Package 'corkscrew'

October 12, 2022

Type Package

Title Preprocessor for Data Modeling

Version 1.1

Date 2015-10-30

Author Navin Loganathan [aut],
Mohan Manivannan [aut],
Santhosh Sasanapuri [aut, cre],
LatentView Analytics [ctb]

Maintainer Santhosh Sasanapuri <santhosh458@gmail.com>

Depends R (>= 3.0.1), ggplot2, gplots, RColorBrewer, igraph, stats, grDevices

Suggests datasets

Description Includes binning categorical variables into lesser number of categories based on t-test, converting categorical variables into continuous features
using the mean of the response variable for the respective categories, understanding the relationship between the response variable and predictor variables
using data transformations.

NeedsCompilation no

Repository CRAN

Date/Publication 2015-10-31 00:56:46

R topics documented:

corkscrew-package *Preprocessor for Data Modeling*

Description

Includes binning categorical variables into lesser number of categories based on t-test, converting categorical variables into continuous features using the mean of the response variable for the respective categories, understanding the relationship between the response variable and predictor variables using data transformations.

Details

Package:	corkscrew
Type:	Package
Version:	1.1
Date:	2015-10-30
Depends:	R (>= 3.0.1), ggplot2, gplots, RColorBrewer, igraph, stats, grDevices
License:	GPL (version 2 or newer)

Author(s)

Navin Loganathan, Mohan Manivannan, Santhosh Sasanapuri, LatentView Analytics

Examples

```
# using transformation
data(airquality)
transformation(names(airquality)[2:4],"Ozone",airquality)

# using ctoc
data(ChickWeight)
# Converting the "Chick" variable into factor from ord.factor for demonstration purposes.
ChickWeight$Chick <- as.factor(as.numeric(ChickWeight$Chick))
# Returns a dataframe with two added columns for "Chick" and "Diet"
head(ctoc(y = "weight", x = c("Chick","Diet"), data = ChickWeight, min.obs = 12))

# using tbin
train = as.data.frame(cbind(runif(1000, 10, 1000),sample(1:40, 1000,TRUE)))
colnames(train) = c("response","state")
train$state = as.factor(train$state)
train.output = tbin(dv = "response",idv = c("state"),train,25,TRUE)
```

`apply.ctoc` *Applying Categorical to Continuous conversion to a new dataframe*

Description

Extrapolating the categorical to continuous conversion that is calculated from one dataframe to another dataframe.

Usage

```
apply.ctoc(y, x, data, newdata, min.obs)
```

Arguments

`y`	Response variable (categorical or continuous).
`x`	Predictor variables in the dataframe which are categorical and need to be converted into continuous.
`data`	Name of the dataframe from which the values of the categories have to be calculated.
`newdata`	Name of the dataframe to which the values of the categories have to be applied.
`min.obs`	The minimum number of observations within a category in a categorical variable to get converted into a continuous feature. All the categories which have observations less than the min.obs will form a different category.

Details

This function is only for categorical variables. The min.obs refers to the minimum number of observations in the "data".

Value

Returns a dataframe with converted features without replacing the original ones.

Author(s)

Santhosh Sasanapuri

See Also

`ctoc`, `tbin`, `apply.tbin`.

Examples

```
data(ChickWeight)
set.seed(2)
sample_ex <- sample(nrow(ChickWeight), size = 289, replace = FALSE, prob = NULL)
train <- ChickWeight[sample_ex,]
test <- ChickWeight[-sample_ex,colnames(ChickWeight) != "weight"]
# Returns the test dataframe with an added column "Diet_cont" by extrapolating it from train
head(apply.ctoc(y = "weight", "Diet", data = train, newdata = test, min.obs = 60))
```

`apply.tbin` *Extrapolate t-test based binning to a new data*

Description

Extrapolates the binning of categorical variables to the new datasets.

Usage

```
apply.tbin(idv, train.output, test)
```

Arguments

`idv`	Predictor variables in the dataframe which are categorical and need to be binned.
`train.output`	The output object of the tbin function.
`test`	A new data set on which binning has to be extrapolated.

Details

This function performs binning on the new dataset based on the output object from the tbin function.

Value

Returns a dataframe which contains the extrapolated variables of the output object from tbin function appended to the new dataset.

Warning

New level error is thrown if the new dataset contains new levels other than what is present in the old dataset.

Author(s)

Mohan Manivannan

See Also

`tbin`, `ctoc`, `apply.ctoc`.

Examples

```
train = as.data.frame(cbind(runif(1000, 10, 1000),sample(1:40, 1000,TRUE)))
colnames(train) = c("response","state")
train$state = as.factor(train$state)
train.output = tbin(dv = "response",idv = c("state"),train,25,TRUE)

# extrapolating the tbin function to a new dataset using apply.tbin
test = as.data.frame(sample(1:40, 100,TRUE))
colnames(test) = c("state")
test$state = as.factor(test$state)
test.output = apply.tbin(idv = c("state"), train.output, test)
```

`ctoc` *Categorical variables into Continuous features*

Description

Converting categorical variables into continuous features using the mean of the response variable for the respective categories without using the index record.

Usage

```
ctoc(y, x, data, min.obs)
```

Arguments

`y`	Response variable (categorical or continuous).
`x`	Predictor variables in the dataframe which are categorical and need to be converted into continuous.
`data`	Name of the dataframe.
`min.obs`	The minimum number of observations within a category in a categorical variable to get converted into a continuous feature. All the categories which have observations less than the min.obs will form a different category.

Details

This function is only for categorical variables.

Value

Returns a dataframe with converted features without replacing the original ones.

Author(s)

Santhosh Sasanapuri

See Also

apply.ctoc, tbin, apply.tbin.

Examples

```
data(ChickWeight)
# Converting the "Chick" variable into factor from ord.factor for demonstration purposes.
ChickWeight$Chick <- as.factor(as.numeric(ChickWeight$Chick))
# Returns a dataframe with two added columns for "Chick" and "Diet"
head(ctoc(y = "weight", x = c("Chick","Diet"), data = ChickWeight, min.obs = 12))
```

---

tbin *t-test based binning*

Description

Bins the different levels of a categorical variable based on the similarity of the response.

Usage

```
tbin(dv, idv, train, min.obs, plot.bin = c(TRUE, FALSE), method = c(1, 2, 3))
```

Arguments

dv	The response variable and it must be continuous.
idv	Predictor variables in the dataframe which are categorical and need to be binned.
train	Name of the dataframe.
min.obs	All the levels with the count of observations less than min.obs are binned into one category, by default the min.obs is set at 30.
plot.bin	If TRUE, then a heat map comparing the binned levels and original levels is plotted, by default the plot.bin is set as FALSE.
method	Three types of community detection is used for binning the levels, the methods are 1.Fastgreedy, 2.Walktrap, 3.edge.betweenness choose the method that suits the needs, by default the method is set at 1.

Details

The levels of a categorical variable are compared with each other and those levels which are having same response levels are binned together as a category.Before comparing the levels, the levels which has less than the minimum observation cut-off are binned to form a small category. Every pair of levels are compared using either a parametric or non-parametric test depending on the normality of the response data. A pairwise comparison matrix is created for each level of a categorical variables which is further processed to form a graph. Then the levels which should be combined together are identified using a community detection algorithm, a community is a collection of levels which have statistically same response level. The newly created variables in the new data set are created by extrapolating the values from the original data set(training).

Value

The tbin output contains the newly created binned categorical variables appended to the original data(training) set. The names of the new variables are sufficed with "_cat" to their corresponding original variables.

Note

The minimum observation cut-off is set to 30, so that the statistical significance of the parametric and non-parametric tests are applicable.

Author(s)

Mohan Manivannan

See Also

apply.tbin, ctoc, apply.ctoc.

Examples

```
train = as.data.frame(cbind(runif(1000, 10, 1000),sample(1:40, 1000,TRUE)))
colnames(train) = c("response","state")
train$state = as.factor(train$state)
train.output = tbin(dv = "response",idv = c("state"),train,25,TRUE)
```

---

transformation *Relationship between the response variables and predictors*

Description

Transformation is used to study the relationship between the two variables. The relationships that are studied include linear, power, log and arctangent.

Usage

```
transformation(x, y, data)
```

Arguments

x	Predictor variables in the dataframe that are to be transformed.
y	Response variable in the dataframe.
data	Name of the dataframe.

Details

Applies only when both the response variable and the predictor variables are continuous.

Value

Returns a list

[[1]]	Summary of the predictor variables' best transformation, correlation and influence studied against the y variable
[[2]]	Summary of the predictor variables' correlation and influence for all the transformations studied against the y variable

Author(s)

Navin Loganathan

Examples

```
data(airquality)
transformation(names(airquality)[2:4],"Ozone",airquality)
```

Index