Package 'caretSDM'

July 10, 2025

Type Package

Title Build Species Distribution Modeling using 'caret'

Version 1.1.0.1

Maintainer Luíz Fernando Esser <luizesser@gmail.com>

Description Use machine learning algorithms and advanced geographic information system tools to build Species Distribution Modeling in a extensible and modern fashion.

License MIT + file LICENSE

BugReports https://github.com/luizesser/caretSDM/issues

Imports caret, checkmate, cli, CoordinateCleaner, data.table, dismo, dplyr, fs, furrr, future, ggplot2, ggspatial, glue, gtools, httr, lwgeom, mapview, methods, parallelly, pdp, pROC, progressr, purrr, raster, rgbif, Rtsne, sf, stars, stats, stringdist, stringr, terra, tidyr, usdm, utils

Encoding UTF-8

LazyData true

RoxygenNote 7.3.2

Suggests e1071, kknn, mda, naivebayes, nnet, here, tibble, withr, roxyglobals, covr, knitr, rmarkdown, testthat (>= 3.0.0)

Config/testthat/edition 3

Depends R (>= 4.2.0)

URL https://luizesser.github.io/caretSDM/

Config/Needs/website rmarkdown

NeedsCompilation no

Author Dayani Bailly [aut] (ORCID: <https://orcid.org/0000-0002-6954-9902>), Edivando Couto [aut] (ORCID: <https://orcid.org/0000-0003-4264-8449>), José Hilário Delconte Ferreira [aut] (ORCID: <https://orcid.org/0000-0002-7116-2600>), Reginaldo Ré [aut] (ORCID: <https://orcid.org/0000-0001-6452-3466>), Valéria Batista [aut] (ORCID: <https://orcid.org/0000-0002-6574-7338>), Luíz Fernando Esser [aut, cre, cph] (ORCID: <https://orcid.org/0000-0003-2982-7223>)

Repository CRAN

Date/Publication 2025-07-10 13:30:02 UTC

Contents

add_predictors *Add predictors to* sdm_area

Description

This function includes new predictors to the sdm_area object.

Usage

```
add_predictors(sa, pred, variables_selected = NULL, gdal = TRUE)

get_predictors(i)
```

Arguments

sa	A sdm_area object.
pred	RasterStack, SpatRaster, stars or sf object with predictors data.
variables_selected	character vector with variables names in pred to be used as predictors. If NULL adds all variables.
gdal	Boolean. Force the use or not of GDAL when available. See details.
i	input_sdm or sdm_area object to retrieve data from.

Details

add_predictors returns a sdm_area object with a grid built upon the x parameter. There are two ways to make the grid and resample the variables in sdm_area: with and without gdal. As standard, if gdal is available in you machine it will be used (gdal = TRUE), otherwise sf/stars will be used.

Value

For add_predictors the same input sdm_area object is returned including the pred data binded to the previous grid. get_predictors retrieves the grid from the i object.

Author(s)

Luíz Fernando Esser (luizesser@gmail.com) and Reginaldo Ré. https://luizfesser.wordpress.com

See Also

sdm_area predictors bioc

Examples

```
# Create sdm_area object:
sa <- sdm_area(parana, cell_size = 25000, crs = 6933)

# Include predictors:
sa <- add_predictors(sa, bioc)

# Retrieve predictors data:
get_predictors(sa)
```

---

add_scenarios *Add scenarios to* sdm_area

Description

This function includes scenarios in the sdm_area object.

Usage

```
add_scenarios(sa, scen = NULL, scenarios_names = NULL, pred_as_scen = TRUE,
                     variables_selected = NULL, stationary = NULL)

set_scenarios_names(i, scenarios_names = NULL)

scenarios_names(i)

get_scenarios_data(i)

select_scenarios(i, scenarios_names = NULL)
```

Arguments

sa	A sdm_area or input_sdm object.
scen	RasterStack, SpatRaster or stars object. If NULL adds predictors as a scenario.
scenarios_names	Character vector with names of scenarios.
pred_as_scen	Logical. If TRUE adds the current predictors as a scenario.
variables_selected	Character vector with variables names in scen to be used as variables. If NULL adds all variables.
stationary	Names of variables from sa that should be used in scenarios as stationary variables.
i	A sdm_area or input_sdm object.

Details

The function add_scenarios adds scenarios to the sdm_area or input_sdm object. If scen has variables that are not present as predictors the function will use only variables present in both objects. stationary variables are those that don't change through the scenarios. It is useful for hidrological variables in fish habitat modeling, for example (see examples below). When adding multiple scenarios in multiple runs, the function will always add a new "current" scenario. To avoid that, set pred_as_scen = FALSE.

Value

add_scenarios returns the input sdm_area or input_sdm object with a new slot called scenarios with scen data as a list, where each slot of the list holds a scenario and each scenario is a sf object. set_scenarios_names sets new names for scenarios in sdm_area/input_sdm object. scenarios_names returns scenarios' names. get_scenarios_data retrieves scenarios data as a list of sf objects. select_scenarios selects scenarios from sdm_area/input_sdm object.

Author(s)

Luíz Fernando Esser (luizesser@gmail.com) https://luizfesser.wordpress.com

See Also

sdm_area input_sdm

Examples

```
# Create sdm_area object:
sa <- sdm_area(parana, cell_size = 100000, crs = 6933)

# Include predictors:
sa <- add_predictors(sa, bioc)

# Include scenarios:
sa <- add_scenarios(sa, scen[1:2]) |> select_predictors(c("bio1", "bio12"))

# Set scenarios names:
sa <- set_scenarios_names(sa, scenarios_names = c("future_1", "future_2",
                                                  "current"))
scenarios_names(sa)

# Get scenarios data:
scenarios_grid <- get_scenarios_data(sa)
scenarios_grid

# Select scenarios:
sa <- select_scenarios(sa, scenarios_names = c("future_1"))

# Setting stationary variables in scenarios:
sa <- sdm_area(rivs[c(1:200),], cell_size = 100000, crs = 6933, lines_as_sdm_area = TRUE) |>
  add_predictors(bioc) |>
  add_scenarios(scen, stationary = c("LENGTH_KM", "DIST_DN_KM"))
```

algorithms *Caret Algorithms*

Description

A `data.frame` with characteristics of each algorithm available in `caretSDM`. Each column is a different characteristics. This can be helpful for more experienced modelers select algorithms. See the source for a selection method using this data.

Usage

```
algorithms
```

Format

'algorithms' A `data.frame` with 230 rows and 60 columns:

X Algorithms names

Further columns Algorithms attributes

Source

<https://topepo.github.io/caret/models-clustered-by-tag-similarity.html>

bioc *Bioclimatic Variables*

Description

A `stars` object with bioclimatic variables (bio1, bio4 and bio12) for the Parana state in Brazil. Data obtained from WorldClim 2.1 at 10 arc-min resolution.

Usage

```
bioc
```

Format

'bioc' A `stars` with 1 attribute and 3 bands:

bio1 Annual Mean Temperature

bio4 Temperature Seasonality

bio12 Annual Precipitation

Source

<https://www.worldclim.org/>

`buffer_sdm` *Create buffer around occurrences*

Description

Create buffer around records in `occ_data` to be used as study area

Usage

```
buffer_sdm(occ_data, size = NULL, crs = NULL)
```

Arguments

`occ_data`	A `data.frame` object with species, decimalLongitude and decimalLatitude columns. Usually the output from `GBIF_data`.
`size`	`numeric`. The distance between the record and the margin of the buffer (i.e. buffer radius).
`crs`	`numeric`. Indicates which EPSG it the `occ_data` in.

Value

A `sf` buffer around `occ_data` records.

Author(s)

Luíz Fernando Esser (luizesser@gmail.com) https://luizfesser.wordpress.com

See Also

`GBIF_data`

Examples

```
# Create sdm_area object:
study_area <- buffer_sdm(occ, size=50000, crs=6933)
plot(study_area)
```

data_clean *Presence data cleaning routine*

Description

Data cleaning wrapper using CoordinateCleaner package.

Usage

```
data_clean(occ, pred = NULL,
           species = NA, lon = NA, lat = NA,
           capitals = TRUE,
           centroids = TRUE,
           duplicated = TRUE,
           identical = TRUE,
           institutions = TRUE,
           invalid = TRUE,
           terrestrial = TRUE,
           independent_test = TRUE)
```

Arguments

occ	A occurrences_sdm object or input_sdm.
pred	A sdm_area object. If occ is a input_sdm object with predictors data, than pred is obtained from it.
species	A character stating the name of the column with species names in occ (see details).
lon	A character stating the name of the column with longitude in occ (see details).
lat	A character stating the name of the column with latitude in occ (see details).
capitals	Boolean to turn on/off the exclusion from countries capitals coordinates (see ?cc_cap)
centroids	Boolean to turn on/off the exclusion from countries centroids coordinates (see ?cc_cen)
duplicated	Boolean to turn on/off the exclusion from duplicated records (see ?cc_dupl)
identical	Boolean to turn on/off the exclusion from records with identical lat/long values (see ?cc_equ)
institutions	Boolean to turn on/off the exclusion from biodiversity institutions coordinates (see ?cc_inst)
invalid	Boolean to turn on/off the exclusion from invalid coordinates (see ?cc_val)
terrestrial	Boolean to turn on/off the exclusion from coordinates falling on sea (see ?cc_sea)
independent_test	Boolean. If occ has independent test data, the data cleaning routine is also applied on it.

Details

If the user does not used GBIF_data function to obtain species records, the function may have problems to find which column from the presences table has species, longitude and latitude information. In this regard, we implemented the parameters species, lon and lat so the use can explicitly inform which columns should be used. If they remain as NA (standard) the function will try to guess which columns are the correct one.

Value

A occurrences_sdm object or input_sdm with cleaned presence data.

Author(s)

Luíz Fernando Esser (luizesser@gmail.com) https://luizfesser.wordpress.com

See Also

GBIF_data occurrences_sdm sdm_area input_sdm predictors

Examples

```
# Create sdm_area object:
sa <- sdm_area(parana, cell_size = 50000, crs = 6933)

# Include predictors:
sa <- add_predictors(sa, bioc) |> select_predictors(c("bio1", "bio12"))

# Create occurrences:
oc <- occurrences_sdm(occ, crs = 6933) |> join_area(sa)

# Create input_sdm:
i <- input_sdm(oc, sa)

# Clean coordinates (terrestrial is set to false to make the run quicker):
i <- data_clean(i, terrestrial = FALSE)
```

GBIF_data *Retrieve Species data from GBIF*

Description

This function is a wrapper to get records from GBIF using rgbif and return a data.frame ready to be used in caretSDM.

Usage

```
GBIF_data(s, file = NULL, as_df = FALSE, ...)
```

Arguments

s	character vector of species names.
file	character with file to save the output. If not informed, data will not be saved on folder.
as_df	Should the output be a dataframe? Default is FALSE, returning a occurrences object.
...	Arguments to pass on rgbif::occ_data().

Value

A data.frame with species occurrences data, or an occurrences object if as_df = FALSE.

Author(s)

Luíz Fernando Esser (luizesser@gmail.com) https://luizfesser.wordpress.com

References

https://www.gbif.org

Examples

```
# Select species names:
s <- c("Araucaria angustifolia", "Salminus brasiliensis")

# Run function:
oc <- GBIF_data(s)
```

---

gcms_ensembles *Ensemble GCMs into one scenario*

Description

An ensembling method to group different GCMs into one SSP scenario

Usage

```
gcms_ensembles(i, gcms = NULL)
```

Arguments

i	A input_sdm object.
gcms	GCM codes in scenarios_names(i) to group scenarios.

Value

A input_sdm object with grouped GCMs.

Author(s)

Luíz Fernando Esser (luizesser@gmail.com) https://luizfesser.wordpress.com

See Also

GBIF_data occurrences_sdm sdm_area input_sdm predictors

Examples

```
# Create sdm_area object:
set.seed(1)
sa <- sdm_area(parana, cell_size = 100000, crs = 6933)

# Include predictors:
sa <- add_predictors(sa, bioc)

# Include scenarios:
sa <- add_scenarios(sa, scen) |> select_predictors(c("bio1", "bio12"))

# Create occurrences:
oc <- occurrences_sdm(occ, crs = 6933) |> join_area(sa)

# Create input_sdm:
i <- input_sdm(oc, sa)

# Pseudoabsence generation:
i <- pseudoabsences(i, method="random", n_set = 2)

# Custom trainControl:
ctrl_sdm <- caret::trainControl(method = "boot",
                                number = 1,
                                classProbs = TRUE,
                                returnResamp = "all",
                                summaryFunction = summary_sdm,
                                savePredictions = "all")

# Train models:
i <- train_sdm(i,
               algo = c("naive_bayes"),
               ctrl=ctrl_sdm,
               variables_selected = c("bio1", "bio12")) |>
  suppressWarnings()

# Predict models:
i  <- predict_sdm(i, th=0.8)

#' # Ensemble GCMs:
i <- gcms_ensembles(i, gcms = c("ca", "mi"))
```

i

---

input_sdm input_sdm

Description

This function creates a new input_sdm object.

Usage

```
input_sdm(...)
```

Arguments

...	Data to be used in SDMs. Can be a occurrences and/or a sdm_area object.

Details

If sdm_area is used, it can include predictors and scenarios. In this case, input_sdm will detect and include as scenarios and predictors in the input_sdm output. Objects can be included in any order, since the function will work by detecting their classes. The returned object is used throughout the whole workflow to apply functions.

Value

A input_sdm object containing:

grid	sf with POLYGON geometry representing the grid for the study area or LINESTRING if sdm_area was built with a LINESTRING sf.
bbox	Four corners for the bounding box (class bbox): minimum value of X, minimum value of Y, maximum value of X, maximum value of Y
cell_size	numeric information regarding the size of the cell used to rescale variables to the study area, representing also the cell size in the grid.
epsg	character information about the EPSG used in all slots from sdm_area.
predictors	character vector with predictors names included in sdm_area.

Author(s)

Luiz Fernando Esser (luizesser@gmail.com) https://luizfesser.wordpress.com

See Also

occurrences_sdm sdm_area

Examples

```
# Create sdm_area object:
sa <- sdm_area(parana, cell_size = 50000, crs = 6933)

# Include predictors:
sa <- add_predictors(sa, bioc) |> select_predictors(c("bio1", "bio4", "bio12"))

# Include scenarios:
sa <- add_scenarios(sa, scen)

# Create occurrences:
oc <- occurrences_sdm(occ, crs = 6933) |> join_area(sa)

# Create input_sdm:
i <- input_sdm(oc, sa)
```

`is_input_sdm` `is_class` *functions to check caretSDM data classes.*

Description

This functions returns a boolean to check caretSDM object classes.

Usage

```
is_input_sdm(x)

is_sdm_area(x)

is_occurrences(x)

is_predictors(x)

is_scenarios(x)

is_models(x)

is_predictions(x)
```

Arguments

x	Object to be tested.

Value

Boolean.

Author(s)

Luíz Fernando Esser (luizesser@gmail.com) https://luizfesser.wordpress.com

Examples

```
# Create sdm_area object:
sa <- sdm_area(parana, cell_size = 25000, crs = 6933)

is_sdm_area(sa)

is_input_sdm(sa)
```

---

join_area *Join Area*

Description

Join cell_id data from sdm_area to a occurrences

Usage

```
join_area(occ, pred)
```

Arguments

occ	A occurrences object or input_sdm.
pred	A sdm_area object to retrieve cell_id from.

Details

This function is key in this SDM workflow. It attaches cell_id values to occ, deletes records outside pred and allows the use of pseudoabsences. This function also tests if CRS from both occ and pred are equal, otherwise the CRS of pred is used to convert occ.

Value

A occurrences object with cell_id to each record.

Author(s)

Luíz Fernando Esser (luizesser@gmail.com) https://luizfesser.wordpress.com

See Also

occurrences_sdm sdm_area input_sdm pseudoabsences

Examples

```
# Create sdm_area object:
sa <- sdm_area(parana, cell_size = 50000, crs = 6933)

# Include predictors:
sa <- add_predictors(sa, bioc) |> select_predictors(c("bio1", "bio4", "bio12"))

# Include scenarios:
sa <- add_scenarios(sa, scen)

# Create occurrences:
oc <- occurrences_sdm(occ, crs = 6933) |> join_area(sa)
```

---

occ *Araucaria angustifolia occurrence data*

Description

A `data.frame` object with Araucaria angustifolia occurrence data obtained from GBIF and filtered with Parana state `sf`.

Usage

```
occ
```

Format

'occ' A `data.frame` with 420 rows and 3 columns (EPSG:6933):

species Species name

decimalLongitude Longitude in meters

decimalLatitude Latitude in meters

Source

<https://www.gbif.org>

occurrences_sdm *Occurrences Managing*

Description

This function creates and manage occurrences objects.

Usage

```
occurrences_sdm(x,
                independent_test = NULL,
                p = 0.1,
                crs = NULL,
                independent_test_crs = NULL,
                ...)

n_records(i)

species_names(i)

get_coords(i)

occurrences_as_df(i)

add_occurrences(oc1, oc2)
```

Arguments

x	A data.frame, tibble or sf with species records.
independent_test	Boolean. If independet_test is TRUE, a fraction of the data is kept for independent testing. Otherwise, the whole dataset x is used. It can also be a data.frame or a sf, with species records to be used as independent test. Structure and names should be identical to those in x.
p	Numeric. Fraction of data to be used as independent test. Standard is 0.1.
crs	Numeric. CRS of x.
independent_test_crs	Numeric. CRS of independent_test if it is a data.frame.
...	A vector with column names addressing the columns with species names, longitude and latitude, respectively, in x.
i	input_sdm or occurrences object.
oc1	A occurrences object to be summed with.
oc2	A occurrences object to be summed with.

Details

x must have three columns: species, decimalLongitude and decimalLatitude. When sf it is only necessary a species column. n_records return the number of presence records to each species. species_names return the species names. get_coords return a data.frame with coordinates of species records. add_occurrences return a occurrences. This function sums two occurrences objects. It can also sum a occurrences object with a data.frame object. occurrences_as_df returns a data.frame with species names and coordinates.

Value

A occurrences object.

Author(s)

Luíz Fernando Esser (luizesser@gmail.com) https://luizfesser.wordpress.com

See Also

input_sdm GBIF_data occ

Examples

```
# Create occurrences:
oc <- occurrences_sdm(occ, crs = 6933)
```

---

parana *Paraná State*

Description

A sf object with a polygon for the Paraná state in Brazil. This is a subset of the brazilian map provided by official government agency (IBGE)

Usage

```
parana
```

Format

'parana' A sf with 1 row and 5 columns:

GID0 State code

CODIGOIB1 State's phone code

NOMEUF2 Name of the state

SIGLAUF3 Abbreviation of the state's name

geom Geometry column of the sf

Source

<https://www.ibge.gov.br/geociencias/cartas-e-mapas/bases-cartograficas-continuas/15759-brasil.html>

`pca_predictors` *Predictors as PCA-axes*

Description

Transform predictors data into PCA-axes.

Usage

```
pca_predictors(i, cumulative_proportion = 0.99)

pca_summary(i)

get_pca_model(i)
```

Arguments

`i`	A `input_sdm` object.
`cumulative_proportion`	A `numeric` with the threshold for cumulative proportion. Standard is 0.99, meaning that axes returned as predictors sum up more than 99 variance.

Details

`pca_predictors` Transform predictors data into PCA-axes. If the user wants to use PCA-axes as future scenarios, then scenarios should be added after the PCA transformation (see examples). `pca_summary` Returns the summary of `prcomp` function. See ?stats::prcomp. `get_pca_model` Returns the model built to calculate PCA-axes.

Value

`input_sdm` object with variables from both `predictors` and `scenarios` transformed in PCA-axes.

Author(s)

Luíz Fernando Esser (luizesser@gmail.com) https://luizfesser.wordpress.com

See Also

`vif_predictors` `sdm_area` `add_scenarios` `add_predictors`

Examples

```
# Create sdm_area object:
sa <- sdm_area(parana, cell_size = 50000, crs = 6933)

# Include predictors:
sa <- add_predictors(sa, bioc) |> select_predictors(c("bio1", "bio12"))

# Create occurrences:
oc <- occurrences_sdm(occ, crs = 6933) |> join_area(sa)

# Create input_sdm:
i <- input_sdm(oc, sa)

# PCA transformation:
i <- pca_predictors(i)
```

---

pdp_sdm *Model Response to Variables*

Description

Obtain the Partial Dependence Plots (PDP) to each variable.

Usage

```
pdp_sdm(i, spp = NULL, algo = NULL, variables_selected = NULL, mean.only = FALSE)

get_pdp_sdm(i, spp = NULL, algo = NULL, variables_selected = NULL)
```

Arguments

i	A input_sdm object.
spp	A character vector with species names to obtain the PDPs. If NULL (standard), the first species in species_names(i) is used.
algo	A character containing the algorithm to obtain the PDP. If NULL (standard) all algorithms are mixed.
variables_selected	A character. If there is a subset of predictors that should be ploted in this, it can be informed using this parameter.
mean.only	Boolean. Should only the mean curve be plotted or a curve to each run should be included? Standard is FALSE.

Value

A plot (for pdp_sdm) or a data.frame (for get_pdp_sdm) with PDP values.

Author(s)

Luíz Fernando Esser (luizesser@gmail.com) https://luizfesser.wordpress.com

See Also

varImp_sdm

Examples

```
# Create sdm_area object:
sa <- sdm_area(parana, cell_size = 100000, crs = 6933)

# Include predictors:
sa <- add_predictors(sa, bioc) |> select_predictors(c("bio1", "bio12"))

# Include scenarios:
sa <- add_scenarios(sa)

# Create occurrences:
oc <- occurrences_sdm(occ, crs = 6933) |> join_area(sa)

# Create input_sdm:
i <- input_sdm(oc, sa)

# Pseudoabsence generation:
i <- pseudoabsences(i, method="bioclim", n_set=3)

# Custom trainControl:
ctrl_sdm <- caret::trainControl(method = "repeatedcv",
                                number = 2,
                                repeats = 1,
                                classProbs = TRUE,
                                returnResamp = "all",
                                summaryFunction = summary_sdm,
                                savePredictions = "all")
# Train models:
i <- train_sdm(i, algo = c("naive_bayes"), ctrl=ctrl_sdm)
# PDP plots:
pdp_sdm(i)
get_pdp_sdm(i)
```

---

plot_occurrences *S3 Methods for plot and mapview*

Description

This function creates different plots depending on the input.

Usage

```
plot_occurrences(i, spp_name = NULL, pa = TRUE)

plot_grid(i)

plot_predictors(i, variables_selected = NULL)

plot_scenarios(i, variables_selected = NULL, scenario = NULL)

plot_predictions(
  i,
  spp_name = NULL,
  scenario = NULL,
  id = NULL,
  ensemble = TRUE,
  ensemble_type = "mean_occ_prob"
)

mapview_grid(i)

mapview_occurrences(i, spp_name = NULL, pa = TRUE)

mapview_predictors(i, variables_selected = NULL)

mapview_scenarios(i, variables_selected = NULL, scenario = NULL)

mapview_predictions(
  i,
  spp_name = NULL,
  scenario = NULL,
  id = NULL,
  ensemble = TRUE,
  ensemble_type = "mean_occ_prob"
)
```

Arguments

i	Object to be plotted. Can be a input_sdm, but also occurrences or sdm_area.
spp_name	A character with species to be plotted. If NULL, the first species is plotted.
pa	Boolean. Should pseudoabsences be plotted together? (not implemented yet.)
variables_selected	A character vector with names of variables to be plotted.
scenario	description
id	The id of models to be plotted (only used when ensemble = FALSE). Possible values are row names of get_validation_metrics(i).
ensemble	Boolean. Should the ensemble be plotted (TRUE)? Otherwise a prediction will be plotted

`ensemble_type` Character of the type of ensemble to be plotted. One of: "mean_occ_prob", "wmean_AUC" or "committee_avg"

Details

We implemented a bestiary of plots to help visualizing the process and results. If you are not familiar with mapview, consider using it to better visualize maps.

Value

The plot or mapview desired.

Author(s)

Luíz Fernando Esser (luizesser@gmail.com) https://luizfesser.wordpress.com

See Also

`WorldClim_data`

`predictors` *Predictors Names Managing*

Description

This function manage predictors names in `sdm_area` objects.

Usage

```
predictors(x)

## S3 method for class 'sdm_area'
predictors(x)

## S3 method for class 'input_sdm'
predictors(x)

set_predictor_names(x, new_names)

## S3 method for class 'input_sdm'
set_predictor_names(x, new_names)

## S3 method for class 'sdm_area'
set_predictor_names(x, new_names)

get_predictor_names(x)

## S3 method for class 'sdm_area'
```

```
get_predictor_names(x)

## S3 method for class 'input_sdm'
get_predictor_names(x)

test_variables_names(sa, scen)

set_variables_names(s1 = NULL, s2 = NULL, new_names = NULL)
```

Arguments

x	A sdm_area or input_sdm object to get/set predictors names.
new_names	A character vector from size length(get_predictor_names(x))
sa	A sdm_area object.
scen	A stars object with scenarios.
s1	A stars object with scenarios.
s2	A stars object with scenarios or a sdm_area object.

Details

This functions is available so users can modify predictors names to better represent them. Use carefully to avoid giving wrong names to the predictors. Useful to make sure the predictors names are equal the names in scenarios. test_variables_names Tests if variables in a stars object (scen argument) matches the given sdm_area object (sa argument). set_variables_names will set s1 object variables names as the s2 object variables names OR assign new names to it.

Value

predictors and get_predictor_names return a character vector with predictors names. test_variables_names returns a logical informing if all variables are equal in both objects (TRUE) or not (FALSE). set_variables_names returns the s1 object with new names provided by s2 or new_names.

Author(s)

Luíz Fernando Esser (luizesser@gmail.com) https://luizfesser.wordpress.com

See Also

parana sdm_area

Examples

```
# Create sdm_area object:
sa <- sdm_area(parana, cell_size = 50000, crs = 6933)

# Include predictors:
sa <- add_predictors(sa, bioc)

# Check predictors' names:
```

```
get_predictor_names(sa)
```

`predict_sdm` *Predict SDM models in new data*

Description

This function projects SDM models to new scenarios

Usage

```
predict_sdm(m,
            scen = NULL,
            metric = "ROC",
            th = 0.9,
            tp = "prob",
            ensembles = TRUE,
            file = NULL,
            add.current = TRUE)

get_predictions(i)

get_ensembles(i)
```

Arguments

`m`	A `input_sdm` or a `models` object.
`scen`	A `scenarios` object or `NULL`. If `NULL` and `m` is a `input_sdm` with a scenarios slot, it will be used.
`metric`	A character containing the metric in which the `th` will be calculated/applied. Default is ROC. See `?mean_validation_metrics` for the metrics available.
`th`	Thresholds for metrics. Can be numeric or a function.
`tp`	Type of output to be retrieved. See details.
`ensembles`	Boolean. Should ensembles be calculated? If `TRUE` a series of ensembles are obtained. See details.
`file`	File to sabe predictions.
`add.current`	If current scenario is not available, predictors will be used as the current scenario.
`i`	A `input_sdm` or a `predictions` object.

Details

tp is a parameter to be passed on caret to retrieve either the probabilities of classes (tp="prob") or the raw output (tp="raw"), which could vary depending on the algorithm used, but usually would be on of the classes (factor vector with presences and pseudoabsences).

When ensembles is set to TRUE, three ensembles are currently implemented. mean_occ_prob is the mean occurrence probability, which is a simple mean of predictions, wmean_AUC is the same mean_occ_prob, but weighted by AUC, and committee_avg is the committee average, as known as majority rule, where predictions are binarized and then a mean is obtained.

get_predictions returns the list of all predictions to all scenarios, all species, all algorithms and all repetitions. Useful for those who wish to implement their own ensemble methods.

get_ensembles returns a matrix of data.frames, where each column is a scenario and each row is a species.

scenarios_names returns the scenarios names in a sdm_area or input_sdm object.

get_scenarios_data returns the data from scenarios in a sdm_area or input_sdm object.

Value

A input_sdm or a predictions object.

Author(s)

Luíz Fernando Esser (luizesser@gmail.com) https://luizfesser.wordpress.com

See Also

sdm_area input_sdm mean_validation_metrics

Examples

```
# Create sdm_area object:
set.seed(1)
sa <- sdm_area(parana, cell_size = 100000, crs = 6933)

# Include predictors:
sa <- add_predictors(sa, bioc) |> select_predictors(c("bio1", "bio12"))

# Include scenarios:
sa <- add_scenarios(sa)

# Create occurrences:
oc <- occurrences_sdm(occ, crs = 6933) |> join_area(sa)

# Create input_sdm:
i <- input_sdm(oc, sa)

# Pseudoabsence generation:
i <- pseudoabsences(i, method="random", n_set=2)

# Custom trainControl:
```

```
ctrl_sdm <- caret::trainControl(method = "boot",
                                number = 1,
                                repeats = 1,
                                classProbs = TRUE,
                                returnResamp = "all",
                                summaryFunction = summary_sdm,
                                savePredictions = "all")

# Train models:
i <- train_sdm(i, algo = c("naive_bayes"), ctrl=ctrl_sdm) |>
  suppressWarnings()

# Predict models:
i  <- predict_sdm(i, th = 0.8)
i
```

`print.input_sdm` *Print method for input_sdm*

Description

Print method for input_sdm

Usage

```
## S3 method for class 'input_sdm'
print(x, ...)
```

Arguments

`x`	input_sdm object
`...`	passed to other methods

Value

Concatenate structured characters to showcase what is stored in the object.

print.models *Print method for models*

Description

Print method for models

Usage

```
## S3 method for class 'models'
print(x, ...)
```

Arguments

x	models object
...	passed to other methods

Value

Concatenate structured characters to showcase what is stored in the object.

print.occurrences *Print method for occurrences*

Description

Print method for occurrences

Usage

```
## S3 method for class 'occurrences'
print(x, ...)
```

Arguments

x	occurrences object
...	passed to other methods

Value

Concatenate structured characters to showcase what is stored in the object.

print.predictions *Print method for predictions*

Description

Print method for predictions

Usage

```
## S3 method for class 'predictions'
print(x, ...)
```

Arguments

x	predictions object
...	passed to other methods

Value

Concatenate structured characters to showcase what is stored in the object.

pseudoabsences *Obtain Pseudoabsences*

Description

This function obtains pseudoabsences given a set of predictors.

Usage

```
pseudoabsences(occ,
               pred = NULL,
               method = "random",
               n_set = 10,
               n_pa = NULL,
               variables_selected = NULL,
               th = 0)

n_pseudoabsences(i)

pseudoabsence_method(i)

pseudoabsence_data(i)
```

Arguments

occ	A occurrences_sdm or input_sdm object.
pred	A sdm_area object. If NULL and occ is a input_sdm, pred will be retrieved from occ.
method	Method to create pseudoabsences. One of: "random", "bioclim" or "mahal.dist".
n_set	numeric. Number of datasets of pseudoabsence to create.
n_pa	numeric. Number of pseudoabsences to be generated in each dataset created. If NULL then the function prevents imbalance by using the same number of presence records (n_records(occ)). If you want to address different sizes to each species, you must provide a named vector (as in n_records(occ)).
variables_selected	A vector with variables names to be used while building pseudoabsences. Only used when method is not "random".
th	numeric Threshold to be applied in bioclim/mahal.dist projections. See details.
i	A input_sdm object.

Details

pseudoabsences is used in the SDM workflow to obtain pseudoabsences, a step necessary for most of the algorithms to run. We implemented three methods so far: "random", which is self-explanatory, "bioclim" and "mahal.dist". The two last are built with the idea that pseudoabsences should be environmentally different from presences. Thus, we implemented two presence-only methods to infer the distribution of the species. "bioclim" uses an envelope approach (bioclimatic envelope), while "mahal.dist" uses a distance approach (mahalanobis distance). th parameter enters here as a threshold to binarize those results. Pseudoabsences are retrieved outside the projected distribution of the species.

n_pseudoabsences returns the number of pseudoabsences obtained per species.

pseudoabsence_method returns the method used to obtain pseudoabsences.

pseudoabsence_data returns a list of species names. Each species name will have a lists with pseudoabsences data from class sf.

Value

A occurrences_sdm or input_sdm object with pseudoabsence data.

Author(s)

Luíz Fernando Esser (luizesser@gmail.com) https://luizfesser.wordpress.com

See Also

link{input_sdm} sdm_area occurrences_sdm

Examples

```
# Create sdm_area object:
sa <- sdm_area(parana, cell_size = 25000, crs = 6933)

# Include predictors:
sa <- add_predictors(sa, bioc) |> select_predictors(c("bio1", "bio4", "bio12"))

# Include scenarios:
sa <- add_scenarios(sa)

# Create occurrences:
oc <- occurrences_sdm(occ, crs = 6933) |> join_area(sa)

# Create input_sdm:
i <- input_sdm(oc, sa)

# Pseudoabsence generation:
i <- pseudoabsences(i, method="bioclim")
```

---

rivs *Hydrologic Variables*

Description

A `sf` LINESTRING object with hydrologic variables (LENGTH_KM and DIST_DN_KM) for the Paraná state in Brazil. Data obtained from HydroSHEDS for river flows >= 10m3/s.

Usage

```
rivs
```

Format

'rivs' A `sf` with 1031 attributes and 2 fiels:

LENGTH_KM Length of the river reach segment, in kilometers.

DIST_DN_KM Distance from the reach outlet, i.e., the most downstream pixel of the reach, to the final downstream location along the river network, in kilometers. This downstream location is either the pour point into the ocean or an endorheic sink.

Source

<https://www.hydrosheds.org/>

salm *Salminus brasiliensis occurrence data*

Description

A `data.frame` object with Salminus brasiliensis occurrence data obtained from GBIF and filtered with Parana state `sf`.

Usage

```
salm
```

Format

'salm' A `data.frame` with 46 rows and 3 columns (EPSG:6933):

species Species name

decimalLongitude Longitude in meters

decimalLatitude Latitude in meters

Source

<https://www.gbif.org>

scen *Bioclimatic Variables*

Description

A `stars` object with bioclimatic variables (bio1, bio4 and bio12) and four future scenarios for the Parana state in Brazil. Data from MIROC6 GCM from WorldClim 2.1 at 10 arc-min resolution.

Usage

```
scen
```

Format

'scen' A `stars` with 4 attribute and 3 bands:

ca_ssp245_2090 Intermediate scenario for the year 2090 and GCM CanESM5

ca_ssp585_2090 Extreme scenario for the year 2090 and GCM CanESM5

mi_ssp245_2090 Intermediate scenario for the year 2090 and GCM MIROC6

mi_ssp585_2090 Extreme scenario for the year 2090 and GCM MIROC6

bio1 Annual Mean Temperature

bio4 Temperature Seasonality

bio12 Annual Precipitation

Source

<https://www.worldclim.org/>

---

sdm_area *Create a* sdm_area *object*

Description

This function creates a new sdm_area object.

Usage

```
sdm_area(x, cell_size = NULL, crs = NULL, variables_selected = NULL,
                gdal = TRUE, crop_by = NULL, lines_as_sdm_area = FALSE)

get_sdm_area(i)
```

Arguments

x	A shape or a raster. Usually a shape from sf class, but rasters from stars, rasterStack or SpatRaster class are also allowed.
cell_size	numeric. The cell size to be used in models.
crs	numeric. Indicates which EPSG should the output grid be in. If NULL, epsg from x is used.
variables_selected	A character vector with variables in x to be used in models. If NULL (standard), all variables in x are used.
gdal	Boolean. Force the use or not of GDAL when available. See details.
crop_by	A shape from sf to crop x.
lines_as_sdm_area	Boolean. If x is a sf with LINESTRING geometry, it can be used to model species distribution in lines and not grid cells.
i	A sdm_area or a input_sdm object.

Details

The function returns a sdm_area object with a grid built upon the x parameter. There are two ways to make the grid and resample the variables in sdm_area: with and without gdal. As standard, if gdal is available in you machine it will be used (gdal = TRUE), otherwise sf/stars will be used. get_sdm_area will return the grid built by sdm_area.

Value

A sdm_area object containing:

grid	sf with POLYGON geometry representing the grid for the study area.
cell_size	numeric information regarding the size of the cell used to rescale variables to the study area, representing also the cell size in the grid.

Author(s)

Luíz Fernando Esser (luizesser@gmail.com) and Reginaldo Ré. https://luizfesser.wordpress.com

See Also

WorldClim_data parana input_sdm, add_predictors

Examples

```
# Create sdm_area object:
sa_area <- sdm_area(parana, cell_size = 50000, crs = 6933)

# Create sdm_area using a subset of rivs (lines):
sa_rivers <- sdm_area(rivs[c(1:100),], cell_size = 100000, crs = 6933, lines_as_sdm_area = TRUE)
```

---

sdm_as_stars sdm_as_X *functions to transform* caretSDM *data into other classes.*

Description

This functions transform data from a caretSDM object to be used in other packages.

Usage

```
sdm_as_stars(x,
             what = NULL,
             spp = NULL,
             scen = NULL,
             id = NULL,
             ens = NULL)

sdm_as_raster(x, what = NULL, spp = NULL, scen = NULL, id = NULL, ens = NULL)

sdm_as_terra(x, what = NULL, spp = NULL, scen = NULL, id = NULL, ens = NULL)
```

Arguments

x	A caretSDM object.
what	Sometimes multiple data inside x could be transformed. This parameter allows users to specify what needs to be converted.It can be one of: "predictors", "scenarios", "predictions" or "ensembles".
spp	character. Which species should be converted?
scen	character. Which scenario should be converted?
id	character. Which id should be converted?
ens	character. Which ensemble should be converted?

Value

The output is the desired class.

Author(s)

Luíz Fernando Esser (luizesser@gmail.com) https://luizfesser.wordpress.com

Examples

```
# Create sdm_area object:
sa <- sdm_area(parana, cell_size = 100000, crs = 6933)

# Include predictors:
sa <- add_predictors(sa, bioc) |> select_predictors(c("bio1", "bio12"))

# Include scenarios:
sa <- add_scenarios(sa)

# Create occurrences:
oc <- occurrences_sdm(occ, crs = 6933) |> join_area(sa)

# Create input_sdm:
i <- input_sdm(oc, sa)

# Pseudoabsence generation:
i <- pseudoabsences(i, method="random", n_set=2)

# Custom trainControl:
ctrl_sdm <- caret::trainControl(method = "boot",
                                number = 1,
                                classProbs = TRUE,
                                returnResamp = "all",
                                summaryFunction = summary_sdm,
                                savePredictions = "all")

# Train models:
i <- train_sdm(i, algo = c("naive_bayes"), ctrl=ctrl_sdm) |>
  suppressWarnings()

# Predict models:
i  <- predict_sdm(i, th=0.8)

# Transform in stars:
sdm_as_stars(i)
```

`select_predictors` *Tidyverse methods for caretSDM objects*

Description

Set of functions to facilitate the use of caretSDM through tidyverse grammatics.

Usage

```
select_predictors(x, ...)

## S3 method for class 'sdm_area'
select(.data, ...)

## S3 method for class 'input_sdm'
select(.data, ...)

## S3 method for class 'sdm_area'
mutate(.data, ...)

## S3 method for class 'input_sdm'
mutate(.data, ...)

## S3 method for class 'sdm_area'
filter(.data, ..., .by, .preserve)

## S3 method for class 'input_sdm'
filter(.data, ..., .by, .preserve)

## S3 method for class 'occurrences'
filter(.data, ..., .by, .preserve)

filter_species(x, spp = NULL, ...)
```

Arguments

x	sdm_area or input_sdm object.
...	character arguments to pass to the given function.
.data	Data to pass to tidyr function.
.by	See ?dplyr::filter.
.preserve	See ?dplyr::filter.
spp	Species to be filtered.

Value

The transformed sdm_area/input_sdm object.

Examples

```
# Create sdm_area object:
sa <- sdm_area(parana, cell_size = 25000, crs = 6933)
```

```
# Include predictors:
sa <- add_predictors(sa, bioc) |> select_predictors(c("bio1", "bio4", "bio12"))
```

---

summary_sdm *Calculates performance across resamples*

Description

This function is used in `caret::trainControl(summaryFunction=summary_sdm)` to calculate performance metrics across resamples.

Usage

```
summary_sdm(data, lev = NULL, model = NULL, custom_fun=NULL)
```

Arguments

`data`	A `data.frame` with observed and predicted values.
`lev`	A `character` vector of factors levels for the response.
`model`	Models names taken from `train` object.
`custom_fun`	A custom function to be applied in models (not yet implemented).

Details

See `?caret::defaultSummary` for more details and options to pass on `caret::trainControl`.

Value

A `input_sdm` or a `predictions` object.

Author(s)

Luíz Fernando Esser (luizesser@gmail.com) https://luizfesser.wordpress.com

See Also

`train_sdm`

Examples

```
# Create sdm_area object:
sa <- sdm_area(parana, cell_size = 100000, crs = 6933)

# Include predictors:
sa <- add_predictors(sa, bioc) |> select_predictors(c("bio1", "bio12"))

# Include scenarios:
sa <- add_scenarios(sa)

# Create occurrences:
oc <- occurrences_sdm(occ, crs = 6933) |> join_area(sa)

# Create input_sdm:
i <- input_sdm(oc, sa)

# Pseudoabsence generation:
i <- pseudoabsences(i, method="bioclim")

# Custom trainControl:
ctrl_sdm <- caret::trainControl(method = "repeatedcv",
                                number = 2,
                                repeats = 1,
                                classProbs = TRUE,
                                returnResamp = "all",
                                summaryFunction = summary_sdm,
                                savePredictions = "all")

# Train models:
i <- train_sdm(i, algo = c("naive_bayes"), ctrl=ctrl_sdm) |>
suppressWarnings()
```

---

train_sdm *Train SDM models*

Description

This function is a wrapper to fit models in caret using caretSDM data.

Usage

```
train_sdm(occ,
          pred = NULL,
          algo,
          ctrl = NULL,
          variables_selected = NULL,
          parallel = FALSE,
          ...)
```

```
get_tune_length(i)

algorithms_used(i)

get_models(i)

get_validation_metrics(i)

mean_validation_metrics(i)
```

Arguments

occ	A occurrences or a input_sdm object.
pred	A predictors object. If occ is a input_sdm object, then pred is obtained from it.
algo	A character vector. Algorithms to be used. For a complete list see (https://topepo.github.io/caret/availabl models.html) or in caretSDM::algorithms.
ctrl	A trainControl object to be used to build models. See ?caret::trainControl.
variables_selected	A vector of variables to be used as predictors. If NULL, predictors names from pred will be used. Can also be a selection method (e.g. 'vif').
parallel	Should a paralelization method be used (not yet implemented)?
...	Additional arguments to be passed to caret::train function.
i	A models or a input_sdm object.

Details

The object algorithms has a table comparing algorithms available. If the function detects that the necessary packages are not available it will ask for installation. This will happen just in the first time you use the algorithm.

get_tune_length return the length used in grid-search for tunning.

algorithms_used return the names of the algorithms used in the modeling process.

get_models returns a list with trained models (class train) to each species.

get_validation_metrics return a list with a data.frame to each species with complete values for ROC, Sensitivity, Specificity, with their respectives Standard Deviations (SD) and TSS to each of the algorithms and pseudoabsence datasets used.

mean_validation_metrics return a list with a tibble to each species summarizing values for ROC, Sensitivity, Specificity and TSS to each of the algorithms used.

Value

A models or a input_sdm object.

Author(s)

Luíz Fernando Esser (luizesser@gmail.com) https://luizfesser.wordpress.com

See Also

input_sdm sdm_area algorithms

Examples

```
# Create sdm_area object:
sa <- sdm_area(parana, cell_size = 100000, crs = 6933)

# Include predictors:
sa <- add_predictors(sa, bioc) |> select_predictors(c("bio1", "bio12"))

# Include scenarios:
sa <- add_scenarios(sa)

# Create occurrences:
oc <- occurrences_sdm(occ, crs = 6933) |> join_area(sa)

# Create input_sdm:
i <- input_sdm(oc, sa)

# Pseudoabsence generation:
i <- pseudoabsences(i, method="bioclim")

# Custom trainControl:
ctrl_sdm <- caret::trainControl(method = "repeatedcv",
                                number = 2,
                                repeats = 1,
                                classProbs = TRUE,
                                returnResamp = "all",
                                summaryFunction = summary_sdm,
                                savePredictions = "all")

# Train models:
i <- train_sdm(i, algo = c("naive_bayes"), ctrl=ctrl_sdm) |>
suppressWarnings()
```

---

tsne_sdm *tSNE*

Description

This function calculates tSNE with presences and pseudoabsences data and returns a list of plots.

Usage

```
tsne_sdm(occ, pred = NULL, variables_selected = NULL)
```

Arguments

occ	A occurrences or input_sdm object.
pred	A predictors object. If occ is of class input_sdm, then pred is retrieved from it.
variables_selected	Variable to be used in t-SNE. It can also be 'vif', if previously calculated.

Value

A list of plots, where each plot is a tSNE for a given pseudoabsence dataset.

Author(s)

Luíz Fernando Esser (luizesser@gmail.com) https://luizfesser.wordpress.com

---

use_mem *MacroEcological Models (MEM) in caretSDM*

Description

This functions sums all species records into one. Should be used before the data cleaning routine.

Usage

```
use_mem(x, add = TRUE, name = "MEM")
```

Arguments

x	A occurrences or input_sdm object containing occurrences.
add	Logical. Should the new MEM records be added to the pool (TRUE) of species or the output should have only the summed records (FALSE)? Standard is TRUE.
name	How should the new records be named? Standard is "MEM".

Value

A input_sdm or occurrences object with MEM data.

Author(s)

Luíz Fernando Esser (luizesser@gmail.com) https://luizfesser.wordpress.com

Examples

```
# Create sdm_area object:
sa <- sdm_area(parana, cell_size = 25000, crs = 6933)

# Include predictors:
sa <- add_predictors(sa, bioc) |> select_predictors(c("bio1", "bio4", "bio12"))

# Include scenarios:
sa <- add_scenarios(sa)

# Create occurrences:
oc <- occurrences_sdm(occ, crs = 6933) |> join_area(sa)

# Create input_sdm:
i <- input_sdm(oc, sa)

# Use MEM:
i <- use_mem(i)
```

---

varImp_sdm *Calculation of variable importance for models*

Description

This function retrieves variable importance as a function of ROC curves to each predictor.

Usage

```
varImp_sdm(m, id = NULL, ...)
```

Arguments

m	A `models` or `input_sdm` object.
id	Vector of model ids to filter varImp calculation.
...	Parameters passing to caret::varImp().

Value

A `data.frame` with variable importance data.

Author(s)

Luíz Fernando Esser (luizesser@gmail.com) https://luizfesser.wordpress.com

Examples

```
# Create sdm_area object:
sa <- sdm_area(parana, cell_size = 100000, crs = 6933)

# Include predictors:
sa <- add_predictors(sa, bioc) |> select_predictors(c("bio1", "bio12"))

# Include scenarios:
sa <- add_scenarios(sa)

# Create occurrences:
oc <- occurrences_sdm(occ, crs = 6933) |> join_area(sa)

# Create input_sdm:
i <- input_sdm(oc, sa)

# Pseudoabsence generation:
i <- pseudoabsences(i, method="bioclim")

# Custom trainControl:
ctrl_sdm <- caret::trainControl(method = "repeatedcv",
                                number = 2,
                                repeats = 1,
                                classProbs = TRUE,
                                returnResamp = "all",
                                summaryFunction = summary_sdm,
                                savePredictions = "all")

# Train models:
i <- train_sdm(i, algo = c("naive_bayes"), ctrl=ctrl_sdm) |>
suppressWarnings()

# Variable importance:
varImp_sdm(i)
```

---

vif_predictors *Calculate VIF*

Description

Apply Variance Inflation Factor (VIF) calculation.

Usage

```
vif_predictors(pred, area = "all", th = 0.5, maxobservations = 5000, variables_selected =
NULL)

vif_summary(i)
```

```
selected_variables(i)
```

Arguments

pred	A input_sdm or predictors object.
area	Character. Which area should be used in vif selection? Standard is "all".
th	Threshold to be applied in VIF routine. See ?usdm::vifcor.
maxobservations	Max observations to use to calculate the VIF.
variables_selected	If there is a subset of predictors that should be used in this function, it can be informed using this parameter. If set to NULL (standard) all variables are used.
i	A input_sdm to retrieve information from.

Details

vif_predictors is a wrapper function to run usdm::vifcor in caretSDM.

Value

A input_sdm or predictors object with VIF data.

Author(s)

Luíz Fernando Esser (luizesser@gmail.com) https://luizfesser.wordpress.com

See Also

get_predictor_names

Examples

```
# Create sdm_area object:
sa <- sdm_area(parana, cell_size = 25000, crs = 6933)

# Include predictors:
sa <- add_predictors(sa, bioc) |> select_predictors(c("bio1", "bio4", "bio12"))

# Include scenarios:
sa <- add_scenarios(sa, scen)

# Create occurrences:
oc <- occurrences_sdm(occ, crs = 6933) |> join_area(sa)

# Create input_sdm:
i <- input_sdm(oc, sa)

# VIF calculation:
i <- vif_predictors(i)
```

```
i

# Retrieve information about vif:
vif_summary(i)
selected_variables(i)
```

`WorldClim_data` *Download WorldClim v.2.1 bioclimatic data*

Description

This function allows to download data from WorldClim v.2.1 (https://www.worldclim.org/data/index.html) considering multiple GCMs, time periods and SSPs.

Usage

```
WorldClim_data(path = NULL,
               period = "current",
               variable = "bioc",
               year = "2090",
               gcm = "mi",
               ssp = "585",
               resolution = 10)
```

Arguments

`path`	Directory path to save downloads.
`period`	Can be "current" or "future".
`variable`	Allows to specify which variables you want to retrieve Possible entries are: "tmax","tmin","prec" and/or "bioc".
`year`	Specify the year you want to retrieve data. Possible entries are: "2030", "2050", "2070" and/or "2090". You can use a vector to provide more than one entry.
`gcm`	GCMs to be considered in future scenarios. You can use a vector to provide more than one entry.

CODE	GCM
ac	ACCESS-CM2
ae	ACCESS-ESM1-5
bc	BCC-CSM2-MR
ca	CanESM5
cc	CanESM5-CanOE
ce	CMCC-ESM2
cn	CNRM-CM6-1
ch	CNRM-CM6-1-HR
cr	CNRM-ESM2-1

ec	EC-Earth3-Veg
ev	EC-Earth3-Veg-LR
fi	FIO-ESM-2-0
gf	GFDL-ESM4
gg	GISS-E2-1-G
gh	GISS-E2-1-H
hg	HadGEM3-GC31-LL
in	INM-CM4-8
ic	INM-CM5-0
ip	IPSL-CM6A-LR
me	MIROC-ES2L
mi	MIROC6
mp	MPI-ESM1-2-HR
ml	MPI-ESM1-2-LR
mr	MRI-ESM2-0
uk	UKESM1-0-LL

`ssp`	SSPs for future data. Possible entries are: "126", "245", "370" and/or "585". You can use a vector to provide more than one entry.
`resolution`	You can select one resolution from the following alternatives: 10, 5, 2.5 OR 30.

Details

This function will create a folder entitled "input_data/WorldClim_data_current" or "input_data/WorldClim_data_future". All the data downloaded will be stored in this folder. Note that, despite being possible to retrieve a lot of data at once, it is not recommended to do so, since the data is very heavy.

Value

If data is not downloaded, the function downloads the data and has no return value. If the data is downloaded, it imports the data as a `stack`.

Author(s)

Luíz Fernando Esser (luizesser@gmail.com) https://luizfesser.wordpress.com

References

https://www.worldclim.org/data/index.html

Examples

```
# download data from multiple periods:
year <- c("2050", "2090")
WorldClim_data(period = "future",
               variable = "bioc",
               year = year,
               gcm = "mi",
               ssp = "126",
```

```
                resolution = 10)

# download data from one specific period
WorldClim_data(period = "future",
                variable = "bioc",
                year = "2070",
                gcm = "mi",
                ssp = "585",
                resolution = 10)
```

`write_ensembles` *Write caretSDM data*

Description

This function exports caretSDM data.

Usage

```
write_ensembles(x, path = NULL, ext = ".tif", centroid = FALSE)

write_predictions(x, path = NULL, ext = ".tif", centroid = FALSE)

write_predictors(x, path = NULL, ext = ".tif", centroid = FALSE)

write_models(x, path = NULL)

write_gpkg(x, file_path, file_name)

## S3 method for class 'sdm_area'
write_gpkg(x, file_path, file_name)

write_occurrences(x, path = NULL, grid = FALSE, ...)

write_pseudoabsences(x, path = NULL, ext = ".csv", centroid = FALSE)

write_grid(x, path = NULL, centroid = FALSE)

write_validation_metrics(x, path = NULL)
```

Arguments

`x`	Object to be written. Can be of class `input_sdm`, `occurrences`, `predictions` or `models`.
`path`	A path with filename and the proper extension (see details) or the directory to save files in.

ext	How it should be saved?
centroid	Should coordinates for the centroids of each cell be included? Standard is FALSE.
file_path	A path to save the sdm_area GeoPackage file.
file_name	The name of the sdm_area GeoPackage file to be saved without extension.
grid	Boolean. Return a grid.
...	Arguments to pass to sf::st_write or write.csv.

Details

ext can be set accordingly to the desired output. Possible values are .tif and .asc for rasters, .csv for for a spreadsheet, but also one of: c("bna", "csv", "e00", "gdb", "geojson", "gml", "gmt", "gpkg", "gps", "gtm", "gxt", "jml", "map", "mdb", "nc", "ods", "osm", "pbf", "shp", "sqlite", "vdv", "xls", "xlsx"). path ideally should only provide the folder. We recommend using: results/what_are_you_writting. So for writting ensembles users are advised to run: path = "results/ensembles"

Value

No return value, called for side effects.

Author(s)

Luíz Fernando Esser (luizesser@gmail.com) https://luizfesser.wordpress.com

Index