Thera Bank - Loan Purchase Modelling

Table of Contents

2 Project Objective

The objective of the report is to explore the data file Thera_Bank_data.csv in R and generate insights about the data set. This exploration report will consists of the following scenarios:

- Exploratory Data Analysis of the data
- Apply appropriate clustering on the data and interpreting the output
- Build appropriate models on both the test and train data (CART and Random Forest)
- Interpreting the model outputs and performing the necessary modifications wherever eligible
- Check the performance of all the models that you have built (test and train)

3 Exploratory Data Analysis – Step by step approach

A Typical Data exploration activity consists of the following steps:

1. Environment Set up and Data Import
2. Variable Identification
3. Visualisation Plots

We shall follow these steps in exploring the provided dataset.

3.1 Environment Set up and Data Import

3.1.1 Set up working Directory

Setting a working directory on starting of the R session makes importing and exporting data files and code files easier. Basically, working directory is the location/ folder on the PC where you have the data, codes etc. related to the project.
Please refer Appendix A for Source Code.

3.1.2 Import and Read the Dataset

The given dataset is in .csv format. Hence, the command 'read.csv' is used for importing the file.
Please refer Appendix A for Source Code.

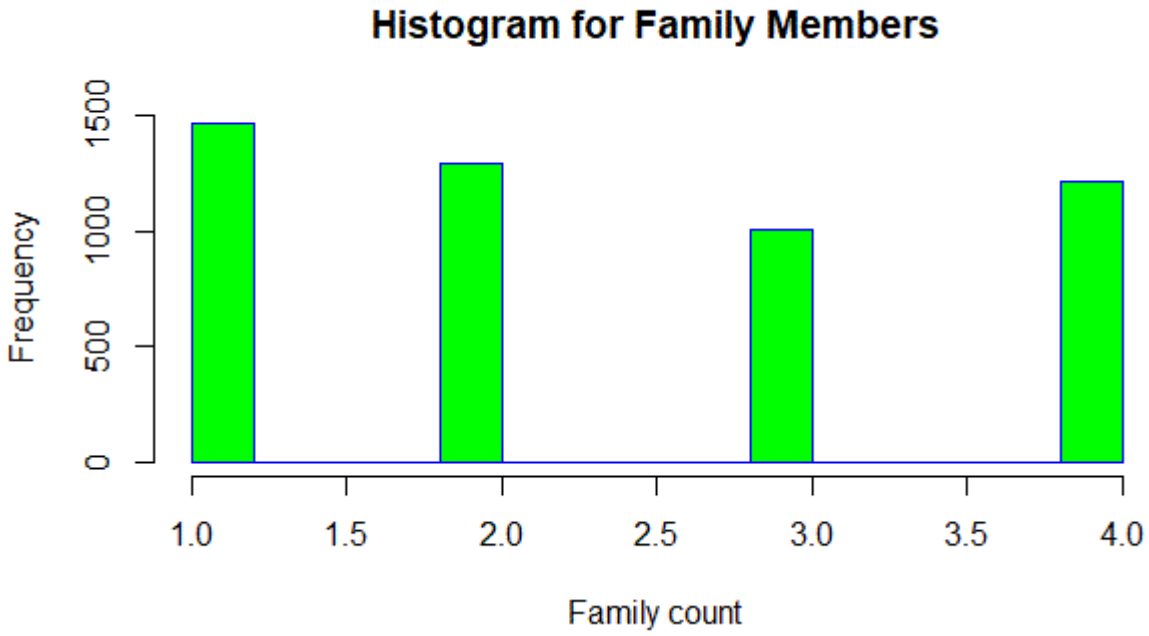
3.2 Variable Identification

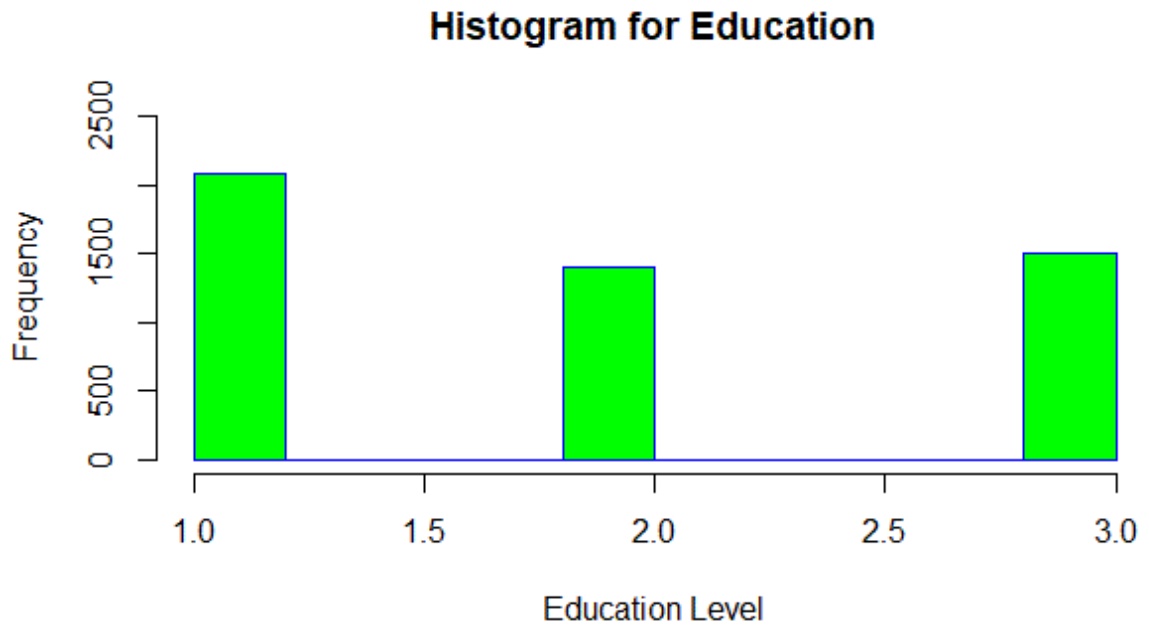
The length and breadth of the data was examined and the names of the data were pulled from the dataset. The data consisted of 14 variables consisting of 5000 employees determining whether or not they took a loan from the bank. The string type of the data was also verified by using the str() function.

3.3 Visualisation Plots

Histogram Plot

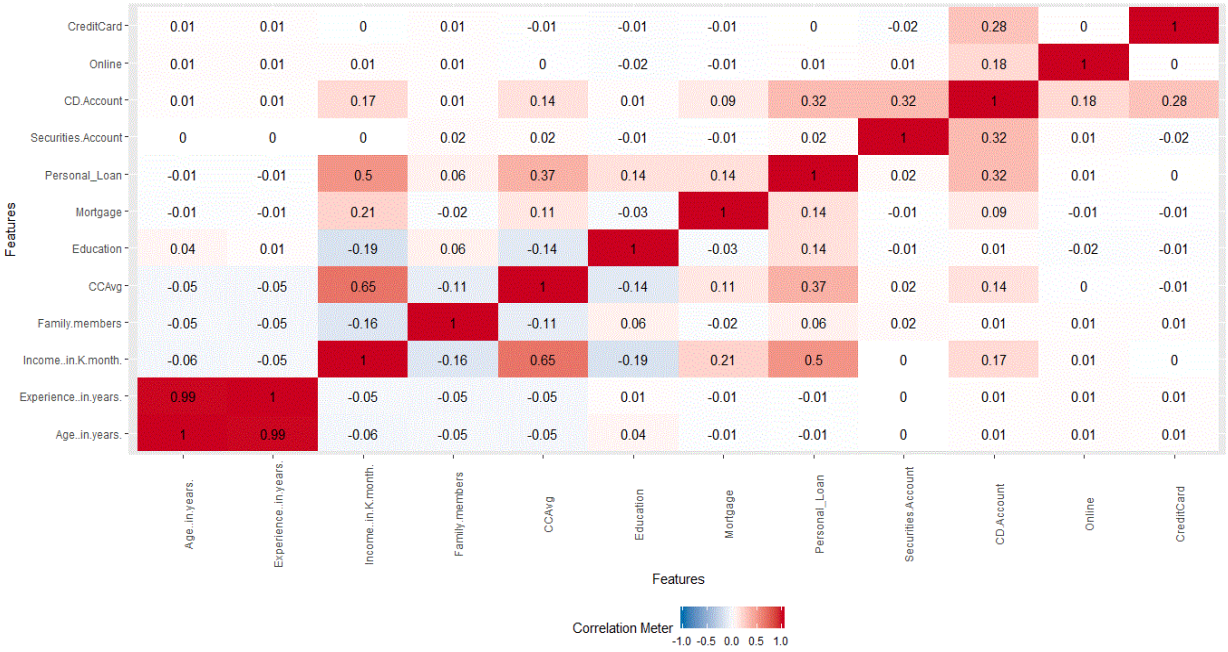
Below are the histogram plots for Family and Education from the dataset.





Corrplot

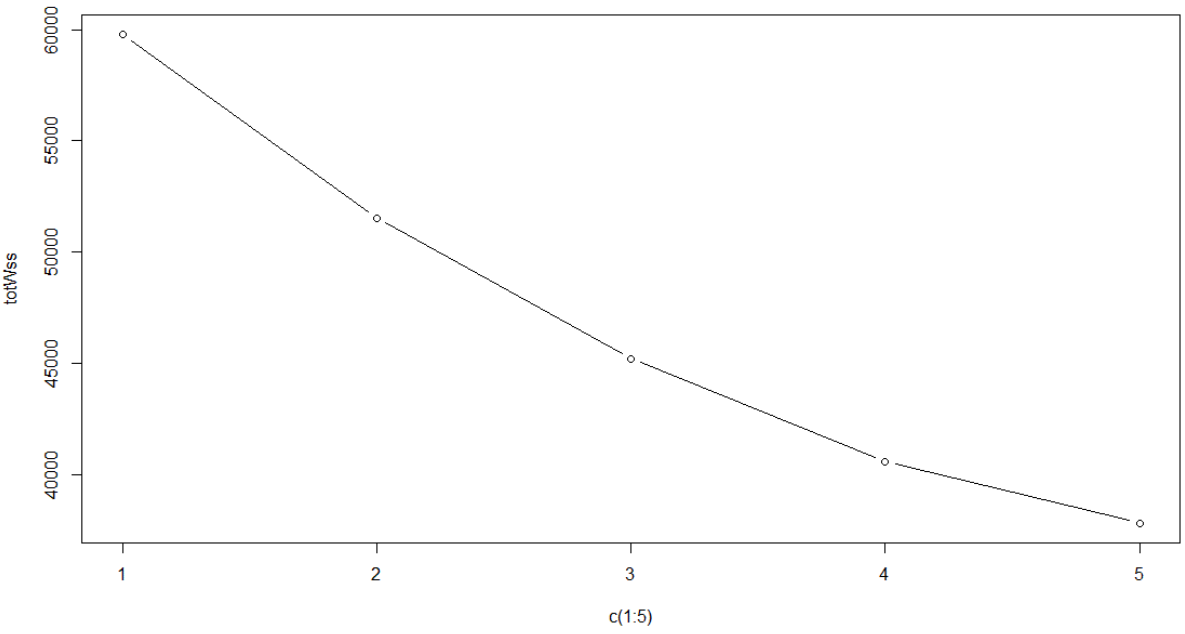
Corrplot was plotted between all the variables to determine the correlation amongst the variables and the output is displayed in the below figure.



4. Performing clustering and interpreting the result

A k-means clustering was performed on the scaled dataset. Initially K-means clustering with 2 cluste rs of sizes 916, 4066 was found and plotted.

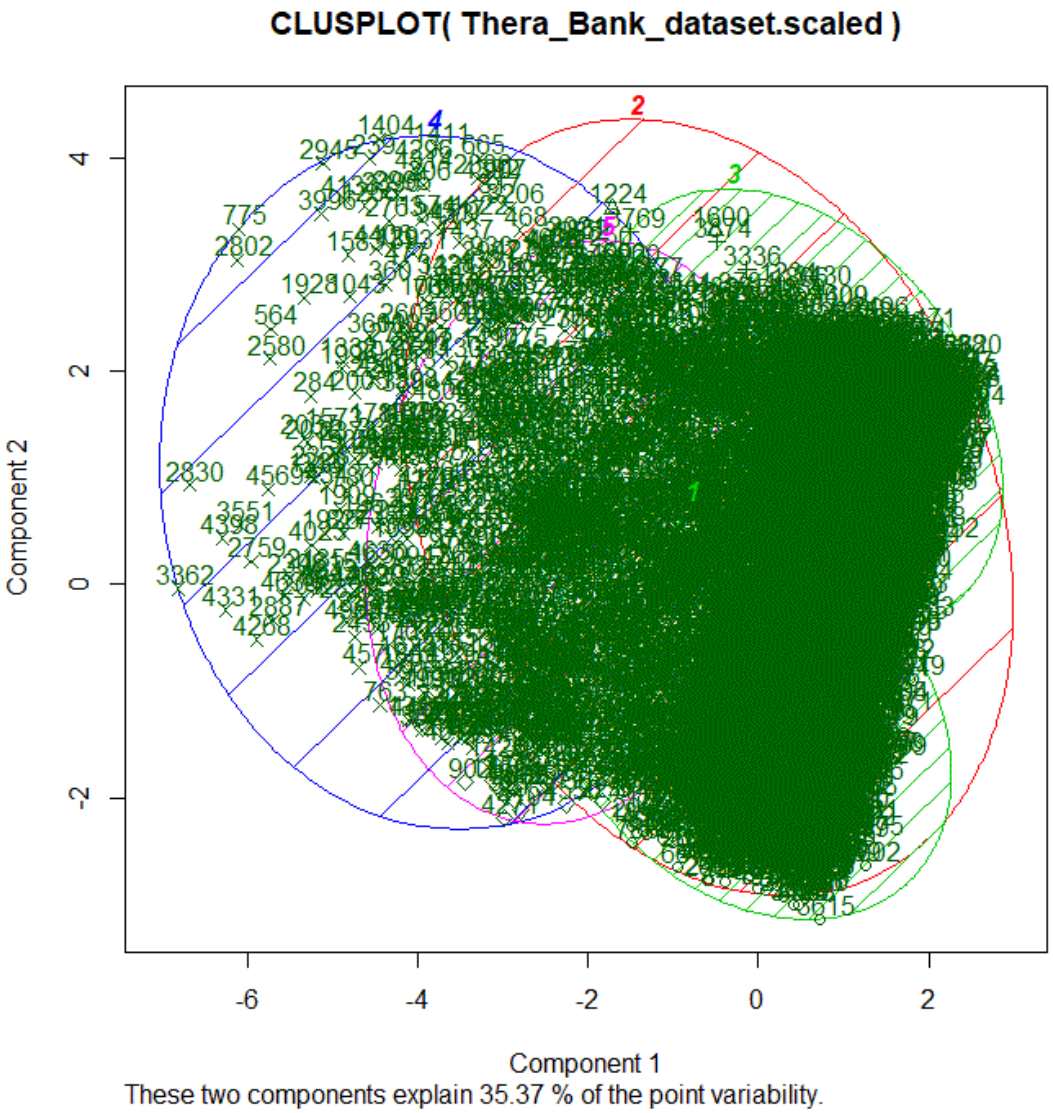
To find the right number of cluster withinss command and NbClust function was used.



According to the majority rule, the best number of clusters was found to be 5.

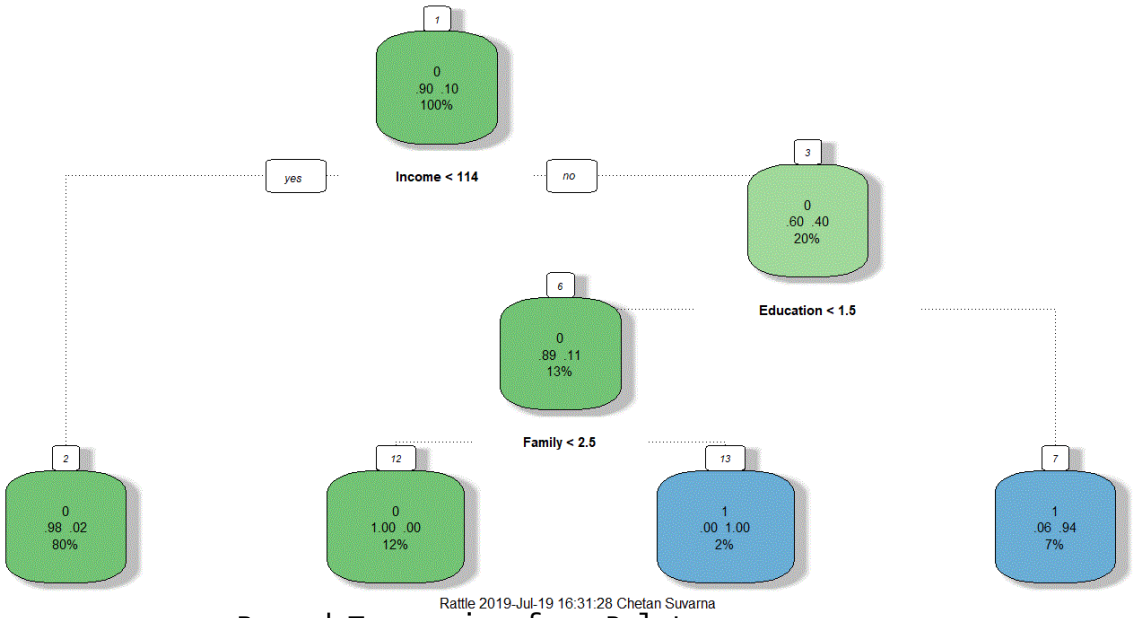
K-means clustering with 5 clusters of sizes 1703, 441, 1790, 478, 570 was found.

Clusters were plotted using clusplot() function. The output is displayed below.



5. CART model with Performance Measures

Using fancyRpartPlot() function a graph is plotted which displays the distributed dataset in the form of a tree. After plotting the complexity plot and pruning the tree at 0.05 a new pruned tree is displayed below.



Pruned Tree using fancyRplot

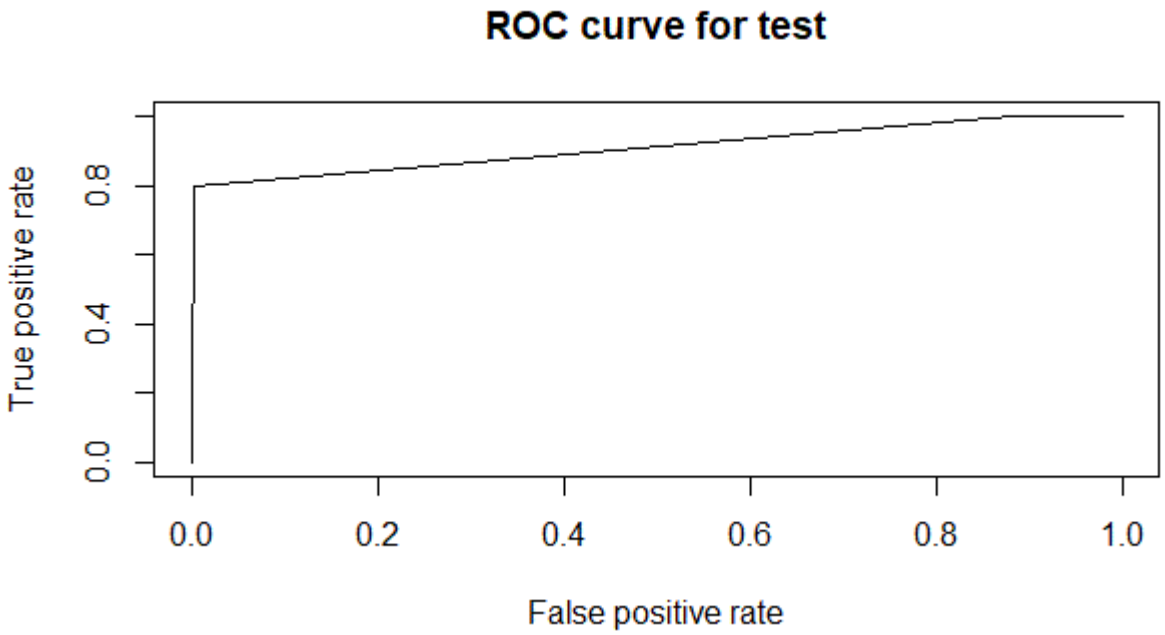
Confusion Metrix

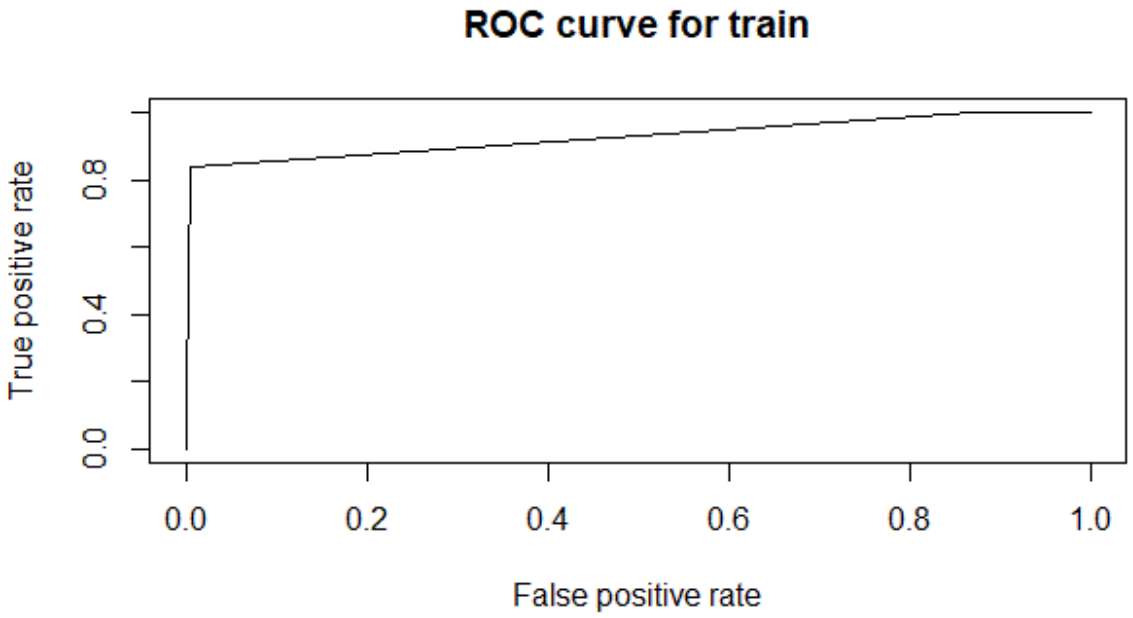
```
CART_CM_train
       0    1
  0 2990   14
  1   52  266
```

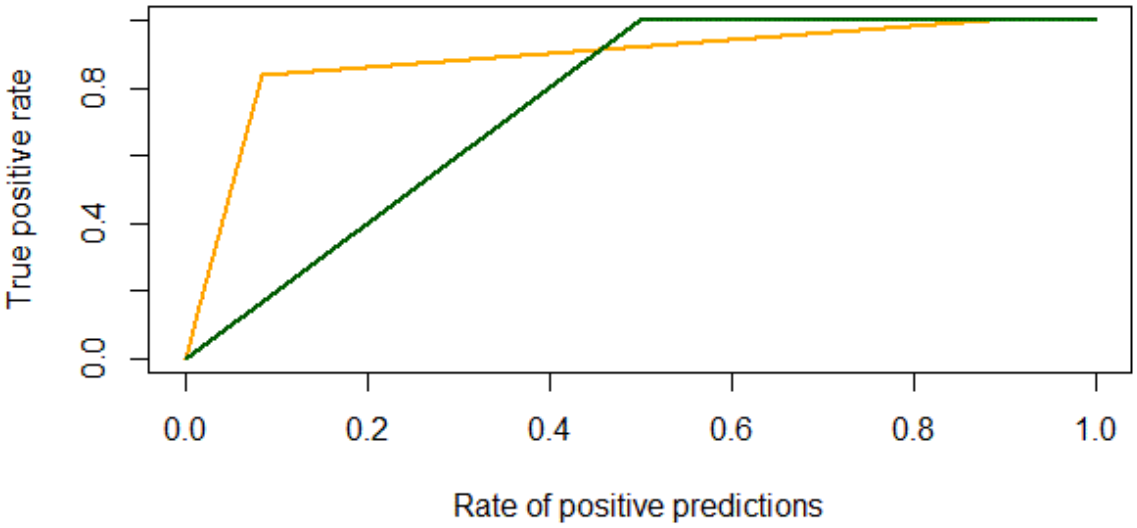
```
CART_CM_test

       0    1
  0 1497    3
  1   32  128
```

Misclassification Rate for train and test are as below:
0.01986755
0.02108434

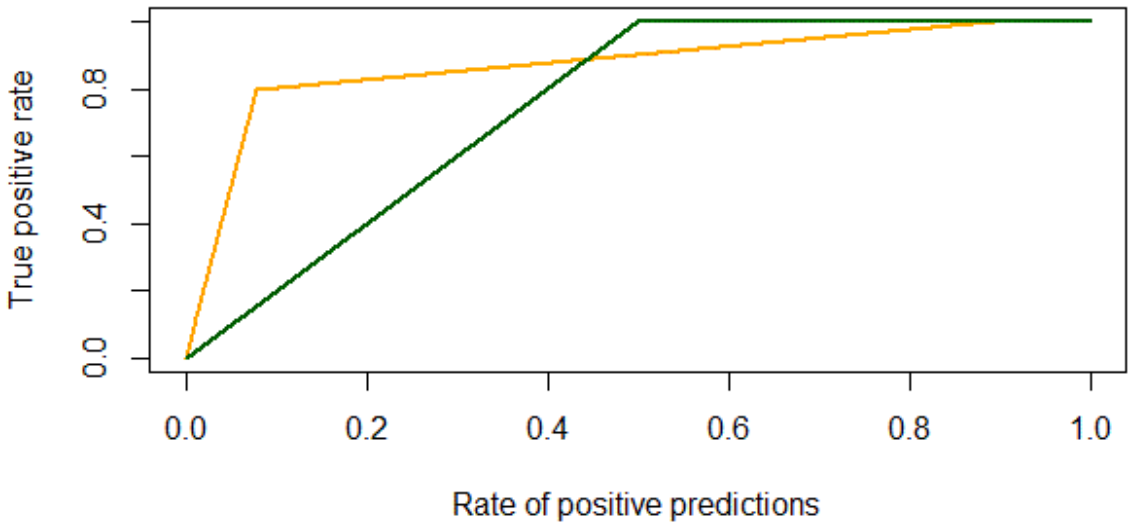
The graphs for all the performance measures have been plotted below.



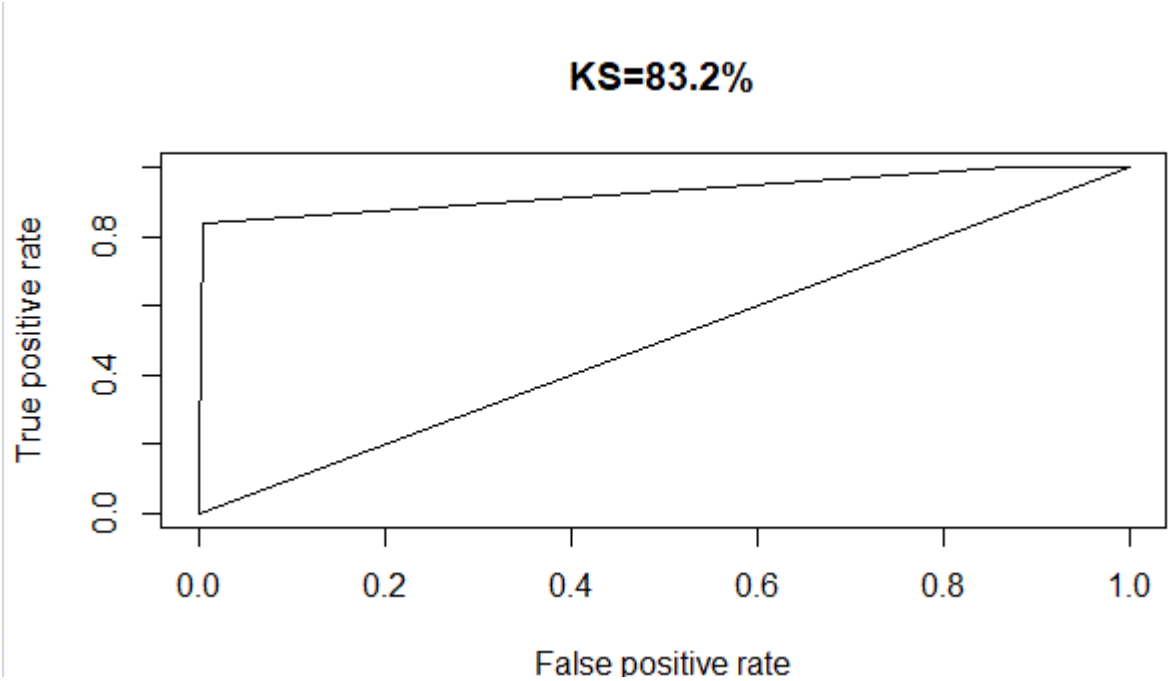




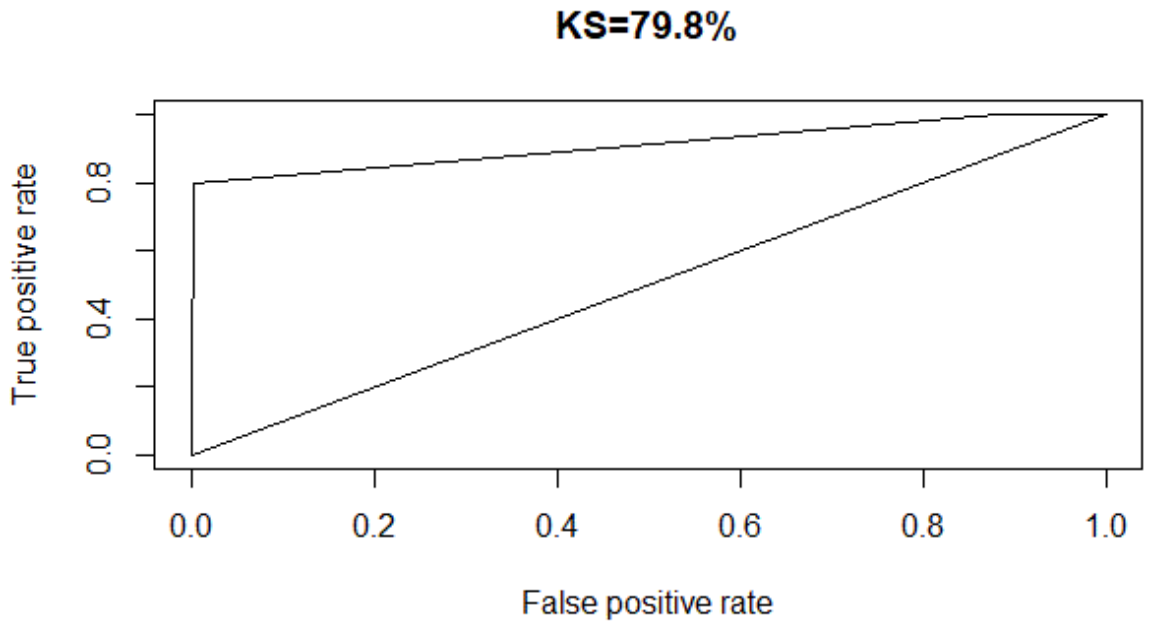
Gains for train



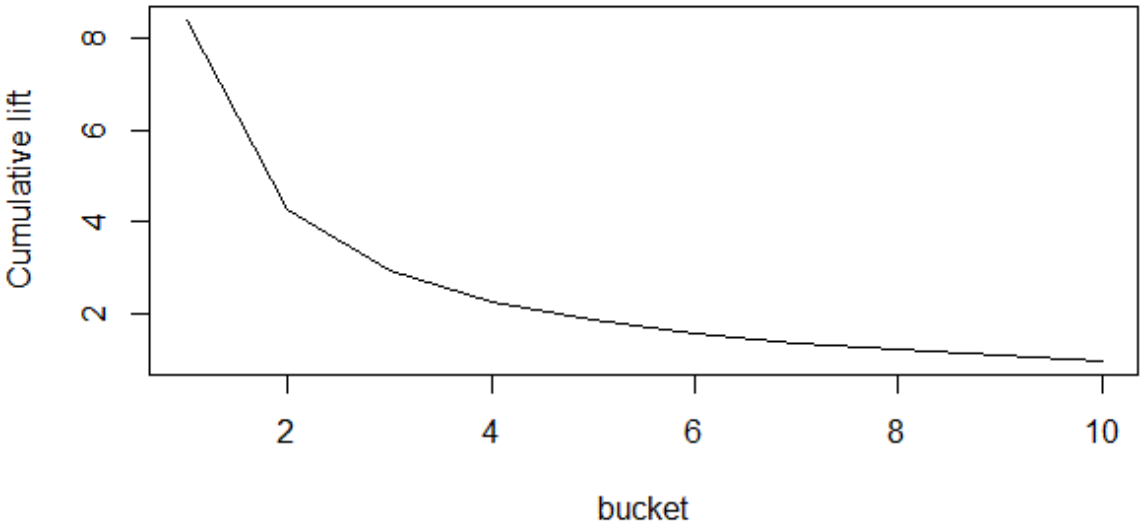
Gains for test



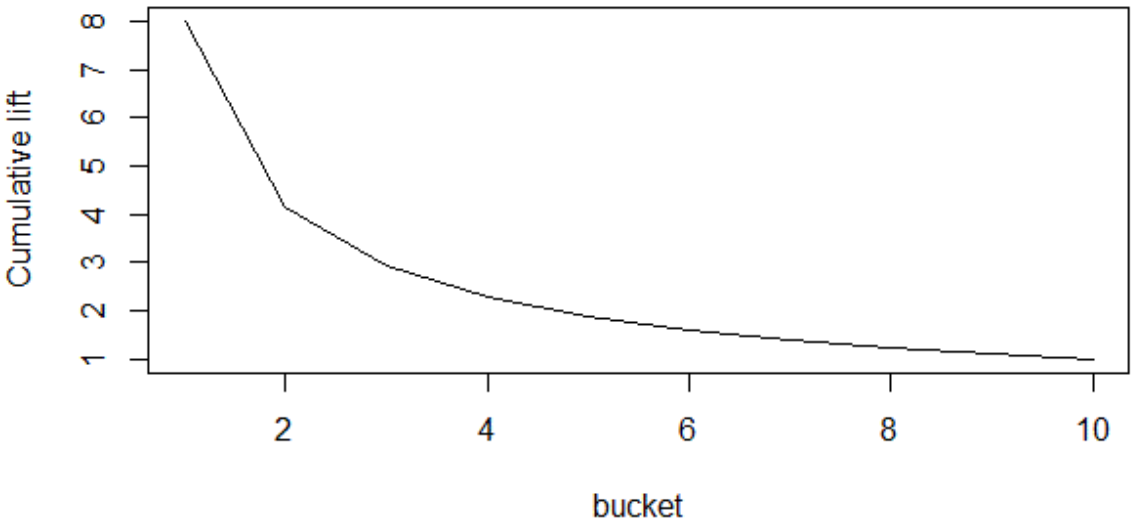
KS train



KS test



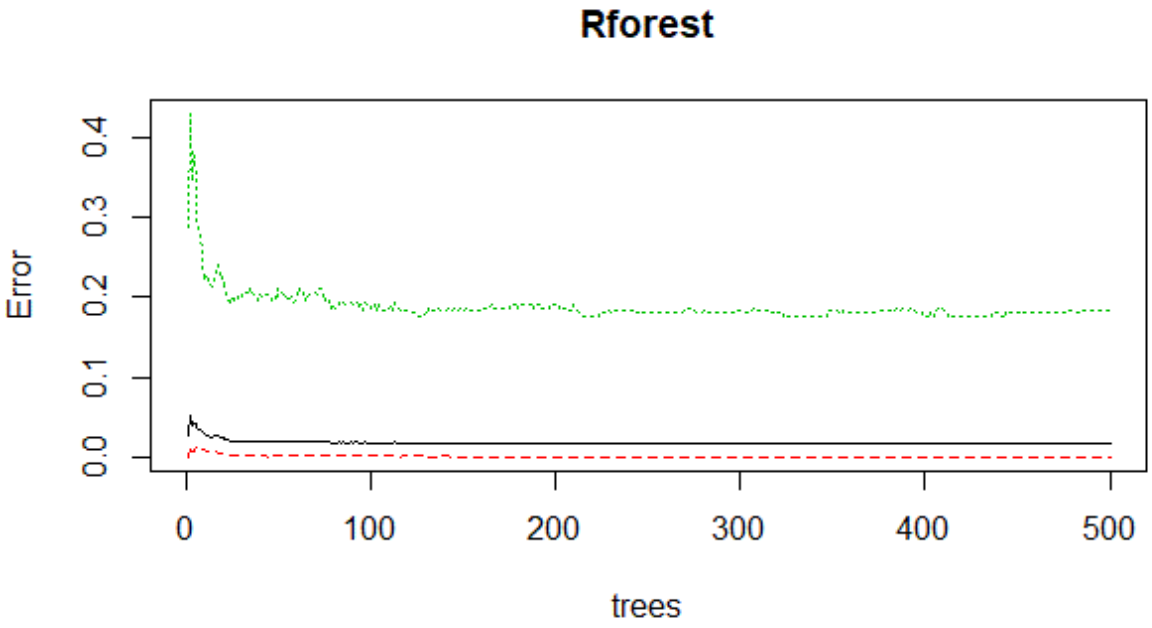
Lift train



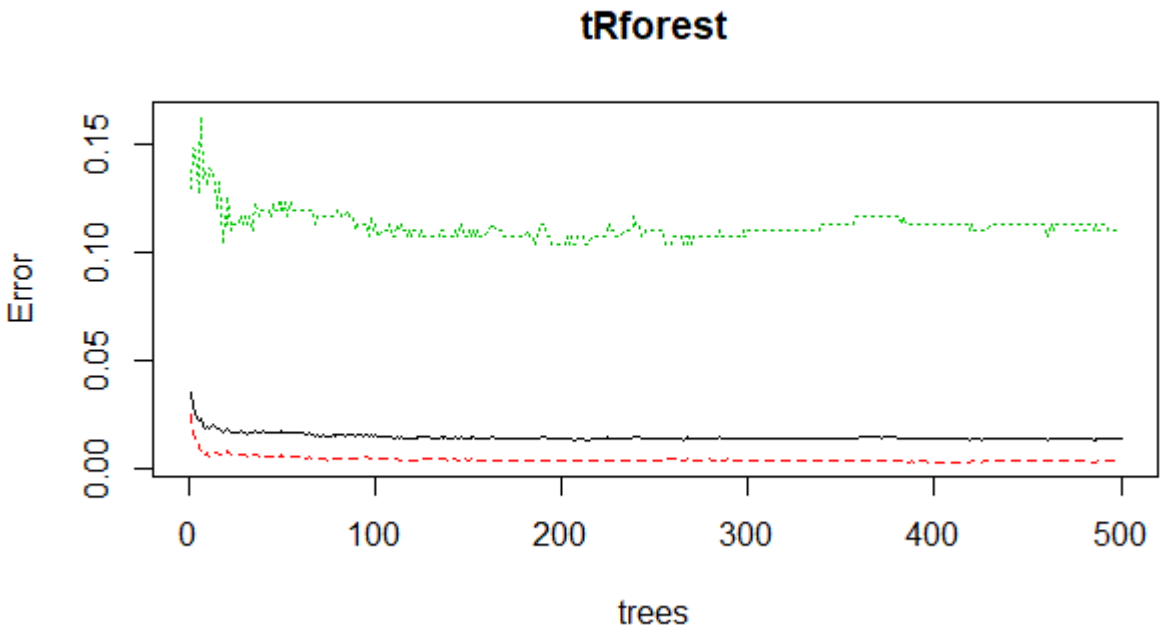
Lift test

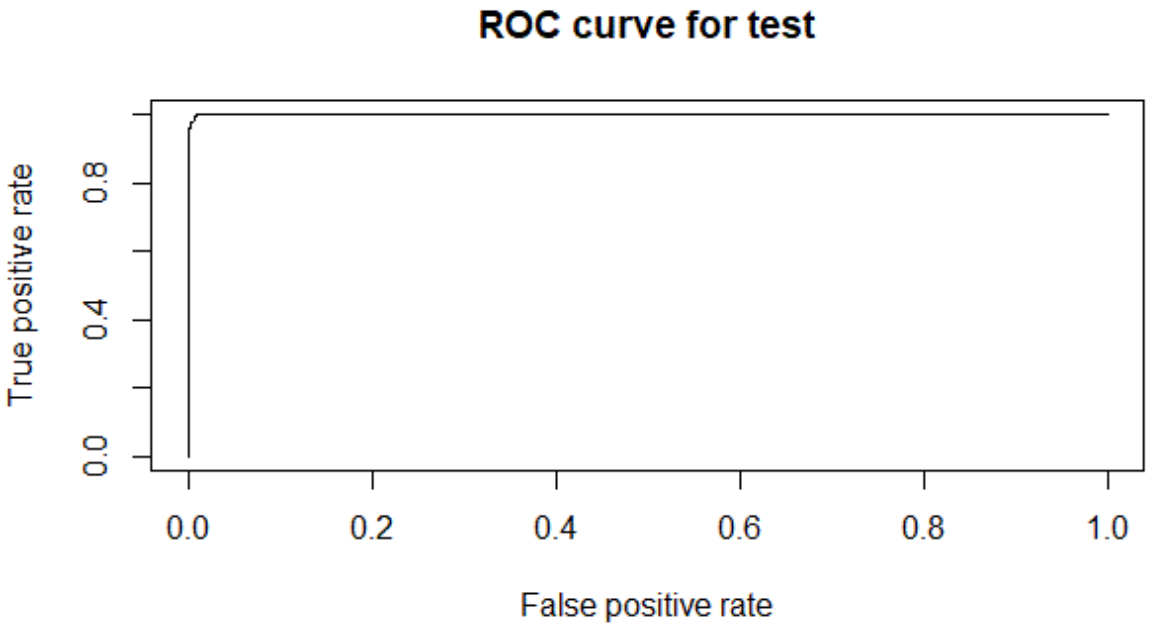
5. Random Forest plots

RF train

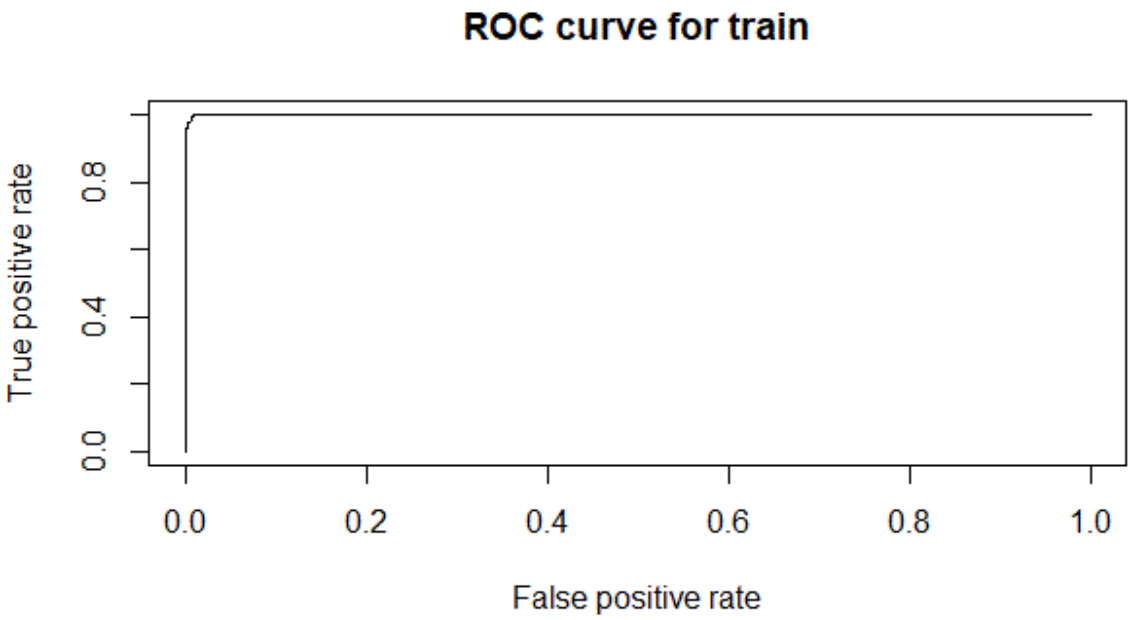


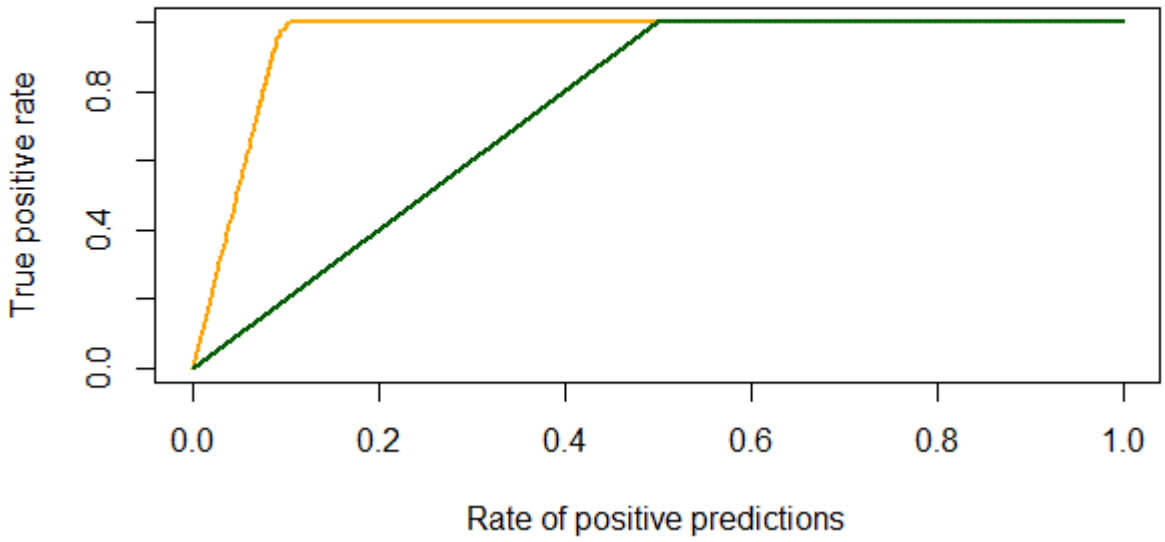
RF test


tRforest
Error
0.00
0.05
0.10
0.15
0
100
200
300
400
500
trees

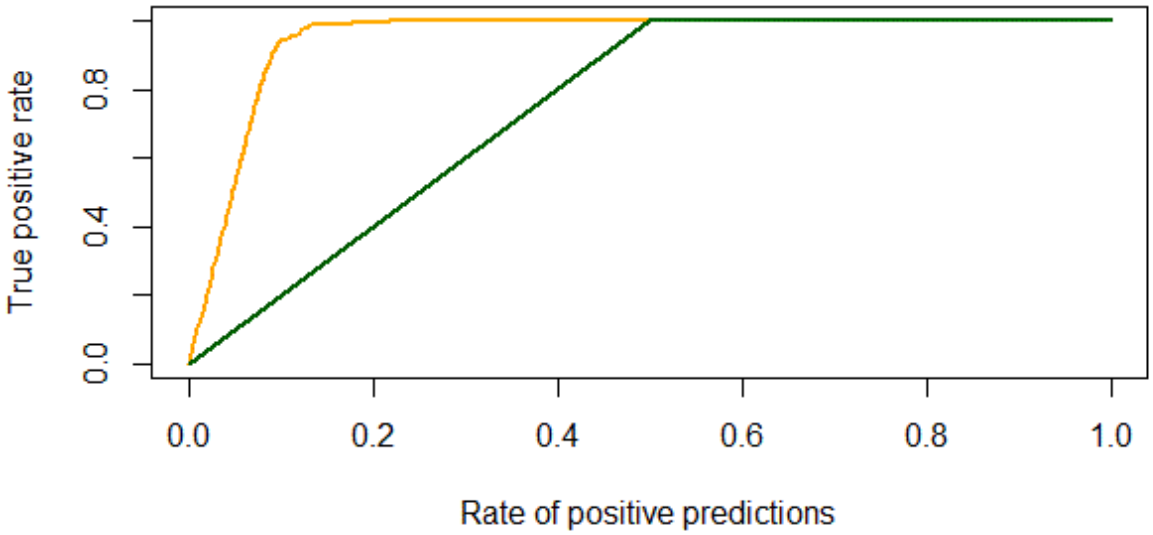

ROC curve for test
True positive rate
0.0
0.4
0.8
0.0
0.2
0.4
0.6
0.8
1.0
False positive rate

Tuned RF


ROC curve for train
True positive rate
0.0
0.4
0.8
0.0
0.2
0.4
0.6
0.8
1.0
False positive rate

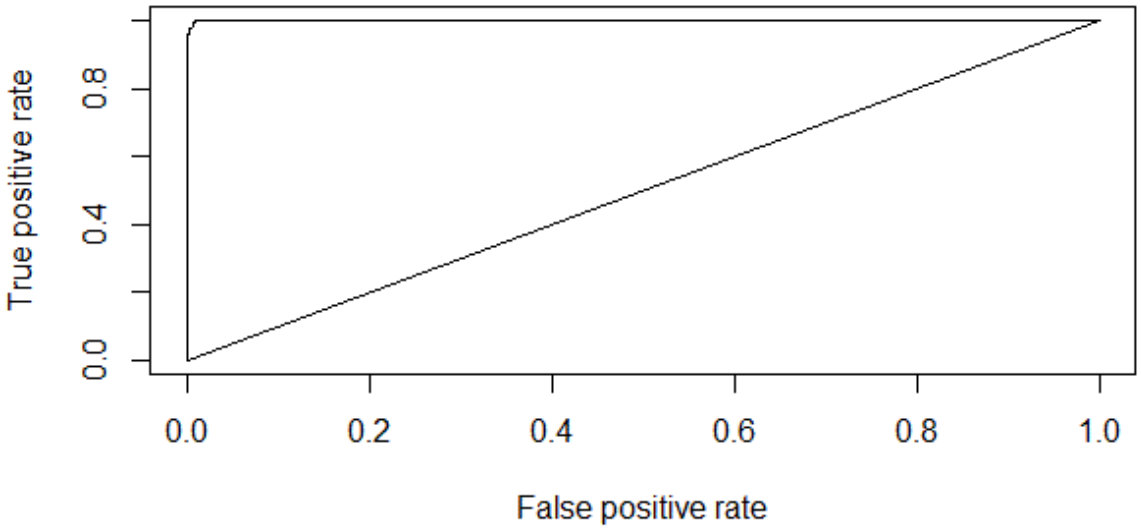

True positive rate
0.0
0.4
0.8
0.0
0.2
0.4
0.6
0.8
1.0
Rate of positive predictions

Gain Train

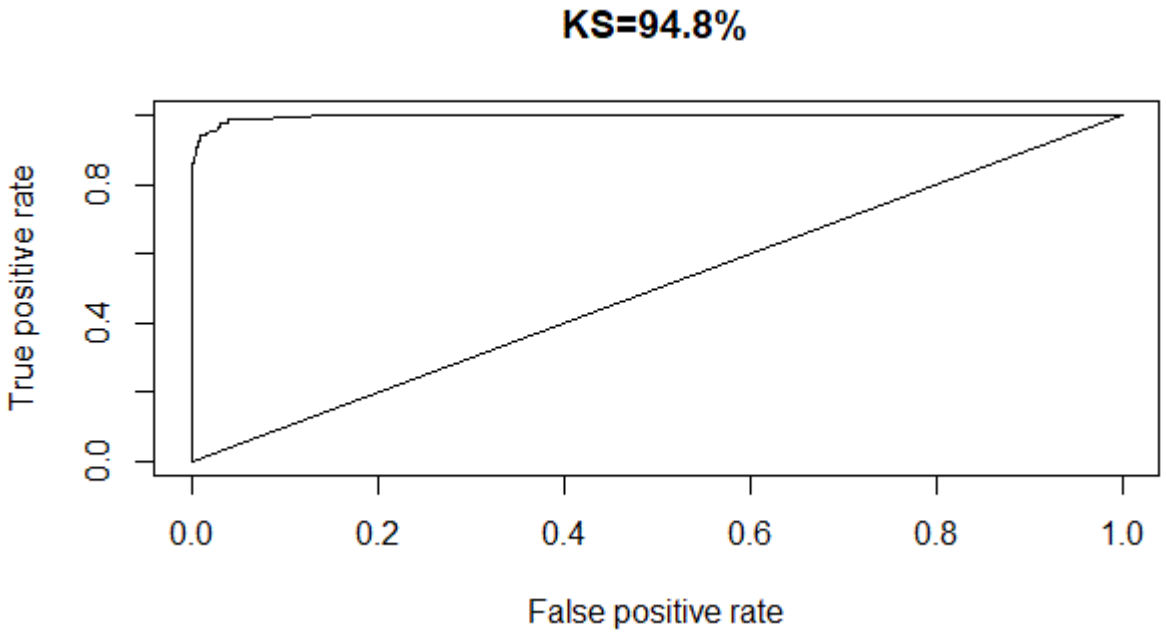


Gain Test

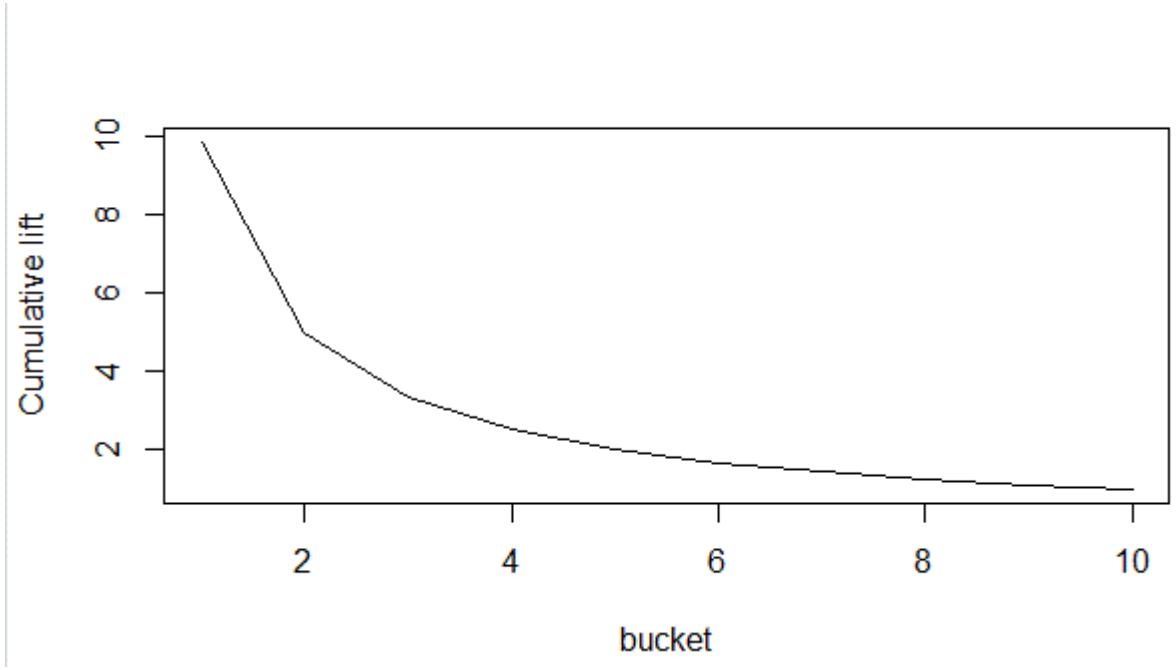
KS=99.2%



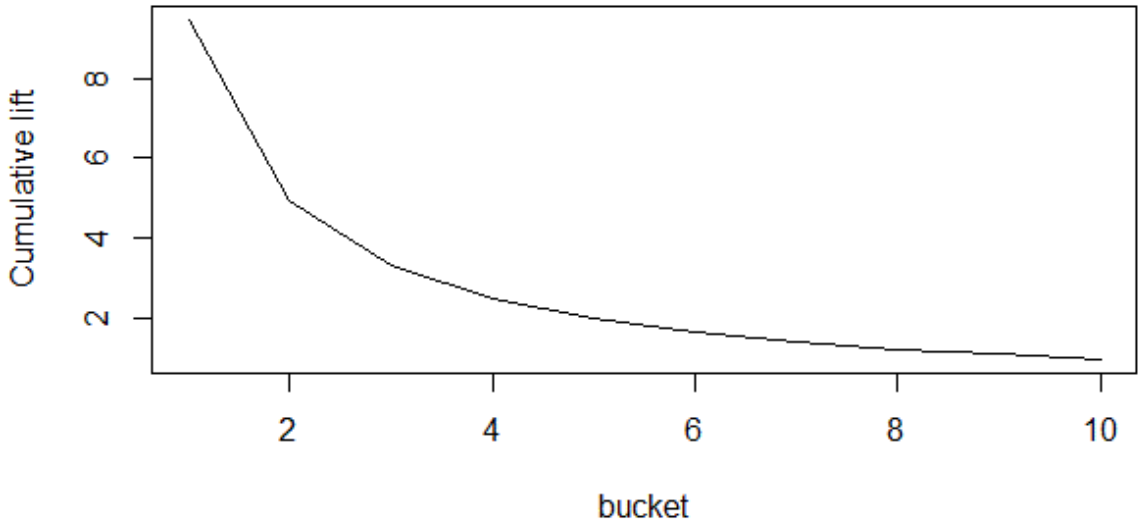
KS train



KS test



Lift Train



Lift test

6. Conclusion

	Type	CART	Random Forest
Accuracy	Train	98.01	99.36
	Test	97.89	98.55
ROC	Train	92.6	99.97
	Test	91.8	99.68
Gini	Train	77.19	90.08
	Test	76.43	89.58
Concordance	Train	85.4	99.97
	Test	82.2	99.68

Based on the above data and the plots obtained we can conclude that the model performs netter on train than test dataset. Also, RF model is relatively better than the CART model as the numbers display a better significance rate in terms of Accuracy, ROC curve, Gini and Concordance level.

5 Appendix A – Source Code

Here is the code produced in RStudio for doing an analysis on Product Service Management.

```
#====================================================================== #
> # Thera Bank - Loan Purchase Modelling  #
> #=====================================================================#
> #=====================================================================#
> #setting up workig directory
> #=====================================================================#
> Thera_Bank_dataset=read.csv('Thera_Bank_data.csv',header = TRUE)
> ##Check the dimension or shape of the data
> dim(Thera_Bank_dataset)
[1] 4982   14
>
> ##View top 5 rows
> Thera_Bank_dataset[1:5,]
  ID Age..in.years. Experience..in.years. Income..in.K.month. ZIP.Code
1  1             25                     1                  49    91107
2  2             45                    19                  34    90089
3  3             39                    15                  11    94720
4  4             35                     9                 100    94112
5  5             35                     8                  45    91330
  Family.members CCAvg Education Mortgage Personal_Loan Securities.Account
1              4   1.6         1        0             0                   1
2              3   1.5         1        0             0                   1
3              1   1.0         1        0             0                   0
4              1   2.7         2        0             0                   0
5              4   1.0         2        0             0                   0
  CD.Account Online CreditCard
1          0      0          0
2          0      0          0
3          0      0          0
4          0      0          0
5          0      0          1
>
> ##Lets see the variable list in the data
> names(Thera_Bank_dataset)
 [1] "ID"                    "Age..in.years."        "Experience..in.years."
 [4] "Income..in.K.month."   "ZIP.Code"              "Family.members"
 [7] "CCAvg"                 "Education"             "Mortgage"
[10] "Personal_Loan"         "Securities.Account"    "CD.Account"
[13] "Online"                "CreditCard"
> hist(Thera_Bank_dataset$Family.members,main="Histogram for Family Members",
+      xlab="Family count",
+      border="blue",
+      col="green",xlim=c(1,4),ylim = c(0,1500))
> hist(Thera_Bank_dataset$Education,main="Histogram for Education",
+      xlab="Education Level",
+      border="blue",
+      col="green",xlim=c(1,3),ylim = c(0,2500))
>
> ##Lets see the datatype of the data set. Can see several factor variables as factors
> str(Thera_Bank_dataset)
'data.frame':  4982 obs. of  14 variables:
 $ ID                  : int  1 2 3 4 5 6 7 8 9 10 ...
 $ 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 9
3023 ...
 $ 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 ...
> ##Convert all variables into factors where necessary. Use the dplyr package
> library(dplyr)
> Thera_Bank_dataset.scaled=scale(Thera_Bank_dataset[,-c(1,5)])
> print(Thera_Bank_dataset.scaled)
      Age..in.years. Experience..in.years. Income..in.K.month. Family.members
 [1,]     -1.77207692           -1.664059998         -0.537197228       1.3971629
 [2,]     -0.02852262           -0.095519426         -0.862999862       0.5254452
 [3,]     -0.55158891           -0.444083998         -1.362563900      -1.2179901
 [4,]     -0.90029977           -0.966930855          0.570531725      -1.2179901
 [5,]     -0.90029977           -1.054071998         -0.624077931       1.3971629
 [6,]     -0.72594434           -0.618366284         -0.971600740       1.3971629
 [7,]      0.66889911            0.601609717         -0.037633191      -0.3462724
 [8,]      0.40736596            0.340186288         -1.123641969      -1.2179901
 [9,]     -0.90029977           -0.879789712          0.157848389       0.5254452
[10,]     -0.98747748           -0.966930855          2.308145770      -1.2179901
[11,]      1.71503169            1.647303432          0.679132603       1.3971629
[12,]     -1.42336606           -1.315495427         -0.624077931       0.5254452
[13,]      0.23301053            0.253045145          0.874614183      -0.3462724
[14,]      1.19196540            1.037315431         -0.732678809       1.3971629
[15,]      1.88938712            1.821585717          0.831173832      -1.2179901
[16,]      1.27914311            0.863033146         -1.123641969      -1.2179901
[17,]     -0.63876662           -0.531225141          1.222136992       1.3971629
[18,]     -0.29005576           -0.182660569          0.157848389       1.3971629
[19,]      0.05865510            0.078762860          2.590508052      -0.3462724
[20,]      0.84325454            0.688750860         -1.145362144      -1.2179901
[21,]      1.01760997            0.601609717         -0.233114771       0.5254452
[22,]     -1.42336606           -1.315495427         -0.254834946      -1.2179901
[23,]     -0.11570033           -0.182660569         -0.667518282      -0.3462724
[24,]     -0.81312205           -0.792648569          1.699980854      -0.3462724
[25,]     -0.20287805           -0.095519426         -0.971600740       0.5254452
[26,]     -0.46441119           -0.356942855          0.201288741       1.3971629
[27,]      0.05865510           -0.008378283          1.830301908      -1.2179901
[28,]      0.93043225            0.863033146         -0.558917404      -1.2179901
[29,]     -0.63876662           -0.618366284          0.983215061      -1.2179901
[30,]      1.19196540            1.298738860         -0.841279686      -1.2179901
[31,]     -0.46441119           -0.356942855         -0.971600740      -1.2179901
[32,]      0.66889911            0.688750860         -0.710958633      -0.3462724
[33,]     -1.33618835           -1.228354284         -1.210522671       0.5254452
[34,]     -1.24901063           -1.315495427         -0.515477053       1.3971629
[35,]      0.23301053            0.340186288          0.157848389       0.5254452
[36,]      1.19196540            1.298738860          1.026655412      -1.2179901
[37,]      0.49454368            0.427327431         -0.059353366      -1.2179901
[38,]     -0.29005576           -0.182660569          1.461058923       0.5254452
[39,]     -0.63876662           -0.618366284          0.136128214       1.3971629
[40,]      1.01760997            1.037315431          0.223008916       0.5254452
[41,]     -0.98747748           -0.966930855         -0.298275297       0.5254452
[42,]     -1.16183292           -1.141213141          1.265577343       1.3971629
[43,]     -0.55158891           -0.444083998         -0.624077931      -1.2179901
[44,]      0.05865510           -0.008378283          0.657412427      -1.2179901
[45,]      1.01760997            0.950174288         -0.472036702       1.3971629
[46,]     -0.55158891           -0.531225141         -0.667518282       0.5254452
[47,]     -0.72594434           -0.705507426          2.612228228       1.3971629
[48,]      0.93043225            0.514468574          0.157848389      -0.3462724
[49,]     -0.46441119           -0.356942855         -0.537197228      -1.2179901
[50,]     -1.16183292           -1.054071998         -1.427724426       1.3971629
[51,]      1.36632083            1.473021146          1.243857168      -1.2179901
[52,]     -1.33618835           -1.228354284         -0.037633191      -1.2179901
[53,]      0.40736596            0.514468574          2.525347526       0.5254452
[54,]     -1.42336606           -1.315495427         -0.645798106      -1.2179901
[55,]     -0.37723348           -0.269801712          1.417618572      -0.3462724
[56,]      0.84325454            0.863033146         -0.971600740       0.5254452
[57,]      0.93043225            0.950174288          1.243857168      -0.3462724
[58,]     -1.24901063           -1.315495427          2.481907174      -0.3462724
[59,]      0.32018824            0.340186288         -0.754398984       0.5254452
```

```
[60,]     0.14583281          0.078762860        1.113536114     -1.2179901
[61,]    -0.29005576         -0.182660569       -1.123641969     -1.2179901
[62,]    -0.29005576         -0.269801712       -0.906440213      1.3971629
[63,]     0.14583281          0.253045145        0.679132603     -0.3462724
[64,]     1.19196540          1.298738860        1.243857168     -1.2179901
[65,]     1.45349854          1.385880003        0.679132603     -0.3462724
[66,]     0.66889911          0.253045145       -0.624077931      1.3971629
[67,]     0.14583281          0.078762860       -0.298275297      0.5254452
[68,]     0.66889911          0.775892003       -1.167082320      1.3971629
[69,]    -0.29005576         -0.182660569        0.896334359     -1.2179901
[70,]     0.66889911          0.775892003       -0.102793717      1.3971629
[71,]    -0.11570033         -0.008378283        1.222136992     -1.2179901
[72,]    -0.37723348         -0.356942855        0.244729092     -1.2179901
[73,]    -1.51054378         -1.489777712        1.330737870     -0.3462724
[74,]    -1.24901063         -1.141213141        1.330737870      1.3971629
[75,]     1.10478768          1.037315431       -1.340843724      0.5254452
[76,]     0.05865510         -0.008378283       -0.971600740      0.5254452
[77,]     0.75607682          0.863033146        1.287297519     -0.3462724
[78,]     0.40736596          0.514468574       -1.188802495     -0.3462724
[79,]     1.27914311          1.385880003       -0.710958633      1.3971629
[80,]     0.14583281          0.165904002       -0.732678809      0.5254452
[81,]    -0.37723348         -0.356942855        0.179568565     -1.2179901
[82,]    -1.07465520         -0.966930855       -0.515477053     -1.2179901
[83,]     0.05865510          0.165904002       -1.210522671     -1.2179901
             CCAvg  Education   Mortgage Personal_Loan Securities.Account
 [1,] -0.19442322 -1.0491006 -0.5557035    -0.3257401          2.9321499
 [2,] -0.25161294 -1.0491006 -0.5557035    -0.3257401          2.9321499
 [3,] -0.53756151 -1.0491006 -0.5557035    -0.3257401         -0.3409782
 [4,]  0.43466365  0.1417445 -0.5557035    -0.3257401         -0.3409782
 [5,] -0.53756151  0.1417445 -0.5557035    -0.3257401         -0.3409782
 [6,] -0.88069980  0.1417445  0.9675427    -0.3257401         -0.3409782
 [7,] -0.25161294  0.1417445 -0.5557035    -0.3257401         -0.3409782
 [8,] -0.93788952  1.3325897 -0.5557035    -0.3257401         -0.3409782
 [9,] -0.76632037  0.1417445  0.4663456    -0.3257401         -0.3409782
[10,]  3.98042599  1.3325897 -0.5557035     3.0693163         -0.3409782
[11,]  0.26309450  1.3325897 -0.5557035    -0.3257401         -0.3409782
[12,] -1.05226895  0.1417445 -0.5557035    -0.3257401         -0.3409782
[13,]  1.06375051  1.3325897 -0.5557035    -0.3257401          2.9321499
[14,]  0.32028422  0.1417445 -0.5557035    -0.3257401         -0.3409782
[15,]  0.03433564 -1.0491006 -0.5557035    -0.3257401          2.9321499
[16,] -0.25161294  1.3325897 -0.5557035    -0.3257401         -0.3409782
[17,]  1.57845795  1.3325897  0.7611674     3.0693163         -0.3409782
[18,]  0.26309450 -1.0491006 -0.5557035    -0.3257401         -0.3409782
[19,]  3.52290827  1.3325897 -0.5557035     3.0693163         -0.3409782
[20,] -0.82351009  0.1417445 -0.5557035    -0.3257401          2.9321499
[21,]  0.03433564  1.3325897 -0.5557035    -0.3257401         -0.3409782
[22,] -0.42318208 -1.0491006  1.9994191    -0.3257401         -0.3409782
[23,] -0.70913066 -1.0491006  1.0461619    -0.3257401          2.9321499
[24,]  1.12094023 -1.0491006  1.0068523    -0.3257401         -0.3409782
[25,] -0.82351009 -1.0491006  0.3975538    -0.3257401         -0.3409782
[26,] -0.99507923  1.3325897 -0.5557035    -0.3257401         -0.3409782
[27,]  0.26309450 -1.0491006 -0.5557035    -0.3257401         -0.3409782
[28,]  0.14871507  1.3325897 -0.5557035    -0.3257401         -0.3409782
[29,]  0.77780194  0.1417445 -0.5557035     3.0693163         -0.3409782
[30,] -0.42318208  1.3325897  0.6432387    -0.3257401         -0.3409782
[31,]  0.03433564  0.1417445 -0.5557035    -0.3257401         -0.3409782
[32,] -0.76632037  1.3325897  1.3409837    -0.3257401         -0.3409782
[33,] -0.59475123  1.3325897 -0.5557035    -0.3257401         -0.3409782
[34,] -0.08004379  1.3325897 -0.5557035    -0.3257401         -0.3409782
[35,] -0.70913066 -1.0491006 -0.5557035    -0.3257401         -0.3409782
[36,]  0.54904308 -1.0491006 -0.5557035    -0.3257401         -0.3409782
[37,] -0.30880265  1.3325897  1.3901207    -0.3257401         -0.3409782
[38,]  1.75002709  1.3325897 -0.5557035     3.0693163          2.9321499
[39,] -0.70913066  1.3325897  2.2451040    -0.3257401         -0.3409782
[40,] -0.19442322  1.3325897 -0.5557035    -0.3257401          2.9321499
[41,]  0.20590479 -1.0491006 -0.5557035    -0.3257401         -0.3409782
[42,] -0.48037180  0.1417445  3.4931831     3.0693163         -0.3409782
[43,] -0.70913066 -1.0491006 -0.5557035    -0.3257401         -0.3409782
[44,]  2.15035510 -1.0491006 -0.5557035    -0.3257401         -0.3409782
```

```
[45,]  0.32028422 -1.0491006 -0.5557035    -0.3257401          -0.3409782
[46,] -0.70913066  0.1417445  0.9478879    -0.3257401          -0.3409782
[47,] -0.99507923  1.3325897  1.5178768     3.0693163           2.9321499
[48,]  1.46407852  1.3325897 -0.5557035    -0.3257401          -0.3409782
[49,] -0.08004379 -1.0491006 -0.5557035    -0.3257401          -0.3409782
[50,] -0.70913066  0.1417445 -0.5557035    -0.3257401           2.9321499
[51,]  0.54904308 -1.0491006 -0.5557035    -0.3257401          -0.3409782
[52,] -1.05226895 -1.0491006  1.4785672    -0.3257401          -0.3409782
[53,]  0.09152535  1.3325897  1.8028713     3.0693163          -0.3409782
[54,] -0.99507923  1.3325897 -0.5557035    -0.3257401          -0.3409782
[55,]  3.46571855 -1.0491006 -0.5557035    -0.3257401          -0.3409782
[56,] -1.05226895  0.1417445 -0.5557035    -0.3257401           2.9321499
[57,] -0.42318208  1.3325897 -0.5557035     3.0693163          -0.3409782
[58,]  1.46407852 -1.0491006  3.9157611    -0.3257401          -0.3409782
[59,] -0.13723351  0.1417445 -0.5557035    -0.3257401           2.9321499
[60,]  2.15035510 -1.0491006  0.5449647    -0.3257401           2.9321499
[61,] -0.53756151 -1.0491006 -0.5557035    -0.3257401          -0.3409782
[62,] -1.10945866  0.1417445 -0.5557035    -0.3257401          -0.3409782
[63,]  0.77780194 -1.0491006 -0.5557035    -0.3257401          -0.3409782
[64,]  1.06375051 -1.0491006 -0.5557035    -0.3257401          -0.3409782
[65,]  0.49185336 -1.0491006  2.7463011    -0.3257401          -0.3409782
[66,]  0.03433564  1.3325897  0.7415126    -0.3257401           2.9321499
[67,]  0.09152535 -1.0491006 -0.5557035    -0.3257401          -0.3409782
[68,] -0.99507923 -1.0491006 -0.5557035    -0.3257401          -0.3409782
[69,]  0.89218137 -1.0491006 -0.5557035    -0.3257401          -0.3409782
[70,] -0.53756151  0.1417445 -0.5557035    -0.3257401          -0.3409782
[71,]  1.75002709 -1.0491006 -0.5557035    -0.3257401          -0.3409782
[72,]  1.17812994  1.3325897 -0.5557035    -0.3257401          -0.3409782
[73,]  0.77780194 -1.0491006 -0.5557035    -0.3257401          -0.3409782
[74,]  1.06375051  0.1417445 -0.5557035     3.0693163          -0.3409782
[75,] -0.93788952  1.3325897 -0.5557035    -0.3257401          -0.3409782
[76,] -0.82351009  0.1417445 -0.5557035    -0.3257401          -0.3409782
[77,]  0.37747393  1.3325897 -0.5557035     3.0693163          -0.3409782
[78,] -0.88069980 -1.0491006  0.6039291    -0.3257401          -0.3409782
[79,] -0.36599237 -1.0491006  1.1542632    -0.3257401          -0.3409782
[80,]  0.43466365  0.1417445 -0.5557035    -0.3257401          -0.3409782
[81,]  1.17812994  1.3325897 -0.5557035    -0.3257401          -0.3409782
[82,]  0.26309450  0.1417445 -0.5557035    -0.3257401          -0.3409782
[83,] -0.59475123  1.3325897 -0.5557035    -0.3257401          -0.3409782
      CD.Account     Online CreditCard
 [1,] -0.2531054 -1.2143381 -0.6453407
 [2,] -0.2531054 -1.2143381 -0.6453407
 [3,] -0.2531054 -1.2143381 -0.6453407
 [4,] -0.2531054 -1.2143381 -0.6453407
 [5,] -0.2531054 -1.2143381  1.5492581
 [6,] -0.2531054  0.8233286 -0.6453407
 [7,] -0.2531054  0.8233286 -0.6453407
 [8,] -0.2531054 -1.2143381  1.5492581
 [9,] -0.2531054  0.8233286 -0.6453407
[10,] -0.2531054 -1.2143381 -0.6453407
[11,] -0.2531054 -1.2143381 -0.6453407
[12,] -0.2531054  0.8233286 -0.6453407
[13,] -0.2531054 -1.2143381 -0.6453407
[14,] -0.2531054  0.8233286 -0.6453407
[15,] -0.2531054 -1.2143381 -0.6453407
[16,] -0.2531054  0.8233286  1.5492581
[17,] -0.2531054 -1.2143381 -0.6453407
[18,] -0.2531054 -1.2143381 -0.6453407
[19,] -0.2531054 -1.2143381 -0.6453407
[20,] -0.2531054 -1.2143381  1.5492581
[21,] -0.2531054  0.8233286 -0.6453407
[22,] -0.2531054  0.8233286 -0.6453407
[23,] -0.2531054 -1.2143381 -0.6453407
[24,] -0.2531054 -1.2143381  1.5492581
[25,] -0.2531054  0.8233286 -0.6453407
[26,] -0.2531054 -1.2143381 -0.6453407
[27,] -0.2531054  0.8233286  1.5492581
[28,] -0.2531054  0.8233286  1.5492581
[29,]  3.9501309  0.8233286  1.5492581
```

```
  [30,] -0.2531054  0.8233286 -0.6453407
  [31,] -0.2531054  0.8233286 -0.6453407
  [32,] -0.2531054 -1.2143381 -0.6453407
  [33,] -0.2531054 -1.2143381 -0.6453407
  [34,] -0.2531054  0.8233286 -0.6453407
  [35,] -0.2531054 -1.2143381 -0.6453407
  [36,] -0.2531054 -1.2143381  1.5492581
  [37,] -0.2531054 -1.2143381 -0.6453407
  [38,]  3.9501309  0.8233286 -0.6453407
  [39,] -0.2531054  0.8233286 -0.6453407
  [40,] -0.2531054 -1.2143381 -0.6453407
  [41,] -0.2531054 -1.2143381 -0.6453407
  [42,] -0.2531054  0.8233286 -0.6453407
  [43,] -0.2531054  0.8233286 -0.6453407
  [44,] -0.2531054  0.8233286  1.5492581
  [45,] -0.2531054 -1.2143381  1.5492581
  [46,] -0.2531054  0.8233286 -0.6453407
  [47,]  3.9501309  0.8233286  1.5492581
  [48,] -0.2531054 -1.2143381  1.5492581
  [49,] -0.2531054 -1.2143381  1.5492581
  [50,] -0.2531054  0.8233286 -0.6453407
  [51,] -0.2531054  0.8233286 -0.6453407
  [52,] -0.2531054 -1.2143381 -0.6453407
  [53,] -0.2531054  0.8233286 -0.6453407
  [54,] -0.2531054  0.8233286 -0.6453407
  [55,] -0.2531054  0.8233286 -0.6453407
  [56,]  3.9501309  0.8233286 -0.6453407
  [57,] -0.2531054 -1.2143381 -0.6453407
  [58,] -0.2531054 -1.2143381 -0.6453407
  [59,] -0.2531054  0.8233286 -0.6453407
  [60,] -0.2531054 -1.2143381 -0.6453407
  [61,] -0.2531054 -1.2143381 -0.6453407
  [62,] -0.2531054  0.8233286 -0.6453407
  [63,] -0.2531054 -1.2143381 -0.6453407
  [64,] -0.2531054  0.8233286  1.5492581
  [65,] -0.2531054 -1.2143381 -0.6453407
  [66,] -0.2531054 -1.2143381 -0.6453407
  [67,] -0.2531054  0.8233286  1.5492581
  [68,] -0.2531054  0.8233286 -0.6453407
  [69,] -0.2531054 -1.2143381  1.5492581
  [70,] -0.2531054  0.8233286 -0.6453407
  [71,] -0.2531054 -1.2143381  1.5492581
  [72,] -0.2531054  0.8233286  1.5492581
  [73,] -0.2531054 -1.2143381  1.5492581
  [74,]  3.9501309  0.8233286  1.5492581
  [75,] -0.2531054 -1.2143381 -0.6453407
  [76,] -0.2531054 -1.2143381 -0.6453407
  [77,] -0.2531054 -1.2143381 -0.6453407
  [78,] -0.2531054  0.8233286 -0.6453407
  [79,] -0.2531054  0.8233286  1.5492581
  [80,] -0.2531054  0.8233286 -0.6453407
  [81,] -0.2531054  0.8233286 -0.6453407
  [82,] -0.2531054 -1.2143381 -0.6453407
  [83,] -0.2531054  0.8233286 -0.6453407
 [ reached getOption("max.print") -- omitted 4899 rows ]
attr(,"scaled:center")
       Age..in.years. Experience..in.years.   Income..in.K.month.
          45.32717784            20.09614613          73.73263749
       Family.members                 CCAvg            Education
           2.39723003             1.93996186           1.88097150
            Mortgage         Personal_Loan    Securities.Account
          56.54636692             0.09594540           0.10417503
         CD.Account                 Online            CreditCard
           0.06021678             0.59594540           0.29405861
attr(,"scaled:scale")
       Age..in.years. Experience..in.years.   Income..in.K.month.
           11.4708214            11.4756356           46.0401435
       Family.members                 CCAvg            Education
            1.1471604             1.7485661            0.8397397
```

```
          Mortgage         Personal_Loan    Securities.Account
          101.7563694             0.2945459             0.3055181
          CD.Account                Online            CreditCard
           0.2379119             0.4907574             0.4556642

> Thera_Bank_dataset <- mutate_if(Thera_Bank_dataset, is.character, as.factor)
> str(Thera_Bank_dataset)
'data.frame':  4982 obs. of  14 variables:
 $ ID                    : int  1 2 3 4 5 6 7 8 9 10 ...
 $ 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 9
3023 ...
 $ 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 ...
>
> #checking na values
> is.na(Thera_Bank_dataset)
           ID Age..in.years. Experience..in.years. Income..in.K.month. ZIP.Code
   [1,] FALSE          FALSE                FALSE               FALSE    FALSE
   [2,] FALSE          FALSE                FALSE               FALSE    FALSE
   [3,] FALSE          FALSE                FALSE               FALSE    FALSE
   [4,] FALSE          FALSE                FALSE               FALSE    FALSE
   [5,] FALSE          FALSE                FALSE               FALSE    FALSE
   [6,] FALSE          FALSE                FALSE               FALSE    FALSE
   [7,] FALSE          FALSE                FALSE               FALSE    FALSE
   [8,] FALSE          FALSE                FALSE               FALSE    FALSE
   [9,] FALSE          FALSE                FALSE               FALSE    FALSE
  [10,] FALSE          FALSE                FALSE               FALSE    FALSE
  [11,] FALSE          FALSE                FALSE               FALSE    FALSE
  [12,] FALSE          FALSE                FALSE               FALSE    FALSE
  [13,] FALSE          FALSE                FALSE               FALSE    FALSE
  [14,] FALSE          FALSE                FALSE               FALSE    FALSE
  [15,] FALSE          FALSE                FALSE               FALSE    FALSE
  [16,] FALSE          FALSE                FALSE               FALSE    FALSE
  [17,] FALSE          FALSE                FALSE               FALSE    FALSE
  [18,] FALSE          FALSE                FALSE               FALSE    FALSE
  [19,] FALSE          FALSE                FALSE               FALSE    FALSE
  [20,] FALSE          FALSE                FALSE               FALSE    FALSE
  [21,] FALSE          FALSE                FALSE               FALSE    FALSE
  [22,] FALSE          FALSE                FALSE               FALSE    FALSE
  [23,] FALSE          FALSE                FALSE               FALSE    FALSE
  [24,] FALSE          FALSE                FALSE               FALSE    FALSE
  [25,] FALSE          FALSE                FALSE               FALSE    FALSE
  [26,] FALSE          FALSE                FALSE               FALSE    FALSE
  [27,] FALSE          FALSE                FALSE               FALSE    FALSE
  [28,] FALSE          FALSE                FALSE               FALSE    FALSE
  [29,] FALSE          FALSE                FALSE               FALSE    FALSE
  [30,] FALSE          FALSE                FALSE               FALSE    FALSE
  [31,] FALSE          FALSE                FALSE               FALSE    FALSE
  [32,] FALSE          FALSE                FALSE               FALSE    FALSE
  [33,] FALSE          FALSE                FALSE               FALSE    FALSE
  [34,] FALSE          FALSE                FALSE               FALSE    FALSE
```

```
[35,] FALSE          FALSE                FALSE              FALSE    FALSE
[36,] FALSE          FALSE                FALSE              FALSE    FALSE
[37,] FALSE          FALSE                FALSE              FALSE    FALSE
[38,] FALSE          FALSE                FALSE              FALSE    FALSE
[39,] FALSE          FALSE                FALSE              FALSE    FALSE
[40,] FALSE          FALSE                FALSE              FALSE    FALSE
[41,] FALSE          FALSE                FALSE              FALSE    FALSE
[42,] FALSE          FALSE                FALSE              FALSE    FALSE
[43,] FALSE          FALSE                FALSE              FALSE    FALSE
[44,] FALSE          FALSE                FALSE              FALSE    FALSE
[45,] FALSE          FALSE                FALSE              FALSE    FALSE
[46,] FALSE          FALSE                FALSE              FALSE    FALSE
[47,] FALSE          FALSE                FALSE              FALSE    FALSE
[48,] FALSE          FALSE                FALSE              FALSE    FALSE
[49,] FALSE          FALSE                FALSE              FALSE    FALSE
[50,] FALSE          FALSE                FALSE              FALSE    FALSE
[51,] FALSE          FALSE                FALSE              FALSE    FALSE
[52,] FALSE          FALSE                FALSE              FALSE    FALSE
[53,] FALSE          FALSE                FALSE              FALSE    FALSE
[54,] FALSE          FALSE                FALSE              FALSE    FALSE
[55,] FALSE          FALSE                FALSE              FALSE    FALSE
[56,] FALSE          FALSE                FALSE              FALSE    FALSE
[57,] FALSE          FALSE                FALSE              FALSE    FALSE
[58,] FALSE          FALSE                FALSE              FALSE    FALSE
[59,] FALSE          FALSE                FALSE              FALSE    FALSE
[60,] FALSE          FALSE                FALSE              FALSE    FALSE
[61,] FALSE          FALSE                FALSE              FALSE    FALSE
[62,] FALSE          FALSE                FALSE              FALSE    FALSE
[63,] FALSE          FALSE                FALSE              FALSE    FALSE
[64,] FALSE          FALSE                FALSE              FALSE    FALSE
[65,] FALSE          FALSE                FALSE              FALSE    FALSE
[66,] FALSE          FALSE                FALSE              FALSE    FALSE
[67,] FALSE          FALSE                FALSE              FALSE    FALSE
[68,] FALSE          FALSE                FALSE              FALSE    FALSE
[69,] FALSE          FALSE                FALSE              FALSE    FALSE
[70,] FALSE          FALSE                FALSE              FALSE    FALSE
[71,] FALSE          FALSE                FALSE              FALSE    FALSE
      Family.members CCAvg Education Mortgage Personal_Loan Securities.Account
 [1,]          FALSE FALSE     FALSE    FALSE         FALSE              FALSE
 [2,]          FALSE FALSE     FALSE    FALSE         FALSE              FALSE
 [3,]          FALSE FALSE     FALSE    FALSE         FALSE              FALSE
 [4,]          FALSE FALSE     FALSE    FALSE         FALSE              FALSE
 [5,]          FALSE FALSE     FALSE    FALSE         FALSE              FALSE
 [6,]          FALSE FALSE     FALSE    FALSE         FALSE              FALSE
 [7,]          FALSE FALSE     FALSE    FALSE         FALSE              FALSE
 [8,]          FALSE FALSE     FALSE    FALSE         FALSE              FALSE
 [9,]          FALSE FALSE     FALSE    FALSE         FALSE              FALSE
[10,]          FALSE FALSE     FALSE    FALSE         FALSE              FALSE
[11,]          FALSE FALSE     FALSE    FALSE         FALSE              FALSE
[12,]          FALSE FALSE     FALSE    FALSE         FALSE              FALSE
[13,]          FALSE FALSE     FALSE    FALSE         FALSE              FALSE
[14,]          FALSE FALSE     FALSE    FALSE         FALSE              FALSE
[15,]          FALSE FALSE     FALSE    FALSE         FALSE              FALSE
[16,]          FALSE FALSE     FALSE    FALSE         FALSE              FALSE
[17,]          FALSE FALSE     FALSE    FALSE         FALSE              FALSE
[18,]          FALSE FALSE     FALSE    FALSE         FALSE              FALSE
[19,]          FALSE FALSE     FALSE    FALSE         FALSE              FALSE
[20,]          FALSE FALSE     FALSE    FALSE         FALSE              FALSE
[21,]          FALSE FALSE     FALSE    FALSE         FALSE              FALSE
[22,]          FALSE FALSE     FALSE    FALSE         FALSE              FALSE
[23,]          FALSE FALSE     FALSE    FALSE         FALSE              FALSE
```

```
[24,]          FALSE FALSE     FALSE    FALSE          FALSE                FALSE
[25,]          FALSE FALSE     FALSE    FALSE          FALSE                FALSE
[26,]          FALSE FALSE     FALSE    FALSE          FALSE                FALSE
[27,]          FALSE FALSE     FALSE    FALSE          FALSE                FALSE
[28,]          FALSE FALSE     FALSE    FALSE          FALSE                FALSE
[29,]          FALSE FALSE     FALSE    FALSE          FALSE                FALSE
[30,]          FALSE FALSE     FALSE    FALSE          FALSE                FALSE
[31,]          FALSE FALSE     FALSE    FALSE          FALSE                FALSE
[32,]          FALSE FALSE     FALSE    FALSE          FALSE                FALSE
[33,]          FALSE FALSE     FALSE    FALSE          FALSE                FALSE
[34,]          FALSE FALSE     FALSE    FALSE          FALSE                FALSE
[35,]          FALSE FALSE     FALSE    FALSE          FALSE                FALSE
[36,]          FALSE FALSE     FALSE    FALSE          FALSE                FALSE
[37,]          FALSE FALSE     FALSE    FALSE          FALSE                FALSE
[38,]          FALSE FALSE     FALSE    FALSE          FALSE                FALSE
[39,]          FALSE FALSE     FALSE    FALSE          FALSE                FALSE
[40,]          FALSE FALSE     FALSE    FALSE          FALSE                FALSE
[41,]          FALSE FALSE     FALSE    FALSE          FALSE                FALSE
[42,]          FALSE FALSE     FALSE    FALSE          FALSE                FALSE
[43,]          FALSE FALSE     FALSE    FALSE          FALSE                FALSE
[44,]          FALSE FALSE     FALSE    FALSE          FALSE                FALSE
[45,]          FALSE FALSE     FALSE    FALSE          FALSE                FALSE
[46,]          FALSE FALSE     FALSE    FALSE          FALSE                FALSE
[47,]          FALSE FALSE     FALSE    FALSE          FALSE                FALSE
[48,]          FALSE FALSE     FALSE    FALSE          FALSE                FALSE
[49,]          FALSE FALSE     FALSE    FALSE          FALSE                FALSE
[50,]          FALSE FALSE     FALSE    FALSE          FALSE                FALSE
[51,]          FALSE FALSE     FALSE    FALSE          FALSE                FALSE
[52,]          FALSE FALSE     FALSE    FALSE          FALSE                FALSE
[53,]          FALSE FALSE     FALSE    FALSE          FALSE                FALSE
[54,]          FALSE FALSE     FALSE    FALSE          FALSE                FALSE
[55,]          FALSE FALSE     FALSE    FALSE          FALSE                FALSE
[56,]          FALSE FALSE     FALSE    FALSE          FALSE                FALSE
[57,]          FALSE FALSE     FALSE    FALSE          FALSE                FALSE
[58,]          FALSE FALSE     FALSE    FALSE          FALSE                FALSE
[59,]          FALSE FALSE     FALSE    FALSE          FALSE                FALSE
[60,]          FALSE FALSE     FALSE    FALSE          FALSE                FALSE
[61,]          FALSE FALSE     FALSE    FALSE          FALSE                FALSE
[62,]          FALSE FALSE     FALSE    FALSE          FALSE                FALSE
[63,]          FALSE FALSE     FALSE    FALSE          FALSE                FALSE
[64,]          FALSE FALSE     FALSE    FALSE          FALSE                FALSE
[65,]          FALSE FALSE     FALSE    FALSE          FALSE                FALSE
[66,]          FALSE FALSE     FALSE    FALSE          FALSE                FALSE
[67,]          FALSE FALSE     FALSE    FALSE          FALSE                FALSE
[68,]          FALSE FALSE     FALSE    FALSE          FALSE                FALSE
[69,]          FALSE FALSE     FALSE    FALSE          FALSE                FALSE
[70,]          FALSE FALSE     FALSE    FALSE          FALSE                FALSE
[71,]          FALSE FALSE     FALSE    FALSE          FALSE                FALSE
      CD.Account Online CreditCard
 [1,]      FALSE  FALSE      FALSE
 [2,]      FALSE  FALSE      FALSE
 [3,]      FALSE  FALSE      FALSE
 [4,]      FALSE  FALSE      FALSE
 [5,]      FALSE  FALSE      FALSE
 [6,]      FALSE  FALSE      FALSE
 [7,]      FALSE  FALSE      FALSE
 [8,]      FALSE  FALSE      FALSE
 [9,]      FALSE  FALSE      FALSE
[10,]      FALSE  FALSE      FALSE
[11,]      FALSE  FALSE      FALSE
[12,]      FALSE  FALSE      FALSE
```

```
 [13,]       FALSE  FALSE       FALSE
 [14,]       FALSE  FALSE       FALSE
 [15,]       FALSE  FALSE       FALSE
 [16,]       FALSE  FALSE       FALSE
 [17,]       FALSE  FALSE       FALSE
 [18,]       FALSE  FALSE       FALSE
 [19,]       FALSE  FALSE       FALSE
 [20,]       FALSE  FALSE       FALSE
 [21,]       FALSE  FALSE       FALSE
 [22,]       FALSE  FALSE       FALSE
 [23,]       FALSE  FALSE       FALSE
 [24,]       FALSE  FALSE       FALSE
 [25,]       FALSE  FALSE       FALSE
 [26,]       FALSE  FALSE       FALSE
 [27,]       FALSE  FALSE       FALSE
 [28,]       FALSE  FALSE       FALSE
 [29,]       FALSE  FALSE       FALSE
 [30,]       FALSE  FALSE       FALSE
 [31,]       FALSE  FALSE       FALSE
 [32,]       FALSE  FALSE       FALSE
 [33,]       FALSE  FALSE       FALSE
 [34,]       FALSE  FALSE       FALSE
 [35,]       FALSE  FALSE       FALSE
 [36,]       FALSE  FALSE       FALSE
 [37,]       FALSE  FALSE       FALSE
 [38,]       FALSE  FALSE       FALSE
 [39,]       FALSE  FALSE       FALSE
 [40,]       FALSE  FALSE       FALSE
 [41,]       FALSE  FALSE       FALSE
 [42,]       FALSE  FALSE       FALSE
 [43,]       FALSE  FALSE       FALSE
 [44,]       FALSE  FALSE       FALSE
 [45,]       FALSE  FALSE       FALSE
 [46,]       FALSE  FALSE       FALSE
 [47,]       FALSE  FALSE       FALSE
 [48,]       FALSE  FALSE       FALSE
 [49,]       FALSE  FALSE       FALSE
 [50,]       FALSE  FALSE       FALSE
 [51,]       FALSE  FALSE       FALSE
 [52,]       FALSE  FALSE       FALSE
 [53,]       FALSE  FALSE       FALSE
 [54,]       FALSE  FALSE       FALSE
 [55,]       FALSE  FALSE       FALSE
 [56,]       FALSE  FALSE       FALSE
 [57,]       FALSE  FALSE       FALSE
 [58,]       FALSE  FALSE       FALSE
 [59,]       FALSE  FALSE       FALSE
 [60,]       FALSE  FALSE       FALSE
 [61,]       FALSE  FALSE       FALSE
 [62,]       FALSE  FALSE       FALSE
 [63,]       FALSE  FALSE       FALSE
 [64,]       FALSE  FALSE       FALSE
 [65,]       FALSE  FALSE       FALSE
 [66,]       FALSE  FALSE       FALSE
 [67,]       FALSE  FALSE       FALSE
 [68,]       FALSE  FALSE       FALSE
 [69,]       FALSE  FALSE       FALSE
 [70,]       FALSE  FALSE       FALSE
 [71,]       FALSE  FALSE       FALSE
 [ reached getOption("max.print") -- omitted 4911 rows ]
> seed=1234
```

```
> set.seed(seed)
>
> clust2=kmeans(x=Thera_Bank_dataset.scaled,centers=2,nstart=5)#5 times iteration will
be performed
> print(clust2)
K-means clustering with 2 clusters of sizes 916, 4066

Cluster means:
  Age..in.years. Experience..in.years. Income..in.K.month. Family.members
1    -0.12635962           -0.10864768           1.5584203    -0.27204319
2     0.02846665            0.02447646          -0.3510853     0.06128666
       CCAvg   Education   Mortgage Personal_Loan Securities.Account CD.Account
1  1.4496812 -0.25347047  0.4463046     1.4459160         0.07709601  0.5912128
2 -0.3265883  0.05710255 -0.1005448    -0.3257401        -0.01736841 -0.1331901
        Online   CreditCard
1  0.024723416  0.025497386
2 -0.005569761 -0.005744123

Clustering vector:
   [1] 2 2 2 2 2 2 2 2 2 1 2 2 2 2 2 2 1 2 1 2 2 2 2 1 2 2 1 2 1 2 2 2 2 2 2 2 2 1 2
  [40] 2 2 1 2 1 2 2 1 2 2 2 2 2 1 2 1 2 1 1 2 1 2 2 2 1 2 2 2 2 2 2 1 2 1 1 2 2 1 2
  [79] 2 2 2 2 2 2 2 2 2 2 2 1 2 2 2 2 2 2 2 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 1
 [118] 2 2 2 2 2 2 2 2 2 2 2 2 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 1 2 2 2 2 2 1 1 2 2 2 2
 [157] 2 1 2 2 2 2 2 2 2 2 2 2 2 1 2 1 2 2 2 2 2 2 2 2 2 1 2 2 2 2 1 1 2 2 2 2 2 1 2
 [196] 1 2 2 2 2 2 2 2 2 2 2 1 2 2 2 2 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 1 2 2 2 2 2 2 2
 [235] 2 2 2 2 2 1 2 2 2 2 1 2 2 2 1 2 2 1 2 2 2 2 2 2 2 1 2 2 2 2 2 2 2 2 2 2 2 2 2
 [274] 2 1 2 2 2 2 2 2 2 2 2 1 2 2 2 2 2 2 2 2 2 1 2 1 1 1 2 2 2 2 2 1 2 2 2 1 2 2 2
 [313] 2 2 2 2 1 1 1 1 2 2 2 2 2 1 2 2 2 2 2 2 1 2 2 2 1 2 2 2 2 2 2 2 2 2 2 1 1 2 1
 [352] 2 2 1 2 2 1 2 2 2 2 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 1 2 2 2 2 2 2 1 1 2
 [391] 2 2 2 2 2 1 2 2 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 1 1 1 2 2 2 2 2 2
 [430] 2 2 2 2 1 2 2 2 2 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 1 2 2 2 1 1 1 2 2
 [469] 2 2 2 2 2 1 2 2 2 1 2 2 2 2 2 2 2 2 2 2 1 2 2 2 2 2 2 2 2 2 2 1 2 2 2 2 1 2 2
 [508] 2 2 2 2 2 2 2 2 2 2 2 2 2 1 2 2 2 2 2 2 2 2 2 2 1 2 2 2 2 1 2 2 2 1 2 2 2 2 2
 [547] 2 2 1 2 2 2 2 2 2 2 2 2 2 2 1 2 2 2 2 1 2 2 2 2 2 2 2 1 2 2 2 2 2 2 1 2 2 1 2
 [586] 2 2 2 2 2 1 1 2 2 2 2 2 2 2 2 2 2 1 2 2 2 2 2 2 1 1 2 2 2 2 2 1 2 2 2 2 2 2 2
 [625] 2 2 1 2 2 2 1 2 2 2 2 2 1 2 2 2 2 2 2 2 2 2 1 2 2 2 2 2 1 2 2 2 2 2 2 1 2 1 2
 [664] 2 2 1 2 2 2 2 2 2 1 2 2 2 2 1 2 2 1 2 2 2 2 2 2 2 2 2 2 1 2 2 2 2 2 2 2 1 1 1
 [703] 2 2 2 2 2 2 2 2 2 2 2 2 2 1 2 2 1 2 2 1 2 2 1 2 2 2 2 2 2 1 1 2 2 1 2 2 2 1 2
 [742] 2 2 1 2 2 2 1 2 2 2 2 2 2 2 2 2 2 2 1 2 2 2 2 1 2 2 1 2 2 1 2 2 1 2 1 1 1 1 1
 [781] 2 2 2 2 2 2 2 2 2 2 2 2 2 1 2 2 2 2 2 1 2 1 2 2 2 2 2 1 2 2 2 2 2 2 2 2 2 2 2
 [820] 2 2 2 2 2 2 2 2 1 1 2 2 2 2 2 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 1 2 2 2 2
 [859] 2 2 1 2 2 2 2 2 2 2 2 2 2 2 2 2 1 1 2 2 2 2 2 2 1 2 2 2 1 2 2 2 2 1 1 2 2 1 2
 [898] 2 1 2 2 2 1 2 2 2 2 2 1 1 2 1 2 2 2 2 2 2 2 2 2 2 1 2 2 2 2 2 2 2 2 1 2 2 2 2
 [937] 2 2 2 2 2 2 2 2 1 1 2 1 2 2 2 2 2 2 1 2 2 2 2 2 1 2 2 2 2 2 1 2 2 2 2 1 2 2 2
 [976] 2 2 1 2 2 2 2 2 2 2 2 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 1
 [ reached getOption("max.print") -- omitted 3982 entries ]

Within cluster sum of squares by cluster:
[1] 15556.42 35974.24
 (between_SS / total_SS =  13.8 %)

Available components:

[1] "cluster"      "centers"      "totss"        "withinss"     "tot.withinss"
[6] "betweenss"    "size"         "iter"         "ifault"
>
> library(cluster)
> clusplot(Thera_Bank_dataset.scaled,clust2$cluster,color=TRUE,shade = TRUE,label = 2,
lines = 1)

#to find right no. of cluster
```

```
> totWss=rep(0,5)
> for (k in 1:5) {
+   set.seed(seed)
+   clust=kmeans(x=Thera_Bank_dataset.scaled,centers=k,nstart=5)
+   totWss[k]=clust$tot.withinss
+ }
> print(totWss)
[1] 59772.00 51530.66 45198.95 40572.55 37820.73
> plot(c(1:5),totWss,type='b')#plot(x,y,type)
> library(NbClust)
> set.seed(seed)
> nc=NbClust(Thera_Bank_dataset[,-c(1,5)],min.nc=4,max.nc=8,method='kmeans')#nc means
no. of clusters
*** : The Hubert index is a graphical method of determining the number of clusters.
                In the plot of Hubert index, we seek a significant knee that correspon
ds to a
                significant increase of the value of the measure i.e the significant p
eak in Hubert
                index second differences plot.

*** : The D index is a graphical method of determining the number of clusters.
                In the plot of D index, we seek a significant knee (the significant pe
ak in Dindex
                second differences plot) that corresponds to a significant increase of
the value of
                the measure.

*******************************************************************
* Among all indices:
* 7 proposed 4 as the best number of clusters
* 12 proposed 5 as the best number of clusters
* 2 proposed 6 as the best number of clusters
* 1 proposed 7 as the best number of clusters
* 1 proposed 8 as the best number of clusters

                   ***** Conclusion *****

* According to the majority rule, the best number of clusters is  5


*******************************************************************
> table(nc$Best.n[1,])# votes for no. of cluster

 0  3  4  5  6  7  8
 2  1  7 12  2  1  1
> #now perform using 5 clusters
> seed=1234
> set.seed(seed)
> clust3=kmeans(x=Thera_Bank_dataset.scaled,centers=5,nstart=5)#5 times iteration will
be performed
> print(clust3)
K-means clustering with 5 clusters of sizes 1703, 441, 1790, 478, 570

Cluster means:
  Age..in.years. Experience..in.years. Income..in.K.month. Family.members
1    -0.88222946          -0.883832079          -0.3769041    0.176348666
2     0.01259522           0.008417629          -0.2754223    0.114294949
3     0.88796355           0.879536412          -0.4355007    0.008745544
4    -0.02378073          -0.023144879           1.5416689    0.188064961
5    -0.14246542          -0.108514158           1.4139604   -0.800483202
       CCAvg   Education   Mortgage Personal_Loan Securities.Account CD.Account
```

```
1 -0.3418840  0.10398430 -0.1057669    -0.3257401        -0.34097822 -0.1914019
2 -0.2189850 -0.04187787 -0.0879997    -0.3257401         2.93214993  0.6523537
3 -0.3745868  0.11845983 -0.1153813    -0.3257401        -0.34097822 -0.1920528
4  1.1259294  0.41827969  0.4394950     3.0693163         0.06987468  0.9691746
5  1.4230123 -1.00104899  0.3778645    -0.3257401        -0.23761628 -0.1424939
       Online  CreditCard
1 -0.04055594 -0.00745083
2  0.01935126 -0.02826528
3  0.03672095  0.02529929
4  0.01764030  0.00661127
5 -0.02391177 -0.04086346

Clustering vector:
   [1] 2 2 1 1 1 1 3 3 1 4 3 1 2 3 2 3 4 1 4 2 3 1 2 5 1 1 5 3 4 3 1 3 1 1 3 5 3 4 1
  [40] 2 1 4 1 5 3 1 4 3 1 2 5 1 4 1 5 2 4 5 2 5 1 1 5 5 5 2 3 3 5 3 5 1 5 4 3 1 4 3
  [79] 3 3 1 1 3 1 1 3 3 1 4 1 1 2 3 1 5 5 3 3 3 3 1 3 2 1 1 1 1 1 3 1 3 1 3 3 3 1 5
 [118] 3 3 3 2 1 3 1 1 2 1 1 4 1 1 3 3 3 3 2 3 3 1 1 1 3 5 3 3 5 3 5 4 3 2 3 1 1 1 1
 [157] 2 4 1 1 2 1 1 1 3 1 1 2 5 3 4 2 3 1 3 3 3 1 1 4 2 1 3 4 5 3 5 3 3 3 5 2 5 3 2
 [196] 4 1 1 1 3 3 1 2 1 1 4 3 1 3 5 3 1 1 1 1 3 1 1 1 3 3 1 1 5 2 3 3 1 1 3 1 1 3 3
 [235] 1 3 3 5 4 1 1 1 4 3 1 1 4 3 5 4 3 1 3 1 3 3 4 3 1 1 2 3 5 3 1 3 1 1 1 2 2 1 1
 [274] 3 5 1 3 1 2 1 1 3 1 4 5 1 1 1 1 3 1 3 5 4 2 5 5 4 3 3 3 1 5 2 3 5 2 1 3 1 4 4
 [313] 1 3 3 4 4 4 4 3 3 5 3 5 3 1 3 5 3 3 1 3 5 1 3 1 5 1 3 2 1 2 4 4 2 4 3 3 5 1 3
 [352] 1 5 1 1 5 3 1 3 4 3 1 3 1 1 3 3 3 5 1 1 1 3 1 3 3 4 1 3 1 1 1 4 5 3 3 3 2 1 3
 [391] 3 1 3 1 4 1 5 3 3 2 5 3 3 3 5 2 1 1 3 1 1 3 1 3 3 4 2 5 3 1 1 1 3 1 3 1 5 3 1
 [430] 3 3 2 4 3 3 2 4 3 3 3 3 3 1 3 3 1 1 2 3 1 3 1 3 5 3 2 4 4 4 3 1 3 1 3 1 3 1 4
 [469] 5 4 2 3 5 5 3 1 4 1 1 3 2 1 3 1 2 3 5 5 1 3 3 1 3 3 3 1 1 1 4 3 3 3 5 3 1 1 1
 [508] 1 1 3 3 1 1 3 3 5 3 1 3 5 1 4 1 3 2 3 1 2 3 1 4 1 3 1 5 1 3 5 1 2 2 3 3 3 3 1
 [547] 3 5 1 3 5 2 3 1 3 1 3 1 3 4 1 1 1 4 1 5 3 1 3 1 5 1 3 3 1 1 1 5 1 1 5 1 5 1 1
 [586] 2 1 3 1 4 5 3 1 3 3 1 3 1 3 1 4 1 1 3 3 3 2 5 5 1 1 3 3 4 1 5 1 1 3 1 1 3 1 1
 [625] 1 4 3 2 3 5 3 1 3 2 5 3 5 3 1 3 2 3 1 4 1 5 3 4 3 1 1 3 3 5 3 4 1 3 2 2 2 3 3
 [664] 1 4 3 1 3 1 3 5 3 3 3 4 3 5 5 1 1 3 1 2 3 1 1 1 5 1 3 1 3 1 1 1 4 5 5 3 5 3 1
 [703] 1 1 2 1 1 3 2 1 3 3 2 3 4 3 3 5 3 3 4 3 5 1 1 3 3 1 4 4 1 3 5 3 1 2 2 1 3 2 1
 [742] 3 5 3 3 3 1 3 3 2 2 3 1 3 1 3 1 4 5 3 1 2 4 2 4 5 3 3 4 3 3 4 1 4 4 5 5 5 2 4
 [781] 3 1 3 3 1 1 3 3 1 1 1 1 5 3 1 3 3 4 3 5 3 3 1 3 1 4 1 3 3 1 3 3 5 1 3 1 2 1 3
 [820] 3 1 3 1 3 1 3 4 5 1 1 1 1 1 2 1 3 3 1 5 1 3 2 3 2 1 1 2 3 3 3 1 4 3 2 3 3 1 3
 [859] 1 3 2 3 1 3 1 1 1 3 1 1 1 3 3 5 5 3 1 1 3 5 4 1 3 4 1 3 1 2 4 3 3 4 1 3 3 5 2
 [898] 3 5 3 3 5 3 3 1 2 4 5 5 5 1 2 1 1 2 3 2 1 1 4 1 3 1 1 2 3 3 5 2 2 2 3 4 4 3 1
 [937] 1 3 1 3 1 1 1 4 5 2 4 3 1 3 3 3 5 1 5 1 2 4 3 3 3 1 5 4 1 5 3 4 2 3 3 3 1 4 3
 [976] 5 3 5 5 3 3 1 1 1 1 4 1 1 1 1 5 3 2 3 3 1 3 2 3 4
 [ reached getOption("max.print") -- omitted 3982 entries ]

Within cluster sum of squares by cluster:
[1] 10238.422  4698.103 10670.780  7173.917  5039.513
 (between_SS / total_SS =  36.7 %)

Available components:

[1] "cluster"      "centers"      "totss"        "withinss"     "tot.withinss"
[6] "betweenss"    "size"         "iter"         "ifault"
> clusplot(Thera_Bank_dataset.scaled,clust3$cluster,color=TRUE,shade = TRUE,label = 2,
lines = 1)
> Thera_Bank_dataset$cluster=clust3$cluster
> print(Thera_Bank_dataset)
   ID Age..in.years. Experience..in.years. Income..in.K.month. ZIP.Code
1   1             25                     1                  49    91107
2   2             45                    19                  34    90089
3   3             39                    15                  11    94720
4   4             35                     9                 100    94112
5   5             35                     8                  45    91330
6   6             37                    13                  29    92121
7   7             53                    27                  72    91711
```

```
8   8              50                  24                  22    93943
9   9              35                  10                  81    90089
10 10              34                   9                 180    93023
11 11              65                  39                 105    94710
12 12              29                   5                  45    90277
13 13              48                  23                 114    93106
14 14              59                  32                  40    94920
15 15              67                  41                 112    91741
16 16              60                  30                  22    95054
17 17              38                  14                 130    95010
18 18              42                  18                  81    94305
19 19              46                  21                 193    91604
20 20              55                  28                  21    94720
21 22              57                  27                  63    90095
22 23              29                   5                  62    90277
23 24              44                  18                  43    91320
24 25              36                  11                 152    95521
25 26              43                  19                  29    94305
26 27              40                  16                  83    95064
27 28              46                  20                 158    90064
28 29              56                  30                  48    94539
29 30              38                  13                 119    94104
30 31              59                  35                  35    93106
31 32              40                  16                  29    94117
32 33              53                  28                  41    94801
33 34              30                   6                  18    91330
34 35              31                   5                  50    94035
35 36              48                  24                  81    92647
36 37              59                  35                 121    94720
37 38              51                  25                  71    95814
38 39              42                  18                 141    94114
39 40              38                  13                  80    94115
40 41              57                  32                  84    92672
41 42              34                   9                  60    94122
42 43              32                   7                 132    90019
43 44              39                  15                  45    95616
44 45              46                  20                 104    94065
45 46              57                  31                  52    94720
46 47              39                  14                  43    95014
47 48              37                  12                 194    91380
48 49              56                  26                  81    95747
49 50              40                  16                  49    92373
50 51              32                   8                   8    92093
51 52              61                  37                 131    94720
52 53              30                   6                  72    94005
53 54              50                  26                 190    90245
54 55              29                   5                  44    95819
55 56              41                  17                 139    94022
56 57              55                  30                  29    94005
57 58              56                  31                 131    95616
58 60              31                   5                 188    91320
59 61              49                  24                  39    90404
60 62              47                  21                 125    93407
61 63              42                  18                  22    90089
62 64              42                  17                  32    94523
63 65              47                  23                 105    90024
64 66              59                  35                 131    91360
65 67              62                  36                 105    95670
66 68              53                  23                  45    95123
   Family.members CCAvg Education Mortgage Personal_Loan Securities.Account
1               4   1.6         1        0             0                   1
```

2	3	1.5	1	0	0	1
3	1	1.0	1	0	0	0
4	1	2.7	2	0	0	0
5	4	1.0	2	0	0	0
6	4	0.4	2	155	0	0
7	2	1.5	2	0	0	0
8	1	0.3	3	0	0	0
9	3	0.6	2	104	0	0
10	1	8.9	3	0	1	0
11	4	2.4	3	0	0	0
12	3	0.1	2	0	0	0
13	2	3.8	3	0	0	1
14	4	2.5	2	0	0	0
15	1	2.0	1	0	0	1
16	1	1.5	3	0	0	0
17	4	4.7	3	134	1	0
18	4	2.4	1	0	0	0
19	2	8.1	3	0	1	0
20	1	0.5	2	0	0	1
21	3	2.0	3	0	0	0
22	1	1.2	1	260	0	0
23	2	0.7	1	163	0	1
24	2	3.9	1	159	0	0
25	3	0.5	1	97	0	0
26	4	0.2	3	0	0	0
27	1	2.4	1	0	0	0
28	1	2.2	3	0	0	0
29	1	3.3	2	0	1	0
30	1	1.2	3	122	0	0
31	1	2.0	2	0	0	0
32	2	0.6	3	193	0	0
33	3	0.9	3	0	0	0
34	4	1.8	3	0	0	0
35	3	0.7	1	0	0	0
36	1	2.9	1	0	0	0
37	1	1.4	3	198	0	0
38	3	5.0	3	0	1	1
39	4	0.7	3	285	0	0
40	3	1.6	3	0	0	1
41	3	2.3	1	0	0	0
42	4	1.1	2	412	1	0
43	1	0.7	1	0	0	0
44	1	5.7	1	0	0	0
45	4	2.5	1	0	0	0
46	3	0.7	2	153	0	0
47	4	0.2	3	211	1	1
48	2	4.5	3	0	0	0
49	1	1.8	1	0	0	0
50	4	0.7	2	0	0	1
51	1	2.9	1	0	0	0
52	1	0.1	1	207	0	0
53	3	2.1	3	240	1	0
54	1	0.2	3	0	0	0
55	2	8.0	1	0	0	0
56	3	0.1	2	0	0	1
57	2	1.2	3	0	1	0
58	2	4.5	1	455	0	0
59	3	1.7	2	0	0	1
60	1	5.7	1	112	0	1
61	1	1.0	1	0	0	0
62	4	0.0	2	0	0	0

```
63              2   3.3         1        0             0                    0
64              1   3.8         1        0             0                    0
65              2   2.8         1      336             0                    0
66              4   2.0         3      132             0                    1
   CD.Account Online CreditCard cluster
1           0      0          0       2
2           0      0          0       2
3           0      0          0       1
4           0      0          0       1
5           0      0          1       1
6           0      1          0       1
7           0      1          0       3
8           0      0          1       3
9           0      1          0       1
10          0      0          0       4
11          0      0          0       3
12          0      1          0       1
13          0      0          0       2
14          0      1          0       3
15          0      0          0       2
16          0      1          1       3
17          0      0          0       4
18          0      0          0       1
19          0      0          0       4
20          0      0          1       2
21          0      1          0       3
22          0      1          0       1
23          0      0          0       2
24          0      0          1       5
25          0      1          0       1
26          0      0          0       1
27          0      1          1       5
28          0      1          1       3
29          1      1          1       4
30          0      1          0       3
31          0      1          0       1
32          0      0          0       3
33          0      0          0       1
34          0      1          0       1
35          0      0          0       3
36          0      0          1       5
37          0      0          0       3
38          1      1          0       4
39          0      1          0       1
40          0      0          0       2
41          0      0          0       1
42          0      1          0       4
43          0      1          0       1
44          0      1          1       5
45          0      0          1       3
46          0      1          0       1
47          1      1          1       4
48          0      0          1       3
49          0      0          1       1
50          0      1          0       2
51          0      1          0       5
52          0      0          0       1
53          0      1          0       4
54          0      1          0       1
55          0      1          0       5
56          1      1          0       2
```

```
57          0      0          0       4
58          0      0          0       5
59          0      1          0       2
60          0      0          0       5
61          0      0          0       1
62          0      1          0       1
63          0      0          0       5
64          0      1          1       5
65          0      0          0       5
66          0      0          0       2
 [ reached 'max' / getOption("max.print") -- omitted 4916 rows ]


> custProfile=aggregate(Thera_Bank_dataset[,2:14],list(Thera_Bank_dataset$
cluster),FUN='mean')
> print(custProfile)
  Group.1 Age..in.years. Experience..in.years. Income..in.K.month. ZIP.Cod
e
1       1       35.20728              9.953611            56.37992 93259.2
3
2       2       45.47166             20.192744            61.05215 93160.2
1
3       3       55.51285             30.189385            53.68212 93084.1
8
4       4       45.05439             19.830544           144.71130 93151.5
3
5       5       43.69298             18.850877           138.83158 93043.2
3
  Family.members    CCAvg Education  Mortgage Personal_Loan Securities.Acc
ount
1       2.599530 1.342155  1.968291  45.78391             0         0.0000
0000
2       2.528345 1.557052  1.845805  47.59184             0         1.0000
0000
3       2.407263 1.284972  1.980447  44.80559             0         0.0000
0000
4       2.612971 3.908724  2.232218 101.26778             1         0.1255
2301
5       1.478947 4.428193  1.040351  94.99649             0         0.0315
7895
  CD.Account    Online CreditCard
1 0.01467998 0.5760423  0.2906635
2 0.21541950 0.6054422  0.2811791
3 0.01452514 0.6139665  0.3055866
4 0.29079498 0.6046025  0.2970711
5 0.02631579 0.5842105  0.2754386

> #cart/rf and PM
> ##Convert all variables into factors where necessary. Use the dplyr pack
age
> ##Convert all variables into factors where necessary. Use the dplyr pack
age
> library(dplyr)
> Thera_Bank_dataset <- mutate_if(Thera_Bank_dataset, is.character, as.fac
tor)
> attach(Thera_Bank_dataset)
The following objects are masked from Thera_Bank_dataset (pos = 3):

    Age..in.years., CCAvg, CD.Account, cluster, CreditCard, Education,
    Experience..in.years., Family.members, ID, Income..in.K.month.,
    Mortgage, Online, Personal_Loan, Securities.Account, ZIP.Code
```

```
The following objects are masked from Thera_Bank_dataset (pos = 4):

    CCAvg, CD.Account, CreditCard, Education, Mortgage, Online

The following objects are masked from Thera_Bank_dataset (pos = 5):

    Age..in.years., CCAvg, CD.Account, cluster, CreditCard, Education,
    Experience..in.years., Family.members, ID, Income..in.K.month.,
    Mortgage, Online, Personal_Loan, Securities.Account, ZIP.Code

The following objects are masked from Thera_Bank_dataset (pos = 6):

    Age..in.years., CCAvg, CD.Account, cluster, CreditCard, Education,
    Experience..in.years., Family.members, Income..in.K.month., Mortgage,
    Online, Securities.Account

The following objects are masked from Thera_Bank_dataset (pos = 20):

    Age..in.years., CCAvg, CD.Account, cluster, CreditCard, Education,
    Experience..in.years., Family.members, ID, Income..in.K.month.,
    Mortgage, Online, Securities.Account, ZIP.Code

> Personal.Loan = as.factor(Personal.Loan)
> table(Personal.Loan)
Personal.Loan
   0    1
4504  478
> library(psych)
> describe(Thera_Bank_dataset[,2:14],na.rm = TRUE,quant = c(0.01,0.05,0.10
,0.25,0.75,0.90,0.95,0.99),IQR=TRUE,check=TRUE)
                      vars    n     mean       sd  median  trimmed     mad
min
Age..in.years.           1 4982    45.33    11.47    45.0    45.37   14.83
23
Experience..in.years.    2 4982    20.10    11.48    20.0    20.12   14.83
-3
Income..in.K.month.      3 4982    73.73    46.04    64.0    68.79   43.00
8
ZIP.Code                 4 4982 93152.52 2123.02 93437.0 93236.32 1968.89
9307
Family.members           5 4982     2.40     1.15     2.0     2.37    1.48
1
CCAvg                    6 4982     1.94     1.75     1.5     1.65    1.33
0
Education                7 4982     1.88     0.84     2.0     1.85    1.48
1
Mortgage                 8 4982    56.55   101.76     0.0    33.03    0.00
0
Personal_Loan            9 4982     0.10     0.29     0.0     0.00    0.00
0
Securities.Account      10 4982     0.10     0.31     0.0     0.01    0.00
0
CD.Account              11 4982     0.06     0.24     0.0     0.00    0.00
0
Online                  12 4982     0.60     0.49     1.0     0.62    0.00
0
CreditCard              13 4982     0.29     0.46     0.0     0.24    0.00
0
                        max range   skew kurtosis    se    IQR Q0.01   Q0.
05
```

```
Age..in.years.            67    44  -0.03    -1.16  0.16    20.0    25    27
.0
Experience..in.years.     43    46  -0.02    -1.12  0.16    20.0    -1     2
.0
Income..in.K.month.      224   216   0.84    -0.04  0.65    59.0    10    18
.0
ZIP.Code               96651 87344 -12.52   486.20 30.08 2697.0 90024 90071
.1
Family.members             4     3   0.15    -1.40  0.02     2.0     1     1
.0
CCAvg                     10    10   1.60     2.64  0.02     1.8     0     0
.1
Education                  3     2   0.23    -1.55  0.01     2.0     1     1
.0
Mortgage                 635   635   2.10     4.75  1.44   101.0     0     0
.0
Personal_Loan              1     1   2.74     5.53  0.00     0.0     0     0
.0
Securities.Account         1     1   2.59     4.71  0.00     0.0     0     0
.0
CD.Account                 1     1   3.70    11.66  0.00     0.0     0     0
.0
Online                     1     1  -0.39    -1.85  0.01     1.0     0     0
.0
CreditCard                 1     1   0.90    -1.18  0.01     1.0     0     0
.0
                         Q0.1   Q0.25   Q0.75    Q0.9 Q0.95    Q0.99
Age..in.years.           30.0    35.0    55.0    61.0    63    65.00
Experience..in.years.     4.0    10.0    30.0    36.0    38    41.00
Income..in.K.month.      22.0    39.0    98.0   145.0   170   193.00
ZIP.Code              90275.2 91911.0 94608.0 95138.0 95670 95929.00
Family.members            1.0     1.0     3.0     4.0     4     4.00
CCAvg                     0.3     0.7     2.5     4.3     6     8.00
Education                 1.0     1.0     3.0     3.0     3     3.00
Mortgage                  0.0     0.0   101.0   200.0   272   431.19
Personal_Loan             0.0     0.0     0.0     0.0     1     1.00
Securities.Account        0.0     0.0     0.0     1.0     1     1.00
CD.Account                0.0     0.0     0.0     0.0     1     1.00
Online                    0.0     0.0     1.0     1.0     1     1.00
CreditCard                0.0     0.0     1.0     1.0     1     1.00
> ##Frequency distribution for categorical variable (Univariate Analysis)
> table(Thera_Bank_dataset[,c(11)])

   0    1
4463  519
> table(Thera_Bank_dataset[,c(12)])

   0    1
4682  300
> table(Thera_Bank_dataset[,c(13)])

   0    1
2013 2969
> table(Thera_Bank_dataset[,c(14)])

   0    1
3517 1465
> ##plotting the corrplot on the dataset
> library(DataExplorer)
> plot_correlation(Thera_Bank_dataset[,-c(1,5,15)])
> ##########################################################################
##############################################
```

```
> ##Build a decision tree using CART technique
> ##############################################################################
##############################################
> library(caTools)
> ## removing unwanted variables.Employee Number, Employee Count, Over18 a
nd Standard Hours
> Thera_Bank_dataset=Thera_Bank_dataset[,-c(1,5,15)]
> Personal.Loan=as.factor(Personal.Loan)
> names(Thera_Bank_dataset)[8]="Personal.Loan"
> names(Thera_Bank_dataset)[1]="Age"
> names(Thera_Bank_dataset)[2]="Experience"
> names(Thera_Bank_dataset)[3]="Income"
> names(Thera_Bank_dataset)[4]="Family"
> names(Thera_Bank_dataset)[5]="CCAvg"
> names(Thera_Bank_dataset)[6]="Education"
> names(Thera_Bank_dataset)[7]="Mortgage"
> names(Thera_Bank_dataset)[9]="Security.Account"
> names(Thera_Bank_dataset)[10]="CD.Account"
> Thera_Bank_dataset=na.omit(Thera_Bank_dataset)
> colSums(is.na(Thera_Bank_dataset))
             Age       Experience           Income           Family
               0                0                0                0
           CCAvg        Education         Mortgage    Personal.Loan
               0                0                0                0
Security.Account       CD.Account           Online       CreditCard
               0                0                0                0
> set.seed(1234)
> ## Spliting the dataset into train and test for development and out of s
ample testing respectively
> sample=sample.split(Thera_Bank_dataset,SplitRatio = 0.7)
> #Use subset function to get the data that is TRUE for the training data
set
> CARTtrain = subset(Thera_Bank_dataset,sample == TRUE)
>
> #Use subset function to get the data that is FALSE for the testing data
set
> CARTtest = subset(Thera_Bank_dataset,sample == FALSE)
> ## Calculate the response rate.
> table(CARTtrain$Personal.Loan)

   0    1
3004  318
> sum(CARTtrain$Personal.Loan== "1")/nrow(CARTtrain)
[1] 0.09572547
> ##Import rpart and rpart.plot library for creating CART model
> library(rpart)
> library(rpart.plot)
> #Define the parameters
> r.ctrl = rpart.control(minsplit=100, minbucket = 10, cp = 0, xval = 10)
> #Building the CART model using the rpart function and the pre defined pa
rameters
> m1 <- rpart(formula = Personal.Loan~., data = CARTtrain, method = "class
", control = r.ctrl)
> m1
n= 3322

node), split, n, loss, yval, (yprob)
      * denotes terminal node

 1) root 3322 318 0 (0.904274533 0.095725467)
   2) Income< 113.5 2654  52 0 (0.980406933 0.019593067)
     4) CCAvg< 2.95 2469   7 0 (0.997164844 0.002835156) *
```

```
     5) CCAvg>=2.95 185  45 0 (0.756756757 0.243243243)
      10) Income< 82.5 72   5 0 (0.930555556 0.069444444) *
      11) Income>=82.5 113  40 0 (0.646017699 0.353982301)
        22) Education< 1.5 58   9 0 (0.844827586 0.155172414) *
        23) Education>=1.5 55  24 1 (0.436363636 0.563636364) *
   3) Income>=113.5 668 266 0 (0.601796407 0.398203593)
     6) Education< 1.5 438  50 0 (0.885844749 0.114155251)
      12) Family< 2.5 388   0 0 (1.000000000 0.000000000) *
      13) Family>=2.5 50   0 1 (0.000000000 1.000000000) *
     7) Education>=1.5 230  14 1 (0.060869565 0.939130435)
      14) Income< 116.5 23   9 0 (0.608695652 0.391304348) *
      15) Income>=116.5 207   0 1 (0.000000000 1.000000000) *
> #Displaying the decision tree
> library(rattle)
> fancyRpartPlot(m1)
> fancyRpartPlot(m1)
> #Examine the complexity plot
> printcp(m1)

Classification tree:
rpart(formula = Personal.Loan ~ ., data = CARTtrain, method = "class",
    control = r.ctrl)

Variables actually used in tree construction:
[1] CCAvg     Education Family    Income

Root node error: 318/3322 = 0.095725

n= 3322

         CP nsplit rel error  xerror     xstd
1 0.3176101      0   1.00000 1.00000 0.053326
2 0.1572327      2   0.36478 0.38679 0.034224
3 0.0157233      3   0.20755 0.22013 0.026031
4 0.0073375      4   0.19182 0.23270 0.026748
5 0.0000000      7   0.16981 0.20126 0.024914
> plotcp(m1)
> #Pruning the tree
> m1_pruned <- prune(m1, cp = 0.05)
> m1_pruned
n= 3322

node), split, n, loss, yval, (yprob)
      * denotes terminal node

 1) root 3322 318 0 (0.90427453 0.09572547)
   2) Income< 113.5 2654  52 0 (0.98040693 0.01959307) *
   3) Income>=113.5 668 266 0 (0.60179641 0.39820359)
     6) Education< 1.5 438  50 0 (0.88584475 0.11415525)
      12) Family< 2.5 388   0 0 (1.00000000 0.00000000) *
      13) Family>=2.5 50   0 1 (0.00000000 1.00000000) *
     7) Education>=1.5 230  14 1 (0.06086957 0.93913043) *
> #Display the new pruned tree
> fancyRpartPlot(m1_pruned)
> ##Use this pruned tree to do the prediction on train as well as test dat
a set
> CARTtrain$CART.Pred = predict(m1_pruned,data=CARTtrain,type="class")
> CARTtrain$CART.Score = predict(m1_pruned,data=CARTtrain,type="prob")[,"1
"]
> CARTtest$CART.Pred = predict(m1_pruned,CARTtest,type="class")
> CARTtest$CART.Score = predict(m1_pruned,CARTtest,type="prob")[,"1"]
> #Confusion matrix
```

```
> ## CART Model Confusion Metrix
> CART_CM_train = table(CARTtrain$Personal.Loan,CARTtrain$CART.Pred)
> CART_CM_train

     0    1
  0 2990   14
  1   52  266
> CART_CM_test = table(CARTtest$Personal.Loan,CARTtest$CART.Pred)
> CART_CM_test

     0    1
  0 1497    3
  1   32  128
> ## Misclassification Rate
> (CART_CM_train[1,2]+CART_CM_train[2,1])/nrow(CARTtrain) #Misclassified i
n train
[1] 0.01986755
> (CART_CM_test[1,2]+CART_CM_test[2,1])/nrow(CARTtest) #Misclassified in t
est
[1] 0.02108434
> ##Accuracy
> (CART_CM_train[1,1]+CART_CM_train[2,2])/nrow(CARTtrain) #Correct classif
ication in train
[1] 0.9801325
> (CART_CM_test[1,1]+CART_CM_test[2,2])/nrow(CARTtest) #Correct classifica
tion in test
[1] 0.9789157
> #Area under the ROC curve for train data
> library(ROCR)
> pred_dtrain <- prediction(CARTtrain$CART.Score, CARTtrain$Personal.Loan)
> perf_dtrain <- performance(pred_dtrain, "tpr", "fpr")
> plot(perf_dtrain,main = "ROC curve for train")
> #Check the value for area under the ROC curve under train
> auc_train_dt <- performance(pred_dtrain,"auc")
> auc_train_dt <- as.numeric(auc_train_dt@y.values)
> auc_train_dt
[1] 0.9268355
> #Area under the ROC curve for test data
> pred_dtest <- prediction(CARTtest$CART.Score, CARTtest$Personal.Loan)
> perf_dtest <- performance(pred_dtest, "tpr", "fpr")
> plot(perf_dtest,main = "ROC curve for test")
> #Check the value for area under the ROC curve under test
> auc_test_dt <- performance(pred_dtest,"auc")
> auc_test_dt <- as.numeric(auc_test_dt@y.values)
> auc_test_dt
[1] 0.9108521
> #Gain chart
> library(gains)
> gain_dtrain <- performance(pred_dtrain, "tpr", "rpp")
> plot(gain_dtrain, col="orange", lwd=2)
> lines(x=c(0, 0.5, 1), y=c(0, 1, 1), col="darkgreen", lwd=2)
> gain_dtest <- performance(pred_dtest, "tpr", "rpp")
> plot(gain_dtest, col="orange", lwd=2)
> lines(x=c(0, 0.5, 1), y=c(0, 1, 1), col="darkgreen", lwd=2)
> #Kolmogorov-Smirnov (KS) statistic and plot
> ks_dtrain <- max(perf_dtrain@y.values[[1]]- perf_dtrain@x.values[[1]])
> plot(perf_dtrain,main=paste0('KS=',round(ks_dtrain*100,1),'%'))
> lines(x = c(0,1),y=c(0,1))
> ks_dtest <- max(perf_dtest@y.values[[1]]- perf_dtest@x.values[[1]])
> plot(perf_dtest,main=paste0('KS=',round(ks_dtest*100,1),'%'))
> lines(x = c(0,1),y=c(0,1))
> #Gini
```

```
> library(ineq)
> ineq(CARTtrain$CART.Score,"gini")
[1] 0.7719529
> ineq(CARTtest$CART.Score,"gini")
[1] 0.7643358
> #Concordance
> install.packages("InformationValue")
Error in install.packages : Updating loaded packages
> library(InformationValue)
> Concordance(actuals=CARTtrain$Personal.Loan,predictedScores = CARTtrain$
CART.Score)
$Concordance
[1] 0.8544331

$Discordance
[1] 0.1455669

$Tied
[1] 0

$Pairs
[1] 955272

> Concordance(actuals=CARTtest$Personal.Loan,predictedScores = CARTtest$CA
RT.Score)
$Concordance
[1] 0.8221042

$Discordance
[1] 0.1778958

$Tied
[1] 2.775558e-17

$Pairs
[1] 240000

> install.packages("InformationValue")
Installing package into 'C:/Users/Chetan Suvarna/Documents/R/win-library/3
.5'
(as 'lib' is unspecified)
Warning in install.packages :
  package 'InformationValue' is in use and will not be installed
> # Lift chart
> library(lift)
>
> plotLift(CARTtrain$CART.Score,CARTtrain$Personal.Loan,cumulative = TRUE)
> plotLift(CARTtest$CART.Score,CARTtest$Personal.Loan,cumulative = TRUE)
> plotLift(CARTtrain$CART.Score,CARTtrain$Personal.Loan,cumulative = TRUE)
> plotLift(CARTtest$CART.Score,CARTtest$Personal.Loan,cumulative = TRUE)
> ##############################################################################
##############################################
> ##Build a Random Forest model
> ##############################################################################
##############################################
> ## Spliting the dataset into train and test for development and out of s
ample testing respectively
> set.seed(1234)
> library(caTools)
> Thera_Bank_dataset=read.csv('Thera_Bank_data.csv',header = TRUE)
> names(Thera_Bank_dataset)
```

```
 [1] "ID"                    "Age..in.years."        "Experience..in.years
."
 [4] "Income..in.K.month."   "ZIP.Code"              "Family.members"
 [7] "CCAvg"                 "Education"             "Mortgage"
[10] "Personal_Loan"         "Securities.Account"    "CD.Account"
[13] "Online"                "CreditCard"
> Thera_Bank_dataset=Thera_Bank_dataset[,-c(1,5)]
> ##Print the model to see the OOB and error rate
> names(Thera_Bank_dataset)[8]="Personal.Loan"
> names(Thera_Bank_dataset)[1]="Age"
> names(Thera_Bank_dataset)[2]="Experience"
> names(Thera_Bank_dataset)[3]="Income"
> names(Thera_Bank_dataset)[4]="Family"
> names(Thera_Bank_dataset)[9]="Security.Account"
> names(Thera_Bank_dataset)[10]="CD.Account"
> Thera_Bank_dataset=na.omit(Thera_Bank_dataset)
> colSums(is.na(Thera_Bank_dataset))
             Age       Experience           Income           Family
               0                0                0                0
           CCAvg        Education         Mortgage    Personal.Loan
               0                0                0                0
Security.Account       CD.Account           Online       CreditCard
               0                0                0                0
> sample1 = sample.split(Thera_Bank_dataset,SplitRatio = 0.7)
> head(sample1)
[1]  TRUE  TRUE  TRUE  TRUE FALSE FALSE
> str(Thera_Bank_dataset)
'data.frame':   4982 obs. of  12 variables:
 $ Age             : int  25 45 39 35 35 37 53 50 35 34 ...
 $ Experience      : int  1 19 15 9 8 13 27 24 10 9 ...
 $ Income          : int  49 34 11 100 45 29 72 22 81 180 ...
 $ Family          : 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   : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 2 ...
 $ Security.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 ...
> #Use subset function to get the data that is TRUE for the training data
set
> RFtrain = subset(Thera_Bank_dataset,sample1 == TRUE)
> table(is.na(RFtrain))

FALSE
39864
> #Use subset function to get the data that is FALSE for the testing data
set
> RFtest = subset(Thera_Bank_dataset,sample1 == FALSE)
> library(randomForest)
> ## set a seed to start the randomness
> seed=1234
> set.seed(seed)
> ##Build the first RF model
> #Node size shall be ~2% of the population. This is similar to minbucket
parameter in CART
> Rforest=randomForest(Personal.Loan~.,data=RFtrain,mtry=5,nodesize=100,im
portance=T)
> ##Print the model to see the OOB and error rate
> print(Rforest)
```

```
Call:
 randomForest(formula = Personal.Loan ~ ., data = RFtrain, mtry = 5,
nodesize = 100, importance = T)
               Type of random forest: classification
                     Number of trees: 500
No. of variables tried at each split: 5

        OOB estimate of  error rate: 1.78%
Confusion matrix:
     0   1 class.error
0 3000   4 0.001331558
1   55 263 0.172955975
> ##Plot the RF to know the optimum number of trees
> plot(Rforest)
> ##Identify the importance of the variables
> importance(Rforest)
                            0          1 MeanDecreaseAccuracy MeanDecreaseG
ini
Age               4.79298105  2.0867683            5.5586290        1.6152
358
Experience        2.73326001  0.1641491            2.8467634        1.0658
123
Income           53.27402672 50.8906153           56.6247687      148.4986
532
Family           52.39523398 32.9560699           51.8932386       59.6076
194
CCAvg            14.63260302 14.5592367           16.2003702       55.3051
817
Education        81.96245172 55.1888817           81.0249530      129.3287
758
Mortgage          5.31843014 -7.7519940           -0.8452025        5.4344
217
Security.Account  0.99693303 -0.8267144            1.0473616        0.0519
255
CD.Account        7.56570446  3.1042304            8.4919056       15.5625
322
Online            0.04858479  0.1602857            0.1272669        0.3256
287
CreditCard        1.67020552  1.4834662            2.9584024        0.2167
128
> ##Tune up the RF model to find out the best mtry
> set.seed(seed)
> tRforest = tuneRF(x=RFtrain[,-(8)],y=RFtrain$Personal.Loan,mtrystart = 6
,stepfactor=1.5,ntree=51,improve=0.0001,
+                   nodesize=10,trace=TRUE,plot=TRUE,doBest=TRUE,importanc
e=TRUE)
mtry = 3  OOB error = 1.48%
Searching left ...
mtry = 2        OOB error = 2.05%
-0.3877551 1e-04
Searching right ...
mtry = 6        OOB error = 1.44%
0.02040816 1e-04
mtry = 11       OOB error = 1.51%
-0.04166667 1e-04
> plot(tRforest)
> ##Build the refined RF model
>
> ##Use this tree to do the prediction on train as well as test data set
> RFtrain$RF.Pred = predict(tRforest,RFtrain,type="class")
> RFtrain$RF.Score = predict(tRforest,RFtrain,type='prob')[,"1"]
> RFtest$RF.Pred = predict(tRforest,RFtest,type="class")
```

```
> RFtest$RF.Score = predict(tRforest,RFtest,type='prob')[,"1"]
> head(RFtrain)
  Age Experience Income Family CCAvg Education Mortgage Personal.Loan
1  25          1     49      4   1.6         1        0             0
2  45         19     34      3   1.5         1        0             0
3  39         15     11      1   1.0         1        0             0
4  35          9    100      1   2.7         2        0             0
7  53         27     72      2   1.5         2        0             0
8  50         24     22      1   0.3         3        0             0
  Security.Account CD.Account Online CreditCard RF.Pred RF.Score
1                1          0      0          0       0    0.000
2                1          0      0          0       0    0.000
3                0          0      0          0       0    0.000
4                0          0      0          0       0    0.006
7                0          0      1          0       0    0.000
8                0          0      0          1       0    0.000
> head(RFtest)
   Age Experience Income Family CCAvg Education Mortgage Personal.Loan
5   35          8     45      4   1.0         2        0             0
6   37         13     29      4   0.4         2      155             0
9   35         10     81      3   0.6         2      104             0
11  65         39    105      4   2.4         3        0             0
17  38         14    130      4   4.7         3      134             1
18  42         18     81      4   2.4         1        0             0
   Security.Account CD.Account Online CreditCard RF.Pred RF.Score
5                 0          0      0          1       0    0.000
6                 0          0      1          0       0    0.000
9                 0          0      1          0       0    0.000
11                0          0      0          0       0    0.020
17                0          0      0          0       1    0.998
18                0          0      0          0       0    0.000
> #Confusion matrix
> ## RF  Model Confusion Metrix
> RF_CM_train = table(RFtrain$Personal.Loan,RFtrain$RF.Pred)
> RF_CM_test = table(RFtest$Personal.Loan,RFtest$RF.Pred)
> RF_CM_train

      0    1
  0 3002    2
  1   19  299
> RF_CM_test

      0    1
  0 1493    7
  1   17  143
> ## Misclassification Rate
> (RF_CM_train[1,2]+RF_CM_train[2,1])/nrow(RFtrain) #Misclassified in trai
n
[1] 0.006321493
> (RF_CM_test[1,2]+RF_CM_test[2,1])/nrow(RFtest) #Misclassified in test
[1] 0.01445783
> ##Accuracy
> (RF_CM_train[1,1]+RF_CM_train[2,2])/nrow(RFtrain) #Correct classificatio
n in train
[1] 0.9936785
> (RF_CM_test[1,1]+RF_CM_test[2,2])/nrow(RFtest) #Correct classification i
n test
[1] 0.9855422
> #Check performance
> library(ROCR)
> pred_rftrain <- prediction(RFtrain$RF.Score,RFtrain$Personal.Loan)
> perf_rftrain <- performance(pred_rftrain, "tpr", "fpr")
```

```
> plot(perf_rftrain,main = "ROC curve")
> plot(perf_rftrain,main = "ROC curve for train")
> pred_rftest <- prediction(RFtest$RF.Score,RFtest$Personal.Loan)
> perf_rftest <- performance(pred_rftest, "tpr", "fpr")
> plot(perf_rftrain,main = "ROC curve for test")
> #Check area under the ROC curve
> auc_train_rf <- performance(pred_rftrain,"auc");
> auc_train_rf <- as.numeric(auc_train_rf@y.values)
> auc_train_rf
[1] 0.9997744
> #Check area under the ROC curve
> auc_test_rf <- performance(pred_rftest,"auc")
> auc_test_rf <- as.numeric(auc_test_rf@y.values)
> auc_test_rf
[1] 0.9968646
> ## List the importance of the variables.
> impVar <- round(randomForest::importance(Rforest), 2)
> impVar[order(impVar[,3], decreasing=TRUE),]
                    0     1 MeanDecreaseAccuracy MeanDecreaseGini
Education       81.96 55.19                81.02           129.33
Income          53.27 50.89                56.62           148.50
Family          52.40 32.96                51.89            59.61
CCAvg           14.63 14.56                16.20            55.31
CD.Account       7.57  3.10                 8.49            15.56
Age              4.79  2.09                 5.56             1.62
CreditCard       1.67  1.48                 2.96             0.22
Experience       2.73  0.16                 2.85             1.07
Security.Account 1.00 -0.83                 1.05             0.05
Online           0.05  0.16                 0.13             0.33
Mortgage         5.32 -7.75                -0.85             5.43
> #Gain chart
> gain_rftrain <- performance(pred_rftrain, "tpr", "rpp")
> plot(gain_rftrain, col="orange", lwd=2)
> lines(x=c(0, 0.5, 1), y=c(0, 1, 1), col="darkgreen", lwd=2)
> gain_rftest <- performance(pred_rftest, "tpr", "rpp")
> plot(gain_rftest, col="orange", lwd=2)
> lines(x=c(0, 0.5, 1), y=c(0, 1, 1), col="darkgreen", lwd=2)
> #Kolmogorov-Smirnov (KS) statistic and plot
> ks_rftrain <- max(perf_rftrain@y.values[[1]]- perf_rftrain@x.values[[1]]
)
> plot(perf_rftrain,main=paste0('KS=',round(ks_rftrain*100,1),'%'))
> lines(x = c(0,1),y=c(0,1))
> ks_rftest <- max(perf_rftest@y.values[[1]]- perf_rftest@x.values[[1]])
> plot(perf_rftest,main=paste0('KS=',round(ks_rftest*100,1),'%'))
> lines(x = c(0,1),y=c(0,1))
> #Gini
> library(ineq)
> ineq(RFtrain$RF.Score,"gini")
[1] 0.9008748
> ineq(RFtest$RF.Score,"gini")
[1] 0.8958084
> #Concordance
> library(InformationValue)
> Concordance(actuals=RFtrain$Personal.Loan,predictedScores = RFtrain$RF.S
core)
$Concordance
[1] 0.9997718

$Discordance
[1] 0.0002282073

$Tied
```

```
[1] 1.021861e-17

$Pairs
[1] 955272

> Concordance(actuals=RFtest$Personal.Loan,predictedScores = RFtest$RF.Sco
re)
$Concordance
[1] 0.9968042

$Discordance
[1] 0.003195833

$Tied
[1] -4.683753e-17

$Pairs
[1] 240000

> # Lift chart
> library(lift)
>
> plotLift(RFtrain$RF.Score,RFtrain$Personal.Loan,cumulative = TRUE)
> plotLift(RFtest$RF.Score,RFtest$Personal.Loan,cumulative = TRUE)
> plotLift(RFtrain$RF.Score,RFtrain$Personal.Loan,cumulative = TRUE)
> q()
> #=============================================================================
#
> #                              THE END                                      #
> #=============================================================================#
```