

Project – 3

**Thera Bank - Loan Purchase Modeling**



**Table of Contents**

**Project objective …………………………….……………………………………………………………………………………….1**

**Exploratory Data Analysis…………………………………………………………………………………………………………2**

**Applying appropriate clustering on the data and interpretation of the output......................….15**

**Build appropriate models on both the test and train data (CART & Random Forest). Interpret all the model outputs and do the necessary modifications wherever eligible (such as pruning) ….24**

**Checking the Performance of all the models (test and train). Using model performance measures…………………………………………………………………………………………………………………………………37**

**Appendix…………………………………………………………………………………………………………………………………47**

**Project objective:**

**Thera Bank - Loan Purchase Modeling**

This case is about a bank (Thera Bank) which has a growing customer base. Majority of these customers are liability customers (depositors) with varying size of deposits. The number of customers who are also borrowers (asset customers) is quite small, and the bank is interested in expanding this base rapidly to bring in more loan business and in the process, earn more through the interest on loans. In particular, the management wants to explore ways of converting its liability customers to personal loan customers (while retaining them as depositors). A campaign that the bank ran last year for liability customers showed a healthy conversion rate of over 9% success. This has encouraged the retail-marketing department to devise campaigns with better target marketing to increase the success ratio with a minimal budget. The department wants to build a model that will help them identify the potential customers who have a higher probability of purchasing the loan. This will increase the success ratio while at the same time reduce the cost of the campaign. The dataset has data on 5000 customers. The data include customer demographic information (age, income, etc.), the customer's relationship with the bank (mortgage, securities account, etc.), and the customer response to the last personal loan campaign (Personal Loan). Among these 5000 customers, only 480 (= 9.6%) accepted the personal loan that was offered to them in the earlier campaign.

As a consultant. It is required to build the best model, which can classify the right customers who have a higher probability of purchasing the loan. The following below are expected:

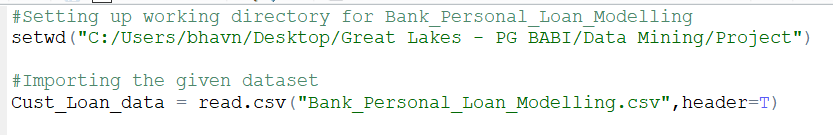
* EDA of the data available. Showcase the results using appropriate graphs.
* Apply appropriate clustering on the data and interpret the output.
* Build appropriate models on both the test and train data (CART & Random Forest). Interpret all the model outputs and do the necessary modifications wherever eligible (such as pruning).
* Check the performance of all the models that you have built (test and train). Use all the model performance measures you have learned so far. Share your remarks on which model performs the best.

**Exploratory Data Analysis:**

**Dataset provided for analysis:**



**Setting up working directory and Importing dataset:**



**Dimension of the data:**

> dim(Cust\_Loan\_data)

[1] 5000 14 **Data set contains 5000 rows and 14 columns**

**Structure of the data:**

> str(Cust\_Loan\_data)

'data.frame': 5000 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 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 ...

|  |  |
| --- | --- |
| **Data Description:** | |
| **ID** | Customer ID |
| **Age** | Customer's age in years |
| **Experience** | Years of professional experience |
| **Income** | Annual income of the customer ($000) |
| **ZIPCode** | Home Address ZIP code. |
| **Family** | Family size of the customer |
| **CCAvg** | Avg. spending on credit cards per month ($000) |
| **Education** | Education Level. 1: Undergrad; 2: Graduate; 3: Advanced/Professional |
| **Mortgage** | Value of house mortgage if any. ($000) |
| **Personal Loan** | Did this customer accept the personal loan offered in the last campaign? |
| **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? |
| **CreditCard** | Does the customer use a credit card issued by the bank? |

**Snapshot of the data:**

head(Cust\_Loan\_data,3)

ID Age..in.years. Experience..in.years. Income..in.K.month. ZIP.Code Family.members CCAvg Education Mortgage

1 1 25 1 49 91107 4 1.6 1 0

2 2 45 19 34 90089 3 1.5 1 0

3 3 39 15 11 94720 1 1.0 1 0

Personal.Loan Securities.Account CD.Account Online CreditCard

1 0 1 0 0 0

2 0 1 0 0 0

3 0 0 0 0 0

**Checking for missing values in dataset:**

> sum(is.na(Cust\_Loan\_data))

[1] 18

> which(is.na(Cust\_Loan\_data))

[1] 25021 25059 25099 25162 25236 25290 25488 25722 26461 26462 27400 27833 28702 29136 29139 29403 29404 29764

> colSums(is.na(Cust\_Loan\_data))

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

0 0 0 0 0

Family.members CCAvg Education Mortgage Personal.Loan

18 0 0 0 0

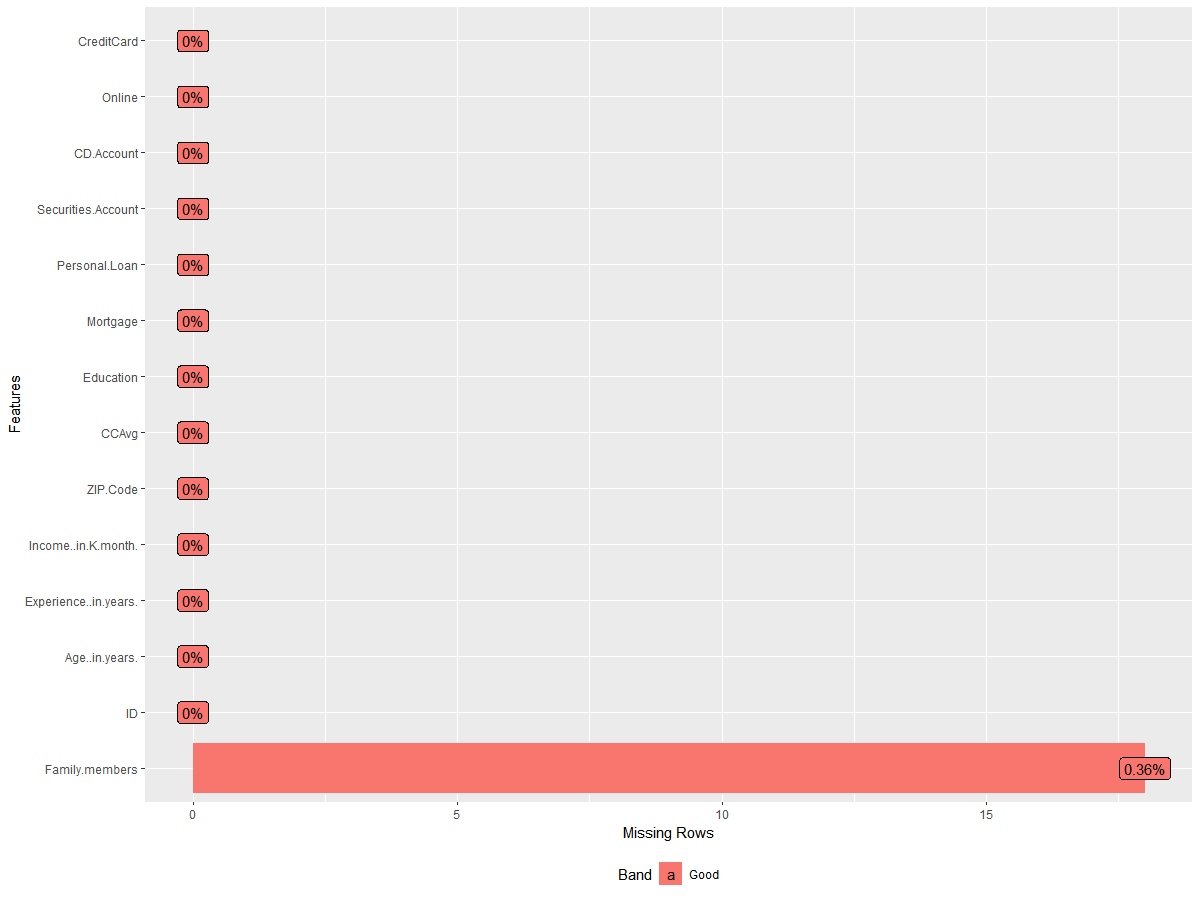
Securities.Account CD.Account Online CreditCard

0 0 0 0

**Graph to show missing values:**

> library(DataExplorer)

> plot\_missing(Cust\_Loan\_data)



**Replacement of NA values with zero in family member columns**

> Cust\_Loan\_data[which(is.na(Cust\_Loan\_data$Family.members))] <- 0

**Crosschecking of NA values replaced**

> sum(is.na(Cust\_Loan\_data$Family.members))

[1] 0

***Observation, Inference & Action!***

* There are 18 missing values in Family members column in dataset.
* This shows either information is not gathered or might be corresponding ID of members don’t have family members.
* Considering the above mentioned, columns with missing data are replaced with “0” in an attempt to cement columns meaningfully in dataset.

**Checking for negative values in dataset:**

**Identifying the rows which have negative values**

> has.neg <- apply(Cust\_Loan\_data, 1, function(row) any(row < 0))

> which(has.neg)

[1] 90 227 316 452 525 537 541 577 584 598 650 671 687 794 890 910

[17] 1174 1429 1523 1906 2103 2431 2467 2546 2619 2718 2849 2877 2963 2981 3077 3131

[33] 3158 3280 3285 3293 3395 3426 3627 3797 3825 3888 3947 4016 4089 4117 4286 4412

[49] 4482 4515 4583 4958

**Identifying total rows which have negative values**

> length(which(has.neg))

[1] 52

**Identifying Columns which have negative values**

> names(Cust\_Loan\_data)[sapply(Cust\_Loan\_data, function(x) min(x))<0]

[1] "Experience..in.years."

**Converting rows with negative to positive values**

> Cust\_Loan\_data$Experience..in.years. = abs(Cust\_Loan\_data$Experience..in.years.)

**Crosschecking the values again for negative values**

> which(has.neg)

integer(0)

***Observation, Inference & Action!***

* 52 negative values are found in the above mentioned rows and in column "Experience..in.years."
* Experience cannot be with negative values (logically).
* All the negative values are changed to positive values.

**Identifying important columns for data analysis:**

> names(Cust\_Loan\_data)

[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"

> cust\_data = Cust\_Loan\_data[-c(1,5)]

ID and Zip code are doesn’t makes any sense to analyze, just representative variables. Hence eliminated from data set for analysis.

|  |
| --- |
| head(cust\_data)  Age..in.years. Experience..in.years. Income..in.K.month. Family.members CCAvg  1 25 1 49 4 1.6  2 45 19 34 3 1.5  3 39 15 11 1 1.0  4 35 9 100 1 2.7  5 35 8 45 4 1.0  6 37 13 29 4 0.4  Education Mortgage Personal.Loan Securities.Account CD.Account Online CreditCard  1 1 0 0 1 0 0 0  2 1 0 0 1 0 0 0  3 1 0 0 0 0 0 0  4 2 0 0 0 0 0 0  5 2 0 0 0 0 0 1  6 2 155 0 0 0 1 0 |
|  |
| |  | | --- | |  | |

**Five-point summary of data for analysis:**

**Age..in.years. Experience..in.years. Income..in.K.month. Family.members**

**Min. :23.00 Min. : 0.00 Min. : 8.00 Min. :0.000**

**1st Qu.:35.00 1st Qu.:10.00 1st Qu.: 39.00 1st Qu.:1.000**

**Median :45.00 Median :20.00 Median : 64.00 Median :2.000**

**Mean :45.34 Mean :20.13 Mean : 73.77 Mean :2.389**

**3rd Qu.:55.00 3rd Qu.:30.00 3rd Qu.: 98.00 3rd Qu.:3.000**

**Max. :67.00 Max. :43.00 Max. :224.00 Max. :4.000**

**CCAvg Education Mortgage Personal.Loan Securities.Account**

**Min. : 0.000 Min. :1.000 Min. : 0.0 Min. :0.000 Min. :0.0000**

**1st Qu.: 0.700 1st Qu.:1.000 1st Qu.: 0.0 1st Qu.:0.000 1st Qu.:0.0000**

**Median : 1.500 Median :2.000 Median : 0.0 Median :0.000 Median :0.0000**

**Mean : 1.938 Mean :1.881 Mean : 56.5 Mean :0.096 Mean :0.1044**

**3rd Qu.: 2.500 3rd Qu.:3.000 3rd Qu.:101.0 3rd Qu.:0.000 3rd Qu.:0.0000**

**Max. :10.000 Max. :3.000 Max. :635.0 Max. :1.000 Max. :1.0000**

**CD.Account Online CreditCard**

**Min. :0.0000 Min. :0.0000 Min. :0.000**

**1st Qu.:0.0000 1st Qu.:0.0000 1st Qu.:0.000**

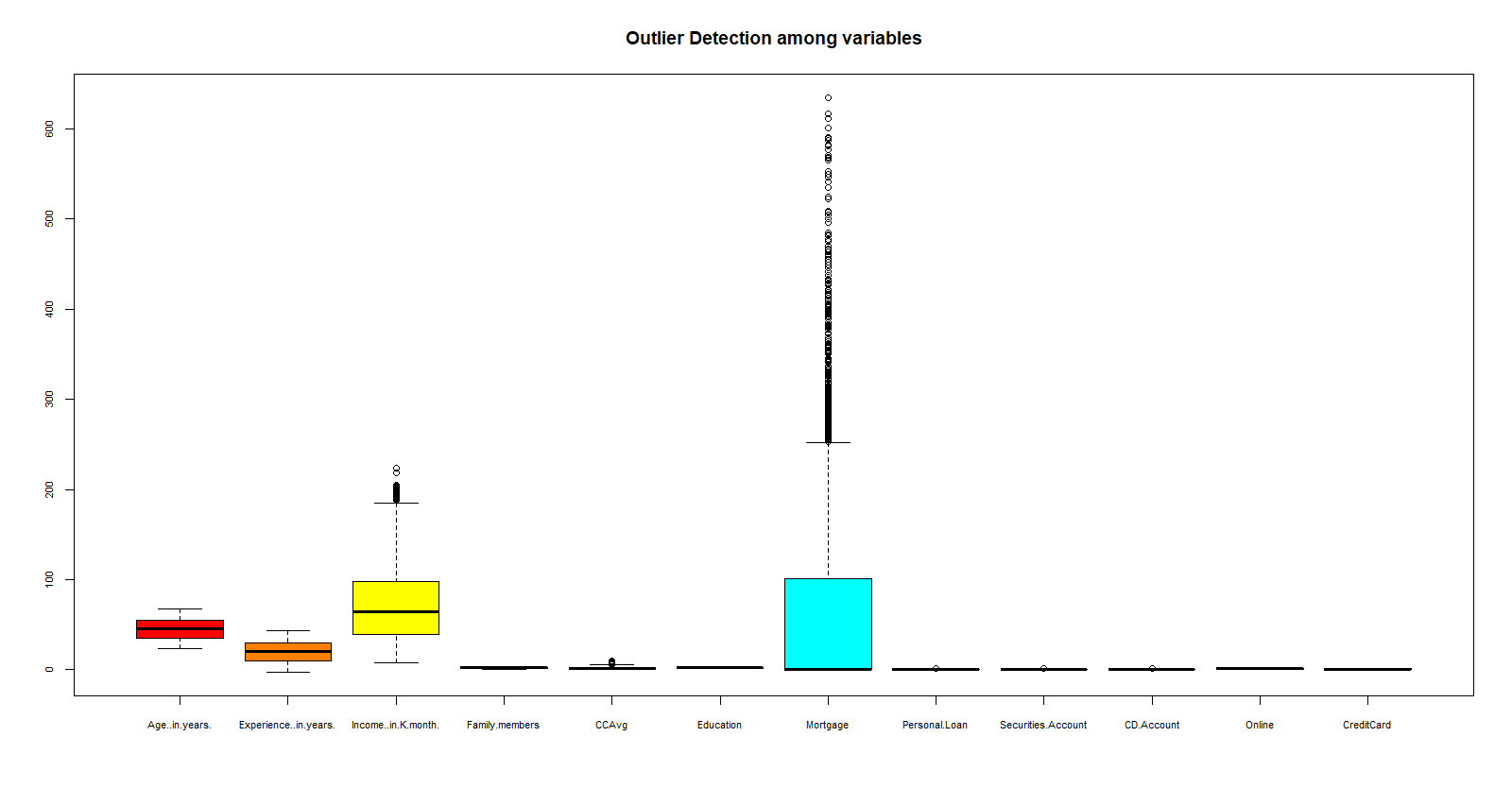
**Median :0.0000 Median :1.0000 Median :0.000**

**Mean :0.0604 Mean :0.5968 Mean :0.294**

**3rd Qu.:0.0000 3rd Qu.:1.0000 3rd Qu.:1.000**

**Max. :1.0000 Max. :1.0000 Max. :1.000**

**Outlier Detection among the data for analysis:**



***Observations from all variables:***

**Age:** Minimum age of bank customer among the sample is 23 and Maximum being 67. Average age among sample is 45.

**Experience:** There are customers with work no experience and highest work experience being 43 years. Average experience among sample is 20 years.

**Annual Income:** Average income is observed to be74,000$ across the sample, lowest being 8000$ and highest being 2,24,000.

**Outlier detection in Annual income: Since there is a drastic higher difference across annual income among the customers. The data with higher values exhibits as an outlier.**

**Family members:** Highest number of family members accompanying a bank customer is 4. On an average, there are two family members.

**Average spending on credit card per month:** Highest amount of bill spent on credit card is 10,000$ and among the sample customers tend to spend nearly 2000$ on credit cards.

**Mortgage:** Average amount on mortgage is 57,000$ and highest being 635,000$. Overall data shows there are high number mortgaged customers.

**Outlier detection in Mortgage: Unequally higher Mortgage amount in majority sample among the customers.**

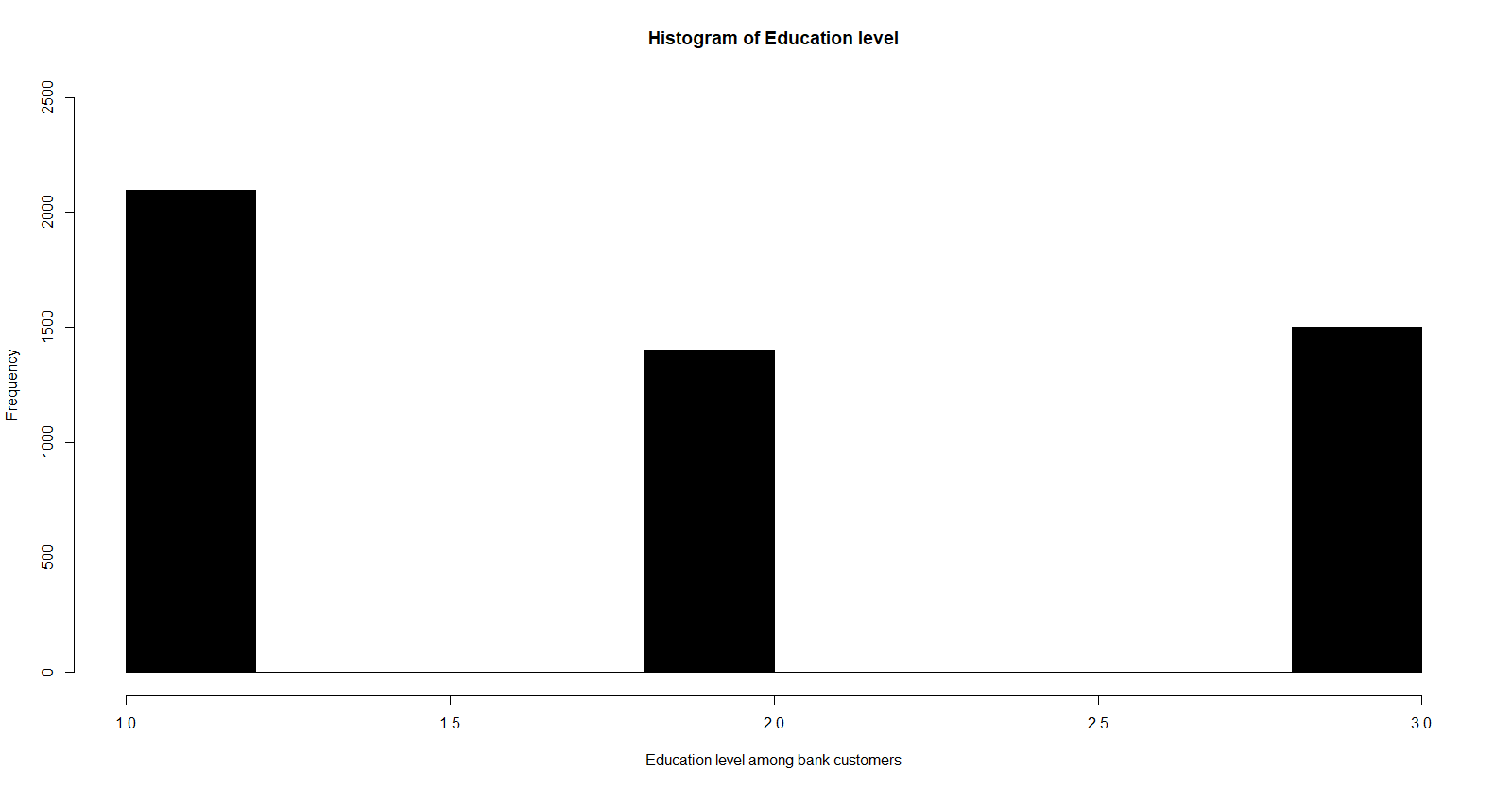
**Education:**

table(cust\_data$Education)

1 2 3

2096 1403 1501

1: Undergrad; 2: Graduate; 3: Advanced/Professional



Higher number bank customers are Undergraduates

**Personal Loan:**

> personal\_loan = as.data.frame(table(Personal.Loan))

> personal\_loan

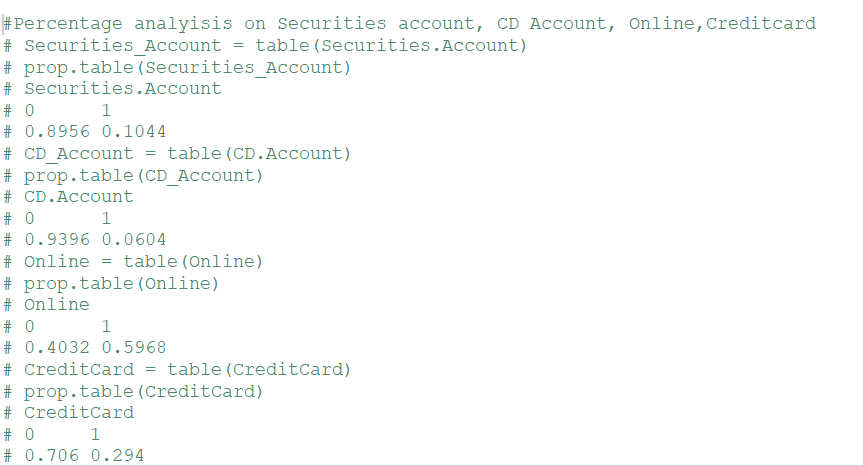
Personal.Loan Freq

1 0 4520

2 1 480

480 (9.6%) customers availed personal loan option given by the bank as per project objective.

**Securities Account, CD Account, Online, Credit Card:**

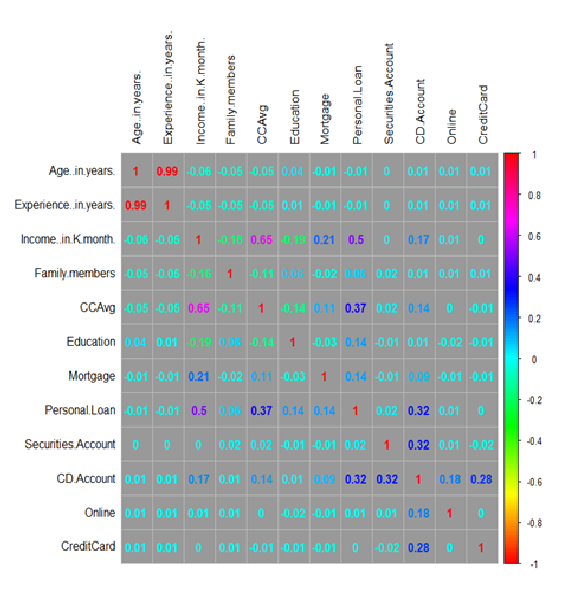


***Insights from the above variables!***

* **10%** of customers from have securities account with bank.
* **6%** of customers have certificate of deposit (CD) account with the bank.
* **60%** of customer use internet-banking facilities.
* **29%** of customer use a credit card issued by the bank.

**Correlation among the bank customer details**

> corrplot(cor\_cust\_data,method = "number",col = c(rainbow(140)),tl.col = "black",tl.cex = 1,bg="grey60")

*\*Please expand for clear observation*

***Observation & Inference!***

* It is very much obvious; age and experience in years are very highly correlated. **(0.99)**
* Spending on credit cards moderately correlated with monthly income.
* There is even weak correlation on customers with deposit account who took personal loan and have Securities Account.
* There is better correlation between (CCAvg) monthly credit card (vs) Income per month and personal loan.

**Applying appropriate clustering on the data and interpretation of the output:**

Among the clustering techniques (unsupervised learning) - Hierarchical clustering and k-Means clustering covered.

When it comes to choose of appropriate clustering. K-Means clustering is preferred.

**Why hierarchical clustering avoided?**

* Considering the size of dataset 5000 rows and 14 columns, (*12,497,500*observations based). It is laborious to construct dendrogram and to identify clusters.
* Processing time and memory in R consumed at higher side.

**K-Means clustering:**

**Identifying the right number of clusters using NbClust:**

> library(NbClust)

> set.seed(seed = 100)

> clust\_size = NbClust(cust\_data,min.nc = 2,max.nc = 6,method = "kmeans")

\*\*\* : 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 corresponds to a

significant increase of the value of the measure i.e the significant peak 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 peak in Dindex

second differences plot) that corresponds to a significant increase of the value of

the measure.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Among all indices:

\* 5 proposed 2 as the best number of clusters

\* 2 proposed 3 as the best number of clusters

**\* 10 proposed 4 as the best number of clusters**

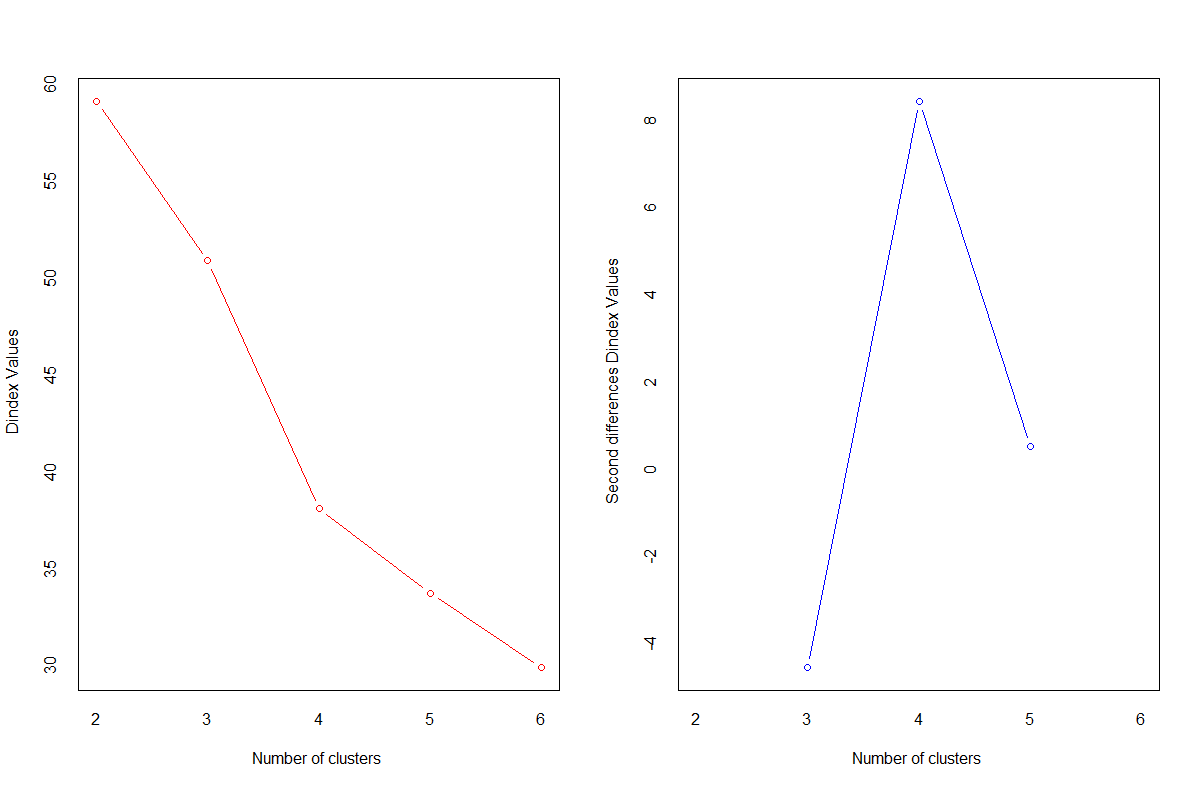
\* 3 proposed 5 as the best number of clusters

\* 4 proposed 6 as the best number of clusters

\*\*\*\*\* Conclusion \*\*\*\*\*

\* According to the majority rule, **the best number of clusters is 4**

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*



Same was tried with other max clusters 5,6,7 but the optimal number of clusters was 4 on all results.

***Inference!***

***Bank customers can be classified into four categories based upon which we can target the potential customers with higher probability of purchasing the loan.***

**Scaling of the dataset (snapshot)**

> Scaled\_cust\_data = scale(cust\_data)

> print(Scaled\_cust\_data)

Age..in.years. Experience..in.years. Income..in.K.month. Family.members

[1,] -1.77423939 -1.67624029 -0.538174951 1.3962869

[2,] -0.02952064 -0.09939388 -0.864022980 0.5297814

[3,] -0.55293627 -0.44980420 -1.363656626 -1.2032295

[4,] -0.90188002 -0.97541966 0.569708351 -1.2032295

[5,] -0.90188002 -1.06302224 -0.625067758 1.3962869

[6,] -0.72740814 -0.62500935 -0.972638990 1.3962869

[7,] 0.66836686 0.60142674 -0.038541305 -0.3367240

[8,] 0.40665905 0.33861900 -1.124701404 -1.2032295

[9,] -0.90188002 -0.88781709 0.156967513 0.5297814

[10,] -0.98911595 -0.97541966 2.307564509 -1.2032295

[11,] 1.71519811 1.65265767 0.678324360 1.3962869

[12,] -1.42529564 -1.32582998 -0.625067758 0.5297814

[13,] 0.23218717 0.25101643 0.873833178 -0.3367240

[14,] 1.19178248 1.03943963 -0.733683768 1.3962869

[15,] 1.88966998 1.82786283 0.830386774 -1.2032295

[16,] 1.27901842 0.86423447 -1.124701404 -1.2032295

[17,] -0.64017220 -0.53740677 1.221404410 1.3962869

[18,] -0.29122845 -0.18699646 0.156967513 1.3962869

[19,] 0.05771530 0.07581127 2.589966135 -0.3367240

[20,] 0.84283873 0.68902932 -1.146424606 -1.2032295

[21,] 0.93007467 0.95183705 -1.059531798 -2.0697349

[22,] 1.01731061 0.60142674 -0.234050123 0.5297814

[23,] -1.42529564 -1.32582998 -0.255773325 -1.2032295

[24,] -0.11675658 -0.18699646 -0.668514162 -0.3367240

[25,] -0.81464408 -0.80021451 1.699314854 -0.3367240

[26,] -0.20399252 -0.09939388 -0.972638990 0.5297814

[27,] -0.46570033 -0.36220162 0.200413917 1.3962869

[28,] 0.05771530 -0.01179131 1.829654065 -1.2032295

[29,] 0.93007467 0.86423447 -0.559898153 -1.2032295

[30,] -0.64017220 -0.62500935 0.982449188 -1.2032295

[31,] 1.19178248 1.30224736 -0.842299778 -1.2032295

***Note: Other rows are omitted.***

**[ reached getOption("max.print") -- omitted 4917 rows ]**

**attr(,"scaled:center")**

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

**45.338400 20.134600 73.774200**

**Family.members CCAvg Education**

**2.388600 1.937938 1.881000**

**Mortgage Personal.Loan Securities.Account**

**56.498800 0.096000 0.104400**

**CD.Account Online CreditCard**

**0.060400 0.596800 0.294000**

**attr(,"scaled:scale")**

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

**11.4631656 11.4151892 46.0337293**

**Family.members CCAvg Education**

**1.1540608 1.7476590 0.8398691**

**Mortgage Personal.Loan Securities.Account**

**101.7138021 0.2946207 0.3058093**

**CD.Account Online CreditCard**

**0.2382503 0.4905893 0.4556375**

**Applying K-means clustering to dataset**

> seed = 1000

> set.seed(seed)

> final\_clust = kmeans(Scaled\_cust\_data, centers = 4,nstart = 5)

> print(final\_clust)

K-means clustering with 4 clusters of sizes 1957, 707, 1861, 475

Cluster means:

Age..in.years. Experience..in.years. Income..in.K.month. Family.members CCAvg

1 0.8713230277 0.865845966 -0.3010846 -0.05999131 -0.3056997

2 -0.1125613568 -0.099269977 1.6419803 -0.12469369 1.6002477

3 -0.8737076362 -0.872424263 -0.2626953 0.08884610 -0.2517846

4 0.0007823686 -0.001463424 -0.1742770 0.08467126 -0.1358937

Education Mortgage Personal.Loan Securities.Account CD.Account Online

1 0.02487370 -0.06751011 -0.3258427 -0.3413892 -0.1956068 0.03235386

2 -0.07050840 0.46767589 1.8921459 -0.1240051 0.5895005 0.02036174

3 0.01628859 -0.08733330 -0.3258427 -0.3413892 -0.2016411 -0.04561381

4 -0.06135041 -0.07579430 -0.1972206 2.9286223 0.7184837 0.01510536

CreditCard

1 0.01754212

2 0.01906650

3 -0.01548928

4 -0.03996714

Clustering vector:

[1] 4 4 3 3 3 3 1 1 3 2 1 3 4 1 4 1 2 3 2 4 1 1 3 4 2 3 3 1 1 2 1 3 1 3 3 1 1 1 2

[40] 3 4 3 2 3 2 1 3 2 1 3 4 1 3 2 3 2 4 2 3 2 4 4 3 3 1 1 1 4 1 1 3 1 2 3 3 2 1 3

[79] 2 1 1 1 3 3 1 3 3 1 1 3 2 3 3 4 1 3 3 1 1 1 1 1 1 3 1 4 3 3 3 3 3 1 3 1 3 1 1

[118] 1 3 3 1 1 1 4 3 1 3 3 4 3 3 2 3 3 1 1 1 1 4 1 1 3 3 3 1 2 1 1 1 1 2 2 1 4 1 3

[157] 3 3 3 4 2 1 3 3 4 3 3 3 1 3 3 4 2 1 2 4 1 3 1 1 1 3 3 2 4 3 1 2 1 1 1 1 1 1 2

[196] 4 2 1 4 2 3 3 3 1 1 3 4 3 3 2 1 3 1 2 1 3 3 3 3 1 3 3 3 1 1 3 3 2 4 1 1 3 3 1

[235] 3 3 3 1 1 3 1 1 3 2 3 3 3 4 1 3 3 2 1 1 2 1 3 1 3 1 1 2 1 3 3 4 1 1 1 3 1 3 3

[274] 3 4 4 3 3 1 3 3 1 3 4 3 3 1 3 2 3 1 3 3 3 3 1 3 1 3 2 4 2 2 2 1 1 1 3 2 4 1 2

[313] 4 3 1 3 2 2 3 1 1 2 4 2 2 1 1 1 1 2 1 3 1 1 1 1 3 1 3 3 1 3 3 3 1 4 3 4 2 2 4

[352] 2 1 1 2 3 1 3 3 3 3 2 1 3 1 2 1 3 1 3 3 1 1 1 3 3 3 3 1 3 1 1 2 3 1 3 3 3 2 2

[391] 1 1 1 4 3 1 1 3 1 3 2 3 2 1 1 4 3 1 1 1 1 4 3 3 1 3 3 1 3 1 1 2 4 2 1 3 3 3 1

[430] 3 1 3 3 1 3 1 1 4 2 1 1 4 2 1 1 1 1 1 3 1 1 3 3 4 1 3 1 3 1 2 1 4 2 2 2 1 3 1

[469] 3 1 3 1 3 2 1 2 4 1 3 2 1 3 2 3 3 1 4 3 3 1 3 4 1 1 2 3 1 1 3 1 1 1 3 3 3 2 1

[508] 1 1 2 1 3 3 3 3 3 1 1 3 3 1 1 3 1 3 1 2 3 2 3 1 4 1 3 4 1 3 2 3 1 3 3 3 1 2 3

[547] 4 4 1 1 1 1 3 1 2 3 1 3 4 1 3 1 3 1 3 1 2 3 3 3 2 3 3 1 3 1 3 2 3 1 1 3 3 3 3

[586] 3 3 1 3 3 3 3 4 3 1 3 2 4 1 3 1 1 3 1 3 1 3 2 3 3 1 1 1 4 2 2 3 3 1 1 2 3 3 3

[625] 3 1 3 3 1 2 3 3 2 1 4 1 2 1 3 1 4 2 1 1 1 3 1 4 1 3 2 3 3 1 2 1 3 3 1 1 3 1 2

[664] 3 1 4 4 4 1 1 3 2 1 3 1 3 1 2 1 1 1 2 1 3 2 3 3 1 3 4 1 3 3 3 3 3 1 3 1 3 3 3

[703] 2 2 2 1 1 1 3 3 3 4 3 3 1 4 3 1 1 4 1 1 2 1 1 2 1 1 2 1 2 3 3 1 1 3 2 2 3 1 2

[742] 1 3 4 4 3 1 4 3 1 3 1 1 1 3 1 1 4 4 1 3 1 3 1 3 2 3 1 3 4 2 4 2 3 1 1 2 1 1 2

[781] 3 2 2 4 2 2 4 2 1 3 1 1 3 3 1 1 3 3 3 3 2 1 3 1 1 2 1 2 1 1 3 1 3 2 3 1 1 3 1

[820] 1 1 3 1 3 4 3 1 1 3 1 3 1 3 1 2 1 3 3 3 3 3 4 3 1 1 3 1 3 1 4 1 4 3 3 4 1 1 1

[859] 3 2 1 4 1 1 3 1 3 1 4 1 3 1 3 3 3 1 3 3 3 1 1 2 1 1 3 3 1 3 2 3 1 2 3 1 3 4 2

[898] 1 1 2 3 1 1 3 4 1 3 1 1 2 1 1 3 4 4 2 3 2 3 4 3 3 4 1 4 3 3 2 3 1 3 3 4 1 1 2

[937] 4 4 4 1 2 2 1 3 3 1 3 1 3 3 3 2 2 4 2 1 3 1 1 1 1 3 1 3 4 2 1 1 1 3 1 2 3 3 1

[976] 2 4 1 1 1 3 2 1 1 1 2 1 1 1 3 3 3 3 2 3 3 3 3 1 1

[ reached getOption("max.print") -- omitted 4000 entries ]

Within cluster sum of squares by cluster:

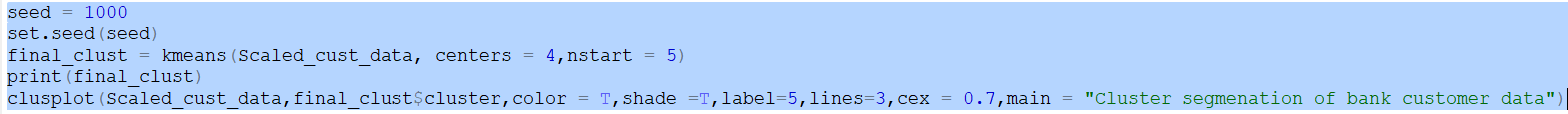
[1] 12590.75 11618.79 11757.49 5510.20

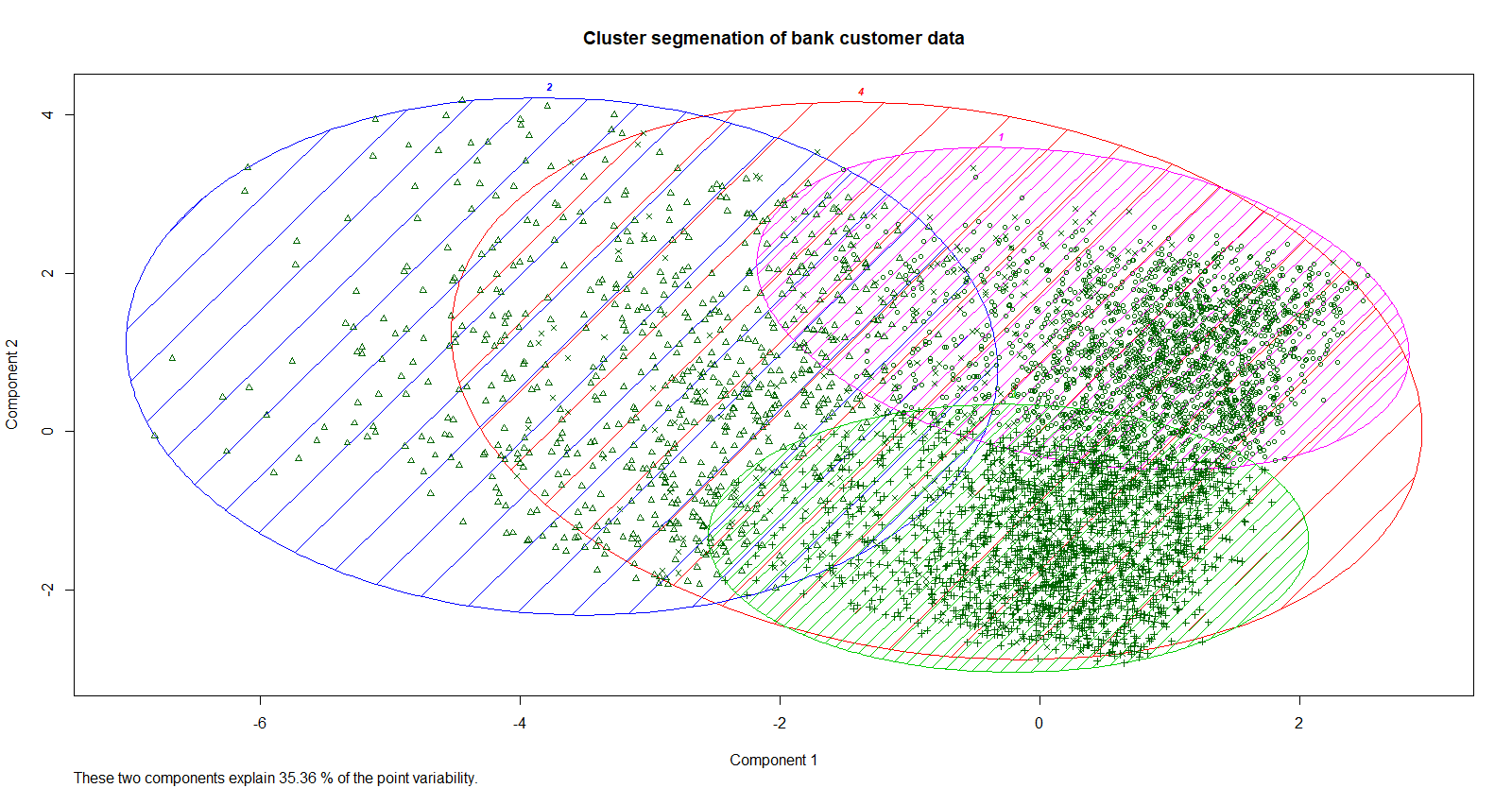
(between\_SS / total\_SS = 30.9 %)

Available components:

[1] "cluster" "centers" "totss" "withinss" "tot.withinss"

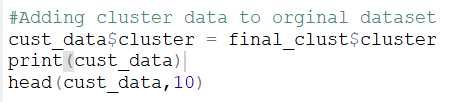
[6] "betweenss" "size" "iter" "ifault"





***Observation & Inference!***

* Most of the customers fall in cluster 1 and cluster 3
* There are distinct kind of customers who belong to cluster 2
* Cluster 4 customers have all attributes of all clusters.



**> head(cust\_data,10)**

**Age..in.years. Experience..in.years. Income..in.K.month. Family.members CCAvg**

**1 25 1 49 4 1.6**

**2 45 19 34 3 1.5**

**3 39 15 11 1 1.0**

**4 35 9 100 1 2.7**

**5 35 8 45 4 1.0**

**6 37 13 29 4 0.4**

**7 53 27 72 2 1.5**

**8 50 24 22 1 0.3**

**9 35 10 81 3 0.6**

**10 34 9 180 1 8.9**

**Education Mortgage Personal.Loan Securities.Account CD.Account Online CreditCard**

**1 1 0 0 1 0 0 0**

**2 1 0 0 1 0 0 0**

**3 1 0 0 0 0 0 0**

**4 2 0 0 0 0 0 0**

**5 2 0 0 0 0 0 1**

**6 2 155 0 0 0 1 0**

**7 2 0 0 0 0 1 0**

**8 3 0 0 0 0 0 1**

**9 2 104 0 0 0 1 0**

**10 3 0 1 0 0 0 0**

**cluster**

**1 4**

**2 4**

**3 3**

**4 3**

**5 3**

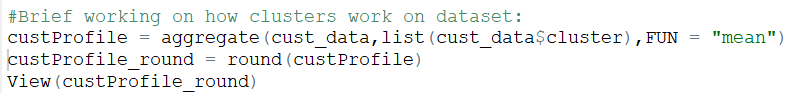
**6 3**

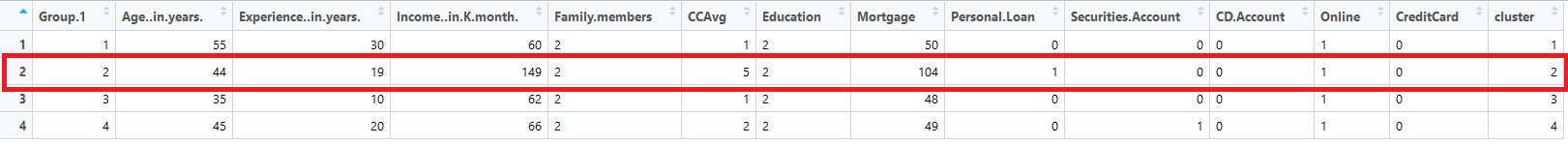
**7 1**

**8 1**

**9 3**

**10 2**



****

*\*The values are rounded off to make data look meaningful among the four buckets of customers.*

***Observation & Inference!***

Target Customers should be of:

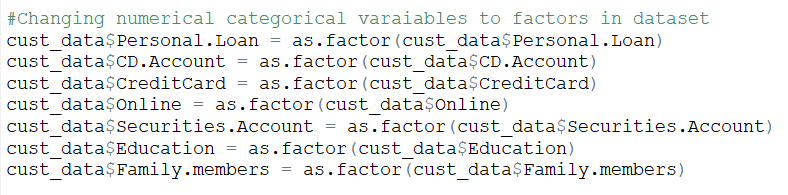
* age with 40 plus years and 15 plus years of experience.
* Monthly income with above 100,000 $
* Should have higher mortgage value 100,000 $
* Average spending on credit card bills of 5000 $.

**Build appropriate models on both the test and train data (CART & Random Forest). Interpret all the model outputs and do the necessary modifications wherever eligible (such as pruning):**

**Personal Loan would be the target or the response variable** i.e. the **Dependent variable** and other variables would be **independent or the predictor variables.**

CART tree which is constructed will fall under classification tree because personal loan is categorical variable.

|  |  |  |
| --- | --- | --- |
| **Variable Class** | **Name of variable** | **Type of variable** |
| **Dependent** | **Personal Loan** | **Categorical** |
| **Independent** | **Age..in.years** | **Continuous** |
| **Experience..in.years** | **Continuous** |
| **Income..in.K.month** | **Continuous** |
| **Family.members** | **Categorical** |
| **CD.Account** | **Categorical** |
| **CreditCard** | **Categorical** |
| **Online** | **Categorical** |
| **Securities.Account** | **Categorical** |
| **Mortgage** | **Continuous** |
| **CCAvg** | **Continuous** |
| **Education** | **Categorical** |



**libraries used in the code and the significance for using it.**

|  |  |
| --- | --- |
| caTools | Building the training and test dataset |
| Rpart | Built CART tree using rpart function and setting the control parameters using rpart.control function. |
| Rattle | Use the fancyRpartPlot function to display the tree |

**Preparing training and test data for CART:**

> library(caTools)

> set.seed(5000)

> Target\_CART = sample.split(cust\_data$Personal.Loan, SplitRatio = 0.7)

> Train\_CART = subset(cust\_data, Target\_CART == TRUE)

> Test\_CART = subset(cust\_data, Target\_CART == FALSE)

> nrow(Train\_CART)

[1] 3500

> nrow(Test\_CART)

[1] 1500

> str(Train\_CART)

'data.frame': 3500 obs. of 12 variables:

$ Age..in.years. : int 25 45 39 35 37 50 35 65 29 67 ...

$ Experience..in.years.: int 1 19 15 8 13 24 10 39 5 41 ...

$ Income..in.K.month. : int 49 34 11 45 29 22 81 105 45 112 ...

$ Family.members : Factor w/ 5 levels "0","1","2","3",..: 5 4 2 5 5 2 4 5 4 2 ...

$ CCAvg : num 1.6 1.5 1 1 0.4 0.3 0.6 2.4 0.1 2 ...

$ Education : Factor w/ 3 levels "1","2","3": 1 1 1 2 2 3 2 3 2 1 ...

$ Mortgage : int 0 0 0 0 155 0 104 0 0 0 ...

$ Personal.Loan : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...

$ Securities.Account : Factor w/ 2 levels "0","1": 2 2 1 1 1 1 1 1 1 2 ...

$ CD.Account : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...

$ Online : Factor w/ 2 levels "0","1": 1 1 1 1 2 1 2 1 2 1 ...

$ CreditCard : Factor w/ 2 levels "0","1": 1 1 1 2 1 2 1 1 1 1 ...

> head(Train\_CART)

Age..in.years. Experience..in.years. Income..in.K.month. Family.members CCAvg

1 25 1 49 4 1.6

2 45 19 34 3 1.5

3 39 15 11 1 1.0

5 35 8 45 4 1.0

6 37 13 29 4 0.4

8 50 24 22 1 0.3

Education Mortgage Personal.Loan Securities.Account CD.Account Online CreditCard

1 1 0 0 1 0 0 0

2 1 0 0 1 0 0 0

3 1 0 0 0 0 0 0

5 2 0 0 0 0 0 1

6 2 155 0 0 0 1 0

8 3 0 0 0 0 0 1

> table(Train\_CART$Personal.Loan)

0 1

3164 336

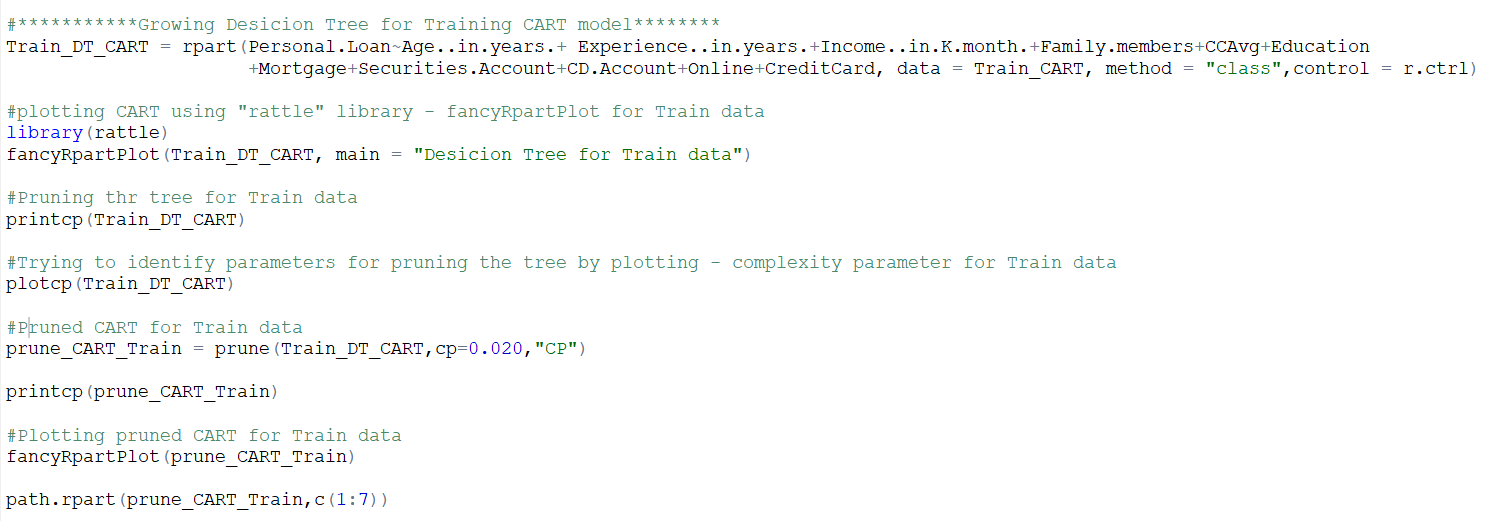
Training and test dataset are split 70% and 30% respectively with 3500 rows and 1500 rows.

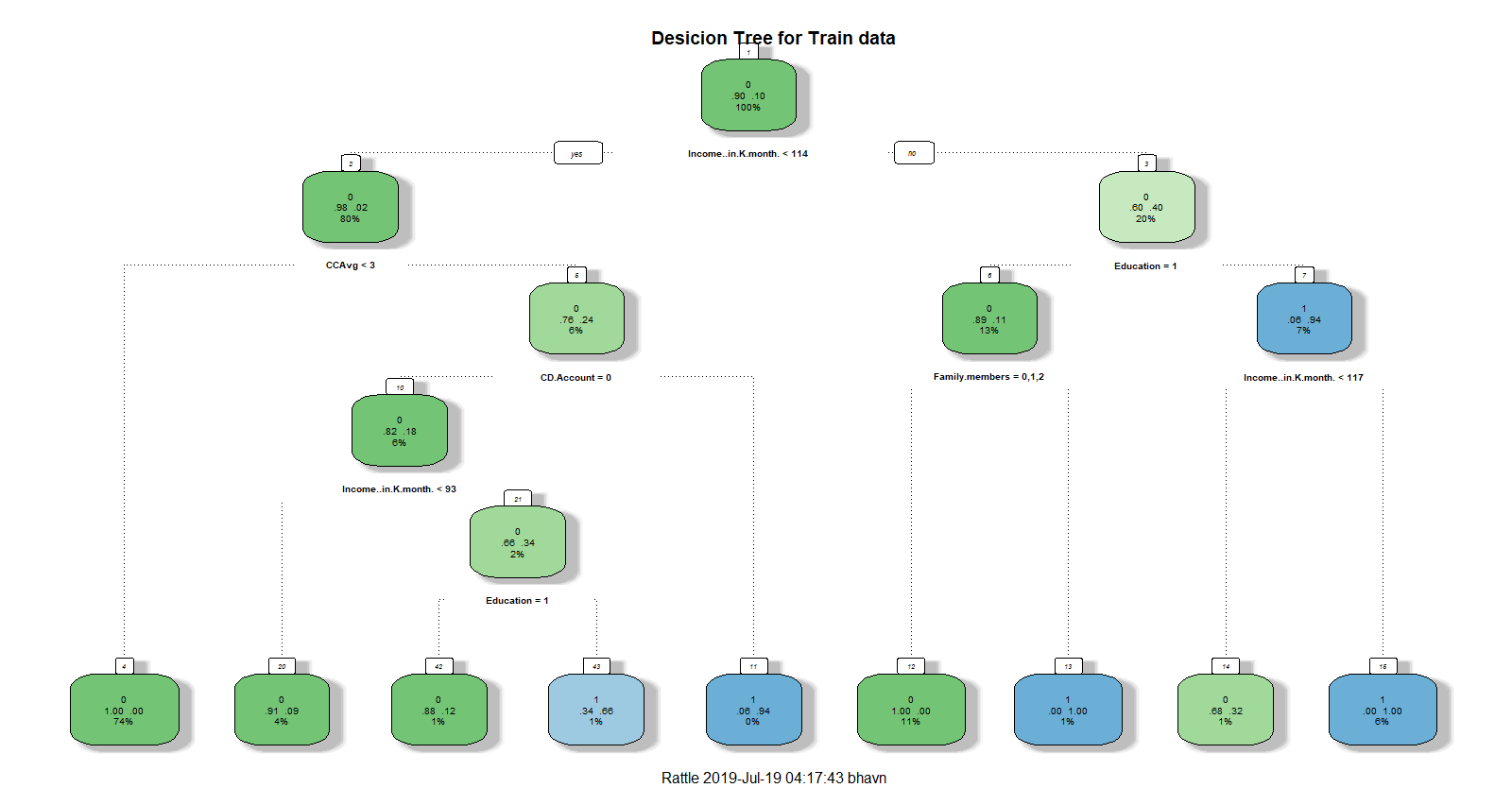


**Control parameters that are used while building the decision tree:**

|  |  |  |
| --- | --- | --- |
| **Parameter** | **Value** | **Significance** |
| minsplit | 50 | If the node will have at least 50 observations then only it will split. |
| minbucket | 10 | The terminal nodes should have at least 10 observations. |
| cp (Complexity Parameter) | 0 | Allowing the full tree to be grown. |
| xval(Cross Validation) | 10 | It will cross validate 10 times. |

**Growing CART Decision Tree for Train data:**





**Pruning the tree for train data: Identifying complexity parameter**

> printcp(Train\_DT\_CART)

Classification tree:

rpart(formula = Personal.Loan ~ Age..in.years. + Experience..in.years. +

Income..in.K.month. + Family.members + CCAvg + Education +

Mortgage + Securities.Account + CD.Account + Online + CreditCard,

data = Train\_CART, method = "class", control = r.ctrl)

Variables actually used in tree construction:

[1] CCAvg CD.Account Education Family.members

[5] Income..in.K.month.

Root node error: 336/3500 = 0.096

n= 3500

CP nsplit rel error xerror xstd

1 0.318452 0 1.00000 1.00000 0.051870

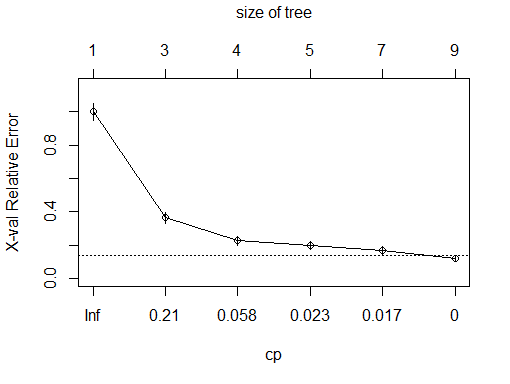
2 0.139881 2 0.36310 0.36012 0.032167

3 0.023810 3 0.22321 0.22321 0.025497

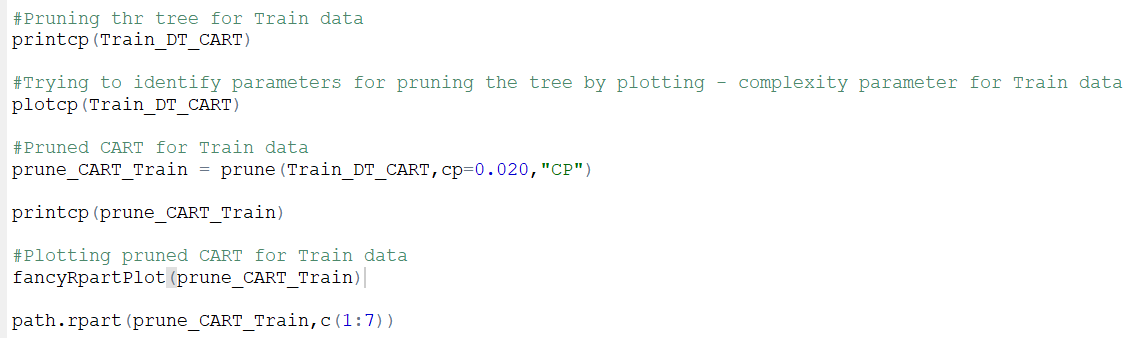
4 0.022321 4 0.19940 0.17857 0.022855

5 0.013393 6 0.15476 0.16071 0.021701

6 0.000000 8 0.12798 0.13393 0.019836

 **Plotting for complexity parameter**

**Deciding complexity parameter for Train data:**



> prune\_CART\_Train = prune(Train\_DT\_CART,cp=0.020,"CP")

> printcp(prune\_CART\_Train)

Classification tree:

rpart(formula = Personal.Loan ~ Age..in.years. + Experience..in.years. +

Income..in.K.month. + Family.members + CCAvg + Education +

Mortgage + Securities.Account + CD.Account + Online + CreditCard,

data = Train\_CART, method = "class", control = r.ctrl)

Variables actually used in tree construction:

[1] CCAvg CD.Account Education Family.members

[5] Income..in.K.month.

Root node error: 336/3500 = 0.096

n= 3500

CP nsplit rel error xerror xstd

1 0.318452 0 1.00000 1.00000 0.051870

2 0.139881 2 0.36310 0.36012 0.032167

3 0.023810 3 0.22321 0.22321 0.025497

4 0.022321 4 0.19940 0.17857 0.022855

5 0.020000 6 0.15476 0.16071 0.021701

**Pruned Tree for Train data:**

> prune\_CART\_Train

n= 3500

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

\* denotes terminal node

1) root 3500 336 0 (0.904000000 0.096000000)

2) Income..in.K.month.< 113.5 2815 60 0 (0.978685613 0.021314387)

4) CCAvg< 2.95 2602 9 0 (0.996541122 0.003458878) \*

5) CCAvg>=2.95 213 51 0 (0.760563380 0.239436620)

10) CD.Account=0 196 35 0 (0.821428571 0.178571429) \*

11) CD.Account=1 17 1 1 (0.058823529 0.941176471) \*

3) Income..in.K.month.>=113.5 685 276 0 (0.597080292 0.402919708)

6) Education=1 441 47 0 (0.893424036 0.106575964)

12) Family.members=0,1,2 394 0 0 (1.000000000 0.000000000) \*

13) Family.members=3,4 47 0 1 (0.000000000 1.000000000) \*

7) Education=2,3 244 15 1 (0.061475410 0.938524590)

14) Income..in.K.month.< 116.5 22 7 0 (0.681818182 0.318181818) \*

15) Income..in.K.month.>=116.5 222 0 1 (0.000000000 1.000000000) \*

> path.rpart(prune\_CART\_Train,c(1:7))

node number: 1

root

node number: 2

root

Income..in.K.month.< 113.5

node number: 3

root

Income..in.K.month.>=113.5

node number: 4

root

Income..in.K.month.< 113.5

CCAvg< 2.95

node number: 5

root

Income..in.K.month.< 113.5

CCAvg>=2.95

node number: 6

root

Income..in.K.month.>=113.5

Education=1

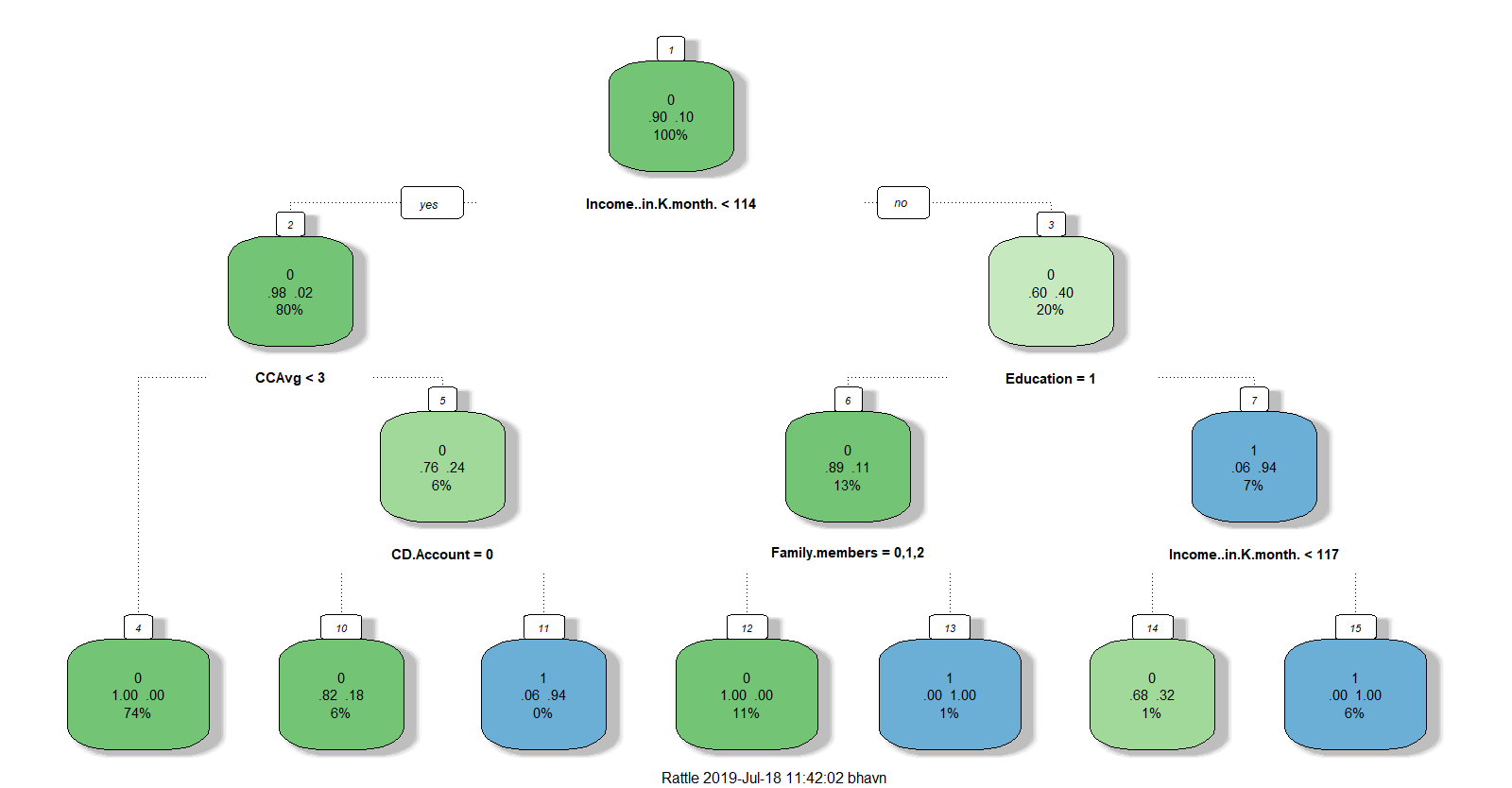
node number: 7

root

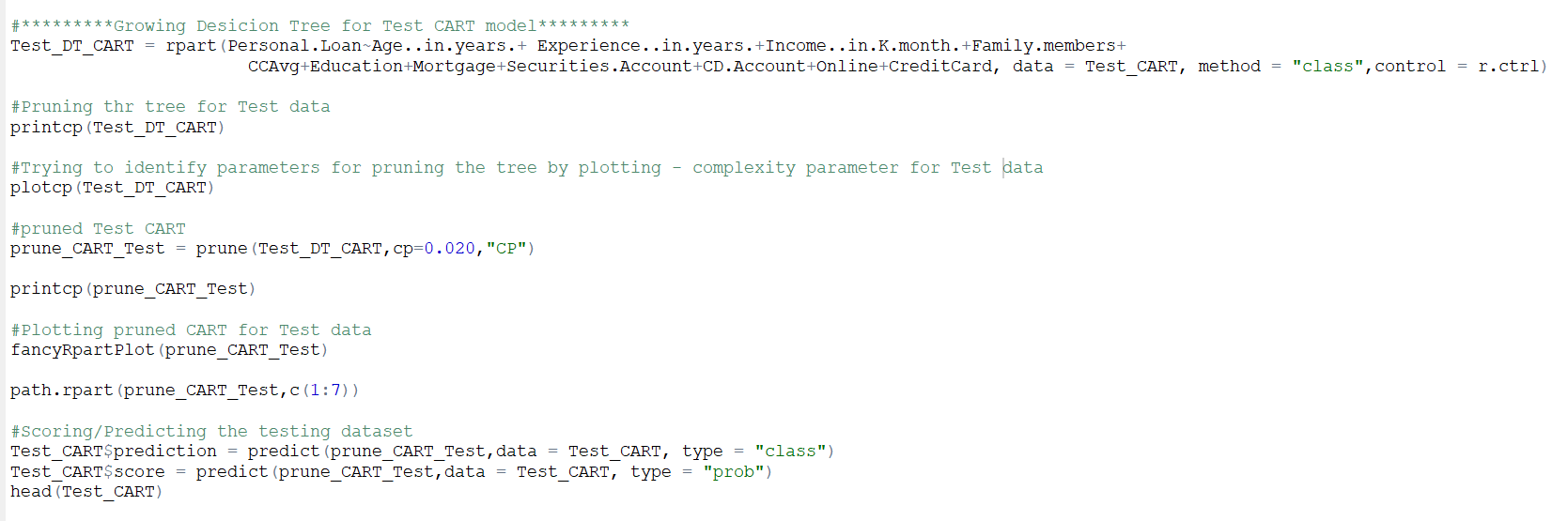
Income..in.K.month.>=113.5

Education=2,3

**Plotting of Pruned decision tree for train data:**



**Following the same procedure for execution of test model and which we will also use for model performance:**

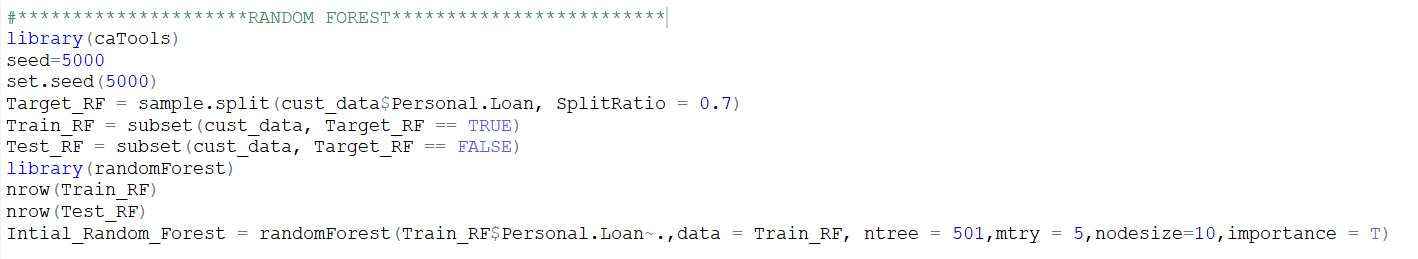


***Observation & Inference for* CART Decision Tree:**

* As per first node, 90% of the customers have income per month less than 114,000$, which is important criterion which should be considered while targeting customers.
* 80% customers spend less than 3000$ as an average spending on credit cards per month.
* 6% customers have a certificate of deposit (CD) account with the bank
* CCAvg, CD.Account , Education ,Family.members ,Income..in.K.month. are critical variables in construction decision tree based on which bank has to decide on issuing personal loans .

**Random Forest:**

**Preparing training and test data for CART:**



> print(Intial\_Random\_Forest)

Call:

randomForest(formula = Train\_RF$Personal.Loan ~ ., data = Train\_RF, ntree = 501, mtry = 3, nodesize = 10, importance = T)

Type of random forest: classification

Number of trees: 501

No. of variables tried at each split: 3

OOB estimate of error rate: 1.29%

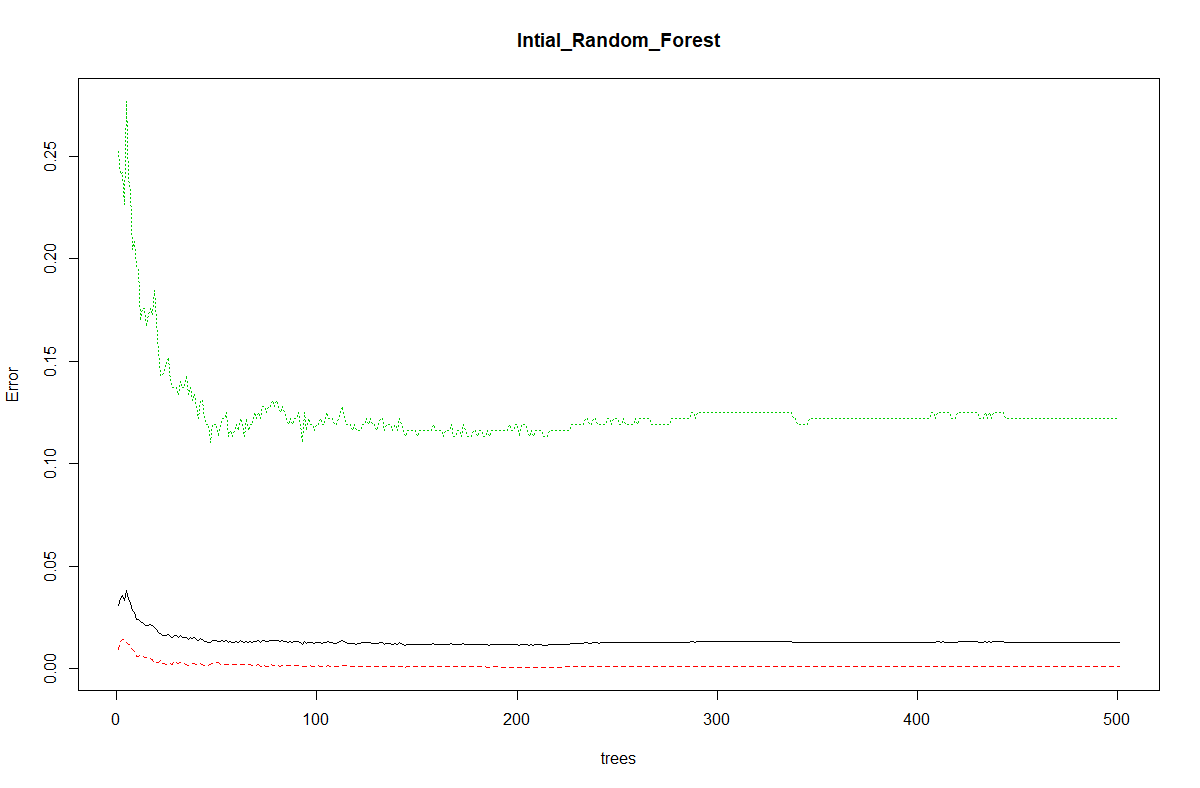
Confusion matrix:

0 1 class.error

0 3160 4 0.001264223

1 41 295 0.122023810

> plot(Intial\_Random\_Forest) **For identifying the right number of trees to build random forest**



> importance(Intial\_Random\_Forest) **Identifying the critical variables which is used to build RF**

0 1 MeanDecreaseAccuracy MeanDecreaseGini

Age..in.years. 16.785100 2.3099077 16.957286 13.129402

Experience..in.years. 15.691579 0.9791646 15.516502 12.750037

Income..in.K.month. 99.650133 80.0787682 106.967040 189.794604

Family.members 64.949156 36.8517205 65.435647 58.406665

CCAvg 32.098724 29.8733543 36.954847 93.365149

Education 102.673020 63.1036195 103.525793 115.656756

Mortgage 9.745540 -2.2180192 8.045072 15.247422

Securities.Account 3.025102 4.8802453 5.830925 1.461378

CD.Account 16.453248 15.2708691 20.818060 32.880052

Online 5.182041 1.2425487 5.253482 2.043560

CreditCard 9.118892 4.5628964 10.598412 3.291220

**Tuning Rf model to find the best mtry**

> set.seed(seed)

> Tuned\_Random\_Forest = tuneRF(x=Train\_RF[,-c(8)],y=Train\_RF$Personal.Loan,mtryStart = 10,stepFactor = 1.5,ntree=151,improve = 0.0001,nodesize=10,trace = T,plot = T,doBest = T,importance = T)

mtry = 10 OOB error = 1.23%

Searching left ...

mtry = 7 OOB error = 1.2%

0.02325581 1e-04

mtry = 5 OOB error = 1.14%

0.04761905 1e-04

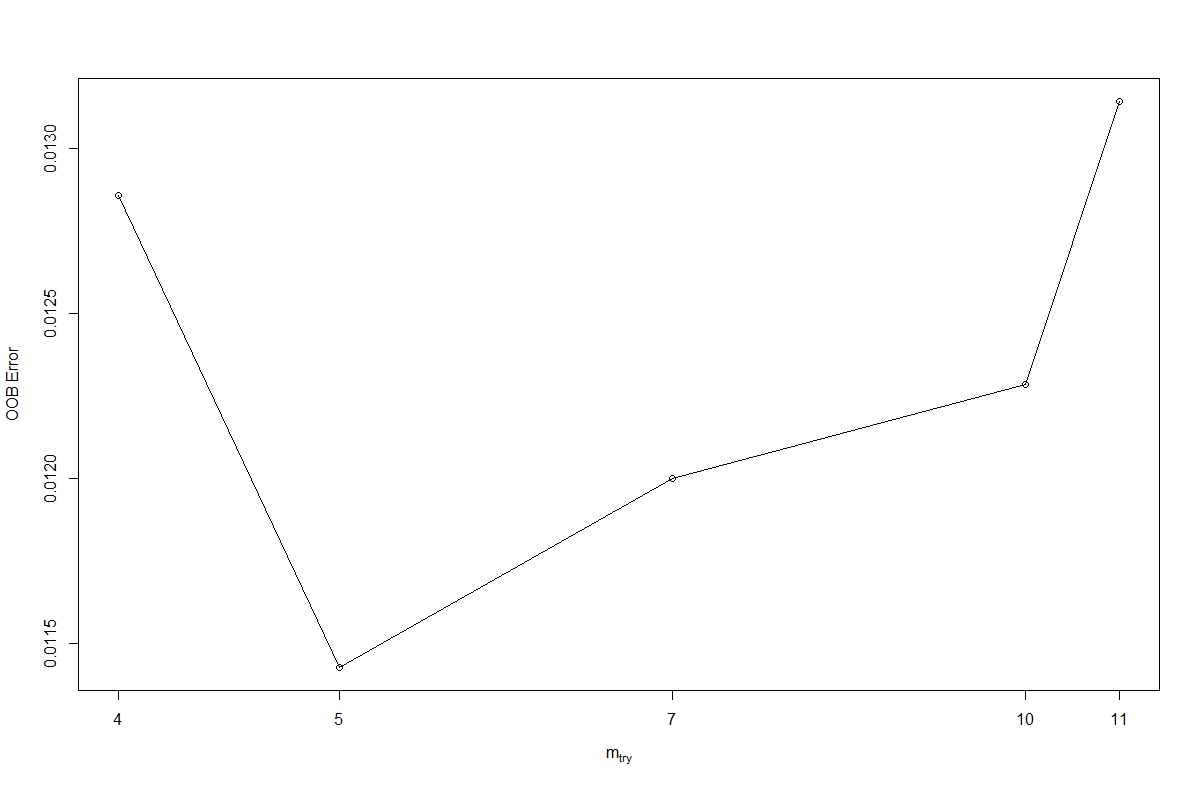
mtry = 4 OOB error = 1.29%

-0.125 1e-04

Searching right ...

mtry = 11 OOB error = 1.31%

-0.15 1e-04



***Observation & Inference for* Random Forest:**

* Likewise, Decision tree Income Annual income of the customer ($000), CCAvg-Avg. spending on credit cards per month ($000), Family size of the customer, Education - Education Level. 1: Undergrad; 2: Graduate; 3: Advanced/Professional – are considered are important variables in constructing Random forest.
* As per, customer demographic information – Thera bank has to target customers with annual income over

117,000 $ dollars, education of over undergraduate level.

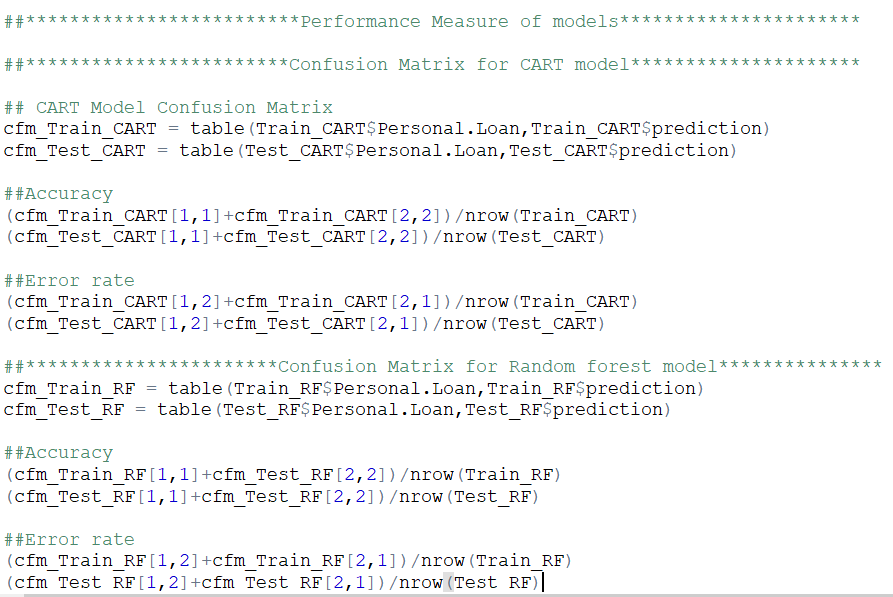
* Higher number of family members > 2 are also target customers.
* Customers who spend on credit card bills with greater than 3000$

**Checking the Performance of all the models (test and train). Using model performance measures:**

From our dataset, we predict target variable which is class output (categorical variable), we will be using the **below** **mentioned Model performance** measures apart from Root Mean square error and Mean absolute error which is used for Regression outputs.

**Confusion matrix, ROC Curves, Kolomogorov-Smirnov chart, AUC, Gini Coefficient, Concordance and discordance ratio**

**Confusion Matrix**



**Confusion Matrix for CART**

cfm\_Train\_CART

0 1

0 3163 1

1 51 285

cfm\_Test\_CART

0 1

0 1349 7

1. 17 127

> ##Accuracy

> (cfm\_Train\_CART[1,1]+cfm\_Train\_CART[2,2])/nrow(Train\_CART)

[1] 0.9851429

> (cfm\_Test\_CART[1,1]+cfm\_Test\_CART[2,2])/nrow(Test\_CART)

[1] 0.984

>

> ##Error rate

> (cfm\_Train\_CART[1,2]+cfm\_Train\_CART[2,1])/nrow(Train\_CART)

[1] 0.01485714

> (cfm\_Test\_CART[1,2]+cfm\_Test\_CART[2,1])/nrow(Test\_CART)

[1] 0.016

**Confusion Matrix for Random forest**

> cfm\_Train\_CART

0 1

0 3163 1

1 51 285

> cfm\_Test\_CART

0 1

0 1349 7

1 17 127

> (cfm\_Train\_RF[1,1]+cfm\_Test\_RF[2,2])/nrow(Train\_RF)

[1] 0.9382857

> (cfm\_Test\_RF[1,1]+cfm\_Test\_RF[2,2])/nrow(Test\_RF)

[1] 0.9846667

> (cfm\_Train\_RF[1,2]+cfm\_Train\_RF[2,1])/nrow(Train\_RF)

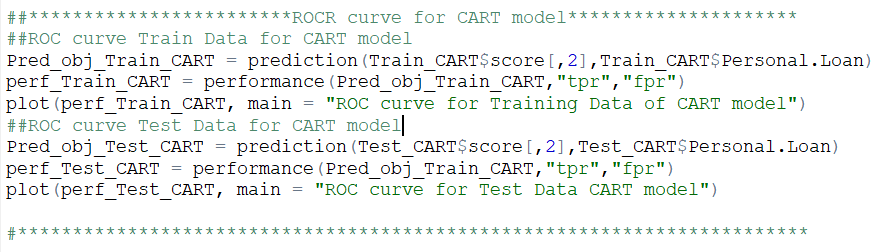
[1] 0.01257143

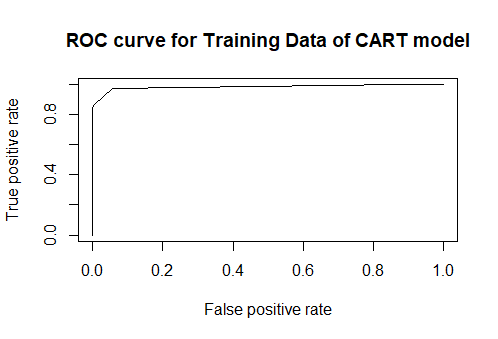
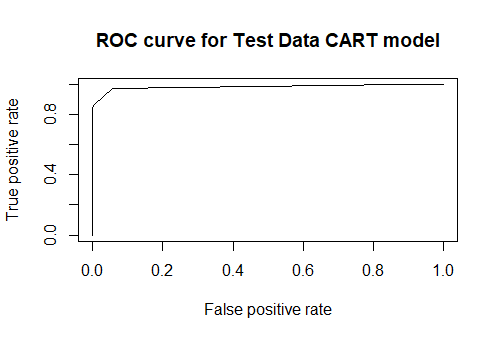
> (cfm\_Test\_RF[1,2]+cfm\_Test\_RF[2,1])/nrow(Test\_RF)

[1] 0.01533333

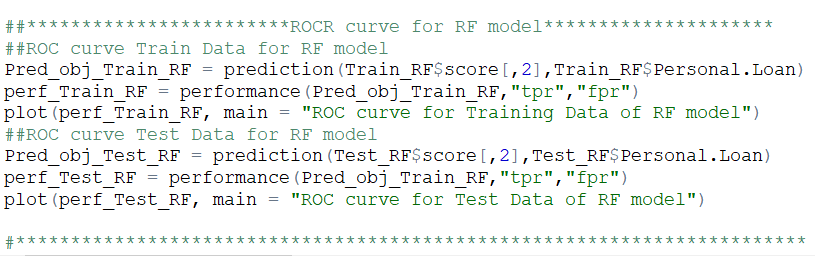
**ROC curve**

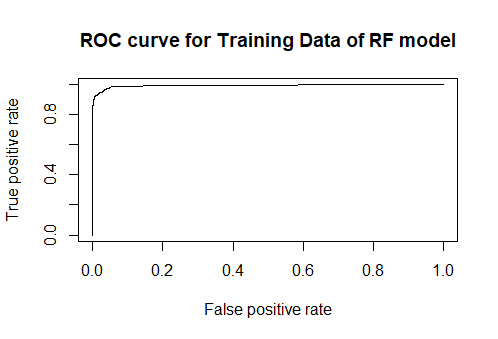
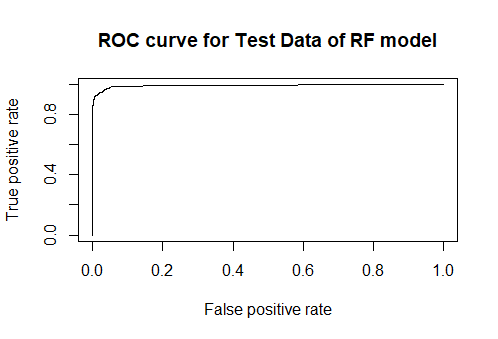
**ROC curve for CART:**



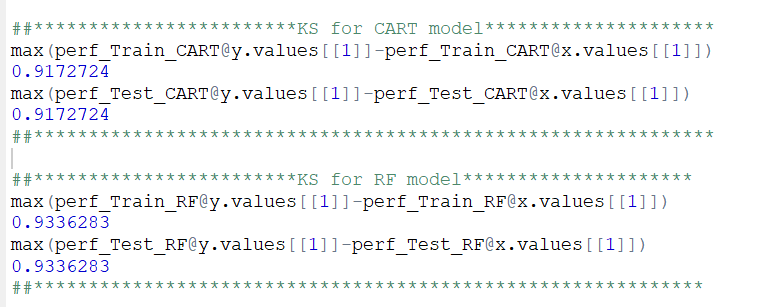
 

**ROC curve for Random forest:**

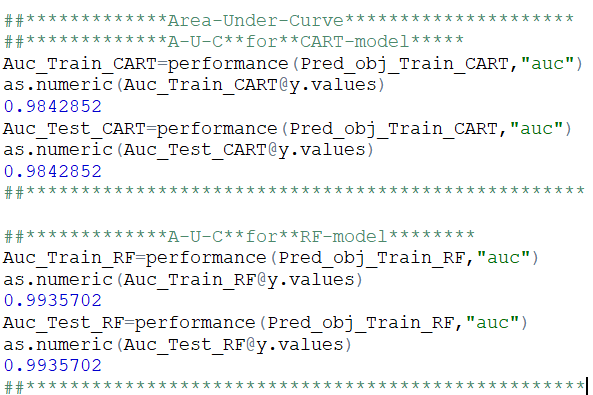


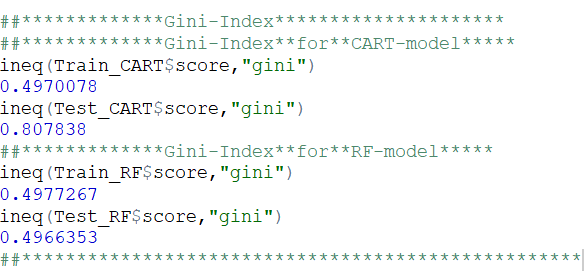
**Kolomogorov-Smirnov chart for CART & Random forest**



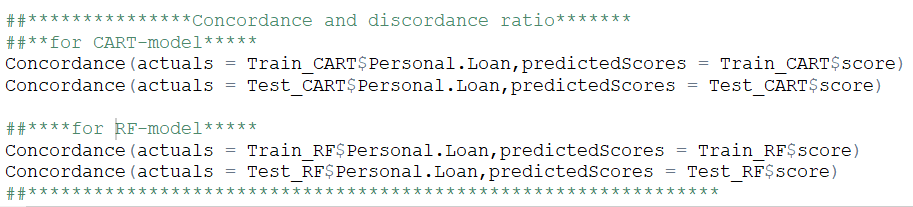
**Area Under Curve for CART & Random forest**



**Gini Coefficient CART & Random forest**



**Concordance and discordance ratio of CART & Random forest:**



> Concordance(actuals = Train\_CART$Personal.Loan,predictedScores = Train\_CART$score)

$Concordance

[1] 0.002031786

$Discordance

[1] 0.9979682

$Tied

[1] 0

$Pairs

[1] 1063104

> Concordance(actuals = Test\_CART$Personal.Loan,predictedScores = Test\_CART$score)

$Concordance

[1] 0.8942355

$Discordance

[1] 0.1057645

$Tied

[1] -1.387779e-17

$Pairs

[1] 195264

> Concordance(actuals = Train\_RF$Personal.Loan,predictedScores = Train\_RF$score)

$Concordance

[1] 0.003944111

$Discordance

[1] 0.9960559

$Tied

[1] 0

$Pairs

[1] 1063104

> Concordance(actuals = Test\_RF$Personal.Loan,predictedScores = Test\_RF$score)

$Concordance

[1] 0.003984349

$Discordance

[1] 0.9960157

$Tied

[1] 0

$Pairs

[1] 195264

***Observation & Inference for* Model performance of CART and Random forest (Train and Test datasets):**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Type of Model** | **Type of Data** | **Scores of Model Performance parameters** | | | | |
| **Confusion Matrix** | **Kolomogorov-Smirnov** | **Area Under Curve** | **Gini Coefficient** | **Concordance and discordance ratio** |
| **CART Decision Tree** | **Training data** | 0.9851429 | 0.9172724 | 0.9842852 | 0.4970078 | 0.9979682 |
| **Test data** | 0.984 | 0.9172724 | 0.9842852 | 0.487838 | 0.8942355 |
| **Random Forest** | **Training data** | 0.9382857 | **0.9336283** | **0.9935702** | **0.4977267** | **0.9960559** |
| **Test data** | 0.9846667 | **0.9336283** | **0.9935702** | **0.4966353** | **0.9960157** |

* From the above data collected over the scores of model performance parameters, we conclude that **Random Forest** is more robust model among the two.
* Model scores look very much satisfactory as they are very accurate on prediction.

**Appendix:**



**EDA, K-means clustering, CART decision tree and Random forest has been built in R and attached is the txt file that contains the code.**