Package 'linkspotter'

July 23, 2020

Type Package

Title Bivariate Correlations Calculation and Visualization

Version 1.3.0

Date 2020-07-23

Description
Compute and visualize using the 'visNetwork' package all the bivariate correlations of a dataframe. Several and different types of correlation coefficients (Pearson's r, Spearman's rho, Kendall's tau, distance correlation, maximal information coefficient and
equal-freq discretization-based maximal normalized mutual information) are used according to the variable couple type (quantitative vs categorical, quantitative vs quantitative, categorical vs categorical).

License MIT + file LICENSE

URL https://github.com/sambaala/linkspotter

BugReports https://github.com/sambaala/linkspotter/issues

Encoding UTF-8

LazyData true

Depends R (>= 3.2.0)

Imports shiny, visNetwork, infotheo, minerva, energy, mclust, rAmCharts, pbapply, ggplot2, dplyr, tidyr, shinybusy

Collate 'linkspotterComplete.R' 'clusterVariables.R' 'linkspotterDurationEstimator.R' 'linkspotterGraph.R' 'linkspotterGraphOnMatrix.R' 'linkspotterOnFile.R' 'linkspotterUI.R' 'corCouplesToMatrix.R' 'matrixToCorCouples.R' 'maxNMI.R' 'BeEFdiscretization_numfact.R' 'BeEFdiscretization_numnum.R' 'multiBivariateCorrelation.R' 'NormalizedMI.R' 'createShinyAppFolder.R' 'EFdiscretization.R' 'non_informative_var.R'

RoxygenNote 7.1.1

Suggests knitr, rmarkdown

VignetteBuilder knitr

NeedsCompilation no

Author Alassane Samba [aut, cre],
Orange [cph]

Maintainer Alassane Samba <alassane.samba@orange.com>

Repository CRAN

Date/Publication 2020-07-23 10:40:02 UTC

R topics documented:

`BeEFdiscretization.numfact`

BeEF: Best Equal-Frequency discretization

Description

Discretize a quantitative variable by optimizing the obtained the Normalized Mutual Information with a target qualitative variable

Usage

```
BeEFdiscretization.numfact(
  continuousY,
  factorX,
  includeNA = T,
  showProgress = F
)
```

Arguments

continuousY	a vector of numeric.
factorX	a vector of factor.
includeNA	a boolean. TRUE to include NA value as a factor level.
showProgress	a boolean to decide whether to show the progress bar.

Value

a factor.

Examples

```
# calculate a correlation dataframe
data(iris)
discreteSepalLength=BeEFdiscretization.numfact(continuousY=iris$Sepal.Length,factorX=iris$Species)
summary(discreteSepalLength)
```

BeEFdiscretization.numnum

BeEF: Best Equal-Frequency discretization (for a couple of quantitative variables)

Description

Discretize two quantitative variables by optimizing the obtained the Normalized Mutual Information

Usage

```
BeEFdiscretization.numnum(
  continuousX,
  continuousY,
  maxNbBins = 100,
  includeNA = T,
  showProgress = F
)
```

Arguments

continuousX	a vector of numeric.
continuousY	a vector of numeric.
maxNbBins	an integer corresponding to the number of bins limitation (for computation time limitation), maxNbBins=100 by default.
includeNA	a boolean. TRUE to include NA value as a factor level.
showProgress	a boolean to decide whether to show the progress bar.

Value

a list of two factors.

Examples

```
# calculate a correlation dataframe
data(iris)
disc=BeEFdiscretization.numnum(iris$Sepal.Length,iris$Sepal.Width)
summary(disc$x)
summary(disc$y)
```

`clusterVariables` *Variable clustering (using Normal Mixture Modeling for Model-Based Clustering : mclust)*

Description

Computation of a variable clustering on a correlation matrix.

Usage

```
clusterVariables(corMatrix, nbCluster = 1:9)
```

Arguments

`corMatrix`	a dataframe corresponding to a correlation matrix
`nbCluster`	an integer or a vector of integers corresponding to the preferred number of cluster for the unsupervised learning.

Value

a dataframe: the first column contains the variable names, the second column the index of the cluster they are affected to.

Examples

```
# calculate a correlation dataframe
data(iris)
corDF <- multiBivariateCorrelation(dataset = iris, corMethods = "MaxNMI")
# tranform to correlation matrix
corMatrix <- corCouplesToMatrix(x1_x2_val = corDF[,c('X1','X2',"MaxNMI")])
# perform the clustering
corGroups <- clusterVariables(corMatrix = corMatrix, nbCluster = 3)
print(corGroups)
```

corCouplesToMatrix *Couples to matrix*

Description

Transform a 2 column correlation dataframe into a correlation matrix

Usage

```
corCouplesToMatrix(x1_x2_val)
```

Arguments

x1_x2_val	a specific dataframe containing correlations values resulting from the function multiBivariateCorrelation() and containing only one coefficient type.

Value

a dataframe corresponding to a correlation matrix.

Examples

```
# calculate a correlation dataframe
data(iris)
corDF<-multiBivariateCorrelation(dataset = iris, corMethods = "MaxNMI")
corMatrix<-corCouplesToMatrix(x1_x2_val = corDF[,c('X1','X2',"MaxNMI")])
print(corMatrix)
corCouples<-matrixToCorCouples(corMatrix,coefName="pearson")
print(corCouples)
```

createShinyAppFolder *Ready-for-deployment shiny app folder creation*

Description

This function creates a shiny app folder containing a shiny app object directly readable by a shiny-server.

Usage

```
createShinyAppFolder(linkspotterObject, folderName)
```

Arguments

linkspotterObject	a linkspotter object, resulting from linkspotterComplete() or linkspotterOnFile() functions.
folderName	a character string corresponding to the name of the shiny app folder to create.

Examples

```
data(iris)
lsOutputIris<-linkspotterComplete(iris)
tmpShinyFolder<-tempdir()
createShinyAppFolder(lsOutputIris,
folderName=file.path(tmpShinyFolder,"myIrisLinkspotterShinyApp1")
)
## Not run:
# launch the shiny app
shiny::runApp(tmpShinyFolder)

## End(Not run)
```

EFdiscretization *EF: Equal-Frequency discretization*

Description

Discretize a quantitative variable with equal frequency binning if possible

Usage

```
EFdiscretization(continuousX, nX, nbdigitsX = 3)
```

Arguments

continuousX	a vector of numeric.
nX	an integer corresponding to the desired number of intervals.
nbdigitsX	number of significant digits to use in constructing levels. Default is 3.

Value

a factor.

Examples

```
data(iris)
disc.Sepal.Length=EFdiscretization(iris$Sepal.Length,5)
summary(disc.Sepal.Length)
```

is.not.informative.variable *Is a vector an non informative variable*

Description

This function determines if a given vector of numeric or factor is a non informative variable or not.

Usage

```
is.not.informative.variable(x, includeNA = T)
```

Arguments

x	a vector of numeric or factor.
includeNA	a boolean. TRUE to include NA value as a factor level.

Examples

```
data(iris)
is.not.informative.variable(iris$Sepal.Length)
```

linkspotterComplete *Linkspotter complete runner*

Description

Computation of correlation matrices, variable clustering and the customizable user interface to visualize them using a graph together with variables distributions and cross plots.

Usage

```
linkspotterComplete(
  dataset,
  targetVar = NULL,
  corMethods = c("pearson", "spearman", "kendall", "mic", "MaxNMI"),
  maxNbBins = 100,
  defaultMinCor = 0.3,
  defaultCorMethod = corMethods[length(corMethods)],
  clusteringCorMethod = defaultCorMethod,
  nbCluster = 1:9,
  printInfo = T,
  appTitle = "Linkspotter",
  htmlTop = "",
  htmlBottom = ""
)
```

Arguments

dataset	the dataframe which variables bivariate correlations are to be analyzed.
targetVar	a vector of character strings corresponding to the names of the target variables. If not NULL, correlation coefficients are computed only with that target variables.
corMethods	a vector of correlation coefficients to compute. The available coefficients are the following : c("pearson","spearman","kendall","mic","distCor","MaxNMI"). It is not case sensitive and still work if only the beginning of the word is put (e.g. pears).
maxNbBins	an integer used if corMethods include 'MaxNMI'. It corresponds to the number of bins limitation (for computation time limitation), maxNbBins=100 by default.
defaultMinCor	a double between 0 and 1. It is the minimal correlation absolute value to consider for the first graph plot.
defaultCorMethod	a string. One of "pearson","spearman","kendall","mic", "distCor" or "MaxNMI". It is the correlation coefficient to consider for the first graph plot.
clusteringCorMethod	a string. One of "pearson","spearman","kendall","mic", "distCor" or "MaxNMI". It is the correlation coefficient to consider for the variables clustering.
nbCluster	an integer. It is the number of clusters to compute.
printInfo	a boolean indicating whether to print on the console some information about the dataset and the estimated computation time.
appTitle	a string taken as the title of the user interface.
htmlTop	a character string that enable to customize your shiny app by adding an HTML code in the HEAD tag.
htmlBottom	a character string that enable to customize your shiny app by adding an HTML code at the end of the BODY tag.

Value

a list containing all the material enabling to analyze correlations:

- computationTime: a string
- run_it: a shiny.appobj object enable to deploy instantly the user interface for a customizable visualization.
- dataset: the initial dataset
- corDF: a the correlation data.frame including values for all coefficients
- corMatrices: a list of correlation matrices
- corGroups: data.frame a data.frame list
- clusteringCorMethod: a character
- defaultMinCor: a numeric
- defaultCorMethod: a string
- corMethods: vector of strings

Examples

```
# run linkspotter on iris example data
data(iris)
lsOutputIris<-linkspotterComplete(iris)
summary(lsOutputIris)
## Not run:
# launch the UI
lsOutputIris$launchShiny(option=list(port=8000))

## End(Not run)
```

---

linkspotterGraph *Linkspotter graph runner*

Description

plot the Linkspotter graph

Usage

```
linkspotterGraph(
  corDF,
  variablesClustering = NULL,
  minCor = 0.3,
 corMethod = colnames(corDF)[-c(1:3, ncol(corDF))][length(colnames(corDF)[-c(1:3,
    ncol(corDF))])],
  smoothEdges = T,
  dynamicNodes = F,
  colorEdgesByCorDirection = F
)
```

Arguments

corDF	a specific dataframe containing correlations values resulting from the function multiBivariateCorrelation()
variablesClustering	a specific dataframe containing the output of the variable clustering resulting from the function clusterVariables()
minCor	a double between 0 and 1. It is the minimal correlation absolute value to consider for the first graph plot.
corMethod	a string. One of "pearson","spearman","kendall","mic", "distCor" or "MaxNMI". It is the correlation coefficient to consider for the first graph plot.
smoothEdges	a boolean. TRUE to let the edges be smooth.
dynamicNodes	a boolean. TRUE to let the graph re-organize itself after any movement.
colorEdgesByCorDirection	a boolean. TRUE to get the edges colored according to the correlation direction (positive-> blue, negative->red or NA->grey).

Value

a visNetwork object corresponding to a dynamic graph for the correlation matrix visualization.

Examples

```
# calculate a correlation dataframe
data(iris)
corDF=multiBivariateCorrelation(dataset = iris)
corMatrix=corCouplesToMatrix(x1_x2_val = corDF[,c('X1','X2',"spearman")])
corGroups=clusterVariables(corMatrix = corMatrix, nbCluster = 3)
# launch the graph
linkspotterGraph(corDF=corDF, variablesClustering=corGroups, minCor=0.3,
corMethod='spearman', colorEdgesByCorDirection=TRUE)
```

---

linkspotterGraphOnMatrix

Linkspotter graph on matrix

Description

Plot the Linkspotter graph from a correlation matrix.

Usage

```
linkspotterGraphOnMatrix(
  corMatrix,
  cluster = FALSE,
  variablesClustering = NULL,
  minCor = 0.3,
  corMethod = "Coef.",
  smoothEdges = T,
  dynamicNodes = F,
  colorEdgesByCorDirection = F
)
```

Arguments

corMatrix	a dataframe corresponding to a matrix of correlation or distance.
cluster	a boolean to decide if to cluster variables or an integer corresponding directly to the number of clusters to consider. If variablesClustering is filled, "cluster" parameter is ignored.
variablesClustering	a specific dataframe containing the output of the variable clustering resulting from the function clusterVariables()
minCor	a double between 0 and 1. It is the minimal correlation absolute value to consider for the first graph plot.

corMethod	a string. One of "pearson","spearman","kendall","mic", "distCor" or "MaxNMI". It is the correlation coefficient to consider for the first graph plot.
smoothEdges	a boolean. TRUE to let the edges be smooth.
dynamicNodes	a boolean. TRUE to let the graph re-organize itself after any movement.
colorEdgesByCorDirection	a boolean. TRUE to get the edges colored according to the correlation direction (positive-> blue, negative->red or NA->grey).

Value

a visNetwork object corresponding to a dynamic graph for the correlation matrix visualization.

Examples

```
# calculate a correlation dataframe
data(iris)
corDF=multiBivariateCorrelation(dataset = iris)
corMatrix=corCouplesToMatrix(x1_x2_val = corDF[,c('X1','X2',"pearson")])
# launch the graph
linkspotterGraphOnMatrix(corMatrix=corMatrix, minCor=0.3)
```

---

linkspotterOnFile *Process Linkspotter on an external file*

Description

This function imports an external dataset, computes its correlation matrices, variable clustering and the customizable user interface to visualize them using a graph.

Usage

```
linkspotterOnFile(
  file,
  corMethods = c("pearson", "spearman", "kendall", "mic", "MaxNMI"),
  defaultMinCor = 0.3,
  defaultCorMethod = corMethods[length(corMethods)],
  clusteringCorMethod = corMethods[length(corMethods)],
  nbCluster = 1:9,
  printInfo = T,
  appTitle = "Linkspotter",
  htmlTop = "",
  htmlBottom = "",
  ...
)
```

Arguments

file	the file containing a structured dataset which the bivariate correlations are to be analyzed.
corMethods	a vector of correlation coefficients to compute. The available coefficients are the following : c("pearson","spearman","kendall","mic","distCor","MaxNMI"). It is not case sensitive and still work if only the beginning of the word is put (e.g. pears).
defaultMinCor	a double between 0 and 1. It is the minimal correlation absolute value to consider for the first graph plot.
defaultCorMethod	a string. One of "pearson","spearman","kendall","mic", "distCor" or "MaxNMI". It is the correlation coefficient to consider for the first graph plot.
clusteringCorMethod	a string. One of "pearson","spearman","kendall","mic", "distCor" or "MaxNMI". It is the correlation coefficient to consider for the variables clustering.
nbCluster	an integer. It is the number of clusters to compute.
printInfo	a boolean indicating whether to print on the console some information about the dataset and the estimated computation time.
appTitle	a string taken as the title of the user interface.
htmlTop	a character string that enable to customize your shiny app by adding an HTML code in the HEAD tag.
htmlBottom	a character string that enable to customize your shiny app by adding an HTML code at the end of the BODY tag.
...	Further arguments to be passed to the used read.csv function.

Value

a list containing all the material enabling to analyze correlations:

- computationTime: a string
- run_it: a shiny.appobj object enable to deploy instantly the user interface for a customizable visualization.
- dataset: the initial dataset
- corDF: a the correlation data.frame including values for all coefficients
- corMatrices: a list of correlation matrices
- corGroups: data.frame a data.frame list
- clusteringCorMethod: a character
- defaultMinCor: a numeric
- defaultCorMethod: a string
- corMethods: vector of strings

Examples

```
# run linkspotter on iris example data
data(iris)
tmpCSV<-tempfile(fileext = '.csv')
write.csv(iris, tmpCSV, row.names = FALSE)
lsOutputIrisFromFile<-linkspotterOnFile(file=tmpCSV)
summary(lsOutputIrisFromFile)
## Not run:
# launch the UI
lsOutputIrisFromFile$launchShiny(options=list(port=8000))

## End(Not run)
```

---

linkspotterUI *Linkspotter user interface runner*

Description

Build the Linkspotter user interface

Usage

```
linkspotterUI(
  dataset,
  corDF,
  variablesClustering = NULL,
  defaultMinCor = 0.3,
  appTitle = "Linkspotter",
  htmlTop = "",
  htmlBottom = "",
  ...
)
```

Arguments

dataset	the dataframe which variables bivariate correlations are contained in corDF
corDF	a specific dataframe containing correlations values resulting from the function multiBivariateCorrelation()
variablesClustering	a specific dataframe containing the output of the variable clustering resulting from the function clusterVariables()
defaultMinCor	a double between 0 and 1. It is the minimal correlation absolute value to consider for the first graph plot.
appTitle	a character string taken as the title of the user interface.
htmlTop	a character string that enable to customize your shiny app by adding an HTML code in the HEAD tag.

htmlBottom	a character string that enable to customize your shiny app by adding an HTML code at the end of the BODY tag.
...	: arguments for 'shiny::shinyApp' function

Value

a 'shiny.appobj' object enable to deploy instantly the user interface for a customizable visualization.

Examples

```
# calculate a correlation dataframe
data(iris)
corDF=multiBivariateCorrelation(dataset = iris)
corMatrix=corCouplesToMatrix(x1_x2_val = corDF[,c('X1','X2',"MaxNMI")])
corGroups=clusterVariables(corMatrix = corMatrix, nbCluster = 3)
## Not run:
# launch the UI
linkspotterUI(dataset=iris, corDF=corDF, variablesClustering=corGroups,
defaultMinCor=0.3,cappTitle="Linkspotter on iris data",
options = list(port=8000)
)

## End(Not run)
```

---

matrixToCorCouples *Matrix to couples*

Description

Transform a correlation matrix into a correlation couples dataframe

Usage

```
matrixToCorCouples(matrix, coefName = "Coef.", sortByDescAbs = F)
```

Arguments

matrix	a dataframe corresponding to a matrix of correlation.
coefName	a string: the name of the coefficient the values of the matrix represent.
sortByDescAbs	a boolean to decide if to sort by descending absolute value of the coefficient.

Value

a dataframe corresponding to all correlation couples from the matrix.

Examples

```
# calculate a correlation dataframe
data(iris)
corDF<-multiBivariateCorrelation(dataset = iris)
corMatrix<-corCouplesToMatrix(x1_x2_val = corDF[,c('X1','X2',"pearson")])
print(corMatrix)
corCouples<-matrixToCorCouples(matrix = corMatrix,coefName="pearson")
print(corCouples)
```

`maxNMI` *Maximal Normalized Mutual Information (MaxNMI)*

Description

Computes the MaxNMI between the two variables whatever their types, by discretizing using Best Equal-Frequency-based discretization (BeEF) if necessary.

Usage

```
maxNMI(x, y, includeNA = T, maxNbBins = 100, showProgress = F)
```

Arguments

`x`	a vector of numeric or factor.
`y`	a vector of numeric or factor.
`includeNA`	a boolean. TRUE to include NA value as a factor level.
`maxNbBins`	an integer corresponding to the number of bins limitation (for computation time limitation), maxNbBins=100 by default.
`showProgress`	a boolean to decide whether to show the progress bar.

Value

a double between 0 and 1 corresponding to the MaxNMI.

Examples

```
# calculate a correlation dataframe
data(iris)
maxNMI(iris$Sepal.Length,iris$Species)
maxNMI(iris$Sepal.Length,iris$Sepal.Width)
```

`multiBivariateCorrelation` *Calculation of all the bivariate correlations in a dataframe*

Description

Computation of a correlation dataframe.

Usage

```
multiBivariateCorrelation(
  dataset,
  targetVar = NULL,
  corMethods = c("pearson", "spearman", "kendall", "mic", "MaxNMI"),
  maxNbBins = 100,
  showProgress = T
)
```

Arguments

`dataset`	the dataframe which variables bivariate correlations are to be analyzed.
`targetVar`	a vector of character strings corresponding to the names of the target variables. If not NULL, correlation coefficients are computed only with that target variables.
`corMethods`	a vector of correlation coefficients to compute. The available coefficients are the following : `c("pearson","spearman","kendall","mic","distCor","MaxNMI")`. It is not case sensitive and still work if only the beginning of the word is put (e.g. `pears`).
`maxNbBins`	an integer used if corMethods include 'MaxNMI'. It corresponds to the number of bins limitation (for computation time limitation), maxNbBins=100 by default.
`showProgress`	a boolean to decide whether to show the progress bar.

Value

a specific dataframe containing correlations values or each specified correlation coefficient.

Examples

```
# run linkspotter on iris example data
data(iris)
corDF<-multiBivariateCorrelation(iris)
print(corDF)
```

NormalizedMI *Maximal Normalized Mutual Information (MaxNMI) function for 2 categorical variables*

Description

Calculate the MaxNMI relationship measurement for 2 categorical variables

Usage

```
NormalizedMI(x, y, includeNA = T)
```

Arguments

x	a vector of factor.
y	a vector of factor.
includeNA	a boolean. TRUE to include NA value as a factor level.

Value

a double between 0 and 1 corresponding to the MaxNMI.

Examples

```
# calculate a correlation dataframe
data(iris)
discreteSepalLength=BeEFdiscretization.numfact(continuousY=iris$Sepal.Length,factorX=iris$Species)
NormalizedMI(iris$Species,discreteSepalLength)
```

Index