Package ‘Rprofet’

January 20, 2025

Type Package

Title WOE Transformation and Scorecard Builder

Version 3.1.1

Maintainer Thomas Brandenburger <thomas.brandenburger@sdstate.edu>

Description
Performs all steps in the credit scoring process. This package allows the user to follow all the necessary steps for building an effective scorecard. It provides the user functions for coarse binning of variables, Weights of Evidence (WOE) transformation, variable clustering, custom binning, visualization, and scaling of logistic regression coefficients. The results will generate a scorecard that can be used as an effective credit scoring tool to evaluate risk. For complete details on the credit scoring process, see Siddiqi (2005, ISBN:047175451X).

License GPL-3

Imports binr, ClustOfVar, ggplot2, gridExtra, plyr, sqldf, stringr,
dplyr, corrplot, kableExtra, purrr, reshape2, ROCit

Suggests ISLR, knitr, MASS, rmarkdown

Encoding UTF-8

RoxygenNote 7.3.1

NeedsCompilation no

Author Thomas Brandenburger [cre],
Eric Stratman [aut],
Krystal Wang [aut]

VignetteBuilder knitr

Repository CRAN

Date/Publication 2024-02-27 17:10:02 UTC

Contents

`BinProfet` *Coarse Binning Variable(s)*

Description

Function that bins selected variable(s) and returns a dataframe with binned values. Uses greedy binning algorithm to perform coarse binning of selected variable(s).

Usage

```
BinProfet(
  data,
  id,
  target,
  varcol,
  min.cat = 4,
  num.bins = 10,
  min.pts.bin = 25,
  bracket = "left",
  special.values = NULL,
  sort_id = FALSE
)
```

Arguments

`data`	Dataframe of that contains ID, binary target and variables to be binned.
`id`	ID variable. See 'Details'.
`target`	The binary target/response variable for WOE. See 'Details'.
`varcol`	Vector of variables to be binned.
`min.cat`	Minimum number of bins.
`num.bins`	Target number of bins. Overridden by the number of levels if varcol is factor.
`min.pts.bin`	Minimum number of observations in a bin.
`bracket`	Indicating if the intervals should be closed on the right or left. Options include left and right.
`special.values`	A vector of values that should have their own bin. See 'Details'.
`sort_id`	Logical. The default is FALSE which does not sort the data by ID column. If TRUE, then data is sorted increasingly by ID column.

Value

A dataframe containing the ID, target, and binned variable(s) with corresponding binned values.

Examples

```
mydata <- ISLR::Default
head(mydata)
mydata$ID <- seq(1:nrow(mydata)) ## make an ID variable
mydata$default <- ifelse(mydata$default=="Yes", 1, 0) ## target coded with 1, 0

## bin balance and income
binned1 <- BinProfet(mydata, id="ID", target="default",
                     varcol = c("balance",  "income"), num.bins = 5)
head(binned1)

## bin categorical variable-------------------
binned2 <- BinProfet(mydata, id="ID", target="default",
                     varcol = "student", num.bins = 5)
head(binned2)
summary(binned2$student_Bins) ## num.bins overriden
```

ScorecardProfet *Scorecard Builder*

Description

Function that fits a logistic regression models and scores points for each bin and calculates observations' total score.

Usage

```
ScorecardProfet(
  object,
  id,
  target,
  GLModel,
  PDO = 100,
  BaseOdds = 10,
  BasePts = 1000,
  reverse = FALSE
)
```

Arguments

`object` A WOEProfet object or a Var_select object that containing dataframes with binned and WOE values.

`id` ID variable.

`target` A binary target variable.

`GLModel` A generalized linear model, glm object.

`PDO` Points to Double Odds.

`BaseOdds` Base Odds.

`BasePts` Base Points.

`reverse` Logical. If FALSE, higher points corresponds to a lower probability of being target.

Value

A scorecard dataframe.

Examples

```
mydata <- ISLR::Default
mydata$ID = seq(1:nrow(mydata)) ## make the ID variable
mydata$default<-ifelse(mydata$default==”Yes”,1,0) ## Creating numeric binary target variable

binned <- BinProfet(mydata, id= ”ID”, target= ”default”, num.bins = 5) ## Binning variables

WOE_dat <- WOEProfet(binned, ”ID”,”default”, 3:5) ## WOE transformation of bins

md <- glm(default ~ student_WOE+balance_WOE+income_WOE, data=WOE_dat$WOE, family=”binomial”)
summary(md)

Score_dat <- ScorecardProfet(object=WOE_dat, id=”ID”, target=”default”, GLModel=md,
                             PDO = 50, BaseOdds = 10, BasePts = 1000, reverse = FALSE)

Score_dat ## Less points means more likely to default
```

ScoreDataProfet *Score a Validation Data Set*

Description

Function that scores the validation set using the scorecard from the ScorecardProfet object created by the training set.

Usage

```
ScoreDataProfet(data, card, id, target)
```

Arguments

`data` The validation data set, which should be binned in the same way as the scorecard in the card argument.

`card` A ScorecardProfet object. The object should be created by using the training set split from the same dataframe as the validation set.

`id` ID variable.

`target` A binary target variable.

Value

A dataframe of scored validation set.

Examples

```
mydata <- ISLR::Default
mydata$ID = seq(1:nrow(mydata)) ## make the ID variable
mydata$default<-ifelse(mydata$default=="Yes",1,0) ## Creating numeric binary target variable

binned <- BinProfet(mydata, id= "ID", target= "default", num.bins = 5) ## Binning variables

WOE_dat <- WOEProfet(binned, "ID","default", 3:5) ## WOE transformation of bins

md <- glm(default ~ student_WOE+balance_WOE+income_WOE, data=WOE_dat$WOE, family="binomial")
summary(md)

Score_card <- ScorecardProfet(object=WOE_dat, id="ID", target="default", GLModel=md,
                               PDO = 50, BaseOdds = 10, BasePts = 1000, reverse = FALSE)
Score_card ## scorecard

## Scoring the data
# variable names needs to be the same as the Attributes on scorecard
colnames(binned)
colnames(binned)[3:5] <- c("student", "balance", "income") #change the variable name
Score_dat = ScoreDataProfet(data=binned, card=Score_card, id="ID", target="default") #scoring data
head(Score_dat)
```

Var_select *Select variables or filter variables by information value*

Description

Function that selects specified variables or filters variables based on information value for WOEProfet object or WOE_StepAIC object.

Usage

```
Var_select(object, id, target, varcol, IVfilter)
```

Arguments

object	WOEProfet object.
id	ID variable.
target	A binary target variable.
varcol	Vector of variables to be selected or removed. Character or numeric.
IVfilter	Threshold of variables' Information Value.

Value

A list with the following components.

Bin	Dataframe with ID, Target, and selected binned variables.
WOE	Dataframe with ID, Target, and WOE values for selected binned variables.
IV	Information value of the selected binned variables.
vars	List containing a dataframe for each variable that consists of Bin, WOE, Target Rate, and observation count.

Examples

```
mydata <- ISLR::Default
mydata$ID = seq(1:nrow(mydata)) ## make the ID variable
mydata$default<-ifelse(mydata$default=="Yes",1,0) ## Creating numeric binary target variable
binned <- BinProfet(mydata, id= "ID", target= "default", num.bins = 5) ## Binning variables
WOE_dat <- WOEProfet(binned, "ID", "default", 3:5) ## WOEProfet object
WOE_dat$IV #IV item, the row index will be used for filtering variables
# To remove the income variable from the WOEProfet object
## Select the first two variables based on the IV item
subWOE1 <- Var_select(WOE_dat, id= "ID", target= "default", varcol= c(1,2))
##  Or remove the third variable based on the IV item
subWOE2 <- Var_select(WOE_dat, id= "ID", target= "default", varcol= -3)
## Filter the WOEProfet object based on variables' information values
subWOE3 <- Var_select(WOE_dat, id= "ID", target= "default", IVfilter = 0.05)
```

WOEclust_hclust *Hierarchical Variable Clustering*

Description

Function that implements hierarchical clustering on the variables to be used as a form of variable selection.

Usage

```
WOEclust_hclust(object, id, target, num_clusts, method = "ward.D")
```

Arguments

object	A WOEProfet object containing dataframes with binned and WOE values.
id	ID variable.
target	A binary target variable.
num_clusts	Number of desired clusters.
method	Clustering method to be used. This should be one of "ward.D", "ward.D2", "single", "average", "mcquitty", "median",or "centroid".

Value

A dataframe indicating the assigned clusters for the predictor variables.

Examples

```
mydata <- ISLR::Default
mydata$ID = seq(1:nrow(mydata)) ## make the ID variable
mydata$default<-ifelse(mydata$default=="Yes",1,0) ## Creating numeric binary target variable
## create two new variables from bivariate normal
sigma <- matrix(c(45000,-3000,-3000, 55000), nrow = 2)
set.seed(10)
newvars <- MASS::mvrnorm(nrow(mydata),
                         mu=c(1000,200), Sigma=sigma)
mydata$newvar1 <- newvars[,1]
mydata$newvar2 <- newvars[,2]
binned <- BinProfet(mydata, id= "ID", target= "default", num.bins = 5) ## Binning variables
WOE_dat <- WOEProfet(binned, "ID","default")
## Cluster variables by WOEClust_hclust
clusters <- WOEclust_hclust(WOE_dat, id="ID", target="default", num_clusts=3)
clusters
```

WOEclust_kmeans *Kmeans Variable Clustering*

Description

Function that implements kmeans variable clusteting to be used as a form of variable selection.

Usage

```
WOEclust_kmeans(object, id, target, num_clusts)
```

Arguments

object	A WOEProfet object containing dataframes with binned and WOE values.
id	ID variable.
target	A binary target variable.
num_clusts	Number of desired clusters.

Value

A dataframe with the name of all the variables to be clustered, the corresponding cluster and the information value for each variable.

Examples

```
mydata <- ISLR::Default
mydata$ID = seq(1:nrow(mydata)) ## make the ID variable
mydata$default<-ifelse(mydata$default=="Yes",1,0) ## Creating numeric binary target variable

## create two new variables from bivariate normal
sigma <- matrix(c(45000,-3000,-3000, 55000), nrow = 2)
set.seed(10)
newvars <- MASS::mvrnorm(nrow(mydata),
                         mu=c(1000,200), Sigma=sigma)

mydata$newvar1 <- newvars[,1]
mydata$newvar2 <- newvars[,2]

binned <- BinProfet(mydata, id= "ID", target= "default", num.bins = 5) ## Binning variables

WOE_dat <- WOEProfet(binned, "ID","default")

## Cluster variables by WOEClust_kmeans
clusters <- WOEclust_kmeans(WOE_dat, id="ID", target="default", num_clusts=3)
clusters
```

WOEplotter *Visualizing WOE and Target Rates*

Description

Function generating three plots: WOE value for each bin, target rate for each bin, and the frequency for each bin.

Usage

```
WOEplotter(data, target, var, color = "#0066CC")
```

Arguments

data	Dataframe containing binned values and a binary target variable.
target	A numeric binary target variable.
var	The desired WOE binned attribute to visualize.
color	A hexadecimal value representing a specific color.

Details

A list of the hexadecimal colors can be found at this link http://www.sthda.com/sthda/RDoc/images/hextable.gif

Examples

```
mydata <- ISLR::Default
mydata$ID = seq(1:nrow(mydata)) ## make the ID variable
mydata$default<-ifelse(mydata$default=="Yes",1,0) ## Creating numeric binary target variable
binned <- BinProfet(mydata, id= "ID", target= "default", num.bins = 5) ## Binning variables
WOEplotter(binned, target= "default", var= "income_Bins")
##--Changing Colors------------------------------
WOEplotter(binned, target= "default", var= "income_Bins", color = "#33FF33")
```

WOEProfet *WOE Transformation*

Description

Function that calculates the WOE for each bin and the information value for each variable.

Usage

```
WOEProfet(data, id, target, varcol)
```

Arguments

data	Dataframe of binned variables.
id	ID variable.
target	A binary target variable.
varcol	Vector of variables to have WOE transformation.

Value

A list with the following components.

Bin	Dataframe with the binned variables and their WOE values.
WOE	Dataframe with the WOE values.
IV	Each attribute and their associated information values.
vars	A list containing the different WOE values for each attribute.

Examples

```
mydata <- ISLR::Default
mydata$ID = seq(1:nrow(mydata)) ## make the ID variable
mydata$default<-ifelse(mydata$default=="Yes",1,0) ## Creating numeric binary target variable
binned <- BinProfet(mydata, id= "ID", target= "default", num.bins = 5) ## Binning variables
WOE_dat <- WOEProfet(binned, "ID", "default", 3:5)
head(WOE_dat$Bin)
head(WOE_dat$WOE)
WOE_dat$IV
head(WOE_dat$vars$income)
```

WOE_customFac *Custom Binning Factor Variables*

Description

Function that bins a factor variable based on user inputted factor levels, plots the information on the new bins, and returns a list contains a dataframe of the newly binned values and id column and more items.

Usage

```
WOE_customFac(
  data,
  var,
  id,
  target,
  new_levels,
  color = "#0066CC",
  plot = FALSE
)
```

Arguments

data	Dataframe containing the target variable and desired factor variables to be binned.
var	A specific factor attribute to be binned.
id	The unique id variable in the dataframe. Must be specified.
target	A binary target variable. Must be specified.
new_levels	A vector the same length as the number of levels for the categorical variable containing the new factor levels. Must be specified.
color	A hexadecimal value representing a specific color.
plot	Logical. The default is FALSE which does not generate the plots.

Value

A list with the following components.

NewBin	Dataframe with the binned variable.
BinWOE	Dataframe with target, binned variable, and WOE values for the bins.
IV	Information value of the newly binned variable.
vars	Dataframe with binned variable, WOE values for the bins, Target Rate for each bin, and observation count for each bin.

Examples

```
mydata <- ISLR::Default
mydata$ID = seq(1:nrow(mydata)) ## make the ID variable
mydata$default <- ifelse(mydata$default=="Yes", 1, 0) ## target coded with 1, 0
## WOE_customFactor
custom1 <- WOE_customFac(data=mydata, var="student", id ="ID", target="default",
                         new_levels=c("Student : No","Student : Yes"))
head(custom1$NewBin)
head(custom1$BinWOE)
custom1$IV
custom1$vars
## --------------------------
mydata$balance_cat <- cut(mydata$balance, breaks = c(-1,400,800,1200,1600,2000,2400,2800),
                          labels = c("Very-Low","Low","Med-Low","Med",
                          "Med-High","High","Very-High"))
custom2 <- WOE_customFac(data=mydata, var="balance_cat", id ="ID", target="default",
                         new_levels=c(1,1,2,2,2,3,3))
head(custom2$NewBin)
head(custom2$BinWOE)
custom2$IV
custom2$vars
```

WOE_customNum *Custom Binning Numeric Variables*

Description

Function that bins a numeric variable based on user inputted breaks, plots the information on the new bins, and returns a list contains a dataframe of the newly binned values and id column and more items.

Usage

```
WOE_customNum(
  data,
  var,
  id,
  target,
  breaks,
  right_bracket = F,
  color = "#0066CC",
  plot = FALSE
)
```

Arguments

data	Dataframe containing the target variable and desired numeric variables to be binned.

var A specific numeric attribute to be binned. Must be specified.

id The unique id variable in the dataframe. Must be specified.

target A binary target variable. Must be specified.

breaks A vector of breakpoints for the desired bins. Must be specified.

right_bracket Logical. Specifying whether the intervals are closed on the right or the left.

color A hexadecimal value representing a specific color.

plot Logical. The default is FALSE which does not generate the plots.

Value

A list with the following components.

NewBin Dataframe with the binned variable.

BinWOE Dataframe with target, binned variable, and WOE values for the bins.

IV Information value of the newly binned variable.

vars Dataframe with binned variable, WOE values for the bins, Target Rate for each bin, and observation count for each bin.

Examples

```
mydata <- ISLR::Default
mydata$ID = seq(1:nrow(mydata)) ## make the ID variable
mydata$default <- ifelse(mydata$default=="Yes", 1, 0) ## target coded with 1, 0

WC_1 <- WOE_customNum(data= mydata, var="balance", id= "ID", target = "default",
                      breaks= seq(0,3000,1000))
head(WC_1$NewBin)
head(WC_1$BinWOE)
WC_1$IV
WC_1$vars

WC_2 <- WOE_customNum(data= mydata, var="income", id= "ID", target = "default",
                      breaks=seq(0,75000, 15000))
head(WC_2$NewBin)
head(WC_2$BinWOE)
WC_2$IV
WC_2$vars
```

Index