

Package ‘varrank’

December 20, 2018

Type Package

Title Heuristics Tools Based on Mutual Information for Variable Ranking

Version 0.2

Author Gilles Kratzer [aut, cre] (<<https://orcid.org/0000-0002-5929-8935>>),
Reinhard Furrer [ctb] (<<https://orcid.org/0000-0002-6319-2332>>)

Maintainer Gilles Kratzer <gilles.kratzer@math.uzh.ch>

Description

A computational toolbox of heuristics approaches for performing variable ranking and feature selection based on mutual information well adapted for multivariate system epidemiology datasets. The core function is a general implementation of the minimum redundancy maximum relevance model. R. Battiti (1994) <[doi:10.1109/72.298224](https://doi.org/10.1109/72.298224)>. Continuous variables are discretized using a large choice of rule. Variables ranking can be learned with a sequential forward/backward search algorithm. The two main problems that can be addressed by this package is the selection of the most representative variable within a group of variables of interest (i.e. dimension reduction) and variable ranking with respect to a set of features of interest.

Encoding UTF-8

License GPL-3

LazyData true

Suggests knitr, rmarkdown, Boruta, DAAG, FSelector, caret, e1071,
mlbench, psych, varSelRF, gplots, entropy, testthat

URL <https://www.math.uzh.ch/pages/varrank/>

Imports stats, FNN, grDevices

VignetteBuilder knitr

RoxygenNote 6.1.0

NeedsCompilation no

Repository CRAN

Date/Publication 2018-12-20 08:50:03 UTC

R topics documented:

discretization	2
entropy.data	4
mi.data	5
plot.varrank	6
print.varrank	8
summary.varrank	9
varrank	10

Index	13
--------------	-----------

discretization	<i>Discretization of a Possibly Continuous Data Frame of Random Variables</i>
----------------	---

Description

This function discretizes data frame of possibly continuous random variables through rules for discretization. The discretization algorithms are unsupervised and univariate. See details for the complete list (the desired number of bins could also be provided).

Usage

```
discretization(data.df = NULL, discretization.method = "cencov", frequency = FALSE)
```

Arguments

data.df	a data frame containing the data to discretize, binary variables must be declared as factors, other as numeric vector. The data frame must be named.
discretization.method	a character vector giving the discretization method to use; see details. If a number is provided, the variable will be discretized by equal binning.
frequency	logical variable to select the output. If set to TRUE a list with the table of count for each bin and the discretized data frame is returned. If set to FALSE only the discretized data frame is returned.

Details

discretization() supports multiple rules for discretization. Below is the list of supported rules. IQR() stands for interquartile range.

fd stands for the Freedman–Diaconis rule. The number of bins is given by

$$\frac{\text{range}(x) * n^{1/3}}{2 * IQR(x)}$$

The Freedman–Diaconis rule is known to be less sensitive than the Scott's rule to outlier.

doane stands for doane's rule. The number of bins is given by

$$1 + \log_2 n + \log_2 \left(1 + \frac{|g|}{\sigma_g} \right)$$

is a modification of Sturges' formula which attempts to improve its performance with non-normal data.

cencov stands for Cencov's rule. The number of bins is given by:

$$n^{1/3}$$

rice stands for Rice' rule. The number of bins is given by:

$$2n^{1/3}$$

terrell-scott stands for Terrell-Scott's rule. The number of bins is given by:

$$(2n)^{1/3}$$

This is known that Cencov, Rice and Terrell-Scott rules over estimates k compared to other rules due to his simplicity.

sturges stands for Sturges's rule. The number of bins is given by:

$$1 + \log_2(n)$$

scott stands for Scott's rule. The number of bins is given by:

$$\text{range}(x) / \sigma(x) n^{-1/3}$$

kmeans apply the classical k-means clustering to one-dimensional continuous data.

Value

the discretized dataframe or a list containing the table of counts for each bin the discretized dataframe

Author(s)

Gilles Kratzer

References

Garcia, S., et al. (2013) A survey of discretization techniques: Taxonomy and empirical analysis in supervised learning. *IEEE Transactions on Knowledge and Data Engineering*, 25.4, 734-750.

Cebeci, Z. and Yıldız, F. (2017) Unsupervised Discretization of Continuous Variables in a Chicken Egg Quality Traits Dataset. *Turkish Journal of Agriculture-Food Science and Technology*, 5.4, 315-320.

Examples

```
rv <- rnorm(n = 100, mean = 0, sd = 2)

entropy.data(freqs.table = discretization(data.df = rv,
discretization.method = "fd",
frequency=TRUE)[[1]])
```

entropy.data	<i>Computes an Empirical Estimation of the Entropy from a Table of Counts</i>
--------------	---

Description

This function empirically estimates the Shannon entropy from a table of counts using the observed frequencies.

Usage

```
entropy.data(freqs.table)
```

Arguments

freqs.table a table of counts.

Details

The general concept of entropy is defined for probability distributions. The 'entropy.data' function estimates empirical entropy from data. The probability is estimated from data using frequency tables. Then the estimates are plug-in in the definition of the entropy to return the so-called empirical entropy. A common known problem of empirical entropy is that the estimations are biased due to the sampling noise. This is also known that the bias will decrease as the sample size increases.

Value

Shannon's entropy estimate on natural logarithm scale.

Author(s)

Gilles Kratzer

Examples

```
library("varrank")

rv <- rnorm(n = 100, mean = 0, sd = 2)

entropy.data(freqs.table = discretization(data.df = rv,
discretization.method = "fd",
frequency = TRUE)[[1]])
```

mi.data	<i>Empirical Estimate of the Mutual Information from a Table of Counts</i>
---------	--

Description

This function estimates the mutual information from observed data

Usage

```
mi.data(X, Y, discretization.method=NULL, k=NULL)
```

Arguments

X a data frame containing only numeric or continuous variables.
Y a data frame containing only numeric or continuous variables.
discretization.method a character vector giving the discretization method to use. See [discretization](#).
k in case of purely continuous dataset, the mutual information can be computed using the k-nearest neighbours.

Details

The mutual information estimation is computed from the observed frequencies through a plugin estimator based on entropy or using the estimator described in A. Kraskov, H. Stogbauer and P.Grassberger (2004) when the data frame is exclusively made of continuous variables.

The plugin estimator is $I(X, Y) = H(X) + H(Y) - H(X, Y)$, where $H()$ is the entropy computed with [entropy.data](#).

Value

Mutual information estimate.

Author(s)

Gilles Kratzer

References

Kraskov, A., Stogbauer, H. and Grassberger, P. (2004) Estimating mutual information. *Physical Review E*, 69:066138, 1–16.

Examples

```
Y <- rnorm(n = 100, mean = 0, sd = 2)
X <- rnorm(n = 100, mean = 5, sd = 2)

mi.data(X = Y, Y = X, discretization.method = "sturges")
```

plot.varrank	<i>Visualization of varrank output</i>
--------------	--

Description

plot method for varrank objects with multiple options.

Usage

```
## S3 method for class 'varrank'
plot(x,
      ## block separation
      colsep = TRUE,
      rowsep = TRUE,
      sepcol = "white",
      sepwidth=c(0.005,0.005),

      ## cell labeling
      cellnote = TRUE,
      notecex = 1.5,
      notecol = "black",
      digitcell = 3,

      ## Row/Column Labelling
      margins = c(6, 6, 4, 2),
      labelsxex = 1.2,

      ## color key + density info
      colkey = NULL,
      densadj = 0.25,
      textlines = 2,

      ## plot labels
      main = NULL,
      maincex = 1,
      ...)
```

Arguments

x	an object of class varrank.
colsep	(optional) a logical parameter to indicate if columns should be separated from others by narrow space of color. The default is TRUE.
rowsep	(optional) a logical parameter to indicate if rows should be separated from others by narrow space of color. The default is TRUE.
sepcol	(optional) the color to use to separate rows or columns. The default is white.

sepwidth	(optional) Vector of length 2 giving the width (colsep) or height (rowsep) the separator box drawn by colsep and rowsep as a function of the width (colsep) or height (rowsep) of a cell. The defaults is <code>c(0.005, 0.005)</code> .
cellnote	(optional) a logical parameter to indicate if the scores should be displayed in cells.
notecex	(optional) numeric scaling factor for scores. The default is 1.5.
notecol	(optional) character string specifying the color for cellnote text. Defaults to "black".
digitcell	(optional) integer that indicate how many digits of the scores should be displayed. The default is 3.
labelsce	the magnification factor to be used for x and y labels relative to the current setting of cex. The default is 1.2.
margins	numerical vector of the form <code>c(bottom, left, top, right)</code> which gives the number of lines of margin to be specified on the four sides of the plot. The default is <code>c(6, 6, 4, 2)</code> .
colkey	specification for the color scheme to be used. The default is a rainbow color scheme.
densadj	numeric scaling value for tuning the kernel width when a density plot is drawn on the color key. (See the <code>adjust</code> parameter for the density function for details.) Defaults is 0.25.
textlines	number of lines to display relevance/redundance in the key. Default is 2.
main	an overall title for the plot. Default is none.
maincex	main magnification to be used for the plot. Default is 1.
...	additional arguments passed to <code>image</code> .

Details

This plot method for `varrank` objects provides an extensible framework for the visualization `varrank` output analysis. The user is allowed to specify block separations, display of scores and the color scheme to be used. The other parameters give a full control on the output. The final rendering depends on the algorithm used. For a ‘forward’ search the key density is on the upper right corner and for a ‘backward’ search the key density is in the bottom left corner. The default color scheme is continuous heat color from blue to red. A popular alternative for creating color palettes is **RColorBrewer**, <https://cran.r-project.org/package=RColorBrewer>.

This plot method is very similar to the `heatmap.2` function from **gplots**, <https://cran.r-project.org/package=gplots>.

Author(s)

Gilles Kratzer

Examples

```
library(mlbench)
data(PimaIndiansDiabetes)
```

```
##forward search for all variables
out <- varrank(data.df = PimaIndiansDiabetes,
  method = "estevez",
  variable.important = "diabetes",
  discretization.method = "sturges",
  algorithm = "forward",scheme="mid")

##default output
plot(x = out)

##typical plot for high dimensional datasets
plot(x = out, colsep = FALSE, rowsep = FALSE, cellnote = FALSE)
```

print.varrank

Methods for Varrank Objects

Description

Methods for computing on varrank objects.

Usage

```
## S3 method for class 'varrank'
print(x,digits=5, ...)
```

Arguments

x	an object of class varrank.
digits	an integer specifying the number of digits to display in the output.
...	additional arguments passed to print.

Details

digits gives the number of digits that will be displayed in the output. If more information are needed. There exists a [summary](#) S3 function that display more details.

Author(s)

Gilles Kratzer

Examples

```
library(mlbench)
data(PimaIndiansDiabetes)

##forward search for all variables
varrank.output <- varrank(data.df = PimaIndiansDiabetes,
  method = "peng",
  variable.important = "diabetes",
  discretization.method = "sturges",
  algorithm = "forward", scheme = "mid")

##Print varrank output

varrank.output
```

summary.varrank

Summary Methods for Varrank Objects

Description

Methods for detailed display of varrank objects.

Usage

```
## S3 method for class 'varrank'
summary(object,digits=3, ...)
```

Arguments

object	an object of class varrank.
digits	an integer specifying the number of digits to display in the output.
...	additional arguments passed to summary.

Details

digits gives the number of digits that will be displayed in the output. This method differs of [print.varrank](#) by the amount of info displayed.

Author(s)

Gilles Kratzer

Examples

```
library(mlbench)
data(PimaIndiansDiabetes)

##forward search for all variables
varrank.output <- varrank(data.df = PimaIndiansDiabetes,
  method = "peng",
  variable.important = "diabetes",
  discretization.method = "sturges",
  algorithm = "forward", scheme = "mid")

##Print summary output of varrank object

summary(varrank.output)
```

varrank	<i>Heuristics Tools Based on Mutual Information for Variable Ranking and Feature Selection</i>
---------	--

Description

This function heuristically estimates the variables ranks based on mutual information with multiple model and multiple search schemes.

Usage

```
varrank(data.df = NULL, variable.important = NULL, method =
  c("battiti", "kwak", "peng", "estevez"), algorithm =
  c("forward", "backward"),
  scheme = c("mid", "miq"),
  discretization.method = NULL, ratio = NULL, n.var =
  NULL, verbose = TRUE)
```

Arguments

data.df	a named data frame with either numeric or factor variables.
variable.important	a list of variables that is the target variables.
method	the method to be used. See ‘Details’. Default is "estevez".
algorithm	the algorithm scheme to be used. Default is "forward".
scheme	the scheme search to be used. "mid" and "miq" stands for the Mutual Information Difference and Quotient schemes, respectively. Those are two ways to combine the relevance and redundancy. They are the two most used mRMRe schemes

discretization.method	a character vector giving the discretization method to use. See discretization .
ratio	parameter to be used in "battiti" and "kwak".
n.var	number of variables to be returned
verbose	logical. Should a progress bar be plotted? As the search scheme.

Details

By default varrank performs a variable ranking based on forward search algorithm using mutual information. The scoring is based on one of the four models implemented. The input dataset can be discrete, continuous or mixed variables. The target variables can be a set. The filter approach based on mutual information is the Minimum Redundancy Maximum Relevance (mRMRe) algorithm. A general formulation of the ensemble of mRMRe techniques is, given a set of features F , a subset of important features C , a candidate feature f_i and possibly some already selected features f_s in S . The local score function for a mid scheme (Mutual Information Difference) is expressed as:

$$g_i(A, C, S, f_i) = MI(f_i; C) - \sum_{f_s} A(f_i, f_s, C) MI(f_i; f_s)$$

Depending of the value method, the value of A and B will be set accordingly to:

battiti defined $A=B$, where B is a user defined parameter (called ratio). Battiti (1994).

kwak defined $A = B MI(f_s; C) / H(f_s)$, where B a user defined parameter (called ratio). Kwak et al. (2002).

peng defined $A=1/|S|$. Peng et al. (2005).

estevez defined $A = 1/|S| \min(H(f_i), H(f_s))$. Estévez et al. (2009).

The search algorithm implemented are a forward search i.e. start with an empty set S and fill in. The the returned list is ranked by decreasing order of relevance. A backward search which start with a full set i.e. all variables except `variable.importance` are considered as selected and the returned list is in increasing order of relevance based solely on mutual information estimation. Thus a backward search will only check for relevance and not for redundancy. `n.var` is optional if it is not provided, all candidate variables will be ranked.

Value

A list with an entry for the variables ranked and another entry for the score for each ranked variables.

Author(s)

Gilles Kratzer

References

Kratzer, G. Furrer, R. "varrank: an R package for variable ranking based on mutual information with applications to system epidemiology"

Examples

```
library(mlbench)
data(PimaIndiansDiabetes)

##forward search for all variables
varrank(data.df = PimaIndiansDiabetes,
  method = "estevez",
  variable.important = "diabetes",
  discretization.method = "sturges",
  algorithm = "forward", scheme = "mid")

##forward search for 3 variables
varrank(data.df = PimaIndiansDiabetes,
  method = "estevez",
  variable.important = "diabetes",
  discretization.method = "sturges",
  algorithm = "forward",
  scheme = "mid",
  n.var=3)

##backward search for all variables
varrank(data.df = PimaIndiansDiabetes,
  method = "peng",
  variable.important = "diabetes",
  discretization.method = "sturges",
  algorithm = "backward",
  scheme = "mid",
  n.var=NULL)
```

Index

discretization, [2](#), [5](#), [11](#)

entropy.data, [4](#), [5](#)

mi.data, [5](#)

plot.varrank, [6](#)

print.varrank, [8](#), [9](#)

summary, [8](#)

summary.varrank, [9](#)

varrank, [10](#)