

# Package ‘caretSDM’

May 8, 2026

**Type** Package

**Title** Build Species Distribution Modeling using 'caret'

**Version** 1.5

**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** blockCV, caret, checkCLI, checkmate, cli, CoordinateCleaner, data.table, dismo, dplyr, ecospat, fs, furrr, future, ggplot2, ggspatial, glue, gtools, htr2, lwgeom, mapview, maxnet, methods, parallelly, pdp, pROC, progressr, purrr, raster, rgbif, Rtsne, sf, stars, stats, stringdist, stringr, terra, tidyr, usdm, utils

**Encoding** UTF-8

**LazyData** true

**RoxygenNote** 7.3.3

**Suggests** bench, biomod2, cito, covr, e1071, earth, gam, here, kkn, knitr, mda, naivebayes, nnet, R.utils, randomForest, rmarkdown, roxygenals, sdm, tibble, withr, xgboost, testthat (>= 3.0.0)

**Config/testthat/edition** 3

**Depends** R (>= 4.2.0)

**URL** <https://luizesser.github.io/caretSDM/>

**Config/Needs/website** rmarkdown

**Config/roxygenals/filename** generated-globals.R

**Config/roxygenals/unique** FALSE

**NeedsCompilation** no

**Author** Luíz Fernando Esser [aut, cre, cph] (ORCID: <https://orcid.org/0000-0003-2982-7223>),  
 Reginaldo Ré [aut] (ORCID: <https://orcid.org/0000-0001-6452-3466>),  
 Marcos R. Lima [aut] (ORCID: <https://orcid.org/0000-0002-5901-0911>),  
 Edivando Couto [aut] (ORCID: <https://orcid.org/0000-0003-4264-8449>),  
 José Hilário Delconte Ferreira [aut] (ORCID: <https://orcid.org/0000-0002-7116-2600>),  
 Valéria Batista [aut] (ORCID: <https://orcid.org/0000-0002-6574-7338>),  
 Dayani Bailly [aut] (ORCID: <https://orcid.org/0000-0002-6954-9902>)

**Repository** CRAN

**Date/Publication** 2026-03-10 16:30:08 UTC

## Contents

|                                 |    |
|---------------------------------|----|
| add_predictors . . . . .        | 3  |
| add_scenarios . . . . .         | 4  |
| algorithms . . . . .            | 6  |
| background . . . . .            | 6  |
| bioc . . . . .                  | 8  |
| buffer_sdm . . . . .            | 9  |
| correlate_sdm . . . . .         | 10 |
| data_clean . . . . .            | 11 |
| GBIF_data . . . . .             | 13 |
| gcms_ensembles . . . . .        | 14 |
| input_sdm . . . . .             | 15 |
| is_input_sdm . . . . .          | 16 |
| join_area . . . . .             | 17 |
| occ . . . . .                   | 18 |
| occurrences_sdm . . . . .       | 19 |
| parana . . . . .                | 20 |
| pca_predictors . . . . .        | 21 |
| pdp_sdm . . . . .               | 22 |
| plot_occurrences . . . . .      | 24 |
| prediction_change_sdm . . . . . | 26 |
| predictors . . . . .            | 27 |
| predict_sdm . . . . .           | 29 |
| print.input_sdm . . . . .       | 31 |
| print.models . . . . .          | 32 |
| print.occurrences . . . . .     | 32 |
| print.predictions . . . . .     | 33 |
| pseudoabsences . . . . .        | 33 |
| rivs . . . . .                  | 35 |
| salm . . . . .                  | 36 |
| scen . . . . .                  | 36 |
| scen_rs . . . . .               | 37 |
| sdm_area . . . . .              | 38 |
| sdm_as_stars . . . . .          | 39 |

*add\_predictors* 3

|                                    |    |
|------------------------------------|----|
| <i>select_predictors</i> . . . . . | 41 |
| <i>summary_sdm</i> . . . . .       | 42 |
| <i>train_sdm</i> . . . . .         | 44 |
| <i>tsne_sdm</i> . . . . .          | 46 |
| <i>tuneGrid_sdm</i> . . . . .      | 47 |
| <i>use_esm</i> . . . . .           | 48 |
| <i>use_mem</i> . . . . .           | 49 |
| <i>varImp_sdm</i> . . . . .        | 50 |
| <i>vif_predictors</i> . . . . .    | 51 |
| <i>WorldClim_data</i> . . . . .    | 53 |
| <i>write_ensembles</i> . . . . .   | 54 |

**Index** 56

---

*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

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

---

**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, crop_area = NULL)

set_scenarios_names(i, scenarios_names = NULL)

scenarios_names(i)

get_scenarios_data(i)

select_scenarios(i, scenarios_names = NULL)
```

**Arguments**

|                              |   |
|------------------------------|---|
| <code>sa</code>              | A <code>sdm_area</code> or <code>input_sdm</code> object.                       |
| <code>scen</code>            | RasterStack, SpatRaster or stars object. If NULL adds predictors as a scenario. |
| <code>scenarios_names</code> | Character vector with names of scenarios.                                       |
| <code>pred_as_scen</code>    | 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.               |
| crop_area          | A sf object to crop the scen object if necessary.  |
| 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)

Luiz 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 characteristic. 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>>

---

background

*Obtain Background data*

---

### Description

This function obtains background data given a set of predictors.

**Usage**

```
background(occ,  
           pred = NULL,  
           n = 10000,  
           n_set = 1,  
           proportion = NULL)  
  
n_background(i)  
  
background_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.  |
| n          | numeric. Number of background records 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)). |
| n_set      | numeric. Number of datasets of background data to create.  |
| proportion | numeric. A number between 0 and 1 representing a proportion of the area to be mapped as background. E.g.: if the whole area has 5,000 cells and proportion is 0.1, then n is set to 500. Standard is NULL. This argument overwrites n.   |
| i          | A input_sdm object.  |

**Details**

background is used in the SDM workflow to obtain background data, a step necessary for MaxEnt algorithm to run. This function helps avoid the use of pseudoabsence data in background algorithms and the use of background data in pseudoabsence algorithms, a very common mistake.

n\_background returns the number of background records obtained per species.

background\_data returns a list of species names. Each species name will have a lists with background data from class sf.

**Value**

A occurrences\_sdm or input\_sdm object with background data.

**Author(s)**

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

**See Also**

[link{input\\_sdm}](#) [pseudoabsences](#) [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 <- background(i, proportion = 1) # All available data is obtained as background data.
```

---

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, mcp = FALSE)
```

## 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.  |
| mcp      | boolean. Should the buffer be applied in each record (FALSE) or in a minimum convex polygon/convex hull (TRUE)? Standard is FALSE. |

## 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)
```

---

|               |  |
|---------------|--|
| correlate_sdm | <i>Correlation between projections</i> |
|---------------|--|

---

### Description

This function aims to unveil the correlation of different algorithms outputs. For that, it uses the predictions on current scenario, but other scenarios can be tested.

### Usage

```
correlate_sdm(i, scenario = "current")
```

### Arguments

|                       |   |
|-----------------------|---|
| <code>i</code>        | A <code>input_sdm</code> object containing predictions.   |
| <code>scenario</code> | A character containing scenario to be tested. Standard is "current". Value must match <code>scenarios_names(i)</code> . |

### Value

A data.frame with pearson correlation between projections.

### Author(s)

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

### Examples

```
if (interactive()) {  
  # 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)

# Check correlations:
correlate_sdm(i)
}

```

---

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 use 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 user 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)

Luiz 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

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

### 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**

|                   |  |
|-------------------|--|
| <code>i</code>    | A <code>input_sdm</code> object.                                 |
| <code>gcms</code> | GCM codes in <code>scenarios_names(i)</code> 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**

```
if (interactive()) {
  # 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(oc, 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(...)
```

```
add_input_sdm(i1, i2)
```

## Arguments

|                  |  |
|------------------|--|
| <code>...</code> | Data to be used in SDMs. Can be a occurrences and/or a <code>sdm_area</code> object. |
| <code>i1</code>  | A <code>input_sdm</code> object.   |
| <code>i2</code>  | A <code>input_sdm</code> 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_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 | <i>Occurrences Managing</i> |
|-----------------|-----------------------------|

---

**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 independent\_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.

|                                   |   |
|-----------------------------------|---|
| <code>crs</code>                  | Numeric. CRS of <code>x</code> .  |
| <code>independent_test_crs</code> | Numeric. CRS of <code>independent_test</code> if it is a <code>data.frame</code> .  |
| <code>...</code>                  | A vector with column names addressing the columns with species names, longitude and latitude, respectively, in <code>x</code> . |
| <code>i</code>                    | <code>input_sdm</code> or occurrences object.   |
| <code>oc1</code>                  | A occurrences object to be summed with.   |
| <code>oc2</code>                  | 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 | <i>Predictors as PCA-axes</i> |
|----------------|-------------------------------|

---

**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**

|                                 |  |
|---------------------------------|--|
| <code>i</code>                  | A <code>input_sdm</code> object.   |
| <code>spp</code>                | A character vector with species names to obtain the PDPs. If <code>NULL</code> (standard), the first species in <code>species_names(i)</code> is used. |
| <code>algo</code>               | A character containing the algorithm to obtain the PDP. If <code>NULL</code> (standard) all algorithms are mixed.                                      |
| <code>variables_selected</code> | A character. If there is a subset of predictors that should be plotted 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](mailto: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="random", 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, pa_id = 1)

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"
)

plot_background(i, variables_selected = NULL)

plot_niche(
  i,
```

```

    spp_name = NULL,
    variables_selected = NULL,
    scenario = NULL,
    id = NULL,
    ensemble = TRUE,
    ensemble_type = "mean_occ_prob",
    raster = FALSE
)

```

### Arguments

|                                 |   |
|---------------------------------|---|
| <code>i</code>                  | Object to be plotted. Can be a <code>input_sdm</code> , but also <code>occurrences</code> or <code>sdm_area</code> .                                      |
| <code>spp_name</code>           | A character with species to be plotted. If <code>NULL</code> , the first species is plotted.  |
| <code>pa</code>                 | Boolean. Should pseudoabsences be plotted together? (not implemented yet.)  |
| <code>pa_id</code>              | The id of pseudoabsences to be plotted (only used when <code>pa = TRUE</code> ). Possible values are numeric values from 1 to number of PA sets.          |
| <code>variables_selected</code> | A character vector with names of variables to be plotted.   |
| <code>scenario</code>           | description   |
| <code>id</code>                 | The id of models to be plotted (only used when <code>ensemble = FALSE</code> ). Possible values are row names of <code>get_validation_metrics(i)</code> . |
| <code>ensemble</code>           | Boolean. Should the ensemble be plotted ( <code>TRUE</code> )? Otherwise a prediction will be plotted   |
| <code>ensemble_type</code>      | Character of the type of ensemble to be plotted. One of: "mean_occ_prob", "wmean_AUC" or "committee_avg"  |
| <code>raster</code>             | Should the niche be extrapolated to a raster covering all possible values in the environmental space?   |

### 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](mailto:luizesser@gmail.com)) <https://luizfesser.wordpress.com>

### See Also

[WorldClim\\_data](#)

---

prediction\_change\_sdm *Prediction Change Analysis*

---

### Description

Provides an automate way for the visualization of projections gain, loss, and stability between different scenarios.

### Usage

```
prediction_change_sdm(i, scenario = NULL, ensemble_type = NULL, species = NULL, th = 0.5)
```

### Arguments

|               |   |
|---------------|---|
| i             | A input_sdm object with projections.  |
| scenario      | Character. One of the scenarios that were projected. Can be ensembles as well.              |
| ensemble_type | Character. Type of ensemble to be used. Standard is NULL, but will return the mean_occ_prob |
| species       | Character. Species to be analyzed. Standard is NULL.  |
| th            | Numeric. Threshold to binarize the ensemble.  |

### Value

A plot with comparison between current and other scenario.

### Author(s)

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

### See Also

[species\\_names](#) [scenarios\\_names](#)

### Examples

```
if (interactive()) {  
  # 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

# Change Analysis
prediction_change_sdm(i, scenario = "_ssp585_2090", ensemble_type = "mean_occ_prob")
}

```

---

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)
```

```
add_predictions(p1, p2)
```

**Arguments**

|                     |   |
|---------------------|---|
| <code>m</code>      | A <code>input_sdm</code> or a <code>models</code> object.   |
| <code>scen</code>   | A <code>scenarios</code> object or <code>NULL</code> . If <code>NULL</code> and <code>m</code> is a <code>input_sdm</code> with a <code>scenarios</code> slot, it will be used.               |
| <code>metric</code> | A character containing the metric in which the <code>th</code> will be calculated/applied. Default is <code>ROC</code> . See <code>?mean_validation_metrics</code> for the metrics available. |
| <code>th</code>     | 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 save 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.  |
| p1          | A predictions object.   |
| p2          | 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

```
if (interactive()) {
  # 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.

---

|                           |                                |
|---------------------------|--------------------------------|
| <code>print.models</code> | <i>Print method for models</i> |
|---------------------------|--------------------------------|

---

**Description**

Print method for models

**Usage**

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

**Arguments**

|                  |                         |
|------------------|-------------------------|
| <code>x</code>   | models object           |
| <code>...</code> | passed to other methods |

**Value**

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

---

|                                |                                     |
|--------------------------------|-------------------------------------|
| <code>print.occurrences</code> | <i>Print method for occurrences</i> |
|--------------------------------|-------------------------------------|

---

**Description**

Print method for occurrences

**Usage**

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

**Arguments**

|                  |                         |
|------------------|-------------------------|
| <code>x</code>   | occurrences object      |
| <code>...</code> | passed to other methods |

**Value**

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

---

|                   |                                     |
|-------------------|-------------------------------------|
| print.predictions | <i>Print method for predictions</i> |
|-------------------|-------------------------------------|

---

**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 | <i>Obtain Pseudoabsences</i> |
|----------------|------------------------------|

---

**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,  
                size = 1,  
                crs = 4326)
```

n\_pseudoabsences(i)

pseudoabsence\_method(i)

pseudoabsence\_data(i)

**Arguments**

|                                 |  |
|---------------------------------|--|
| <code>occ</code>                | A <code>occurrences_sdm</code> or <code>input_sdm</code> object.   |
| <code>pred</code>               | A <code>sdm_area</code> object. If <code>NULL</code> and <code>occ</code> is a <code>input_sdm</code> , <code>pred</code> will be retrieved from <code>occ</code> .  |
| <code>method</code>             | Method to create pseudoabsences. One of: "random", "bioclim", "mahal.dist" or "buffer_sdm".  |
| <code>n_set</code>              | numeric. Number of datasets of pseudoabsence to create.  |
| <code>n_pa</code>               | numeric. Number of pseudoabsences to be generated in each dataset created. If <code>NULL</code> then the function prevents imbalance by using the same number of presence records ( <code>n_records(occ)</code> ). If you want to address different sizes to each species, you must provide a named vector (as in <code>n_records(occ)</code> ). |
| <code>variables_selected</code> | A vector with variables names to be used while building pseudoabsences. Only used when <code>method</code> is not "random".  |
| <code>th</code>                 | numeric Threshold to be applied in bioclim/mahal.dist projections. See details.  |
| <code>size</code>               | numeric The distance between the record and the margin of the buffer (i.e. buffer radius).   |
| <code>crs</code>                | numeric Indicates which EPSG it the size in.   |
| <code>i</code>                  | A <code>input_sdm</code> 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", "mahal.dist" and "buffer\_sdm". 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)**

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

**See Also**

`link{input_sdm}` [background 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="random")
```

---

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  $\geq 10\text{m}^3/\text{s}$ .

## 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/>>

---

scen\_rs

*Bioclimatic Variables*

---

**Description**

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

**Usage**

scen\_rs

**Format**

## 'scen\_rs' A stars with 5 attribute and 3 bands:

**current** Current scenario with the average values for the years 1970-2000

**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)
```

```
add_sdm_area(sa1, sa2)
```

### 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.   |
| sa1                | A sdm_area object.  |
| sa2                | A sdm_area 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. add\_sdm\_area will sum two sdm\_area objects. As geoprocessing in caretSDM is performed using sf objects, add\_sdm\_area simply applies a rbind in the two different areas.

**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)**

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

**Examples**

```
if (interactive()) {
  # 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)

summary_sdm_presence_only(data, lev, threshold)

validate_on_independent_data(model, data_independent, obs_col_name)
```

**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). |
| threshold        | Threshold for presence-only models.                              |
| data_independent | independent data.frame to calculate metrics.                     |
| obs_col_name     | The name of the column with observed values.                     |

**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 = "random")

# 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)
```

```
add_models(m1, m2)
```

**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 ( <a href="https://topepo.github.io/caret/available_models.html">https://topepo.github.io/caret/available_models.html</a> ) or in caretSDM::algorithms. |
| ctrl               | A trainControl object to be used to build models. See ?caret::trainControl and details.  |
| 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.  |
| m1                 | A models object.   |
| m2                 | A models 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. `caret::trainControl` holds multiple resources for validation and model tuning. Make sure to understand its parameters beforehand. As it is a key function in the modeling process, we also implemented spatial crossvalidation on it. You can set methods to be `cv_spatial` or `cv_cluster` and `train_sdm` will detect that and apply the method according to `blockCV` package.

`get_tune_length` return the length used in grid-search for tuning.

`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 respective 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 = "random")
```

```
# 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>

---

|              |                                      |
|--------------|--------------------------------------|
| tuneGrid_sdm | <i>Retrieve tuneGrid from models</i> |
|--------------|--------------------------------------|

---

**Description**

This function aims to retrieve the tune grid used to build models.

**Usage**

```
tuneGrid_sdm(i)
```

**Arguments**

*i*                    A `input_sdm` object containing models.

**Value**

A list with `data.frames` each one representing the table of a given model.

**Author(s)**

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

**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()

# Retrieve tuneGrid from model:
tuneGrid_sdm(i)

```

---

use\_esm

*Ensemble of Small Models (ESM) in caretSDM*


---

### Description

This functions set parameters to run a ESM when running `train_sdm`.

### Usage

```
use_esm(x, spp = NULL, n_records = 20)
```

### Arguments

|                        |   |
|------------------------|---|
| <code>x</code>         | A occurrences or <code>input_sdm</code> object containing occurrences.  |
| <code>spp</code>       | A vector of species names containing the species which the ESM must be applied. Standard is <code>NULL</code> . |
| <code>n_records</code> | Numeric. Number of species records to apply the ESM. Standard is 20.  |

### Details

We supply two different ways to apply the ESM. If species names are provided, then ESM will be applied only in given species. If a number of species records is provided, then ESM will be applied in every species with number of records bellow the given threshold. As standard, `use_esm` will be apply to every species with less then 20 records.

### Value

A `input_sdm` or `occurrences` object with ESM parameters.

### Author(s)

Luíz Fernando Esser ([luizesser@gmail.com](mailto: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", "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_esm(i, n_records = 999)
```

---

use\_mem

*MacroEcological Models (MEM) in caretSDM*

---

**Description**

This function 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

|                  |   |
|------------------|---|
| <code>m</code>   | A models or input_sdm object.                     |
| <code>id</code>  | Vector of model ids to filter varImp calculation. |
| <code>...</code> | 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 = "random")

# 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.  |
| 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. 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.

**Author(s)**

Luíz Fernando Esser (luizesser@gmail.com) [<https://luizfesser.wordpress.com>](<https://luizfesser.wordpress.com>)

**References**

[<https://www.worldclim.org/data/index.html>](<https://www.worldclim.org/data/index.html>)

**Examples**

```
## download data from multiple periods:
#year <- c("2050", "2090")
#WorldClim_data(path = "",
#               period = "future",
#               variable = "bioc",
#               year = year,
#               gcm = "mi",
#               ssp = "126",
#               resolution = 10)

## download data from one specific period
#WorldClim_data(path = "",
#               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 <code>input_sdm</code> , <code>occurrences</code> , <code>predictions</code> or <code>models</code> . |
| 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 <code>FALSE</code> .   |
| file_path | A path to save the <code>sdm_area</code> GeoPackage file.   |
| file_name | The name of the <code>sdm_area</code> GeoPackage file to be saved without extension.  |
| grid      | Boolean. Return a grid.   |
| ...       | Arguments to pass to <code>sf::st_write</code> or <code>write.csv</code> .  |

### Details

`ext` can be set accordingly to the desired output. Possible values are `.tif` and `.asc` for rasters, `.csv` 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_writing`. So for writing ensembles users are advised to run: `path = "results/ensembles"`

### Value

No return value, called for side effects.

### Author(s)

Luiz Fernando Esser ([luizesser@gmail.com](mailto:luizesser@gmail.com)) <https://luizfesser.wordpress.com>

# Index

## \* datasets

- algorithms, 6
  - bioc, 8
  - occ, 18
  - parana, 20
  - rivs, 35
  - salm, 36
  - scen, 36
  - scen\_rs, 37
- add\_input\_sdm (input\_sdm), 15
- add\_models (train\_sdm), 44
- add\_occurrences (occurrences\_sdm), 19
- add\_predictions (predict\_sdm), 29
- add\_predictors, 3, 22, 39
- add\_scenarios, 4, 22
- add\_sdm\_area (sdm\_area), 38
- algorithms, 6, 45
- algorithms\_used (train\_sdm), 44
- background, 6, 34
- background\_data (background), 6
- bioc, 4, 8
- buffer\_sdm, 9
- correlate\_sdm, 10
- data\_clean, 11
- filter.input\_sdm (select\_predictors), 41
- filter.occurrences (select\_predictors), 41
- filter.sdm\_area (select\_predictors), 41
- filter.species (select\_predictors), 41
- GBIF\_data, 9, 12, 13, 14, 20
- gcms\_ensembles, 14
- get\_coords (occurrences\_sdm), 19
- get\_ensembles (predict\_sdm), 29
- get\_models (train\_sdm), 44
- get\_pca\_model (pca\_predictors), 21
- get\_pdp\_sdm (pdp\_sdm), 22
- get\_predictions (predict\_sdm), 29
- get\_predictor\_names, 52
- get\_predictor\_names (predictors), 27
- get\_predictors (add\_predictors), 3
- get\_scenarios\_data (add\_scenarios), 4
- get\_sdm\_area (sdm\_area), 38
- get\_tune\_length (train\_sdm), 44
- get\_validation\_metrics (train\_sdm), 44
- input\_sdm, 5, 12, 14, 15, 18, 20, 30, 39, 45
- is\_input\_sdm, 16
- is\_models (is\_input\_sdm), 16
- is\_occurrences (is\_input\_sdm), 16
- is\_predictions (is\_input\_sdm), 16
- is\_sdm\_area (is\_input\_sdm), 16
- join\_area, 17
- mapview\_grid (plot\_occurrences), 24
- mapview\_occurrences (plot\_occurrences), 24
- mapview\_predictions (plot\_occurrences), 24
- mapview\_predictors (plot\_occurrences), 24
- mapview\_scenarios (plot\_occurrences), 24
- mean\_validation\_metrics, 30
- mean\_validation\_metrics (train\_sdm), 44
- mutate.input\_sdm (select\_predictors), 41
- mutate.sdm\_area (select\_predictors), 41
- n\_background (background), 6
- n\_pseudoabsences (pseudoabsences), 33
- n\_records (occurrences\_sdm), 19
- occ, 18, 20
- occurrences\_as\_df (occurrences\_sdm), 19
- occurrences\_sdm, 7, 12, 14, 16, 18, 19, 34
- parana, 20, 29, 39

- pca\_predictors, 21
- pca\_summary (pca\_predictors), 21
- pdp\_sdm, 22
- plot\_background (plot\_occurrences), 24
- plot\_grid (plot\_occurrences), 24
- plot\_niche (plot\_occurrences), 24
- plot\_occurrences, 24
- plot\_predictions (plot\_occurrences), 24
- plot\_predictors (plot\_occurrences), 24
- plot\_scenarios (plot\_occurrences), 24
- predict\_sdm, 29
- prediction\_change\_sdm, 26
- predictors, 4, 12, 14, 27
- print.input\_sdm, 31
- print.models, 32
- print.occurrences, 32
- print.predictions, 33
- pseudoabsence\_data (pseudoabsences), 33
- pseudoabsence\_method (pseudoabsences), 33
- pseudoabsences, 7, 18, 33
- rivs, 35
- salm, 36
- scen, 36
- scen\_rs, 37
- scenarios\_names, 26
- scenarios\_names (add\_scenarios), 4
- sdm\_area, 4, 5, 12, 14, 16, 18, 22, 29, 30, 38, 45
- sdm\_as\_raster (sdm\_as\_stars), 39
- sdm\_as\_stars, 39
- sdm\_as\_terra (sdm\_as\_stars), 39
- select.input\_sdm (select\_predictors), 41
- select.sdm\_area (select\_predictors), 41
- select\_predictors, 41
- select\_scenarios (add\_scenarios), 4
- selected\_variables (vif\_predictors), 51
- set\_predictor\_names (predictors), 27
- set\_scenarios\_names (add\_scenarios), 4
- set\_variables\_names (predictors), 27
- species\_names, 26
- species\_names (occurrences\_sdm), 19
- summary\_sdm, 42
- summary\_sdm\_presence\_only (summary\_sdm), 42
- test\_variables\_names (predictors), 27
- train\_sdm, 43, 44
- tsne\_sdm, 46
- tuneGrid\_sdm, 47
- use\_esm, 48
- use\_mem, 49
- validate\_on\_independent\_data (summary\_sdm), 42
- varImp\_sdm, 23, 50
- vif\_predictors, 22, 51
- vif\_summary (vif\_predictors), 51
- WorldClim\_data, 25, 39, 53
- write\_ensembles, 54
- write\_gpkg (write\_ensembles), 54
- write\_grid (write\_ensembles), 54
- write\_models (write\_ensembles), 54
- write\_occurrences (write\_ensembles), 54
- write\_predictions (write\_ensembles), 54
- write\_predictors (write\_ensembles), 54
- write\_pseudoabsences (write\_ensembles), 54
- write\_validation\_metrics (write\_ensembles), 54