Package: GeoFIS (via r-universe)

September 4, 2024

Type Package

Title Spatial Data Processing for Decision Making

Version 1.1.0

Author Serge Guillaume [aut], Jean-Luc Lablée [aut, cre], INRAE [cph] (National Research Institute for Agriculture, Food and Environment, France)

Maintainer Jean-Luc Lablée <jean-luc.lablee@inrae.fr>

URL https://www.geofis.org

Description Methods for processing spatial data for decision-making. This package is an R implementation of methods provided by the open source software GeoFIS <https://www.geofis.org> (Leroux et al. 2018) <doi:10.3390/agriculture8060073>. The main functionalities are the management zone delineation (Pedroso et al. 2010) <doi:10.1016/j.compag.2009.10.007> and data aggregation (Mora-Herrera et al. 2020) <doi:10.1016/j.compag.2020.105624>.

License CeCILL

Encoding UTF-8

Depends R (>= 4.0.0), sp, data.tree, FisPro (>= 1.1.0)

Imports methods, utils, stats, Rdpack, foreach, R6, Rcpp, magrittr, sf, nnls, itertools, purrr

SystemRequirements GNU make, C++17, gmp, mpfr

RdMacros Rdpack

NeedsCompilation yes

LinkingTo Rcpp, BH, FisPro

Suggests testthat, rlang, knitr, rmarkdown, RColorBrewer, R.rsp

RoxygenNote 7.3.2

VignetteBuilder knitr, R.rsp

Date/Publication 2024-09-03 08:50:10 UTC

Repository https://j-ll.r-universe.dev

RemoteUrl https://github.com/cran/GeoFIS

RemoteRef HEAD

RemoteSha 3ba87ff64942a90596d16ad810f9e91f2e702eb5

Contents

AggregFis *Class "AggregFis"*

Description

The Fis aggregation operator to be used in Fusion

Slots

`fis` FisPro::Fis object, The Fis to be used in the aggregation operator

`output_index` integer value, The index (1-based index) of the output in the Fis to be used in the aggregation

See Also

NewAggregFis

Aggregation using linguistic rules

AggregFunction *Class "AggregFunction"*

Description

The functional aggregation operator to be used in Fusion

Slots

`func` Function, The function used for the aggregation

See Also

NewAggregFunction

`AggregOwa` *Class "AggregOwa"*

Description

The OWA aggregation operator to be used in Fusion

Slots

`weights` numeric vector, The weights of the OWA aggregation operator (the sum of the weights must be equal to 1 without negative values)

See Also

NewAggregOwa

Aggregation using numerical operators

`AggregWam` *Class "AggregWam"*

Description

The WAM aggregation operator to be used in Fusion

Slots

`weights` numeric vector, The weights of the WAM aggregation operator (the sum of the weights must be equal to 1 without negative values)

See Also

NewAggregWam

Aggregation using numerical operators

conductivity_2014 *Soil conductivity 2014 dataset*

Description

The soil conductivity of a vine plot in year 2014

Usage

```
data(conductivity_2014)
```

Format

sp::SpatialPointsDataFrame object with 353 observations and 1 attribute:

`conduct` numeric value, The soil conductivity

conductivity_border *Border dataset*

Description

The soil conductivity border of a vine plot

Usage

```
data(conductivity_border)
```

Format

sp::SpatialPolygonsDataFrame object with 1 polygon delimiting the border of the vine plot:

`id` integer value, The id of the polygon

DataInZone *Classify data in management zones*

Description

Classify data in management zones of maps obtained with the Zoning algorithms.

Usage

```
DataInZone(data, maps, use_id = FALSE)
```

Arguments

data	sp::SpatialPointsDataFrame or sp::SpatialMultiPointsDataFrame object, The input data.
maps	sp::SpatialPolygonsDataFrame object, or a list of sp::SpatialPolygonsDataFrame, The map or list of maps to process.
use_id	boolean, Use the id attribute of the zone in the map if TRUE, or the order index (1-based indexed) of the zone in the map if FALSE (the default).

Value

sp::SpatialPointsDataFrame or sp::SpatialMultiPointsDataFrame object depending of the type of input data. Return table of membership of each point in the zones for each map, the input data and a column for each map processed (named "map{number of zones in the map}") with the class of each point.

EuclideanDistance *The "Euclidean" distance*

Description

Function to create an "Euclidean" distance
To be used with the Zoning `combine_distance` or `attribute_distance` field

Usage

```
EuclideanDistance()
```

Value

Euclidean distance object

Fusion *Class "Fusion"*

Description

The main class to perform data fusion
More information is available in the vignette "Data Fusion with GeoFIS"

Active bindings

`aggregate` data.tree::Node object, or a list of data.tree::Node, The node(s) to aggregate

Methods

Public methods:

- `Fusion$new()`
- `Fusion$perform()`
- `Fusion$output()`

Method `new()`**:** The constructor to build an object of class Fusion.

Usage:

```
Fusion$new(source)
```

Arguments:

`source` data.frame or sp::Spatial*DataFrame object of sp::sp package
Keep only numeric attributes

Method `perform()`**:** Perform the data fusion

Usage:

```
Fusion$perform()
```

Method `output()`**:** Get the output aggregated data (same object type as data source)

Usage:

```
Fusion$output()
```

Returns: data.frame or sp::Spatial*DataFrame object

References

Guillaume S, Bates T, Lablee J, Betts T, Taylor J (2020). "Combining Spatial Data Layers Using Fuzzy Inference Systems: Application to an Agronomic Case Study." In *Proceedings of the 6th International Conference on Geographical Information Systems Theory, Applications and Management (GISTAM 2020)*, 62-71. ISBN 978-989-758-425-1.

Mora-Herrera DY, Guillaume S, Snoeck D, Zuniga Escobar O (2020). "A fuzzy logic based soil chemical quality index for cacao." *Computers and Electronics in Agriculture*, **177**, 105624. doi:10.1016/j.compag.2020.105624.

See Also

NewFusion

Data Fusion documentation

Examples

```
# more information about this example in the vignette "Data Fusion with GeoFIS"
# section "Learning illustration"

library(GeoFIS)

data(fusion_cars)

fusion <- NewFusion(fusion_cars)
a <- NewFusionInput("a", NewMfTrapezoidalInf(4, 20), "A")
v <- NewFusionInput("v", NewMfTrapezoidalSup(100, 500), "V")
s <- NewFusionInput("s", NewMfTrapezoidalSup(120, 220), "S")
c <- NewFusionInput("c", NewMfTrapezoidalInf(6, 16), "C")
owa_aggreg <- NewFusionAggreg("score", NewAggregOwa(c(1, 0, 0, 0)), a, v, s, c)
fusion$aggregate <- owa_aggreg
fusion$perform()
score <- fusion$output()["score"]
print(score)
```

FusionLabel *Class "FusionLabel"*

Description

Defines the allowed labels for the FisPro::Mfs of the fuzzy inputs or output in the FisPro::Fis "Fusion"

Active bindings

`very_low` character vector (read-only), The very_low label

`low` character vector (read-only), The low label

`average` character vector (read-only), The average label

`high` character vector (read-only), The high label

`very_high` character vector (read-only), The very_high label

Methods

Public methods:

- `FusionLabel$get_labels()`

Method `get_labels()`**:** Get the allowed labels depending on the granularity in the FisPro::Fis
for granularity 2, allowed labels are: [low, high]
for granularity 3, allowed labels are: [low, average, high]
for granularity 4, allowed labels are: [very_low, low, high, very_high]
for granularity 5, allowed labels are: [very_low, low, average, high, very_high]

Usage:

```
FusionLabel$get_labels(granularity)
```

Arguments:

`granularity` integer value, The granularity of the fuzzy inputs or output in the FisPro::Fis (value in range [2, 5])

Returns: character vector, The allowed labels for the granularity

---

`fusion_cars` *Fusion Cars dataset*

Description

Illustration dataset for data fusion numerical operators learning

Usage

```
data(fusion_cars)
```

Format

data.frame object with four cars described by four attributes:

`A` numeric value, the acceleration time (s) from 0 to 100 km/h

`V` numeric value, the volume of the trunk (l)

`S` numeric value, the maximum speed (km/h)

`C` numeric value, the gas consumption (l per 100 km)

`FuzzyDistance` *The "Fuzzy" distance*

Description

Function to create a "Fuzzy" distance
The fuzzy distance function is based on a fuzzy partition that allows for integrating expert knowledge into distance calculations
To be used with the Zoning `attribute_distance` field

Usage

```
FuzzyDistance(fisin)
```

Arguments

`fisin`	FisPro::FisIn object, The partition used for the fuzzy distance (must be a standardized fuzzy partition)

Value

Fuzzy distance object

References

Guillaume S, Charnomordic B, Loisel P (2013). "Fuzzy partitions: a way to integrate expert knowledge into distance calculations." *International Journal of Information Sciences*, **245**, 76-95. doi:10.1016/j.ins.2012.07.045.

Guillaume S, Charnomordic B (2013). "Fuzzy partition-based distance practical use and implementation." In CFP12FUZ-USB ICN (ed.), *IEEE International Conference on Fuzzy Systems, paper F-1136*.

`LearnOwaWeights` *Learn the OWA weights*

Description

Learn the OWA weights using a non-negative least-square optimization method with the constraint that the sum of weights must be equal to 1. The input values are previously sorted in increasing order. The resulting weights are given from min to max. More information is available in the vignette "Data Fusion with GeoFIS", section "Learning illustration".

Usage

```
LearnOwaWeights(data, target, digits = 3)
```

Arguments

`data`	data.frame or numeric matrix, The input data (all columns must be in range [0, 1])
`target`	numeric vector, The target data (must be in range [0, 1])
`digits`	integer value, The number of digits to which weights are to be rounded (default is 3)

Value

numeric vector, The OWA weights

LearnWamWeights *Learn the WAM weights*

Description

Learn the WAM weights using a non-negative least-square optimization method with the constraint that the sum of weights must be equal to 1.
More information is available in the vignette "Data Fusion with GeoFIS", section "Learning illustration".

Usage

```
LearnWamWeights(data, target, digits = 3)
```

Arguments

data	data.frame or numeric matrix, The input data (all columns must be in range [0, 1])
target	numeric vector, The target data (must be in range [0, 1])
digits	integer value, The number of digits to which weights are to be rounded (default is 3)

Value

numeric vector, The WAM weights

MaximumDistance *The "Maximum" distance*

Description

Function to create a "Maximum" distance
To be used with the Zoning `zone_distance` field

Usage

```
MaximumDistance()
```

Value

Maximum distance object

MeanDistance *The "Mean" distance*

Description

Function to create a "Mean" distance
To be used with the Zoning `zone_distance` field

Usage

```
MeanDistance()
```

Value

Mean distance object

MinimumDistance *The "Minimum" distance*

Description

Function to create a "Minimum" distance
To be used with the Zoning `zone_distance` field

Usage

```
MinimumDistance()
```

Value

Minimum distance object

MinkowskiDistance *The "Minkowski" distance*

Description

Function to create a "Minkowski" distance
To be used with the Zoning `combine_distance` field

Usage

```
MinkowskiDistance(power = 2)
```

Arguments

power	numeric value, The power of the Minkowski distance The default value is 2 (equivalent to euclidean distance)

Value

Minkowski distance object

NewAggregFis *Create object of class "AggregFis"*

Description

Function to create an aggregation operator of class AggregFis to be used in Fusion

Usage

```
NewAggregFis(fis, output_index = 1)
```

Arguments

fis	FisPro::Fis object, The Fis to be used in the aggregation operator
output_index	integer value, The index (1-based index) of the output in the Fis to be used in the aggregation (the default is 1)

Value

AggregFis object

See Also

Aggregation using linguistic rules

NewAggregFunction *Create object of class "AggregFunction"*

Description

Function to create an aggregation operator of class AggregFunction to be used in Fusion

Usage

```
NewAggregFunction(func)
```

Arguments

func	The function to be used for the aggregation

NewAggregOwa *Create object of class "AggregOwa"*

Description

Function to create an aggregation operator of class AggregOwa to be used in Fusion

Usage

```
NewAggregOwa(weights)
```

Arguments

weights	numeric vector, The weights of the OWA aggregation operator (the sum of the weights must be equal to 1 without negative values)

See Also

Aggregation using numerical operators

NewAggregWam *Create object of class "AggregWam"*

Description

Function to create an aggregation operator of class AggregWam to be used in Fusion

Usage

```
NewAggregWam(weights)
```

Arguments

weights	numeric vector, The weights of the WAM aggregation operator (the sum of the weights must be equal to 1 without negative values)

See Also

Aggregation using numerical operators

NewFisFusion *Create object of class "Fis" to be used in data fusion*

Description

Function to create object of class FisPro::Fis to be used in AggregFis

Usage

```
NewFisFusion(
  fis_name,
  input_names,
  input_granularities,
  output_name,
  output_conclusions
)
```

Arguments

fis_name	character vector, The name of the Fis
input_names	character vector, The Fis inputs names
input_granularities	integer vector, The granularity (number of membership functions) for each Fis input (granularity must be in range [2, 5])
output_name	character vector, The name of the Fis output
output_conclusions	numeric or character vector, The conclusions of the rules in the Fis the rules are generated according to the granularity of each input, in the lexicographic order of inputs Mfs (prod(input_granularities) rules are generated) if numeric vector, a crisp output FisPro::FisOutCrisp will be added to the Fis (all output conclusions must be be in range [0, 1]) if character vector, a fuzzy output FisPro::FisOutFuzzy will be added to the Fis, the output_conclusions contains the labels of Mfs in the fuzzy output (labels defined on FusionLabel) the length of output_conclusions must be equal to the number of generated rules.

Value

FisPro::Fis object

See Also

Aggregation using linguistic rules

NewFusion *Create object of class "Fusion"*

Description

Function to create object of class Fusion

Usage

```
NewFusion(...)
```

Arguments

...	arguments of Fusion constructor

Value

Fusion object

NewFusionAggreg *Create an aggregation node to be used in data fusion*

Description

Function to create an aggregation node to be used in Fusion

Usage

```
NewFusionAggreg(name, aggreg, ...)
```

Arguments

name	character vector, The name of the node
aggreg	Aggreg object, The aggregation operator to be used to compute the aggregation of satisfaction degrees must be an AggregWam, AggregOwa, AggregFis or AggregFunction object
...	data.tree::Node objects, The nodes to aggregate can be an input node built with NewFusionInput or an aggregate node built with NewFusionAggreg for a hierarchical aggregation structure

Value

data.tree::Node object

See Also

Aggregation of the degrees

NewFusionInput *Create an input node to be used in data fusion*

Description

Function to create an input node to be used in Fusion

Usage

```
NewFusionInput(name, mf, attribute = name)
```

Arguments

name	character vector, The name of the node
mf	FisPro::Mf object, The membership function to be used to compute the satisfaction degree of the input
attribute	character vector, The attribute name in the source dataset (default is the same as name)

Value

data.tree::Node object

See Also

From raw data to satisfaction degrees

NewZoning *Create object of class "Zoning"*

Description

Function to create object of class Zoning

Usage

```
NewZoning(...)
```

Arguments

...	arguments of Zoning constructor

Value

Zoning object

tolima *Tolima dataset*

Description

Soil experimental data in three municipalities of Tolima department in Colombia (Mora-Herrera et al. 2020)

Usage

```
data(tolima)
```

Format

data.frame object with 30 observations and 8 attributes:

`Cadmium` numeric value, Cadmium in Soil (ppm)

`pH` numeric value, pH Soil (°pH)

`OM` numeric value, Organic Matter (%)

`P` numeric value, Available Phosphorus (ppm)

`K` numeric value, Exchangeable Potassium (meq/100 g)

`BalanceGap` numeric value, Balance Gap (%)

`Ngap_N_OpN` numeric value, N Gap (N/Ntarget)

`Base_S` numeric value, Base Saturation (%)

References

Mora-Herrera DY, Guillaume S, Snoeck D, Zúñiga Escobar O (2020). "A fuzzy logic based soil chemical quality index for cacao." *Computers and Electronics in Agriculture*, **177**, 105624. doi:10.1016/j.compag.2020.105624.

ZoneArea *The "Area" smallest zone*

Description

Function to create an "Area" smallest zone
To be used with the Zoning `smallest_zone` field

Usage

```
ZoneArea(area)
```

Arguments

area	numeric value, The minimum area of the zone to retain the zone in the Zoning process

Value

Area Smallest zone object

ZoneSize *The "Size" smallest zone*

Description

Function to create a "Size" smallest zone
To be used with the Zoning smallest_zone field

Usage

```
ZoneSize(number_of_points)
```

Arguments

number_of_points	integer value, The minimum number of points in the zone to retain the zone in the Zoning process

Value

Size Smallest zone object

Zoning *Class "Zoning"*

Description

The main class to perform zoning
A complete use-case example is described in the vignette "Zoning with GeoFIS"

Active bindings

`border` sp::SpatialPolygons object, The border used to limit the processed area, or `NULL` if the Convex Hull of data source is used
Only data points within the border polygon are processed
The default value is `NULL`

`neighborhood` numeric value, The minimum edge length shared by two Voronoi polygons for being considered as neighbors
or `NULL` if all contiguous Voronoi polygons are considered as neighbors
The default value is `NULL`

`attribute_distance` list of Distance object (write-only), The functions used to compute the distance between two data points in the attribute space
The length of the list must be equal to the number of zonable attributes, the distance objects are treated in the order of zonable attributes
In case of a single attribute into the zonable dataset, the list is optional and a single Distance object can be provided
Allowed distance objects: EuclideanDistance, FuzzyDistance or `NULL` if the attribute should not be used in the zoning process
The default value is a list of EuclideanDistance
See Zoning documentation main parameters univariate distance

`combine_distance` Distance object (write-only), The function used to combine attribute distances in case of multivariate zoning
Allowed distance objects: EuclideanDistance or MinkowskiDistance
The default value is EuclideanDistance See Zoning documentation main parameters multivariate combination

`zone_distance` Distance object (write-only), The function used to compute the distance between 2 zones
Allowed distance objects: MaximumDistance, MinimumDistance or MeanDistance
The default value is MaximumDistance
The pair of zones to be merged are those for which the `zone_distance` is minimum.
See Zoning documentation main parameters between zone distance

`smallest_zone` Smallest zone object (write-only), This criterion is used to determine the smallest size for a zone (number of points or area) to be kept in the final map
Allowed Smallest zone objects: ZoneSize or ZoneArea
The default value is ZoneSize with 1 point

Methods

Public methods:

- `Zoning$new()`
- `Zoning$zonable_data()`
- `Zoning$perform_voronoi()`
- `Zoning$voronoi_map()`
- `Zoning$perform_neighborhood()`

- Zoning$neighborhood_map()
- Zoning$perform_zoning()
- Zoning$map_size()
- Zoning$map()
- Zoning$maps()

Method `new()`**:** Constructor, create a new instance of Zoning

Usage:

```
Zoning$new(source, warn = TRUE)
```

Arguments:

`source` sp::SpatialPointsDataFrame or sp::SpatialMultiPointsDataFrame object, The data source

`warn` logical value, Show warnings if TRUE, default value is TRUE

Method `zonable_data()`**:** Get the zonable data
Keep only the attributes that can be used in the zoning process, meaning numeric atributes, without missing values and with a range that is not limited to a unique value
The last condition is required by the min-max standardization process

Usage:

```
Zoning$zonable_data()
```

Returns: sp::SpatialPointsDataFrame object

Method `perform_voronoi()`**:** Compute the Voronoi diagram

Usage:

```
Zoning$perform_voronoi()
```

Method `voronoi_map()`**:** Get the Voronoi map

Usage:

```
Zoning$voronoi_map()
```

Returns: sp::SpatialPolygons object

Method `perform_neighborhood()`**:** Identify adjacent polygons in the voronoi tesselation

Usage:

```
Zoning$perform_neighborhood()
```

Method `neighborhood_map()`**:** Get the neighborhood map

Usage:

```
Zoning$neighborhood_map()
```

Returns: sp::SpatialLinesDataFrame object

Method `perform_zoning()`**:** Perform the zoning

Usage:

```
Zoning$perform_zoning()
```

Method `map_size()`**:** Get the number of maps with different number of zones available after perform zoning

Usage:

```
Zoning$map_size()
```

Returns: integer value

Method `map()`**:** Get the map corresponding to a number of zones

Usage:

```
Zoning$map(number_of_zones)
```

Arguments:

`number_of_zones` integer value, The number of zones in the map

Returns: sp::SpatialPolygonsDataFrame object

Method `maps()`**:** Get the maps corresponding to a number of zones

Usage:

```
Zoning$maps(number_of_zones)
```

Arguments:

`number_of_zones` integer vector, The number of zones in each map

Returns: list of sp::SpatialPolygonsDataFrame object

References

Pedroso M, Taylor J, Tisseyre B, Charnomordic B, Guillaume S (2010). "A segmentation algorithm for the delineation of management zones." *Computer and Electronics in Agriculture*, **70**(1), 199-208. doi:10.1016/j.compag.2009.10.007.

Guillaume S, Charnomordic B, Loisel P (2013). "Fuzzy partitions: a way to integrate expert knowledge into distance calculations." *International Journal of Information Sciences*, **245**, 76-95. doi:10.1016/j.ins.2012.07.045.

Guillaume S, Charnomordic B (2013). "Fuzzy partition-based distance practical use and implementation." In CFP12FUZ-USB ICN (ed.), *IEEE International Conference on Fuzzy Systems, paper F-1136*.

See Also

NewZoning

Zoning documentation

Index