to save S4 Ensemble.SDM and Stacked.SDM class objects.

Usage

```
save.esdm(
  esdm,
  name = strsplit(esdm@name, ".", fixed = TRUE)[[1]][1],
  path = getwd(),
  verbose = TRUE,
  GUI = FALSE
)

## S4 method for signature 'Ensemble.SDM'
save.esdm(
  esdm,
  name = strsplit(esdm@name, ".Ensemble.SDM", fixed = TRUE)[[1]][1],
  path = getwd(),
  verbose = TRUE,
  GUI = FALSE
)

save.stack(stack, name = "Stack", path = getwd(), verbose = TRUE, GUI = FALSE)

## S4 method for signature 'Stacked.SDM'
save.stack(stack, name = "Stack", path = getwd(), verbose = TRUE, GUI = FALSE)
```

Arguments

esdm	Ensemble.SDM. Ensemble SDM to be saved.
name	character. Folder name of the model to save.
path	character. Path to the directory chosen to save the SDM, by default the path to the current directory.
verbose	logical. If set to true, allows the function to print text in the console.
GUI	logical. Don't take that argument into account (parameter for the user interface).
stack	Stacked.SDM. SSDM to be saved.

Value

Nothing in R environment. Creates folders, tables and rasters associated to the SDM. Tables are in .csv and rasters in .grd/.gri.

See Also

load.model

SSDM *SSDM: Stacked species distribution modelling*

Description

SSDM is a package to map species richness and endemism based on Stacked Species Distribution Models (SSDM). It provides tools to build SDM, i.e. a single species fitted with a single algorithm, Ensemble SDM (ESDM), i.e. a single species fitted with multiple algorithms, SSDM several species with one or more algorithms. The package includes numerous modelling algorithms, and specifiable ensemble aggregating and stacking methods. This package also provides tools to evaluate and explore models such as variable importance, algorithm accuracy, and between-algorithm correlation, and tools to map results such as habitat suitability maps, binary maps, between-algorithm variance maps. For ease of use, the SSDM package provides a user-friendly graphical interface (gui).

Details

The SSDM package provides five categories of functions (that you can find in details below): Data preparation, Modelling main functions, Model main methods, Model classes, and Miscellaneous.

Data preparation

load_occ Load occurrence data

load_var Load environmental variables

Modelling main functions

modelling Build an SDM using a single algorithm

ensemble_modelling Build an SDM that assembles multiple algorithms

stack_modelling Build an SSDMs that assembles multiple algorithms and species

Model main methods

ensemble,Algorithm.SDM-method Build an ensemble SDM

stacking,Ensemble.SDM-method Build an SSDM

update,Stacked.SDM-method Update a previous SSDM with new occurrence data

Model classes

Algorithm.SDM S4 class to represent SDMs

Ensemble.SDM S4 class to represent ensemble SDMs

Stacked.SDM S4 class to represent SSDMs

Miscellaneous

`gui` User-friendly interface for SSDM package

`plot.model` Plot SDMs

`save.model` Save SDMs

`load.model` Load SDMs

Author(s)

Maintainer: Sylvain Schmitt <sylvain.m.schmitt@gmail.com>

Authors:

- Robin Pouteau
- Dimitri Justeau
- Florian de Boissieu
- Lukas Baumbach
- Philippe Birnbaum

See Also

Useful links:

- https://github.com/sylvainschmitt/SSDM
- Report bugs at https://github.com/sylvainschmitt/SSDM/issues

---

`Stacked.SDM-class` *An S4 class to represent SSDMs*

Description

This is an S4 class to represent SSDMs that assembles multiple algorithms (including generalized linear model, general additive model, multivariate adaptive splines, generalized boosted regression model, classification tree analysis, random forest, maximum entropy, artificial neural network, and support vector machines) built for multiple species. It is obtained with `stack_modelling` or `stacking`.

Slots

`name` character. Name of the SSDM (by default 'Species.SSDM').

`diversity.map` raster. Local species richness map produced by the SSDM.

`endemism.map` raster. Endemism map produced by the SSDM (see Crisp et al (2011) in references).

`uncertainty` raster. Between-algorithm variance map.

`evaluation` data frame. Evaluation of the SSDM (AUC, Kappa, omission rate, sensitivity, specificity, proportion of correctly predicted occurrences).

`variable.importance` data frame. Relative importance of each variable in the SSDM.

`algorithm.correlation` data frame. Between-algorithm correlation matrix.

`esdms` list. List of ensemble SDMs used in the SSDM.

`parameters` data frame. Parameters used to build the SSDM.

`algorithm.evaluation` data frame. Evaluation of the algorithm averaging the metrics of all SDMs (AUC, Kappa, omission rate, sensitivity, specificity, proportion of correctly predicted occurrences).

References

M. D. Crisp, S. Laffan, H. P. Linder & A. Monro (2001) "Endemism in the Australian flora" *Journal of Biogeography* 28:183-198 http://biology-assets.anu.edu.au/hosted_sites/Crisp/pdfs/Crisp2001_endemism.pdf

See Also

Ensemble.SDM an S4 class to represent ensemble SDMs, and Algorithm.SDM an S4 class to represent SDMs.

`stacking` *Stack different ensemble SDMs in an SSDM*

Description

This is a function to stack several ensemble SDMs in an SSDM. The function takes as inputs several S4 Ensemble.SDM class objects produced with `ensemble_modelling` or `ensemble` functions. The function returns an S4 Stacked.SDM class object containing the local species richness map, the between-algorithm variance map, and all evaluation tables coming with (model evaluation, algorithm evaluation, algorithm correlation matrix and variable importance), and a list of ensemble SDMs for each species (see `ensemble_modelling`).

Usage

```
stacking(
  esdm,
  ...,
  name = NULL,
  method = "pSSDM",
  rep.B = 1000,
  Env = NULL,
  range = NULL,
  endemism = c("WEI", "Binary"),
  eval = TRUE,
  uncertainty = TRUE,
  verbose = TRUE,
  GUI = FALSE
```

```
)

## S4 method for signature 'Ensemble.SDM'
stacking(
  esdm,
  ...,
  name = NULL,
  method = "pSSDM",
  rep.B = 1000,
  Env = NULL,
  range = NULL,
  endemism = c("WEI", "Binary"),
  eval = TRUE,
  uncertainty = TRUE,
  verbose = TRUE,
  GUI = FALSE
)
```

Arguments

esdm, ...	character. Ensemble SDMs to be stacked.
name	character. Optional name given to the final SSDM produced (by default 'Species.SDM').
method	character. Define the method used to create the local species richness map (see details below).
rep.B	integer. If the method used to create the local species richness is the random bernoulli (**Bernoulli**), rep.B parameter defines the number of repetitions used to create binary maps for each species.
Env	raster object. Stacked raster object of environmental variables (can be processed first by load_var). Needed only for stacking method using probability ranking from richness (**PRR**).
range	integer. Set a value of range restriction (in pixels) around presences occurrences on habitat suitability maps (all further points will have a null probability, see Crisp et al (2011) in references). If NULL, no range restriction will be applied.
endemism	character. Define the method used to create an endemism map (see details below).
eval	logical. If set to FALSE, disable stack evaluation.
uncertainty	logical. If set to TRUE, generates an uncertainty map and an algorithm correlation matrix.
verbose	logical. If set to TRUE, allows the function to print text in the console.
GUI	logical. Don't take that argument into account (parameter for the user interface).

Details

Methods: Choice of the method used to compute the local species richness map (see Calabrese et al. (2014) and D'Amen et al (2015) for more informations, see reference below):

pSSDM sum probabilities of habitat suitability maps

Bernoulli draw repeatedly from a Bernoulli distribution

bSSDM sum the binary map obtained with the thresholding (depending on the metric of the ESDM).

MaximumLikelihood adjust species richness of the model by linear regression

PRR.MEM model richness with a macroecological model (MEM) and adjust each ESDM binary map by ranking habitat suitability and keeping as much as predicted richness of the MEM

PRR.pSSDM model richness with a pSSDM and adjust each ESDM binary map by ranking habitat suitability and keeping as much as predicted richness of the pSSDM

Endemism: Choice of the method used to compute the endemism map (see Crisp et al. (2001) for more information, see reference below):

NULL No endemism map

WEI (Weighted Endemism Index) Endemism map built by counting all species in each cell and weighting each by the inverse of its range

CWEI (Corrected Weighted Endemism Index) Endemism map built by dividing the weighted endemism index by the total count of species in the cell.

First string of the character is the method either WEI or CWEI, and in those cases second string of the vector is used to precise range calculation, whether the total number of occurrences **'NbOcc'** whether the surface of the binary map species distribution **'Binary'**.

Value

an S4 Stacked.SDM class object viewable with the `plot.model` function.

References

M. D'Amen, A. Dubuis, R. F. Fernandes, J. Pottier, L. Pelissier, & A Guisan (2015) "Using species richness and functional traits prediction to constrain assemblage predicitions from stacked species distribution models" *Journal of Biogeography* 42(7):1255-1266 http://doc.rero.ch/record/235561/files/pel_usr.pdf

J.M. Calabrese, G. Certain, C. Kraan, & C.F. Dormann (2014) "Stacking species distribution models and adjusting bias by linking them to macroecological models." *Global Ecology and Biogeography* 23:99-112 https://onlinelibrary.wiley.com/doi/full/10.1111/geb.12102

M. D. Crisp, S. Laffan, H. P. Linder & A. Monro (2001) "Endemism in the Australian flora" *Journal of Biogeography* 28:183-198 http://biology-assets.anu.edu.au/hosted_sites/Crisp/pdfs/Crisp2001_endemism.pdf

C. Liu, P. M. Berry, T. P. Dawson, R. & G. Pearson (2005) "Selecting thresholds of occurrence in the prediction of species distributions." *Ecography* 28:85-393 https://www.researchgate.net/publication/230246974_Selecting_Thresholds_of_Occurrence_in_the_Prediction_of_Species_Distributions

See Also

`stack_modelling` to build SSDMs.

Examples

```
## Not run:
# Loading data
data(Env)
data(Occurrences)
Occ1 <- subset(Occurrences, Occurrences$SPECIES == 'elliptica')
Occ2 <- subset(Occurrences, Occurrences$SPECIES == 'gracilis')

# SSDM building
ESDM1 <- ensemble_modelling(c('CTA', 'SVM'), Occ1, Env, rep = 1,
                            Xcol = 'LONGITUDE', Ycol = 'LATITUDE',
                            name = 'elliptica', ensemble.thresh = c(0.6))
ESDM2 <- ensemble_modelling(c('CTA', 'SVM'), Occ2, Env, rep = 1,
                            Xcol = 'LONGITUDE', Ycol = 'LATITUDE',
                            name = 'gracilis', ensemble.thresh = c(0.6))
SSDM <- stacking(ESDM1, ESDM2)

# Results plotting
plot(SSDM)

## End(Not run)
```

`stack_modelling` *Build an SSDM that assembles multiple algorithms and species.*

Description

This is a function to build an SSDM that assembles multiple algorithm and species. The function takes as inputs an occurrence data frame made of presence/absence or presence-only records and a raster object for data extraction and projection. The function returns an S4 Stacked.SDM class object containing the local species richness map, the between-algorithm variance map, and all evaluation tables coming with (model evaluation, algorithm evaluation, algorithm correlation matrix and variable importance), and a list of ensemble SDMs for each species (see `ensemble_modelling`).

Usage

```
stack_modelling(
  algorithms,
  Occurrences,
  Env,
  Xcol = "Longitude",
  Ycol = "Latitude",
  Pcol = NULL,
  Spcol = "SpeciesID",
  rep = 10,
  name = NULL,
  save = FALSE,
```

```
  path = getwd(),
  PA = NULL,
  cv = "holdout",
  cv.param = c(0.7, 1),
  final.fit.data = "all",
  bin.thresh = "SES",
  metric = NULL,
  thresh = 1001,
  axes.metric = "Pearson",
  uncertainty = TRUE,
  tmp = FALSE,
  SDM.projections = FALSE,
  ensemble.metric = c("AUC"),
  ensemble.thresh = c(0.75),
  weight = TRUE,
  method = "pSSDM",
  rep.B = 1000,
  range = NULL,
  endemism = c("WEI", "Binary"),
  verbose = TRUE,
  GUI = FALSE,
  cores = 0,
  parmode = "species",
  ...
)
```

Arguments

algorithms	character. Choice of the algorithm(s) to be run (see details below).
Occurrences	data frame. Occurrence table (can be processed first by load_occ).
Env	raster object. Raster object of environmental variables (can be processed first by load_var).
Xcol	character. Name of the column in the occurrence table containing Latitude or X coordinates.
Ycol	character. Name of the column in the occurrence table containing Longitude or Y coordinates.
Pcol	character. Name of the column in the occurrence table specifying whether a line is a presence or an absence. A value of 1 is presence and value of 0 is absence. If NULL presence-only dataset is assumed.
Spcol	character. Name of the column containing species names or IDs.
rep	integer. Number of repetitions for each algorithm.
name	character. Optional name given to the final Ensemble.SDM produced.
save	logical. If set to true, the SSDM is automatically saved.
path	character. If save is true, the path to the directory in which the ensemble SDM will be saved.

PA — list(nb, strat) defining the pseudo-absence selection strategy used in case of presence-only dataset. If PA is NULL, recommended PA selection strategy is used depending on the algorithm (see details below).

cv — character. Method of cross-validation used to evaluate the ensemble SDM (see details below).

cv.param — numeric. Parameters associated with the method of cross-validation used to evaluate the ensemble SDM (see details below).

final.fit.data — strategy used for fitting the final/evaluated Algorithm.SDMs: 'holdout'= use same train and test data as in (last) evaluation, 'all'= train model with all data (i.e. no test data) or numeric (0-1)= sample a custom training fraction (left out fraction is set aside as test data)

bin.thresh — character. Classification threshold (threshold) used to binarize model predictions into presence/absence and compute the confusion matrix (see details below).

metric — (deprecated) character. Classification threshold (SDMTools::optim.thresh) used to binarize model predictions into presence/absence and compute the confusion matrix (see details below). This argument is only kept for backwards compatibility, if possible please use bin.thresh instead.

thresh — (deprecated) integer. Number of equally spaced thresholds in the interval 0-1 (SDMTools::optim.thresh). Only needed when metric is set.

axes.metric — Metric used to evaluate variable relative importance (see details below).

uncertainty — logical. If set to true, generates an uncertainty map and an algorithm correlation matrix.

tmp — logical. If set to true, the habitat suitability map of each algorithms is saved in a temporary file to release memory. But beware: if you close R, temporary files will be deleted. To avoid any loss you can save your SSDM with save.model. Depending on number, resolution and extent of models, temporary files can take a lot of disk space. Temporary files are written in R environment temporary folder.

SDM.projections — logical. If FALSE (default), the Algorithm.SDMs inside the 'sdms' slot will not contain projections (for memory saving purposes).

ensemble.metric — character. Metric(s) used to select the best SDMs that will be included in the ensemble SDM (see details below).

ensemble.thresh — numeric. Threshold(s) associated with the metric(s) used to compute the selection.

weight — logical. Choose whether or not you want the SDMs to be weighted using the selection metric or, alternatively, the mean of the selection metrics.

method — character. Define the method used to create the local species richness map (see details below).

rep.B — integer. If the method used to create the local species richness is the random bernoulli (**Bernoulli**), rep.B parameter defines the number of repetitions used to create binary maps for each species.

range	integer. Set a value of range restriction (in pixels) around presences occurrences on habitat suitability maps (all further points will have a null probability, see Crisp et al (2011) in references). If NULL, no range restriction will be applied.
endemism	character. Define the method used to create an endemism map (see details below).
verbose	logical. If set to true, allows the function to print text in the console.
GUI	logical. Don't take that argument into account (parameter for the user interface).
cores	integer. Specify the number of CPU cores used to do the computing. You can use detectCores) to automatically use all the available CPU cores.
parmode	character. Parallelization mode: along 'species', 'algorithms' or 'replicates'. Defaults to 'species'.
...	additional parameters for the algorithm modelling function (see details below).

Details

algorithms 'all' allows you to call directly all available algorithms. Currently, available algorithms include Generalized linear model (**GLM**), Generalized additive model (**GAM**), Multivariate adaptive regression splines (**MARS**), Generalized boosted regressions model (**GBM**), Classification tree analysis (**CTA**), Random forest (**RF**), Maximum entropy (**MAXENT**), Artificial neural network (**ANN**), and Support vector machines (**SVM**). Each algorithm has its own parameters settable with the **...** (see each algorithm section below to set their parameters).

"PA" list with two values: **nb** number of pseudo-absences selected, and **strat** strategy used to select pseudo-absences: either random selection or disk selection. We set default recommendation from Barbet-Massin et al. (2012) (see reference).

cv **Cross-validation** method used to split the occurrence dataset used for evaluation: **holdout** data are partitioned into a training set and an evaluation set using a fraction (*cv.param[1]*) and the operation can be repeated (*cv.param[2]*) times, **k-fold** data are partitioned into k (*cv.param[1]*) folds being k-1 times in the training set and once the evaluation set and the operation can be repeated (*cv.param[2]*) times, **LOO** (Leave One Out) each point is successively taken as evaluation data.

metric Choice of the metric used to compute the binary map threshold and the confusion matrix (by default SES as recommended by Liu et al. (2005), see reference below): **Kappa** maximizes the Kappa, **CCR** maximizes the proportion of correctly predicted observations, **TSS** (True Skill Statistic) maximizes the sum of sensitivity and specificity, **SES** uses the sensitivity-specificity equality, **LW** uses the lowest occurrence prediction probability, **ROC** minimizes the distance between the ROC plot (receiving operating curve) and the upper left corner (1,1).

axes.metric Choice of the metric used to evaluate the variable relative importance (difference between a full model and one with each variable successively omitted): **Pearson** (computes a simple Pearson's correlation *r* between predictions of the full model and the one without a variable, and returns the score *1-r*: the highest the value, the more influence the variable has on the model), **AUC**, **Kappa**, **sensitivity**, **specificity**, and **prop.correct** (proportion of correctly predicted occurrences).

ensemble.metric Ensemble metric(s) used to select SDMs: **AUC**, **Kappa**, **sensitivity**, **specificity**, and **prop.correct** (proportion of correctly predicted occurrences).

method Choice of the method used to compute the local species richness map (see Calabrese et al. (2014) and D'Amen et al (2015) for more informations, see reference below): **pSSDM** sum probabilities of habitat suitability maps, **Bernoulli** drawing repeatedly from a Bernoulli distribution, **bSSDM** sum the binary map obtained with the thresholding (depending on the metric, see metric parameter), **MaximumLikelihood** adjust species richness using maximum likelihood parameter estimates on the logit-transformed occurrence probabilities (see Calabrese et al. (2014)), **PRR.MEM** model richness with a macroecological model (MEM) and adjust each ESDM binary map by ranking habitat suitability and keeping as much as predicted richness of the MEM, **PRR.pSSDM** model richness with a pSSDM and adjust each ESDM binary map by ranking habitat suitability and keeping as much as predicted richness of the pSSDM

endemism Choice of the method used to compute the endemism map (see Crisp et al. (2001) for more information, see reference below): **NULL** No endemism map, **WEI** (Weighted Endemism Index) Endemism map built by counting all species in each cell and weighting each by the inverse of its range, **CWEI** (Corrected Weighted Endemism Index) Endemism map built by dividing the weighted endemism index by the total count of species in the cell. First string of the character is the method either WEI or CWEI, and in those cases second string of the vector is used to precise range calculation, whether the total number of occurrences **'NbOcc'** whether the surface of the binary map species distribution **'Binary'**.

... See algorithm in detail section

Value

an S4 Stacked.SDM class object viewable with the `plot.model` function.

Generalized linear model (GLM)

Uses the `glm` function from the package 'stats'. You can set parameters by supplying `glm.args=list(arg1=val1,arg2=val2` (see `glm` for all settable arguments). The following parameters have defaults:

test character. Test used to evaluate the SDM, default 'AIC'.

control list (created with `glm.control`). Contains parameters for controlling the fitting process. Default is `glm.control(epsilon = 1e-08, maxit = 500)`. 'epsilon' is a numeric and defines the positive convergence tolerance (eps). 'maxit' is an integer giving the maximal number of IWLS (Iterative Weighted Last Squares) iterations.

Generalized additive model (GAM)

Uses the `gam` function from the package 'mgcv'. You can set parameters by supplying `gam.args=list(arg1=val1,arg2=val` (see `gam` for all settable arguments). The following parameters have defaults:

test character. Test used to evaluate the model, default 'AIC'.

control list (created with `gam.control`). Contains parameters for controlling the fitting process. Default is `gam.control(epsilon = 1e-08, maxit = 500)`. 'epsilon' is a numeric used for judging the conversion of the GLM IRLS (Iteratively Reweighted Least Squares) loop. 'maxit' is an integer giving the maximum number of IRLS iterations to perform.

Multivariate adaptive regression splines (MARS)

Uses the `earth` function from the package 'earth'. You can set parameters by supplying `mars.args=list(arg1=val1,arg2=` (see `earth` for all settable arguments). The following parameters have defaults:

degree integer. Maximum degree of interaction (Friedman's mi) ; 1 meaning build an additive model (i.e., no interaction terms). By default, set to 2.

Generalized boosted regressions model (GBM)

Uses the `gbm` function from the package 'gbm'. You can set parameters by supplying `gbm.args=list(arg1=val1,arg2=val2` (see `gbm` for all settable arguments). The following parameters have defaults:

distribution character. Automatically detected from the format of the presence column in the occurrence dataset.

n.trees integer. The total number of trees to fit. This is equivalent to the number of iterations and the number of basis functions in the additive expansion. By default, set to 2500.

n.minobsinnode integer. minimum number of observations in the trees terminal nodes. Note that this is the actual number of observations, not the total weight. By default, set to 1.

cv.folds integer. Number of cross-validation folds to perform. If cv.folds>1 then gbm - in addition to the usual fit - will perform a cross-validation. By default, set to 3.

shrinkage numeric. A shrinkage parameter applied to each tree in the expansion (also known as learning rate or step-size reduction). By default, set to 0.001.

bag.fraction numeric. Fraction of the training set observations randomly selected to propose the next tree in the expansion.

train.fraction numeric. Training fraction used to fit the first gbm. The remainder is used to compute out-of-sample estimates of the loss function. By default, set to 1 (since evaluation/holdout is done with `SSDM::evaluate`.

n.cores integer. Number of cores to use for parallel computation of the CV folds. By default, set to 1. If you intend to use this, please set `ncores=0` to avoid conflicts.

Classification tree analysis (CTA)

Uses the `rpart` function from the package 'rpart'. You can set parameters by supplying `cta.args=list(arg1=val1,arg2=va` (see `rpart` for all settable arguments). The following parameters have defaults:

control list (created with `rpart.control`). Contains parameters for controlling the rpart fit. The default is `rpart.control(minbucket=1, xval=3)`. 'mibucket' is an integer giving the minimum number of observations in any terminal node. 'xval' is an integer defining the number of cross-validations.

Random Forest (RF)

Uses the `randomForest` function from the package 'randomForest'. You can set parameters by supplying `cta.args=list(arg1=val1,arg2=val2)` (see `randomForest` all settable arguments). The following parameters have defaults:

ntree integer. Number of trees to grow. This should not be set to a too small number, to ensure that every input row gets predicted at least a few times. By default, set to 2500.

nodesize integer. Minimum size of terminal nodes. Setting this number larger causes smaller trees to be grown (and thus take less time). By default, set to 1.

Maximum Entropy (MAXENT)

Uses the `maxent` function from the package 'dismo'. Make sure that you have correctly installed the maxent.jar file in the folder ~\R\library\version\dismo\java available at https://biodiversityinformatics.amnh.org/open_source/maxent/. As with the other algorithms, you can set parameters by supplying `maxent.args=list(arg1=val1,arg2=val2)`. Mind that arguments are passed from dismo to the MAXENT software again as an argument list (see `maxent` for more details). No specific defaults are set with this method.

Artificial Neural Network (ANN)

Uses the `nnet` function from the package 'nnet'. You can set parameters by supplying `ann.args=list(arg1=val1,arg2=val`
(see `nnet` for all settable arguments). The following parameters have defaults:

size integer. Number of units in the hidden layer. By default, set to 6.

maxit integer. Maximum number of iterations, default 500.

Support vector machines (SVM)

Uses the `svm` function from the package 'e1071'. You can set parameters by supplying `svm.args=list(arg1=val1,arg2=va`
(see `svm` for all settable arguments). The following parameters have defaults:

type character. Regression/classification type SVM should be used with. By default, set to "eps-regression".

epsilon float. Epsilon parameter in the insensitive loss function, default 1e-08.

cross integer. If an integer value k>0 is specified, a k-fold cross-validation on the training data is performed to assess the quality of the model: the accuracy rate for classification and the Mean Squared Error for regression. By default, set to 3.

kernel character. The kernel used in training and predicting. By default, set to "radial".

gamma numeric. Parameter needed for all kernels, default `1/(length(data) -1)`.

Warning

Depending on the raster object resolution the process can be more or less time and memory consuming.

References

M. D'Amen, A. Dubuis, R. F. Fernandes, J. Pottier, L. Pelissier, & A Guisan (2015) "Using species richness and functional traits prediction to constrain assemblage predicitions from stacked species distribution models" *Journal of Biogeography* 42(7):1255-1266 http://doc.rero.ch/record/235561/files/pel_usr.pdf

M. Barbet-Massin, F. Jiguet, C. H. Albert, & W. Thuiller (2012) "Selecting pseudo-absences for species distribution models: how, where and how many?" *Methods Ecology and Evolution* 3:327-338 http://onlinelibrary.wiley.com/doi/10.1111/j.2041-210X.2011.00172.x/full

J.M. Calabrese, G. Certain, C. Kraan, & C.F. Dormann (2014) "Stacking species distribution models and adjusting bias by linking them to macroecological models." *Global Ecology and Biogeography* 23:99-112 https://onlinelibrary.wiley.com/doi/full/10.1111/geb.12102

M. D. Crisp, S. Laffan, H. P. Linder & A. Monro (2001) "Endemism in the Australian flora" *Journal of Biogeography* 28:183-198 http://biology-assets.anu.edu.au/hosted_sites/Crisp/pdfs/Crisp2001_endemism.pdf

C. Liu, P. M. Berry, T. P. Dawson, R. & G. Pearson (2005) "Selecting thresholds of occurrence in the prediction of species distributions." *Ecography* 28:85-393 https://www.researchgate.net/publication/230246974_Selecting_Thresholds_of_Occurrence_in_the_Prediction_of_Species_Distributions

See Also

modelling to build simple SDMs.

Examples

```
## Not run:
# Loading data
data(Env)
data(Occurrences)

# SSDM building
SSDM <- stack_modelling(c('CTA', 'SVM'), Occurrences, Env, rep = 1,
                        Xcol = 'LONGITUDE', Ycol = 'LATITUDE',
                        Spcol = 'SPECIES')

# Results plotting
plot(SSDM)

## End(Not run)
```

---

update,Stacked.SDM-method

Update a previous SSDM

Description

Update a previous SSDM with new occurrence data. The function takes as inputs updated or new occurrence data from one species, previous environmental variables, and an S4 Stacked.SDM class object containing a previously built SSDM.

Usage

```
## S4 method for signature 'Stacked.SDM'
update(
  object,
```

```
  Occurrences,
  Env,
  Xcol = "Longitude",
  Ycol = "Latitude",
  Pcol = NULL,
  Spname = NULL,
  name = stack@name,
  save = FALSE,
  path = getwd(),
  thresh = 1001,
  tmp = FALSE,
  verbose = TRUE,
  GUI = FALSE,
  ...
)
```

Arguments

object	Stacked.SDM. The previously built SSDM.
Occurrences	data frame. New or updated occurrence table (can be processed first by load_occ).
Env	raster object. Environment raster object (can be processed first by load_var).
Xcol	character. Name of the column in the occurrence table containing Latitude or X coordinates.
Ycol	character. Name of the column in the occurrence table containing Longitude or Y coordinates.
Pcol	character. Name of the column in the occurrence table specifying whether a line is a presence or an absence. A value of 1 is presence and value of 0 is absence. If NULL presence-only dataset is assumed.
Spname	character. Name of the new or updated species.
name	character. Optional name given to the final SSDM produced, by default it's the name of the previous SSDM.
save	logical. If set to true, the model is automatically saved.
path	character. Name of the path to the directory to contain the saved SSDM.
thresh	numeric. A single integer value representing the number of equal interval threshold values between 0 and 1.
tmp	logical. If set to true, the habitat suitability map of each algorithm is saved in a temporary file to release memory. But beware: if you close R, temporary files will be deleted To avoid any loss you can save your model with save.model.
verbose	logical. If set to true, allows the function to print text in the console.
GUI	logical. Don't take that argument into account (parameter for the user interface).
...	additional parameters for the algorithm modelling function (see details below).

Value

an S4 Stacked.SDM class object viewable with the plot.model function.

See Also

stack_modelling to build SSDMs.

Examples

```
## Not run:
update(stack, Occurrences, Env, Spname = 'NewSpecie')

## End(Not run)
```

Index

∗ **datasets**
Env, 13
Occurrences, 27

Algorithm.SDM, *3*, *6*, *22*, *24*, *28*, *32*, *34*
Algorithm.SDM-class, 3

detectCores, *5*, *8*, *30*, *40*

earth, *10*, *25*, *42*
ensemble, 3, *6*, *34*
ensemble,Algorithm.SDM-method, *32*
ensemble,Algorithm.SDM-method *(*ensemble*)*, 3
Ensemble.SDM, *3*, *5*, *7*, *10*, *17*, *28*, *31*, *32*, *34*
Ensemble.SDM-class, 6
ensemble_modelling, *5*, *6*, 7, *17*, *19*, *27*, *32*, *34*, *37*
Env, 13
evaluate, 14
evaluate,Algorithm.SDM-method *(*evaluate*)*, 14
evaluate,MAXENT.SDM-method *(*evaluate*)*, 14
evaluate,Stacked.SDM-method *(*evaluate*)*, 14

gam, *10*, *24*, *41*
gam.control, *10*, *25*, *41*
gbm, *11*, *25*, *42*
glm, *10*, *24*, *41*
glm.control, *10*, *24*, *41*
gui, 16, *32*, *33*

load.model, 17, *32*, *33*
load_esdm *(*load.model*)*, 17
load_occ, *8*, 17, *19*, *22*, *32*, *38*, *45*
load_stack *(*load.model*)*, 17
load_var, *8*, *15*, *18*, 19, *20*, *22*, *32*, *35*, *38*, *45*

mapDiversity, 20
mapDiversity,Stacked.SDM-method *(*mapDiversity*)*, 20
maxent, *12*, *26*, *43*
modelling, *3*, *12*, *17*, *19*, 21, *30*, *32*, *44*

nnet, *12*, *26*, *43*

Occurrences, 27

plot,SDM,ANY-method *(*plot.model*)*, 28
plot,Stacked.SDM,ANY-method *(*plot.model*)*, 28
plot.model, *5*, *10*, *24*, 28, *33*, *36*, *41*, *45*
project, 29
project,Algorithm.SDM-method *(*project*)*, 29
project,Ensemble.SDM-method *(*project*)*, 29
project,MAXENT.SDM-method *(*project*)*, 29
project,Stacked.SDM-method *(*project*)*, 29

randomForest, *11*, *26*, *42*
read.csv, *18*
rpart, *11*, *25*, *42*
rpart.control, *11*, *25*, *42*
runApp, *16*

save.esdm *(*save.model*)*, 31
save.esdm,Ensemble.SDM-method *(*save.model*)*, 31
save.model, *9*, *17*, 31, *33*, *39*, *45*
save.stack *(*save.model*)*, 31
save.stack,Stacked.SDM-method *(*save.model*)*, 31
SSDM, 32
SSDM-package *(*SSDM*)*, 32
stack_modelling, *12*, *17*, *19*, *27*, *30*, *32*, *33*, *36*, 37, *46*
Stacked.SDM, *3*, *6*, *17*, *28*, *31*, *32*, *34*, *36*, *37*, *41*, *44*, *45*
Stacked.SDM-class, 33

stacking, *21*, *33*, 34
stacking,Ensemble.SDM-method, *32*
stacking,Ensemble.SDM-method
(stacking), 34
sum,Algorithm.SDM-method (ensemble), 3
svm, *12*, *26*, *43*

threshold, *8*, *15*, *23*, *39*

update,Stacked.SDM-method, *32*, 44