Package ‘biokNN’

November 26, 2021

Type Package

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

Version 0.1.1

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

R topics documented:

`biokNN_impute` *Impute multilevel dataset*

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,
  varNames,
  nIter = 10,
  alpha = 0.5,
  k = 10,
  distance = "gower"
)
```

Arguments

`data`	A dataframe with missing values
`className`	name of the variable that contains the classes
`varNames`	vector with the names of the variables to be imputed
`nIter`	number of iterations, default = 10
`alpha`	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",
              varNames = c("y"),
              nIter = 10,
              alpha = 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,
  varNames,
  m = 5,
  nIter = 10,
  alpha = 0.5,
  k = 10,
  distance = "gower"
)
```

Arguments

`data`	A dataframe with missing values
`className`	name of the variable that contains the classes
`varNames`	vector with the names of the variables to be imputed
`m`	number of imputations
`nIter`	number of iterations, default = 10
`alpha`	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",
               varNames = c("y"),
               m = 3,
               nIter = 10,
               alpha = 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 requiered by the biokNN method where the first value is the weighting parameter and the second the number of neighbors

Usage

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

Arguments

`data`	A dataframe with missing values
`className`	name of the variable that contains the classes
`varNames`	vector with the names of the variables to be imputed
`prop_valid`	proportion of missing values
`nIter`	number of iterations, default = 10
`distance`	distance function used to get the k-nearest neighbors
`alpha_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,
         className = "class",
         varNames = c("y"),
         prop_valid = 0.3,
         alpha_space = c(0.5, 0.7, 0.9),
         k_space = c(10, 15),
         print = TRUE)
```

`create_multilevel` *Generate multilevel dataset*

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

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

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")
```

`normalize` *Normalize dataset*

Description

This function returns a dataset with normalized values for numerical variables

Usage

```
normalize(df_miss)
```

Arguments

`data`	A dataframe with missing values

Value

A dataframe with normalized values for the numerical variables

Examples

```
data(data_example)
normalize(data_example)
```

`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` *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

```
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