Package ‘linkspotter’

July 22, 2018

Type Package

Title Bivariate Correlations Calculation and Visualization

Version 1.2.0

Date 2018-07-18

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

Encoding UTF-8

LazyData true

Depends R (>= 3.2.0)

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

Collate 'linkspotterComplete.R' 'boostedBoxplot.R'
    'clusterVariables.R' 'emptyVar.R'
    'linkspotterDurationEstimator.R' 'linkspotterGraph.R'
    'linkspotterGraphOnMatrix.R' 'linkspotterOnFile.R'
    'linkspotterUI.R' 'corCouplesToMatrix.R' 'matrixToCorCouples.R'
    'maxNML.R' 'BeEFdiscretization_numfact.R'
    'BeEFdiscretization_numnum.R' 'multiBivariateCorrelation.R'
    'NormalizedMI.R' 'uselessVar.R' 'createShinyAppFolder.R'

RoxygenNote 6.0.1

Suggests knitr, rmarkdown

VignetteBuilder knitr

NeedsCompilation no

Author Alassane Samba [aut, cre], Orange [cph]
BeEFdiscretization.numfact

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

**Repository**  CRAN

**Date/Publication**  2018-07-22 12:20:03 UTC

---

**R topics documented:**

- BeEFdiscretization.numfact
- BeEFdiscretization.numnum
- clusterVariables
- corCouplesToMatrix
- createShinyAppFolder
- linkspotterComplete
- linkspotterGraph
- linkspotterGraphOnMatrix
- linkspotterOnFile
- linkspotterUI
- matrixToCorCouples
- maxNMI
- multiBivariateCorrelation
- NormalizedMI

---

**Description**

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

**Usage**

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

**Arguments**

- `continuousY`  a vector of numeric.
- `factorX`  a vector of factor.
- `includeFactorNA`  a boolean. TRUE to include NA value as a factor level.
- `showProgress`  a boolean to decide whether to show the progress bar.
**BeEFdiscretization.numnum**

**Value**

a double between 0 and 1 corresponding to the MaxNMI.

**Examples**

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

---

**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**

```r
BeEFdiscretization.numnum(continuousX, continuousY, maxNbbins = 100, showProgress = F)
```

**Arguments**

- `continuousX` a vector of numeric.
- `continuousY` a vector of numeric.
- `maxNbbins` an integer corresponding to the number of bin 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**

```r
# 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 clusters 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")
# transform 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)
createShinyAppFolder

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

Description

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

Usage

createShinyAppFolder(linkspotterShinyAppObject, folderName)

Arguments

linkspotterShinyAppObject: a shiny.appobj object, resulting from linkspotterUI(), linkspotterComplete()$run_it or linkspotterOnFile()$run_it functions.

cfolderName: a character string corresponding to the name of the shiny app folder to create.

Examples

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

## End(Not run)

---

**linkspotterComplete**  
**Linspotter 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**

```r
linkspotterComplete(dataset, corMethods = c("pearson", "spearman", "kendall", "mic", "MaxNMI"), 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.

- **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.
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

```r
# 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  
*Lin*spotter graph runner

Description

Run the linkSpotter graph

Usage

```r
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)
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)
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

```r
# 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**

Run the linkSpotter user interface

**Usage**

```r
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

```r
# calculate a correlation dataframe
data(iris)
corDF = multBivariateCorrelation(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, capptTitle = "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

```r
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.
**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**

```r
maxNMI(x, y, maxNbins = 100, showProgress = F)
```

**Arguments**

- `x`: a vector of numeric or factor.
- `y`: a vector of numeric or factor.
- `maxNbins`: an integer corresponding to the number of bin limitation (for computation time limitation), `maxNbins`=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**

```r
# 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)
```

```r
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**

```r
multiBivariateCorrelation(dataset, corMethods = c("pearson", "spearman", "kendall", "mic", "MaxNMI"), showProgress = T)
```

**Arguments**

- **dataset**: the dataframe which variables 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`).
- **showProgress**: a boolean to decide whether to show the progress bar.

**Value**

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

**Examples**

```r
# 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**

```r
NormalizedMI(x, y, includeNA = T)
```
**Normalized MI**

**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**

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

BeEFdiscretization.numfact, 2
BeEFdiscretization.numnum, 3

clusterVariables, 4
corCouplesToMatrix, 4
createShinyAppFolder, 5

linkspotterComplete, 6
linkspotterGraph, 7
linkspotterGraphOnMatrix, 8
linkspotterOnFile, 9
linkspotterUI, 11

matrixToCorCouples, 12
maxNMI, 13
multiBivariateCorrelation, 14

NormalizedMI, 14