Package ‘GeoFIS’

April 23, 2022

Type Package

Title Spatial Data Processing for Decision Making

Version 1.0.3

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


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 (>= 1.0.0), rgeos, nnls

SystemRequirements GNU make, C++14, gmp, mpfr

RdMacros Rdpack

NeedsCompilation yes

LinkingTo Rcpp, BH, FisPro

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

RoxygenNote 7.1.1

VignetteBuilder knitr, R.rsp

Repository CRAN

Date/Publication 2022-04-23 08:00:13 UTC
AggregFis

R topics documented:

<table>
<thead>
<tr>
<th>Class</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AggregFis</td>
<td>The Fis aggregation operator to be used in Fusion</td>
</tr>
</tbody>
</table>

**Slots**

- **fis** `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
AggregFunction

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

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**

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

- **id** *integer* value, The id of the polygon

---

**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  Node object, or a list of 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$\texttt{new(source)}

*Arguments:*

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

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

*Usage:*

Fusion$\texttt{perform()}

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

*Usage:*

Fusion$\texttt{output()}

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

**References**


# FusionLabel

## See Also

- `NewFusion`
- Data Fusion documentation

## Examples

```r
# 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 Mfs of the fuzzy inputs or output in the 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 \texttt{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]

\textit{Usage:}
\texttt{FusionLabel$\textunderscore get\_labels(granularity)}

\textit{Arguments:}
granularity\hspace{1em}integer value, The granularity of the fuzzy inputs or output in the \texttt{Fis} (value in range \([2, 5]\])

\textit{Returns:} character vector, The allowed labels for the granularity

---

\textbf{fusion\_cars} \hspace{1em} \textit{Fusion Cars dataset}

\textbf{Description}
Illustration dataset for data fusion numerical operators learning

\textbf{Usage}
\texttt{data(fusion\_cars)}

\textbf{Format}
\texttt{data.frame} object with four cars described by four attributes:

\texttt{A} numeric value, the acceleration time (s) from 0 to 100 km/h
\texttt{V} numeric value, the volume of the trunk (l)
\texttt{S} numeric value, the maximum speed (km/h)
\texttt{C} numeric value, the gas consumption (l per 100 km)

---

\textbf{FuzzyDistance} \hspace{1em} \textit{The "Fuzzy" distance}

\textbf{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 \texttt{Zoning attribute\_distance} field
Usage

FuzzyDistance(fisin)

Arguments

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

Value

Fuzzy distance object

References


GeoFIS

GeoFIS package

Description

*GeoFIS* is an open source software that provides methods for processing spatial data for decision making through a user-friendly interface (Leroux et al. 2018). This R package implements two main functionalities: management zone delineation (Pedroso et al. 2010) and data aggregation (Mora-Herrera et al. 2020; Guillaume et al. 2020). All the mentioned publications are available from the *GeoFIS* web site.

Author(s)

*GeoFIS* Team <contact@geofis.org>

References


LearnOwaWeights


See Also

https://www.geofis.org

---

**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**

```r
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** - 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 **Fis** to be used in **AggregFis**

**Usage**

```r
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 **FisOutCrisp** will be added to the Fis (all output conclusions must be in range \([0, 1]\)). If character vector, a fuzzy output **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**

**Fis** object

**See Also**

*Aggregation using linguistic rules*
### NewFusion

Create object of class "Fusion"

**Description**

Function to create object of class `Fusion`

**Usage**

```r
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**

```r
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
- `...` `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**

- `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** 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**

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 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**


---

### 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)
**ZoneSize**

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

Usage:
Zoning$new(source, warn = TRUE)

Arguments:
source SpatialPointsDataFrame or 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 attributes, 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: 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: SpatialPolygons object

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

Usage:
Zoning$perform_neighborhood()

Method `neighborhood_map()`: Get the neighborhood map

Usage:
Zoning$neighborhood_map()

Returns: 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:  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 SpatialPolygonsDataFrame object

References


See Also
NewZoning
Zoning documentation
Index

* datasets
  conductivity_2014, 4
  conductivity_border, 5
  fusion_cars, 8
  tolima, 18

  AggregFis, 2, 13, 15, 16
  AggregFunction, 3, 13, 16
  AggregOwa, 3, 14, 16
  AggregWam, 4, 14, 16

  character, 7, 8, 15–17
  conductivity_2014, 4
  conductivity_border, 5

  data.frame, 6, 8, 10, 11, 18

  EuclideanDistance, 5, 20

  Fis, 2, 7, 8, 13, 15
  FisIn, 9
  FisOutCrisp, 15
  FisOutFuzzy, 15
  Fusion, 2–4, 6, 6, 13, 14, 16, 17
  fusion_cars, 8
  FusionLabel, 7, 15
  FuzzyDistance, 8, 20

  GeoFIS, 9
  GeoFIS-package (GeoFIS), 9

  integer, 2, 5, 8, 10, 11, 13, 15, 19, 22

  LearnOwaWeights, 10
  LearnWamWeights, 11
  list, 6, 20, 22
  logical, 21

  MaximumDistance, 11, 20
  MeanDistance, 12, 20
  Mf, 7, 17

  MinimumDistance, 12, 20
  MinkowskiDistance, 12, 20

  NewAggregFis, 3, 13
  NewAggregFunction, 3, 13
  NewAggregOwa, 3, 14
  NewAggregWam, 4, 14
  NewFisFusion, 15
  NewFusion, 7, 16
  NewFusionAggreg, 16, 16
  NewFusionInput, 16, 17
  NewZoning, 17, 22
  Node, 6, 16, 17
  numeric, 3, 4, 8, 10, 11, 13–15, 18–20

  sp, 6
  Spatial, 6
  SpatialLinesDataFrame, 21
  SpatialMultiPointsDataFrame, 21
  SpatialPointsDataFrame, 4, 21
  SpatialPolygons, 20, 21
  SpatialPolygonsDataFrame, 5, 22

  tolima, 18

  ZoneArea, 18, 20
  ZoneSize, 19, 20
  Zoning, 5, 8, 11, 12, 17–19, 19, 21

23