Package 'Information'

October 12, 2022

Type Package

Title Data Exploration with Information Theory (Weight-of-Evidence and Information Value)

Version 0.0.9

Date 2016-04-08

Description Performs exploratory data analysis and variable screening for binary classification models using weight-of-evidence (WOE) and information value (IV). In order to make the package as efficient as possible, aggregations are done in data.table and creation of WOE vectors can be distributed across multiple cores. The package also supports exploration for uplift models (NWOE and NIV).

Depends R (>= 3.1.2)

License GPL (>= 3)

Imports data.table, ggplot2, grid, plyr, utils, iterators, doParallel, parallel, foreach

LazyData true

VignetteBuilder knitr

Suggests knitr, reshape2, ClustOfVar, rmarkdown

RoxygenNote 5.0.1

NeedsCompilation no

Author Larsen Kim [aut, cre]

Maintainer Larsen Kim <kblarsen4@gmail.com>

Repository CRAN

Date/Publication 2016-04-09 00:24:08

R topics documented:

`Aggregate` *(helper function)Aggregate data for WOE/NWOE calculations*

Description

`Aggregate` returns aggregated data for the WOE and NWOE functions

Usage

```
Aggregate(data, x, y, breaks, trt)
```

Arguments

`data`	input data frame
`x`	variable to be aggregated
`y`	dependent variable
`breaks`	breaks for binning
`trt`	binary treatment variable (for net lift only)

`CheckInputs` *(helper function) Check user inputs and convert factors to characters. Provide readable reasons if errors are found.*

Description

`CheckInputs` Checks user inputs and converts factors to characters. Returns the altered dataframes as a list. Provides readable reasons if errors are found.

Usage

```
CheckInputs(train, valid, trt, y)
```

Arguments

`train`	training data.
`valid`	validation dataset (default is NULL)
`trt`	treatment indicator
`y`	dependent variable

`create_infotables` *Create WOE/NWOE tables and rank variables by IV/NIV*

Description

`create_infotables` returns WOE or NWOE tables (as data.frames), and a data.frame with IV or NIV values for all predictive variables.

Usage

```
create_infotables(data = NULL, valid = NULL, y = NULL, bins = 10,
  trt = NULL, ncore = NULL, parallel = TRUE)
```

Arguments

`data`	input data.frame for analysis (this is typically your training dataset).
`valid`	validation data.frame (default is NULL). Must have the same variables as the training dataset.
`y`	dependent variable.
`bins`	number of bins (default is 10).
`trt`	binary treatment variable for uplift analysis (Default is NUL).
`ncore`	number of cores used. Default is to use available cores - 1.
`parallel`	set to TRUE for parallel processing. Number of cores is determined by the ncore parameter.

Examples

```
##------------------------------------------------------------
## WOE analysis, no cross validation
##------------------------------------------------------------
library(Information)
data(train, package="Information")
train <- subset(train, TREATMENT==1)
IV <- Information::create_infotables(data=train, y="PURCHASE", parallel=FALSE)
print(head(IV$Summary), row.names=FALSE)
print(IV$Tables$N_OPEN_REV_ACTS, row.names=FALSE)
closeAllConnections()
```

`Information` *Data exploration with information theory (weight-of-evidence and information value)*

Description

The information package performs exploratory data analysis and variable screening for binary classification models using information theory (WOE and IV).

The package also supports exploratory analysis and variable screening for uplift models (NWOE and NIV).

Note that the only functions you will need to use are create_infotables() and plot_infotables():

- create_infotables() creates WOE or NWOE tables and outputs a variable-strength summary data.frame (IV or NIV)

- plot_infotables() creates WOE or NWOE bar charts for one or more variables

Details

Given a data.frame with a set of predictive variables and a binary response variable, create_infotables() will cycle through all variables and create NWOE or WOE tables. It will also rank all variables by their respective IV or NIV values and return the results in a data.frame.

The package needs minimal inputs. You do not have to explicitly specify which variables to evaluate or provide bins: create_infotables() will process all variables in the dataset and generate appropriate bins for WOE/NWOE analysis.

If requested, calculations can be distributed across multiple cores for better performance.

Note that NWOE analysis is only for uplift models. Thus, for NWOE analysis, you must have a "treatment" and a "control" group in your dataset. The treatment and control groups should identified by a binary indicator variable (1/0).

For regular WOE analysis, on the other hand, all you need is a binary response variable (dependent variable).

You can cross validate your IV or NIV values by supplying a validation dataset. This will produce penalized IV/NIV values.

#' To learn more about the Information package, start with the vignette: `browseVignettes(package = "Information")`

Author(s)

Kim Larsen (kblarsen4 at gmail.com)

Examples

```
##------------------------------------------------------------
## WOE analysis, no validation
##------------------------------------------------------------
library(Information)
```

```
data(train, package="Information")
train <- subset(train, TREATMENT==1)
IV <- Information::create_infotables(data=train, y="PURCHASE", parallel=FALSE)

print(head(IV$Summary), row.names=FALSE)
print(IV$Tables$N_OPEN_REV_ACTS, row.names=FALSE)

# Plotting a single variable
Information::plot_infotables(IV, "N_OPEN_REV_ACTS")

# Plotting multiple variables
Information::plot_infotables(IV, IV$Summary$Variable[1:4], same_scale=TRUE)

# If the goal is to plot multiple variables individually, as opposed to a comparison-grid, we can
# loop through the variable names and create individual plots
## Not run:
names <- names(IV$Tables)
plots <- list()
for (i in 1:length(names)){
  plots[[i]] <- plot_infotables(IV, names[i])
}
# Showing the top 18 variables
plots[1:18]

## End(Not run)

# We can speed up create_infotables() by setting parallel=TRUE (default setting)
# If we leave ncore as the default, ncore is set to available clusters - 1
## Not run:
train <- subset(train, TREATMENT==1)
IV <- Information::create_infotables(data=train, y="PURCHASE")

## End(Not run)
closeAllConnections()
```

`is.binary` *(helper function) Calculate cross validation penalty*

Description

`is.binary` returns TRUE if a variable is binary, and FALSE otherwise

Usage

```
is.binary(x)
```

Arguments

`x`	a numeric vector

MultiPlot *(helper function) Plot mutiple WOE vectors on one page*

Description

`MultiPlot` creates a multiple WOE or NWOE bar charts on the same page for a specified vector of variables.

Usage

```
MultiPlot(information_object, variables, same_scales = FALSE)
```

Arguments

`information_object`	object from the information package
`variables`	vector of variables that you want to compare
`same_scales`	set to TRUE if all plots should have the same limits on the y-axes (default is FALSE)

NWOE *Create WOE table (helper function)*

Description

`WOE` returns NWOE tables from a data.frame prepared by Information::Aggregate(). This is only for net lift models.

Usage

```
NWOE(t, x)
```

Arguments

`t`	table prepared by the Aggregate function
`x`	variable

`penalty` *(helper function) Calculate cross validation penalty*

Description

`penalty` returns the weighted cross validation penalty.

Usage

```
penalty(t, v, d_net_lift)
```

Arguments

`t`	training data (data.frame)
`v`	valdation data (data.frame)
`d_net_lift`	is it a net lift model? (1=yes, 0=no)

`plot_infotables` *Create bar charts for WOE or NWOE vectors*

Description

`plot_infotable` creates WOE or NWOE bar charts for one or more variables. For multi-variable plots, bar charts will be displayed in a grid for comparison.

Usage

```
plot_infotables(information_object = NULL, variables = NULL,
  same_scales = FALSE, show_values = FALSE)
```

Arguments

`information_object`	object generated by the information package.
`variables`	vector of one more variables. For multi-variable plots, bar charts will be displayed in a grid.
`same_scales`	if set to TRUE, all plots will have the same limits on the y-axes (default is FALSE).
`show_values`	if set to TRUE, values will be displayed on the bar chart (default is FALSE).

Examples

```
##------------------------------------------------------------
## WOE plots
##------------------------------------------------------------
library(Information)
data(train, package="Information")
train <- subset(train, TREATMENT==1)
IV <- Information::create_infotables(data=train, y="PURCHASE", parallel=FALSE)

# Plotting a single variable
Information::plot_infotables(IV, "N_OPEN_REV_ACTS")

# Plotting multiple variables in a grid for comparison
Information::plot_infotables(IV, IV$Summary$Variable[1:4], same_scale=TRUE)

# If the goal is to plot multiple variables individually, as opposed to a comparison-grid, we can
# loop through the variable names and create individual plots
## Not run:
names <- names(IV$Tables)
plots <- list()
for (i in 1:length(names)){
  plots[[i]] <- plot_infotables(IV, names[i])
}
# Showing the top 18 variables
plots[1:18]

## End(Not run)

# We can speed up the creation of the information tables by invoking the parallel option (default)
# If we leave ncore as the default, create_infotables() will set ncore to available clusters - 1
## Not run:
train <- subset(train, TREATMENT==1)
IV <- Information::create_infotables(data=train, y="PURCHASE")

## End(Not run)
closeAllConnections()
```

`SinglePlot` *(helper function) Plot a WOE or NWOE vector*

Description

`SinglePlot` creates a bar chart of the WOE or NWOE pattern for a specified variable

Usage

```
SinglePlot(information_object, variable, show_values = FALSE)
```

Arguments

`information_object`	object from the information package
`variable`	variable for which we want to see the WOE pattern
`show_values`	if set to TRUE, values will be displayed on the bar chart (default is FALSE)

`train` *Training dataset*

Description

The data is from a historical marketing campaign. It contains 68 predictive variables and 4,972 records. In addition, it contains a treatment indicator and a purchase indicator.

Usage

```
train
```

Format

A data frame with 10000 rows and 70 variables:

- TREATMENT: equals 1 if the person received the marketing offer, and 0 if the person was in the control group
- PURCHASE: equals 1 if the person accepted the offer, and 0 otherwise
- UNIQUE_ID: unique identifier
- AGE: age of the person
- D_REGION_X: 1 if the person lives in region X, 0 otherwise (3 regions: A, B, C)
- Other variables are from credit bureau data (e.g., N_OPEN_REV_ACTS = number of open revolving accounts)

`valid` *Validation dataset*

Description

The data is from a historical marketing campaign. It contains 68 predictive variables and 5,060 records. In addition, it contains a treatment indicator and a purchase indicator.

Usage

```
valid
```

Format

A data frame with 10000 rows and 70 variables:

- TREATMENT: equals 1 if the person received the marketing offer, and 0 if the person was in the control group
- PURCHASE: equals 1 if the person accepted the offer, and 0 otherwise
- UNIQUE_ID: unique identifier
- AGE: age of the person
- D_REGION_X: 1 if the person lives in region X, 0 otherwise (3 regions: A, B, C)
- Other variables are from credit bureau data (e.g., N_OPEN_REV_ACTS = number of open revolving accounts)

`WOE` *Create WOE tables from aggregated data (helper function)*

Description

`WOE` returns WOE tables from data frames prepared by Aggregate()

Usage

```
WOE(t, x)
```

Arguments

`t`	table prepared by the Aggregate function
`x`	variable

Index