

**MINI PROJECT**

* **Bank Personal Loan**

**Model Report**

**By – Nikhil Rawal**



**Project Overview**

Machine learning is the scientific study of algorithms and statistical models that computer systems use in order to perform a specific task effectively without using explicit instructions, relying on patterns and inference instead. It is seen as a subset of artificial intelligence. Machine learning algorithms build a mathematical model based on sample data, known as "training data", in order to make predictions or decisions without being explicitly programmed to perform the task. Machine learning algorithms are used in a wide variety of applications, such as email filtering, and computer vision, where it is infeasible to develop an algorithm of specific instructions for performing the task. Machine learning is closely related to computational statistics, which focuses on making predictions using computers. The study of mathematical optimization delivers methods, theory and application domains to the field of machine learning. Data mining is a field of study within machine learning, and focuses on exploratory data analysis through unsupervised learning. In its application across business problems, machine learning is also referred to as predictive analytics.

The analysis is carried out in the R environment for statistical computing and

visualisation, which is an open-source dialect of the S statistical computing

language. It is free, runs on most computing platforms, and contains contributions

The objective of the Project is to target the market of Personal Loans for the Bank by the given data.

**Project Approach**

* **Data Exploration**
* **Understanding the Attribute**
* **Data Visualisation**
* **Data Partition**
* **Cart – Model building**
* **Cart – training data Model Performance**
* **Cart – Holdout**

**---------------------------------------------------------------------------------------------**

* **Data Exploration**
* Set working directory
* #setwd("F:/r data machine learning")

#getwd()

>"F:/r data machine learning"

* Read Input File
* mydata=read.csv("Bank Personal Loan Dataset.csv", header = T)
* Names of the columns

>names(mydata)

"ID" "Age..in.years."

"Experience..in.years." "Income..in.K.month."

"ZIP.Code" "Family.members"

"CCAvg" "Education"

"Mortgage" "Personal.Loan"

"Securities.Account" "CD.Account"

"Online" "CreditCard"

\*Hence, the column name ‘ID’ is just the column number, and do not have any use and explanatory power . So, we can drop it .

#mydata=mydata[,2:13]

"Age..in.years." "Experience..in.years."

"Income..in.K.month." "ZIP.Code"

"Family.members" "CCAvg"

"Education" "Mortgage"

"Personal.Loan" "Securities.Account"

"CD.Account" "Online"

"CreditCard"

#Head(mydata)

>

|  |
| --- |
| Age in years. Experience in years. Income..in.K.month. ZIP.Code |
| 25 1 49 91107 |
| 45 19 34 90089 |
| 39 15 11 94720 |
| 35 9 100 94112 |
| 35 8 45 91330 |
| 37 13 29 92121 |

|  |
| --- |
| Family members CCAvg Education Mortgage Personal.Loan |
| 4 1.6 1 0 0 |
| 3 1.5 1 0 0 |
| 1 1.0 1 0 0 |
| 1 2.7 2 0 0 |
| 4 1.0 2 0 0 |
| 4 0.4 2 155 0 |

|  |
| --- |
| Securities Account CD Account Online CreditCard |
| 1 1 0 0 0 |
| 2 1 0 0 0 |
| 3 0 0 0 0 |
| 4 0 0 0 0 |
| 5 0 0 0 1 |
| 6 0 0 1 0 |

* Dimension of data

# dim(mydata)

5000 13

* Structure of data

# str(mydata)

* 'data.frame': 5000 obs. of 13 variables:
* $ Age..in.years. : int 25 45 39 35 35 37 53 50 35 34 ...
* $ Experience..in.years.: int 1 19 15 9 8 13 27 24 10 9 ...
* $ Income..in.K.month. : int 49 34 11 100 45 29 72 22 81 180 ...
* $ ZIP.Code : int 91107 90089 94720 94112 91330 92121 91711 93943 90089 93023 ...
* $ Family.members : int 4 3 1 1 4 4 2 1 3 1 ...
* $ CCAvg : num 1.6 1.5 1 2.7 1 0.4 1.5 0.3 0.6 8.9 ...
* $ Education : int 1 1 1 2 2 2 2 3 2 3 ...
* $ Mortgage : int 0 0 0 0 0 155 0 0 104 0 ...
* $ Personal.Loan : int 0 0 0 0 0 0 0 0 0 1 ...
* $ Securities.Account : int 1 1 0 0 0 0 0 0 0 0 ...
* $ CD.Account : int 0 0 0 0 0 0 0 0 0 0 ...
* $ Online : int 0 0 0 0 0 1 1 0 1 0 ...
* $ CreditCard : int 0 0 0 0 1 0 0 1 0 0 ...
* Summary of data

#Summary(mydata)

|  |
| --- |
| Age in years Experience in years Income in K.month. |
| Min. :23.00 Min. :-3.0 Min. : 8.00 |
| 1st Qu.:35.00 1st Qu.:10.0 1st Qu.: 39.00 |
| Median :45.00 Median :20.0 Median : 64.00 |
| Mean :45.34 Mean :20.1 Mean : 73.77 |
| 3rd Qu.:55.00 3rd Qu.:30.0 3rd Qu.: 98.00 |
| Max. :67.00 Max. :43.0 Max. :224.00 |

|  |
| --- |
| ZIP.Code Family.members CCAvg Education |
| Min. : 9307 Min. :1.000 Min. : 0.000 Min. :1.000 |
| 1st Qu.:91911 1st Qu.:1.000 1st Qu.: 0.700 1st Qu.:1.000 |
| Median :93437 Median :2.000 Median : 1.500 Median :2.000 |
| Mean :93153 Mean :2.397 Mean : 1.938 Mean :1.881 |
| 3rd Qu.:94608 3rd Qu.:3.000 3rd Qu.: 2.500 3rd Qu.:3.000 |
| Max. :96651 Max. :4.000 Max. :10.000 Max. :3.000 |

|  |
| --- |
| Mortgage Personal.Loan Securities.Account CD.Account |
| Min. : 0.0 Min. :0.000 Min. :0.0000 Min. :0.0000 |
| 1st Qu.: 0.0 1st Qu.:0.000 1st Qu.:0.0000 1st Qu.:0.0000 |
| Median : 0.0 Median :0.000 Median :0.0000 Median :0.0000 |
| Mean : 56.5 Mean :0.096 Mean :0.1044 Mean :0.0604 |
| 3rd Qu.:101.0 3rd Qu.:0.000 3rd Qu.:0.0000 3rd Qu.:0.0000 |

|  |
| --- |
| Online CreditCard |
| Min. :0.0000 Min. :0.000 |
| 1st Qu.:0.0000 1st Qu.:0.000 |
| Median :1.0000 Median :0.000 |
| Mean :0.5968 Mean :0.294 |
| 3rd Qu.:1.0000 3rd Qu.:1.000 |

* Understanding the Attribute

The binary category have five variables as below:

* Personal Loan - Did this customer accept the personal loan offered in the last campaign? **This is our target variable**
* Securities Account - Does the customer have a securities account with the bank?
* CD Account - Does the customer have a certificate of deposit (CD) account with the bank?
* Online - Does the customer use internet banking facilities?
* Credit Card - Does the customer use a credit card issued by UniversalBank?

Interval variables are as below:

* Age - Age of the customer
* Experience - Years of experience
* Income - Annual income in dollars
* CCAvg - Average credit card spending
* Mortage - Value of House Mortgage

Ordinal Categorical Variables are:

* Family - Family size of the customer
* Education - education level of the customer

The nominal variable is :

* ID
* Zip Code
* **Target variable : Personal Loan**
* Table(Personal.loan)

Personal loan

0 1

4520 480

**Prop.table(table(Personal.loan))**

0 1

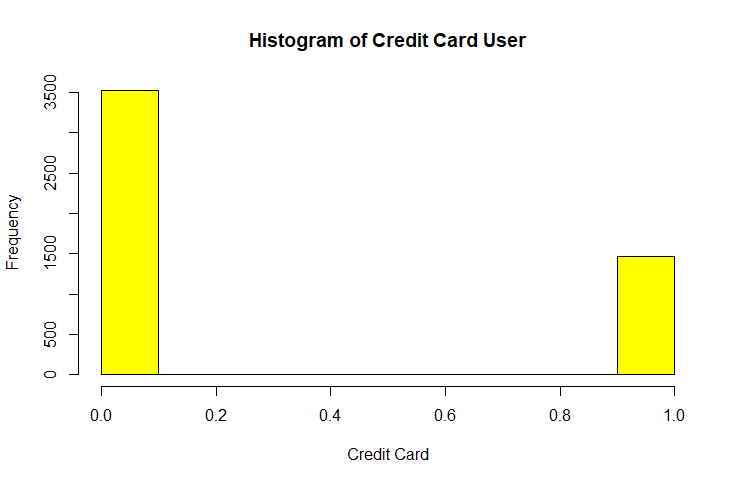
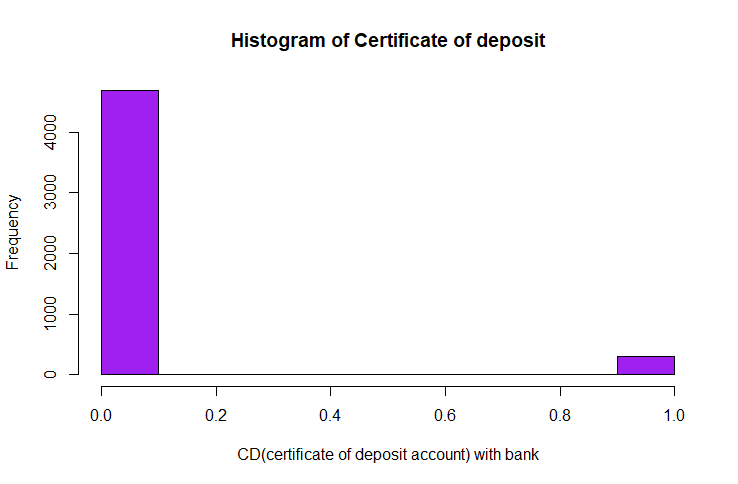
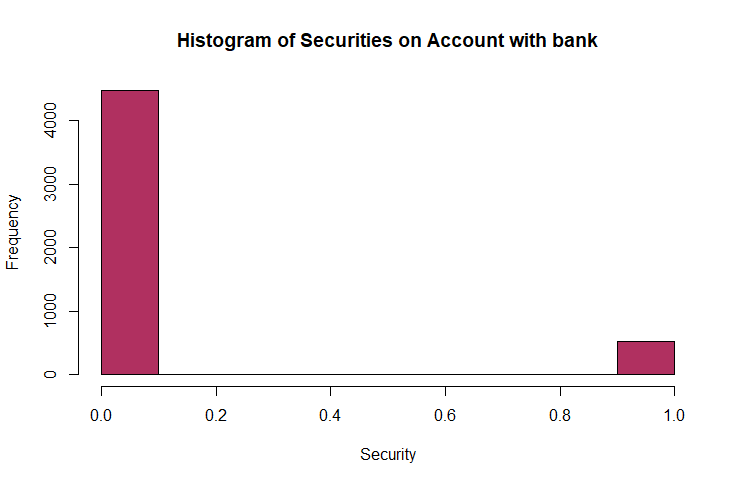
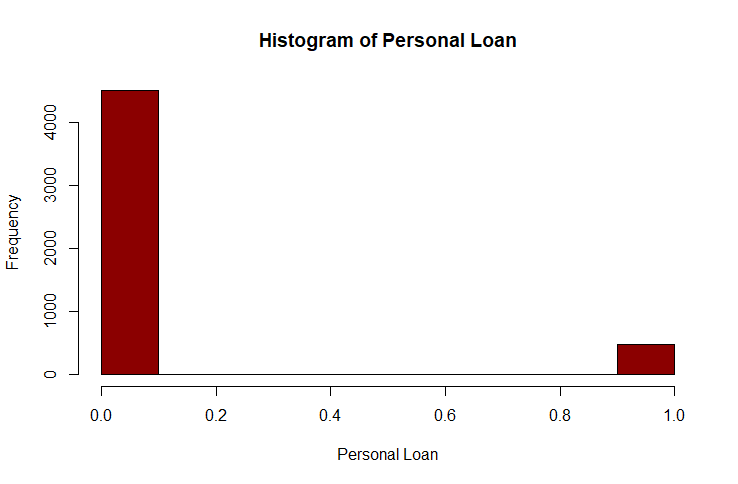
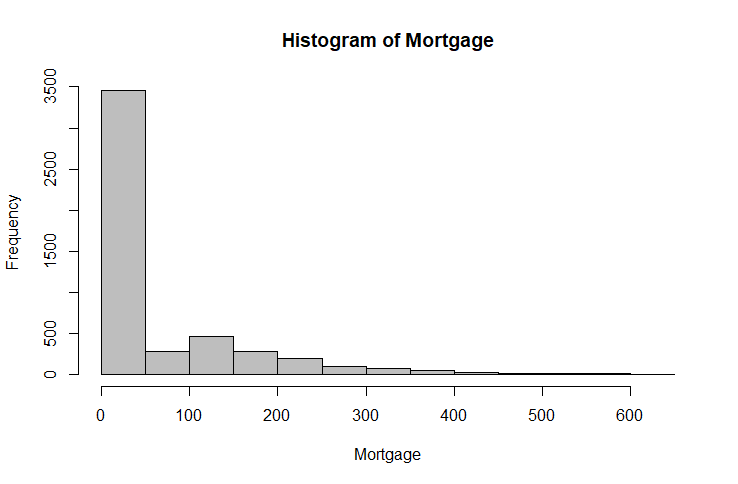
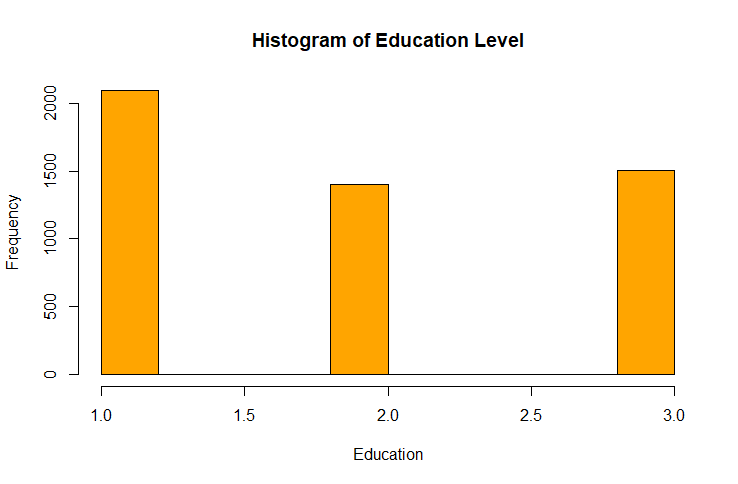
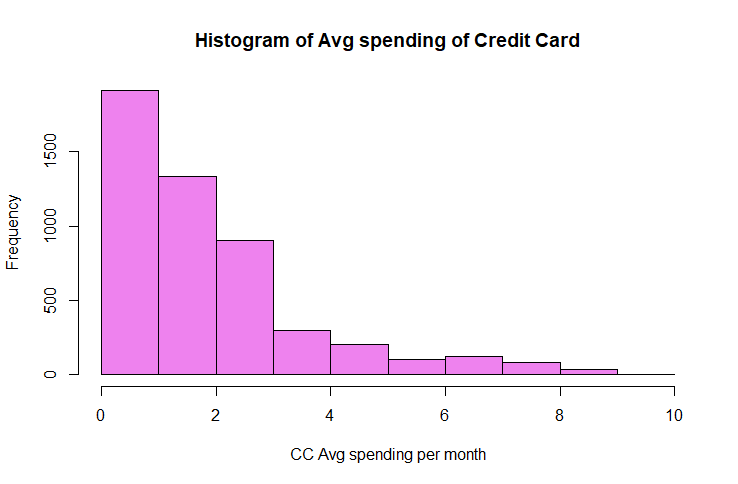
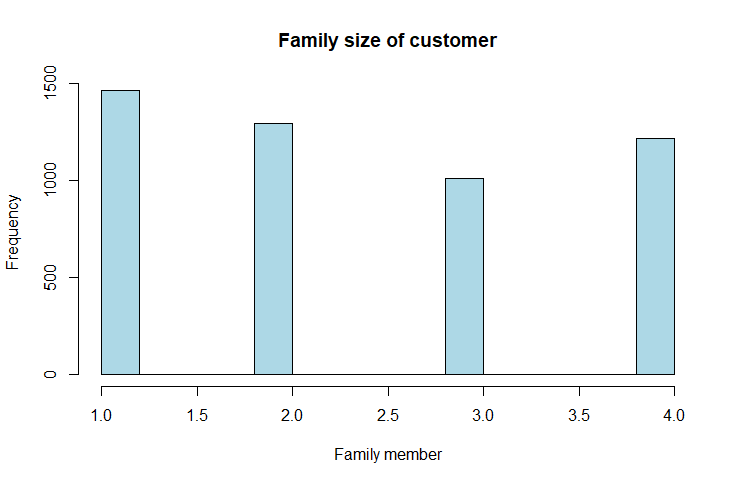
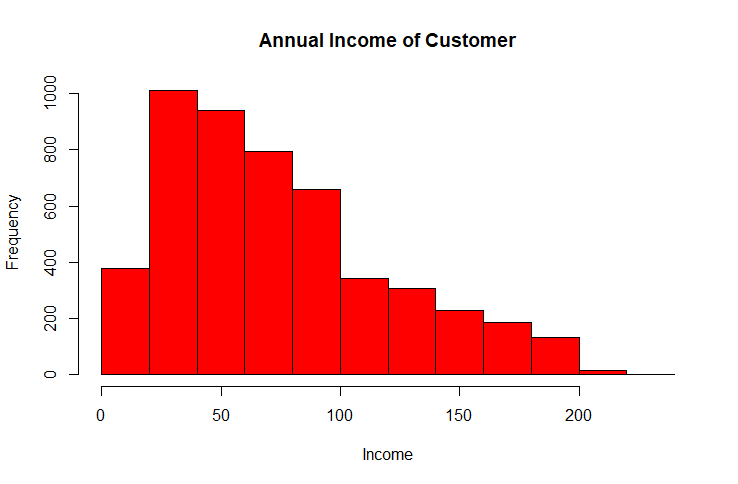
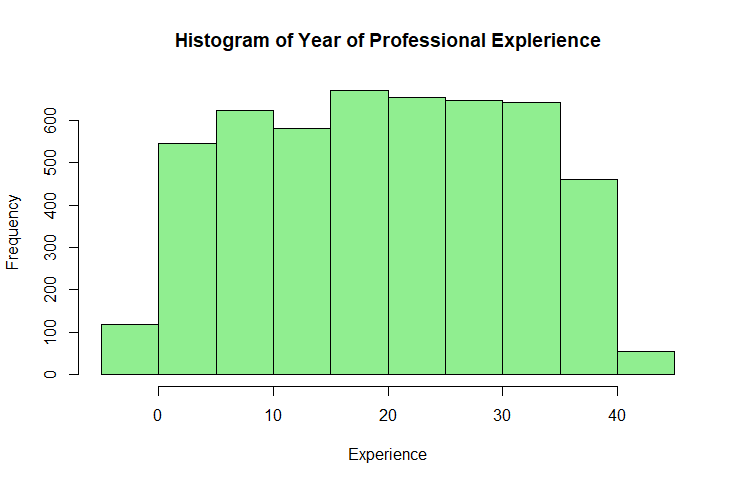
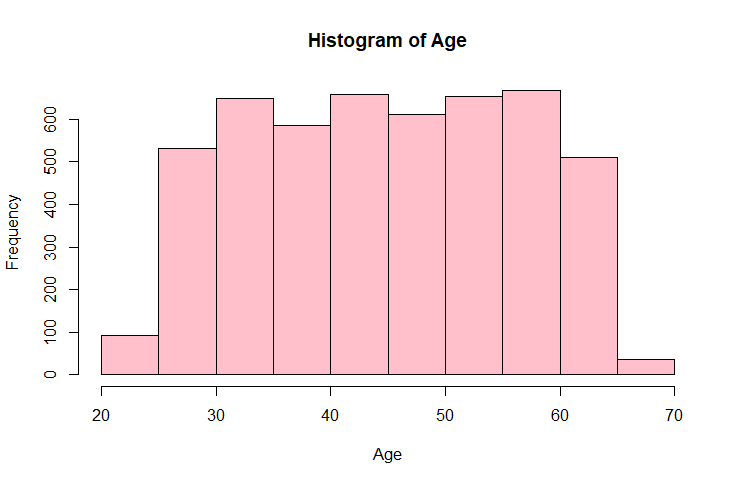
0.904 0.096

**Key observations:**

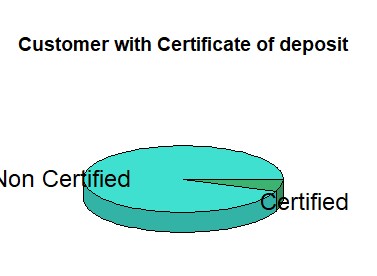
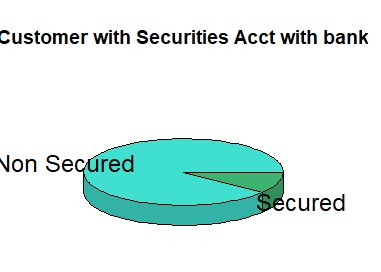
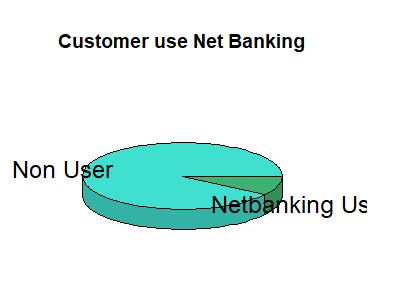
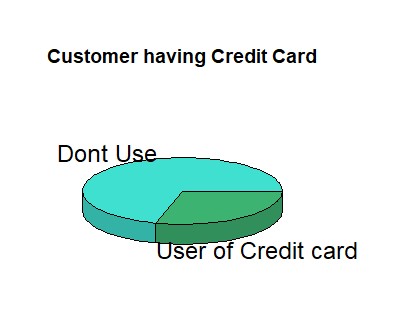
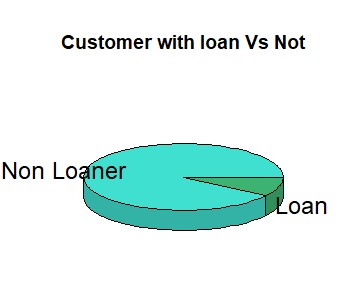
**Number of Rows and Columns:**  
> The number of rows in the dataset is 5,000  
> The number of columns (Features) in the dataset is 14

**Proportion of Responders Vs Non Responders:**  
> Customers with Personal Loans: 480 (9.6%)  
> Total Non-Responder Records: 4520 (90.4%)

* **Data Visulaisation**
* **Histogram of All Variable**



* Pie chart of binary Attributes

* Data Partition

**# Splitting data in Train and Test dataset**

indexes=sample(1:nrow(mydata), size = 0.3\*nrow(mydata))

mytest= mydata[indexes,]

dim(test)

mytrain=mydata[-indexes,]

dim(mydata)

Hence, After Splitting the data

* Dim(mytest)

1500 14

* Dim(mytrain)

3500 14

**#Check if the partition is correct**

-table(mytrain$Personal.Loan)

0 1

3146 354

- table(mytest$Personal.Loan)

0 1

1374 126

-prop.table((table(mytrain$Personal.Loan)))

0 1

0.8988571 0.1011429

-prop.table((table(mytest$Personal.Loan)))

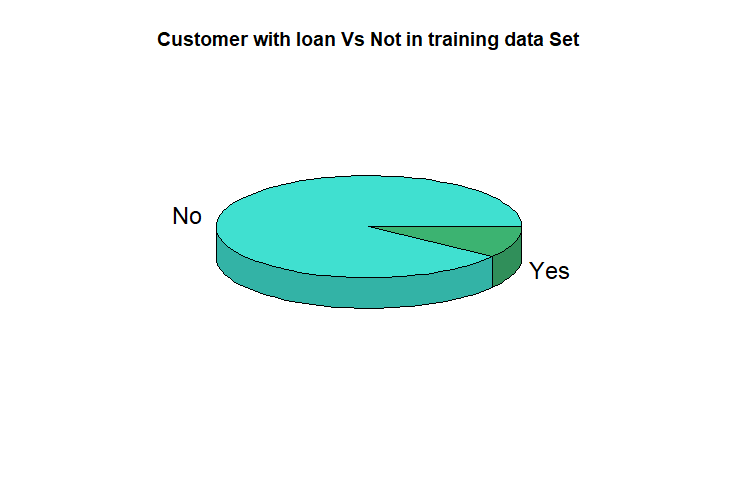
0 1

0.916 0.084

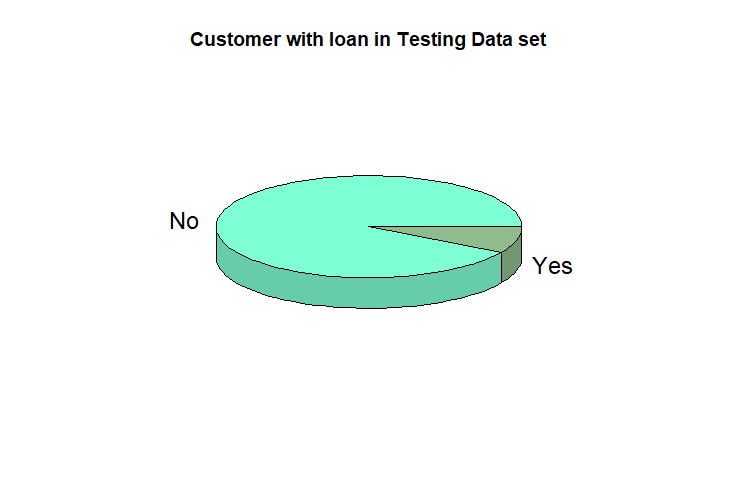
# Pie chart after Data Partition

* Comparison on Customers having Personal Loan or Not

Training data set



Testing data set



----------------------------------------------

***Model Building - Cart***

The CART algorithm is structured as a sequence of questions, the answers to which determine what the next question, if any should be. The result of these questions is a tree like structure where the ends are terminal nodes at which point there are no more questions.

# **Setting the control parameter**

**r.ctrl = rpart.control(minsplit=100, minbucket = 10, cp = 0, xval = 10)**

#Creating the Rpart and model1

-M1

n= 3500

node), split, n, loss, yval, (yprob)

\* denotes terminal node

1) root 3500 354 0 (0.89885714 0.10114286)

2) Income..in.K.month.< 108.5 2705 45 0 (0.98336414 0.01663586)

4) CCAvg< 2.95 2509 0 0 (1.00000000 0.00000000) \*

5) CCAvg>=2.95 196 45 0 (0.77040816 0.22959184)

10) CD.Account< 0.5 180 33 0 (0.81666667 0.18333333) \*

11) CD.Account>=0.5 16 4 1 (0.25000000 0.75000000) \*

3) Income..in.K.month.>=108.5 795 309 0 (0.61132075 0.38867925)

6) Education< 1.5 496 54 0 (0.89112903 0.10887097)

12) Family.members< 2.5 436 0 0 (1.00000000 0.00000000) \*

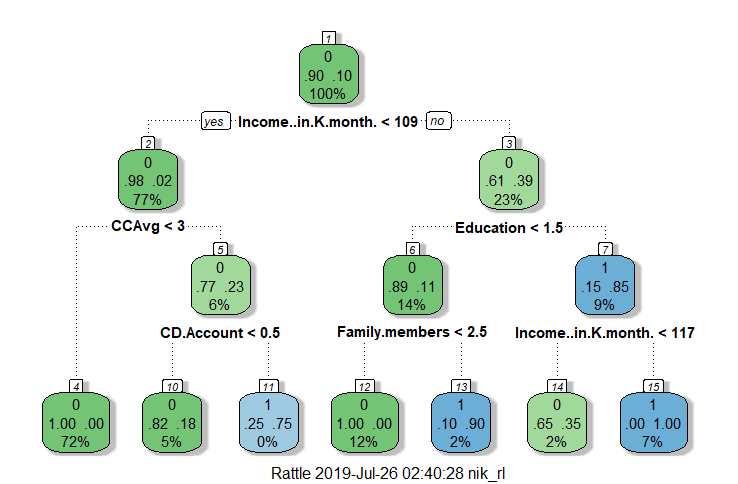
13) Family.members>=2.5 60 6 1 (0.10000000 0.90000000) \*

7) Education>=1.5 299 44 1 (0.14715719 0.85284281)

14) Income..in.K.month.< 116.5 68 24 0 (0.64705882 0.35294118) \*

15) Income..in.K.month.>=116.5 231 0 1 (0.00000000 1.00000000)

* Fancy Rpartplot



* Printcp

Classification tree:

rpart(formula = Personal.Loan ~ ., data = mytrain[, -1], method = "class",

control = r.ctrl)

Variables actually used in tree construction:

[1] CCAvg CD.Account Education

[4] Family.members Income..in.K.month.

Root node error: 354/3500 = 0.10114

n= 3500

CP nsplit rel error xerror xstd

1 0.298023 0 1.00000 1.00000 0.050390

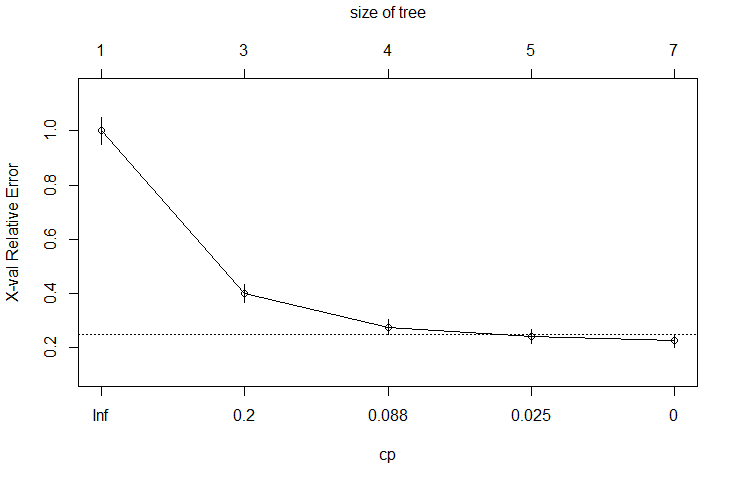
2 0.135593 2 0.40395 0.40113 0.032972

3 0.056497 3 0.26836 0.27684 0.027570

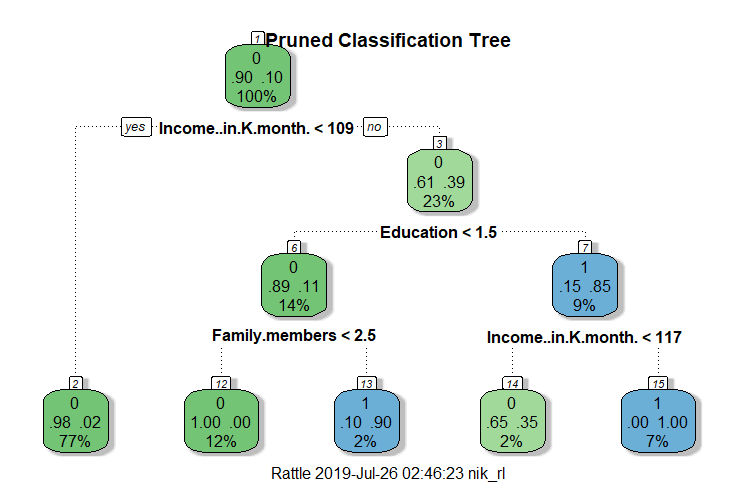
4 0.011299 4 0.21186 0.24294 0.025873

5 0.000000 6 0.18927 0.22599 0.024976

* Tree Performance Graph



* Final Cart Model tree



* Prediction Score

ID Age..in.years. Experience..in.years. Income..in.K.month.

1 25 1 49

2 45 19 34

3 39 15 11

4 35 9 100

5 35 8 45

6 37 13 29

ZIP.Code Family.members CCAvg Education Mortgage Personal.Loan

91107 4 1.6 1 0 0

90089 3 1.5 1 0 0

94720 1 1.0 1 0 0

94112 1 2.7 2 0 0

91330 4 1.0 2 0 0

92121 4 0.4 2 155 0

Securities.Account CD.Account Online CreditCard predict.class

1 0 0 0 0

1 0 0 0 0

0 0 0 0 0

0 0 0 0 0

0 0 0 1 0

0 0 1 0 0

predict.score.0 predict.score.1 deciles

0.98336414 0.01663586 9

0.98336414 0.01663586 9

0.98336414 0.01663586 9

0.98336414 0.01663586 9

0.98336414 0.01663586 9

0.98336414 0.01663586 9

* Performance on Training data set

The following model performance measures will be calculated on the training set to gauge the goodness of the model:

-Rank ordering

-Ks

-Area

-Gini

-Rank ordering

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Column1** | **Column2** | **Column3** | **Column4** | **Column5** | **Column6** | **Column7** | **Column8** | **Column9** | **Column10** | **Column11** |
|  | **deciles** | **cnt** | **cnt\_resp** | **cnt\_non\_resp** | **rrate** | **cum\_resp** | **cum\_non\_resp** | **cum\_rel\_resp** | **cum\_rel\_non\_resp** | **ks** |
|  |  |  |  |  |  |  |  |  |  |  |
| 1 | 10 | 359 | 309 | 50 | 86.10% | 309 | 50 | 87.30% | 1.60% | 85.7 |
| 2 | 9 | 2705 | 45 | 2660 | 1.70% | 354 | 2710 | 100.00% | 86.10% | 13.86 |
| 3 | 2 | 436 | 0 | 436 | 0.00% | 354 | 3146 | 100.00% | 100.00% | 0 |

-KS and Area Curve

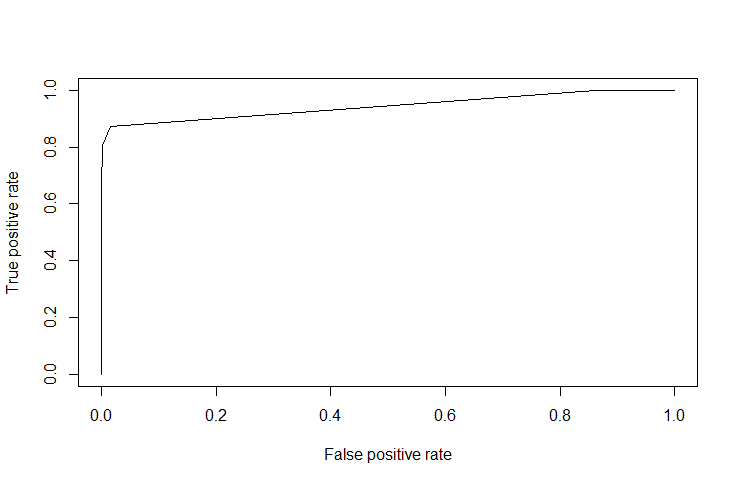
predict.class

Personal.Loan 0 1

0 3140 6

1 69 285

-Plot



* KS

0.8569882

* Auc

0.9434903

* Gini

0.7972688

Cart Model Performance

KS – 85.69%

AUC- 94.34&

Gini- 79.72%

**With ks – 85%, Auc-94%, & Gini coeffeicient ass 79% indicates to be an effective and good model.**

--------------------------------------------------------------------------

Hold Out Sample

* Model building with Oversampled data
* Syntax for the node path

node number: 2

root

Income..in.K.month.< 108.5

node number: 14

root

Income..in.K.month.>=108.5

Education>=1.5

Income..in.K.month.< 116.5

* Scoring Holdout Sample
* Head (mytest)

ID Age..in.years. Experience..in.years. Income..in.K.month.

3501 51 26 90

3502 65 39 105

3503 32 8 58

3504 29 3 53

3505 46 20 15

3506 64 39 103

ZIP.Code Family.members CCAvg Education Mortgage Personal.Loan

94110 1 2.8 2 0 0

91380 4 1.7 3 0 0

95616 3 2.0 1 90 0

95814 4 2.1 3 0 0

95370 4 0.6 3 0 0

90304 1 0.8 3 0 0

Securities.Account CD.Account Online CreditCard predict.class

0 0 1 1 0

1 0 1 0 0

0 0 1 0 0

0 0 1 0 0

1 0 1 0 0

0 0 1 1 0

predict.score.0 predict.score.1 deciles

0.98336414 0.01663586 10

0.98336414 0.01663586 10

0.98336414 0.01663586 10

0.98336414 0.01663586 10

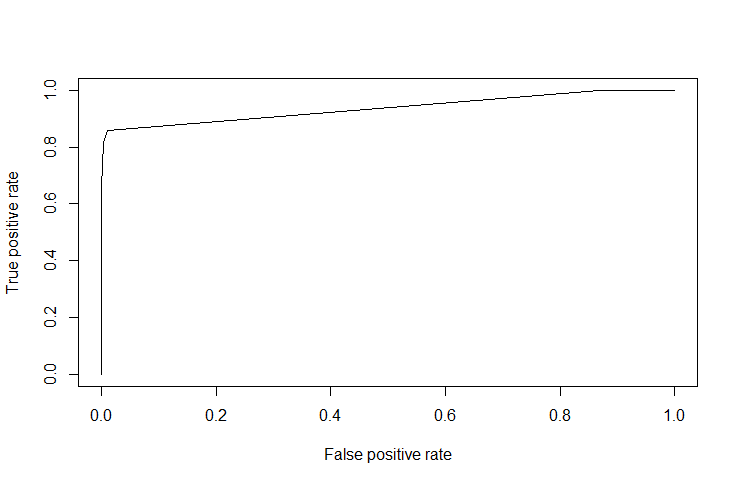
0.98336414 0.01663586 10

0.98336414 0.01663586 10

Holdout Ranking

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **deciles** | **cnt** | **cnt\_resp** | **cnt\_non\_resp** | **rrate** | **cum\_resp** | **cum\_non\_resp** | **cum\_rel\_resp** | **cum\_rel\_non\_resp** | **ks** |
|  |  |  |  |  |  |  |  |  |  |  |
| **1** | 10 | 1319 | 126 | 1193 | 9.55% | 126 | 1193 | 100% | 86.80% | 13.17 |
| **2** | 2 | 181 | 0 | 181 | 0.00% | 126 | 1374 | 100% | 100.00% | 0 |

**Holdout test**

****

**predict.class**

Personal.Loan 0 1

0 1370 4

1 23 103

* **AUC**

0.9367621

* **KS**

0.8469536

* **Gini**

0.7899441

**Summary – Cart Model Performance (holdout Sample)**

1. **Ks : 84%**
2. **Auc: 93%**
3. **Gini: 78%**

Hence, Ks with 84%, Auc with 93%, and Gini with 78%; indicated to be a helpful and good model in performance.

**Cart Conclusion**

**Measures Train Test Deviation**

--------------------------------------------------------------------------------------------------------------

**KS 85.69% 84% 1.69%**

**AUC 94.34% 93% 1.34%**

**GINI 79.72% 78% 1.72%**

**--------------------------------------------------------------------------------------------**

**Conclusion**

* If we compare with the train data set, there is just minor difference between the resulting tests.
* It can be observed that most of the Model Performance values for Training & Testing sets are around 1%
* Hence, the model is performing well.
* The percentage deviation between Training and Testing Dataset also is reasonably under control, suggesting a robust model.

**---------------------------------------------------------------------------------------------------------------------------**

**Thank You**

**Source Code**

#Library(“plotrix”)

#Library(“ggplot2”)

#Library(“crplot”)

#library(caret”)

#library(rpart)

#library(rpart.plot)

#library(corrplot)

#library(rattle)

#library(RColorBrewer)

#library(data.table)

#library(scales)

#library(ROCR)

#library(ineq)

--------------------------------------------------------------------------

setwd("F:/r data machine learning")

getwd()

fulldata=read.csv("bankdata.csv", header = T)

mytrain=read.csv("Bank Personal Loan Dataset.csv", header = T)

mytest=read.csv("bankholdout.csv", header = T)

mytrain

mytest

head(mytrain)

head(mytest)

summary(mytrain)

variable.names("ID", "Age","Experience","Income","ZIP Code","Family members","CCAvg","Education","Mortgage","Personal Loan","Securities Account","CD Account","Online","CreditCard")

dim(mytest)

dim(mytrain)

names(mytrain)

names(mytest)

head(mytrain)

attach(mytrain)

attach(mytest)

table(mytrain$Personal.Loan)

table(mytest$Personal.Loan)

prop.table(table(mytrain$Personal.Loan))

prop.table(table(mytest$Personal.Loan))

colSums(is.na(fulldata))

table(fulldata$Personal.Loan)

prop.table(table(fulldata$Personal.Loan))

prop.table(table(fulldata$Personal.Loan))

table(mytrain$Personal.Loan)

table(mytest$Personal.Loan)

prop.table((table(mytrain$Personal.Loan)))

prop.table((table(mytest$Personal.Loan)))

Data Visualisation

#Graph

hist(mydata$Age..in.years., col = 'pink', main = 'Histogram of Age', xlab = 'Age')

hist(mydata$Experience..in.years., col = 'light green', main = 'Histogram of Explerience',xlab = 'Experience')

hist(mydata$Income..in.K.month., col = 'red', main = 'Histogram of Income',xlab = 'Income')

hist(mydata$Family.members, col = 'light blue', main = 'Histogram of Family member',xlab = 'Family member')

hist(mydata$CCAvg, col = 'violet', main = 'Histogram of Avg spending of Credit Card',xlab = 'CC Avg spending per month')

hist(mydata$Education, col = 'orange', main = 'Histogram of Education Level',xlab = 'Education')

hist(mydata$Mortgage, col = 'grey', main = 'Histogram of Mortgage',xlab = 'Mortgage')

hist(mydata$Personal.Loan, col = 'Dark Red', main = 'Histogram of Personal Loan', xlab = 'Personal Loan')

hist(mydata$Securities.Account, col = 'maroon', main = 'Histogram of Securities on Account with bank', xlab = 'Security')

hist(mydata$CD.Account, col = 'purple', main = 'Histogram of Certificate of deposit', xlab = 'CD(certificate of deposit account) with bank')

hist(mydata$Online, col = 'dark green', main = 'Histogram of Online Bankers', xlab='online Banking')

hist(mydata$CreditCard, col = 'yellow', main = 'Histogram of Credit Card User', xlab='Credit Card')

#Pie Chart

library("plotrix")

pie3D(prop.table((table(fulldata$Personal.Loan))),

main='Customer with loan Vs Not',

#explode=0.1,

labels=c("Non Loaner", "Loan"),

col = c("Turquoise", "Medium Sea Green")

)

pie3D(prop.table((table(fulldata$Securities.Account))),

main='Customer with Securities Acct with bank',

#explode=0.1,

labels=c("Non Secured", "Secured"),

col = c("Turquoise", "Medium Sea Green")

)

pie3D(prop.table((table(fulldata$CD.Account))),

main='Customer with Certificate of deposit',

#explode=0.1,

labels=c("Non Certified", "Certified"),

col = c("Turquoise", "Medium Sea Green")

)

pie3D(prop.table((table(fulldata$Personal.Loan))),

main='Customer use Net Banking',

#explode=0.1,

labels=c("Non User", "Netbanking User"),

col = c("Turquoise", "Medium Sea Green")

)

pie3D(prop.table((table(fulldata$CreditCard))),

main='Customer having Credit Card',

#explode=0.1,

labels=c("Dont Use", "User of Credit card"),

col = c("Turquoise", "Medium Sea Green")

)

#Data Partition

indexes=sample(1:nrow(mydata), size = 0.3\*nrow(mydata))

test= mydata[indexes,]

dim(test)

train=mydata[-indexes,]

dim(mydata)

#piechart after data partition

par(mfrow=c(1,1))

pie3D(prop.table((table(mytrain$Personal.Loan))),

main='Customer with loan Vs Not in training data Set',

#explode=0.1,

labels=c("No", "Yes"),

col = c("Turquoise", "Medium Sea Green")

)

pie3D(prop.table((table(mytest$Personal.Loan))),

main='Customer with loan in Testing Data set',

#explode=0.1,

labels=c("No", "Yes"),

col = c("Aquamarine", "Dark Sea Green")

)

------------------------------------

**Cart**

## loading the library

library(rpart)

library(rpart.plot)

table(mytrain$Personal.Loan)

## Target Rate

sum(mytrain$Personal.Loan)/5000

## setting the control paramter inputs for rpart

r.ctrl = rpart.control(minsplit=100, minbucket = 10, cp = 0, xval = 10)

## calling the rpart function to build the tree

##m1 <- rpart(formula = Ta ~ ., data = CTDF.dev[which(CTDF.dev$Holding\_Period>10),-1], method = "class", control = r.ctrl)

m1 <- rpart(formula = Personal.Loan ~ .,

data = mytrain[,-1], method = "class",

control = r.ctrl)

m1

install.packages("rattle")

install.packages("RcolorBrewer")

library(rattle)

##install.packages

library(RColorBrewer)

fancyRpartPlot(m1)

## to find how the tree performs

printcp(m1)

plotcp(m1)

##rattle()

## Pruning Code

ptree<- prune(m1, cp= 0.025 ,"CP")

printcp(ptree)

fancyRpartPlot(ptree, uniform=TRUE, main="Pruned Classification Tree",

)

## Let's use rattle to see various model evaluation measures

##rattle()

View(mytrain)

## Scoring syntax

?predict

mytrain$predict.class <- predict(ptree, mytrain, type="class")

mytrain$predict.score <- predict(ptree, mytrain, type="prob")

View(mytrain)

head(mytrain)

## deciling code

decile <- function(x){

deciles <- vector(length=10)

for (i in seq(0.1,1,.1)){

deciles[i\*10] <- quantile(x, i, na.rm=T)

}

return (

ifelse(x<deciles[1], 1,

ifelse(x<deciles[2], 2,

ifelse(x<deciles[3], 3,

ifelse(x<deciles[4], 4,

ifelse(x<deciles[5], 5,

ifelse(x<deciles[6], 6,

ifelse(x<deciles[7], 7,

ifelse(x<deciles[8], 8,

ifelse(x<deciles[9], 9, 10

))))))))))

}

class(mytrain$predict.score)

## deciling

mytrain$deciles <- decile(mytrain$predict.score[,2])

View(mytrain)

head(mytrain)

## Ranking code

install.packages("data.table")

install.packages("scales")

library(data.table)

library(scales)

tmp\_DT = data.table(mytrain)

rank <- tmp\_DT[, list(

cnt = length(Personal.Loan),

cnt\_resp = sum(Personal.Loan),

cnt\_non\_resp = sum(Personal.Loan == 0)) ,

by=deciles][order(-deciles)]

rank$rrate <- round(rank$cnt\_resp / rank$cnt,4);

rank$cum\_resp <- cumsum(rank$cnt\_resp)

rank$cum\_non\_resp <- cumsum(rank$cnt\_non\_resp)

rank$cum\_rel\_resp <- round(rank$cum\_resp / sum(rank$cnt\_resp),4);

rank$cum\_rel\_non\_resp <- round(rank$cum\_non\_resp / sum(rank$cnt\_non\_resp),4);

rank$ks <- abs(rank$cum\_rel\_resp - rank$cum\_rel\_non\_resp) \* 100;

rank$rrate <- percent(rank$rrate)

rank$cum\_rel\_resp <- percent(rank$cum\_rel\_resp)

rank$cum\_rel\_non\_resp <- percent(rank$cum\_rel\_non\_resp)

View(rank)

install.packages("ROCR")

install.packages("ineq")

library(ROCR)

library(ineq)

pred <- prediction(mytrain$predict.score[,2], mytrain$Personal.Loan)

perf <- performance(pred, "tpr", "fpr")

plot(perf)

KS <- max(attr(perf, 'y.values')[[1]]-attr(perf, 'x.values')[[1]])

auc <- performance(pred,"auc");

auc <- as.numeric(auc@y.values)

gini = ineq(mytrain$predict.score[,2], type="Gini")

with(mytrain, table(Personal.Loan, predict.class))

auc

KS

gini

View(rank)

#oversampled dataset

## Syntax to get the node path

tree.path <- path.rpart(ptree, node = c(2, 14))

nrow(mytest)

## Scoring Holdout sample

mytest$predict.class <- predict(ptree, mytest, type="class")

mytest$predict.score <- predict(ptree, mytest, type="prob")

mytest$deciles <- decile(mytest$predict.score[,2])

View(mytest)

head(mytest)

## Ranking code

tmp\_DT = data.table(mytest)

h\_rank = tmp\_DT[, list(

cnt = length(Personal.Loan),

cnt\_resp = sum(Personal.Loan),

cnt\_non\_resp = sum(Personal.Loan == 0)) ,

by=deciles][order(-deciles)]

h\_rank$rrate <- round(h\_rank$cnt\_resp / h\_rank$cnt,4);

h\_rank$cum\_resp <- cumsum(h\_rank$cnt\_resp)

h\_rank$cum\_non\_resp <- cumsum(h\_rank$cnt\_non\_resp)

h\_rank$cum\_rel\_resp <- round(h\_rank$cum\_resp / sum(h\_rank$cnt\_resp),4);

h\_rank$cum\_rel\_non\_resp <- round(h\_rank$cum\_non\_resp / sum(h\_rank$cnt\_non\_resp),4);

h\_rank$ks <- abs(h\_rank$cum\_rel\_resp - h\_rank$cum\_rel\_non\_resp)\*100;

h\_rank$rrate <- percent(h\_rank$rrate)

h\_rank$cum\_rel\_resp <- percent(h\_rank$cum\_rel\_resp)

h\_rank$cum\_rel\_non\_resp <- percent(h\_rank$cum\_rel\_non\_resp)

View(h\_rank)

head(h\_rank)

pred <- prediction(mytest$predict.score[,2], mytest$Personal.Loan)

perf <- performance(pred, "tpr", "fpr")

KS <- max(attr(perf, 'y.values')[[1]]-attr(perf, 'x.values')[[1]])

auc <- performance(pred,"auc");

auc <- as.numeric(auc@y.values)

gini = ineq(mytest$predict.score[,2], type="Gini")

with(mytest, table(Personal.Loan, predict.class))

auc

KS

gini

# Thank You

----------------------------------------------------

The END

----------------------------------------------------