

Package ‘biokNN’

October 12, 2022

Type Package

Title Bi-Objective k-Nearest Neighbors Imputation for Multilevel Data

Version 0.1.0

Depends R (>= 2.10)

Maintainer Maximiliano Cubillos <mcub@econ.au.dk>

Description The bi-objective k-nearest neighbors method (biokNN) is an imputation method designed to estimate missing values on data with a multilevel structure. The original algorithm is an extension of the k-nearest neighbors method proposed by Bertsimas et al. (2017) (<<https://jmlr.org/papers/v18/17-073.html>>) using a bi-objective approach. A brief description of the method can be found in Cubillos (2021) (<<https://pure.au.dk/portal/files/214627979/biokNN.pdf>>). The 'biokNN' package provides an R implementation of the method for datasets with continuous variables (e.g. employee productivity, student grades) and a categorical class variable (e.g. department, school). Given an incomplete dataset with such structure, this package produces complete datasets using both single and multiple imputation, including visualization tools to better understand the pattern of the missing values.

License GPL (>= 2)

URL <https://github.com/mcubillos3/biokNN>

BugReports <https://github.com/mcubillos3/biokNN/issues>

Suggests knitr, rmarkdown, testthat

Encoding UTF-8

LazyData true

Imports dplyr, cluster, mice, stats, magrittr, ggplot2, tidyr, desc, lme4, mitml

RoxygenNote 7.1.1

NeedsCompilation no

Author Maximiliano Cubillos [aut, cre]
(<<https://orcid.org/0000-0002-2826-9728>>)

Repository CRAN

Date/Publication 2021-04-22 07:20:15 UTC

R topics documented:

biokNN.impute	2
biokNN.impute.mi	3
calibrate	4
create.multilevel	5
data.example	6
missing.plot	6
pattern.plot	7
target.boxplot	8

Index	9
--------------	----------

biokNN.impute	<i>Impute multilevel dataset</i>
---------------	----------------------------------

Description

This function returns a dataframe with a complete dataset, where the missing values are imputed using a bi-objective kNN method. It assumes that the class variable is complete and its name is known, and the rest of the variables are numerical.

Usage

```
biokNN.impute(
  data,
  className,
  nIter = 10,
  weight = 0.5,
  k = 10,
  distance = "gower"
)
```

Arguments

data	A dataframe with missing values
className	name of the variable that contains the classes
nIter	number of iterations, default = 10
weight	weight of the kNN values in the objective function, default = 0.5
k	number of nearest neighbours, default = 10
distance	distance function used to get the k-nearest neighbors

Value

A dataframe with the imputed data

Examples

```
data(data.example)
complete.data <- biokNN.impute(data.example,
  className = "class",
  nIter = 10,
  weight = 0.9,
  k = 15,
  distance = "gower")
```

biokNN.impute.mi *Multiple imputation for a multilevel dataset*

Description

This function returns a list of m complete datasets, where the missing values are imputed using a bi-objective kNN method. It assumes that the class variable name is known, and the rest of the variables are numerical.

Usage

```
biokNN.impute.mi(
  data,
  className,
  m = 5,
  nIter = 10,
  weight = 0.5,
  k = 10,
  distance = "gower"
)
```

Arguments

data	A dataframe with missing values
className	name of the variable that contains the classes
m	number of imputations
nIter	number of iterations, default = 10
weight	weight of the kNN values in the objective function, default = 0.5
k	number of nearest neighbours, default = 10
distance	distance function used to get the k-nearest neighbors

Value

A dataframe with the imputed data

Examples

```

data(data.example)
complete.data.mi <- biokNN.impute.mi(data.example,
  className = "class",
  m = 3,
  nIter = 10,
  weight = 0.9,
  k = 15,
  distance = "gower")
# View completed data sets
str(complete.data.mi)

```

calibrate

Calibrate parameters

Description

This function returns a vector with the two parameters required by the biokNN method where the first value is the weighting parameter and the second the number of neighbors

Usage

```

calibrate(
  data,
  className,
  prop_valid = 0.1,
  nIter = 10,
  distance = "gower",
  weight_space = NULL,
  k_space = NULL,
  print = FALSE
)

```

Arguments

data	A dataframe with missing values
className	name of the variable that contains the classes
prop_valid	proportion of missing values
nIter	number of iterations, default = 10
distance	distance function used to get the k-nearest neighbors
weight_space	vector with the calibration values to test for the weight parameter
k_space	vector with the calibration values to test for the number of neighbors
print	option to print the RMSE values of the parameters used for calibration (print = TRUE).

Value

A dataframe with the imputed data

Examples

```
data(data.example)
calibrate(data.example,
          "class",
          prop_valid = 0.3,
          weight_space = c(0.5, 0.7, 0.9),
          k_space = c(10, 15),
          print = TRUE)
```

<code>create.multilevel</code>	<i>Generate multilevel dataset</i>
--------------------------------	------------------------------------

Description

This function returns a dataframe with a multilevel structure. It generates a dataframe using a varying intercepts/varying slopes linear regression with a single target variable y .

Usage

```
create.multilevel(
  nClass = 10,
  nVars = 1,
  classMean = 10,
  classSD = 0,
  beta0 = 0,
  tau0 = 1,
  beta = c(1),
  tau = c(1),
  sigma2 = 1
)
```

Arguments

<code>nClass</code>	number of classes
<code>nVars</code>	number of independent variables (X)
<code>classMean</code>	average number of observations per class
<code>classSD</code>	standard deviation of the number of observations per class
<code>beta0</code>	intercept parameter
<code>tau0</code>	variance of the parameter between classes
<code>beta</code>	vector with the slope parameters, one for each independent variable
<code>tau</code>	vector with the variance of the slope parameters, one for each independent variable
<code>sigma2</code>	error variance

Value

A dataframe with the multilevel dataset

Examples

```
df <- create.multilevel(nClass = 20,
                        nVars = 1,
                        classMean = 10,
                        classSD = 2)
```

data.example

Example data set with missing values and multilevel structure

Description

This is a generated dataset containing a class variable, a dependent variable y, and an independent variable X. The data contains missing values in both y and X, assuming a Missing Completely at Random (MCAR) pattern and a 30

Usage

data.example

Format

An object of class `data.frame` with 100 rows and 3 columns.

Fields

y: Object of class "numeric", dependent variable
X: Object of class "numeric", independent variable
class: Object of class "Factor", class variable

missing.plot

Plot number of missing values by class

Description

This function returns a dataframe with a multilevel structure. It generates a dataframe using a varying intercepts/varying slopes linear regression with a single target variable y.

Usage

```
missing.plot(df, class)
```

Arguments

df dataframe with missing values
class name of the variable containing classes

Value

A barplot with the number of missing values by class, by variable

Examples

```
data(data.example)  
missing.plot(data.example, "class")
```

pattern.plot *Plot pattern of missing values by class*

Description

This function returns a dataframe with a multilevel structure. It generates a dataframe using a varying intercepts/varying slopes linear regression with a single target variable y.

Usage

```
pattern.plot(df, class)
```

Arguments

df dataframe with missing values
class name of the variable containing classes

Value

A plot with the patter of missing values by class, by variable

Examples

```
data(data.example)  
pattern.plot(data.example, "class")
```

target.boxplot	<i>Plot pattern of missing values by class</i>
----------------	--

Description

This function returns a dataframe with a multilevel structure. It generates a dataframe using a varying intercepts/varying slopes linear regression with a single target variable y.

Usage

```
target.boxplot(df, y, class)
```

Arguments

df	dataframe with missing values
y	target variable
class	name of the variable containing classes

Value

A boxplot for each class of the target variable

Examples

```
data(data.example)  
target.boxplot(data.example, y, "class")
```


Index

* datasets

data.example, 6

biokNN.impute, 2

biokNN.impute.mi, 3

calibrate, 4

create.multilevel, 5

data.example, 6

missing.plot, 6

pattern.plot, 7

target.boxplot, 8