Nov
2017

Mini Project 4

Lekshmi S Vijay

The document consist of analysis of data provided is from a Personal Loans Campaign executed by MyBank using predictive algorithms -CART, Random Forest and Neural Network using R.

LEKSHMI S VIJAY
BACP
Great Lakes Institute of Management

GREAT LAKES
INSTITUTE OF MANAGEMENT, CHENNAI

Table of Contents

Introduction

The data provided is from a Personal Loans Campaign executed by MyBank. The targeted customer were 20000 with an offer of Personal Loans at 10% interest rate. There were 2512 customers out of 20000 responded expressing their need for Personal Loan. These customers are labelled as **Target = 1** and remaining customers are labelled as **Target = 0.** Hence the response rate of the data is 12%. The variables used in the data are:

Column Name	Description
CUST_ID	Customer ID - Unique ID
TARGET	Target Field - 1: Responder, 0: Non-Responder
AGE	Age of the customer in years
GENDER	Gender
BALANCE	Average Monthly Balance
OCCUPATION	Occupation
AGE_BKT	Age Bucket
SCR	Generic Marketing Score
HOLDING_PERIOD	Ability to hold money in the account (Range 0 - 31)
ACC_TYPE	Account Type - Saving / Current
ACC_OP_DATE	Account Open Date
LEN_OF_RLTN_IN_MNTH	Length of Relationship in Months
NO_OF_L_CR_TXNS	No. of Credit Transactions
NO_OF_L_DR_TXNS	No. of Debit Transactions
TOT_NO_OF_L_TXNS	Total No. of Transaction
NO_OF_BR_CSH_WDL_DR_TXNS	No. of Branch Cash Withdrawal Transactions
NO_OF_ATM_DR_TXNS	No. of ATM Debit Transactions
NO_OF_NET_DR_TXNS	No. of Net Debit Transactions
NO_OF_MOB_DR_TXNS	No. of Mobile Banking Debit Transactions
NO_OF_CHQ_DR_TXNS	No. of Cheque Debit Transactions
FLG_HAS_CC	Has Credit Card - 1: Yes, 0: No
AMT_ATM_DR	Amount Withdrawn from ATM
AMT_BR_CSH_WDL_DR	Amount cash withdrawn from Branch
AMT_CHQ_DR	Amount debited by Cheque Transactions
AMT_NET_DR	Amount debited by Net Transactions
AMT_MOB_DR	Amount debited by Mobile Banking Transactions
AMT_L_DR	Total Amount Debited
FLG_HAS_ANY_CHGS	Has any banking charges
AMT_OTH_BK_ATM_USG_CHGS	Amount charged by way of the Other Bank ATM usage
AMT_MIN_BAL_NMC_CHGS	Amount charged by way Minimum Balance not maintained
NO_OF_IW_CHQ_BNC_TXNS	Amount charged by way Inward Cheque Bounce
NO_OF_OW_CHQ_BNC_TXNS	Amount charged by way Outward Cheque Bounce
AVG_AMT_PER_ATM_TXN	Avg. Amt withdrawn per ATM Transaction
AVG_AMT_PER_CSH_WDL_TXN	Avg. Amt withdrawn per Cash Withdrawal Transaction
AVG_AMT_PER_CHQ_TXN	Avg. Amt debited per Cheque Transaction
AVG_AMT_PER_NET_TXN	Avg. Amt debited per Net Transaction
AVG_AMT_PER_MOB_TXN	Avg. Amt debited per Mobile Banking Transaction
FLG_HAS_NOMINEE	Has Nominee - 1: Yes, 0: No
FLG_HAS_OLD_LOAN	Has any earlier loan - 1: Yes, 0: No
random	Random Number

The data was split in to training 70% (N=14000) and testing data 30% (N=6000)to undergo three predictive model -Classification & Regression model(CART), Random Forest and Neural Networking.

The account open date **(**ACC_OP_DATE) was nullified to get better result as we had another variable which was denoting the length of relationships in months (LEN_OF_RLTN_IN_MNTH) which also denotes for how long customer having relationship with the bank. Further the date was in different format and was converted to one format YYYY-DD-MM using "lubridate" library. But it only worked with CART and RF and not in neural network . To maintain the uniformity of the analysis in order to compare the different model with same variables and in addition the length of relationship with bank was available as another variable; the account open date column was nullified in the analysis.

The response rate of Training was 0.12 and testing data 0.12 in all the three techniques. This denotes both training and testing data are balanced. The summary of training data and testing data is give in table 1 and table 2.
The structure of the data is given below:

```
'data.frame':  14000 obs. of  39 variables:
 $ CUST_ID                 : Factor w/ 20000 levels "C1","C10","C100",..:
4239 4965 16663 5173 5593 5709 17037 9785 17861 5721 ...
 $ TARGET                  : int  0 1 0 0 0 0 0 0 0 1 ...
 $ AGE                     : int  21 33 38 28 52 51 55 38 45 53 ...
 $ GENDER                  : Factor w/ 3 levels "F","M","O": 1 2 2 2 2 2 2
2 1 2 ...
 $ BALANCE                 : num  1.74e+05 6.97 2.71e+05 2.59e+06 5.19e+05
...
 $ OCCUPATION              : Factor w/ 4 levels "PROF","SAL","SELF-
EMP",..: 4 2 1 2 2 1 4 1 1 1 ...
 $ AGE_BKT                 : Factor w/ 7 levels "<25",">50","26-30",..: 1
4 5 3 2 2 2 5 6 2 ...
 $ SCR                     : int  642 315 318 232 183 105 850 996 266 940
...
 $ HOLDING_PERIOD          : int  7 8 25 9 7 8 14 4 17 2 ...
 $ ACC_TYPE                : Factor w/ 2 levels "CA","SA": 2 2 1 2 2 2 2 1
1 2 ...
 $ LEN_OF_RLTN_IN_MNTH     : int  50 91 78 168 171 205 147 211 207 94 ...
 $ NO_OF_L_CR_TXNS         : int  2 6 38 1 6 6 7 30 2 7 ...
 $ NO_OF_L_DR_TXNS         : int  2 6 4 1 5 6 5 28 2 6 ...
 $ TOT_NO_OF_L_TXNS        : int  4 12 42 2 11 12 12 58 4 13 ...
 $ NO_OF_BR_CSH_WDL_DR_TXNS: int  1 2 3 1 1 3 2 12 1 2 ...
 $ NO_OF_ATM_DR_TXNS       : int  0 1 1 0 1 1 1 3 0 1 ...
 $ NO_OF_NET_DR_TXNS       : int  0 1 0 0 1 1 1 7 0 1 ...
 $ NO_OF_MOB_DR_TXNS       : int  0 0 0 0 0 0 0 2 0 0 ...
 $ NO_OF_CHQ_DR_TXNS       : int  1 2 0 0 2 1 1 4 1 2 ...
 $ FLG_HAS_CC              : int  1 0 1 0 1 0 0 0 0 0 ...
 $ AMT_ATM_DR              : int  0 13300 11500 0 6600 4800 17300 36000 0
17100 ...
 $ AMT_BR_CSH_WDL_DR       : int  799820 534720 31180 657540 726260 484030
356960 881980 757820 222450 ...
 $ AMT_CHQ_DR              : int  0 15330 0 0 10250 0 0 58620 0 41050 ...
 $ AMT_NET_DR              : num  0 843601 0 0 539503 ...
 $ AMT_MOB_DR              : int  0 0 0 0 0 0 0 153604 0 0 ...
 $ AMT_L_DR                : num  799820 1406951 42680 657540 1282613 ...
```

```
 $ FLG_HAS_ANY_CHGS        : int  0 1 0 0 0 0 0 0 0 1 ...
 $ AMT_OTH_BK_ATM_USG_CHGS : int  0 0 0 0 0 0 0 0 0 0 ...
 $ AMT_MIN_BAL_NMC_CHGS    : int  0 170 0 0 0 0 0 0 0 0 ...
 $ NO_OF_IW_CHQ_BNC_TXNS   : int  0 0 0 0 0 0 0 0 0 1 ...
 $ NO_OF_OW_CHQ_BNC_TXNS   : int  0 0 0 0 0 0 0 0 0 0 ...
 $ AVG_AMT_PER_ATM_TXN     : num  0 13300 11500 0 6600 4800 17300 12000 0
17100 ...
 $ AVG_AMT_PER_CSH_WDL_TXN : num  799820 267360 10393 657540 726260 ...
 $ AVG_AMT_PER_CHQ_TXN     : num  0 7665 0 0 5125 ...
 $ AVG_AMT_PER_NET_TXN     : num  0 843601 0 0 539503 ...
 $ AVG_AMT_PER_MOB_TXN     : num  0 0 0 0 0 ...
 $ FLG_HAS_NOMINEE         : int  1 1 1 1 0 1 1 1 1 1 ...
 $ FLG_HAS_OLD_LOAN        : int  1 0 1 1 0 0 0 0 0 0 ...
 $ random                  : num  0.725 0.389 0.577 0.287 0.76 ...
```

Table 1 training data summary

```
> summary(CART.train)
    CUST_ID          TARGET            AGE        GENDER       BALANCE          OCCUPATION
 C1     :    1   1   :  1745      Min.   :21.00   F: 3776   Min.   :      0   PROF    :3746
 C10    :    1   0   :  12255     1st Qu.:30.00   M:10096   1st Qu.:  64359   SAL     :4066
 C100   :    1                    Median :38.00   O:  128   Median : 232649   SELF-EMP:2555
 C1000  :    1                    Mean   :38.48             Mean   : 509441   SENP    :3633
 C10001 :    1                    3rd Qu.:47.00             3rd Qu.: 655308
 C10002 :    1                    Max.   :55.00             Max.   :8360431
 (Other):13994
  AGE_BKT          SCR        HOLDING_PERIOD  ACC_TYPE   LEN_OF_RLTN_IN_MNTH NO_OF_L_CR_TXNS
 <25  :1186   Min.   :100.0   Min.   : 1.00   CA: 2995   Min.   : 29.0       Min.   : 0.00
 >50  :2161   1st Qu.:228.0   1st Qu.: 7.00   SA:11005   1st Qu.: 79.0       1st Qu.: 6.00
 26-30:2413   Median :366.0    Median :15.00              Median :126.0       Median :10.00
 31-35:2386   Mean   :441.3   Mean   :14.95              Mean   :125.4       Mean   :12.34
 36-40:1989   3rd Qu.:645.0   3rd Qu.:22.00              3rd Qu.:173.0       3rd Qu.:14.00
 41-45:2139   Max.   :998.0   Max.   :31.00              Max.   :221.0       Max.   :75.00
 46-50:1726
 NO_OF_L_DR_TXNS  TOT_NO_OF_L_TXNS NO_OF_BR_CSH_WDL_DR_TXNS NO_OF_ATM_DR_TXNS
NO_OF_NET_DR_TXNS
 Min.   : 0.000   Min.   :  0.00   Min.   : 0.000           Min.   : 0.000    Min.   : 0.000
 1st Qu.: 2.000   1st Qu.:  9.00   1st Qu.: 1.000           1st Qu.: 0.000    1st Qu.: 0.000
 Median : 5.000   Median : 14.00   Median : 1.000           Median : 1.000    Median : 0.000
 Mean   : 6.655   Mean   : 18.99   Mean   : 1.884           Mean   : 1.033    Mean   : 1.181
 3rd Qu.: 7.000   3rd Qu.: 21.00   3rd Qu.: 2.000           3rd Qu.: 1.000    3rd Qu.: 1.000
 Max.   :74.000   Max.   :149.00   Max.   :15.000           Max.   :25.000    Max.   :22.000

 NO_OF_MOB_DR_TXNS NO_OF_CHQ_DR_TXNS   FLG_HAS_CC       AMT_ATM_DR     AMT_BR_CSH_WDL_DR
 Min.   : 0.000    Min.   : 0.000    Min.   :0.0000   Min.   :     0   Min.   :     0
 1st Qu.: 0.000    1st Qu.: 0.000    1st Qu.:0.0000   1st Qu.:     0   1st Qu.:   840
```

```
Median : 0.000    Median : 2.000    Median :0.0000   Median :  6900   Median :335260
Mean   : 0.419    Mean   : 2.138    Mean   :0.3074   Mean   : 11040   Mean   :377608
3rd Qu.: 0.000    3rd Qu.: 4.000    3rd Qu.:1.0000   3rd Qu.: 15900   3rd Qu.:675820
Max.   :25.000    Max.   :15.000    Max.   :1.0000   Max.   :199300   Max.   :999930

  AMT_CHQ_DR        AMT_NET_DR       AMT_MOB_DR        AMT_L_DR       FLG_HAS_ANY_CHGS
Min.   :      0   Min.   :     0   Min.   :     0   Min.   :      0   Min.   :0.0000
1st Qu.:      0   1st Qu.:     0   1st Qu.:     0   1st Qu.: 229690   1st Qu.:0.0000
Median :  23540   Median :     0   Median :     0   Median : 690172   Median :0.0000
Mean   : 126466   Mean   :236058   Mean   : 22314   Mean   : 773486   Mean   :0.1135
3rd Qu.:  72182   3rd Qu.:472295   3rd Qu.:     0   3rd Qu.:1077922   3rd Qu.:0.0000
Max.   :4928640   Max.   :999854   Max.   :199120   Max.   :6514921   Max.   :1.0000

AMT_OTH_BK_ATM_USG_CHGS AMT_MIN_BAL_NMC_CHGS NO_OF_IW_CHQ_BNC_TXNS NO_OF_OW_CHQ_BNC_TXNS
Min.   :  0.000         Min.   :  0.000      Min.   :0.00000       Min.   :0.00000
1st Qu.:  0.000         1st Qu.:  0.000      1st Qu.:0.00000       1st Qu.:0.00000
Median :  0.000         Median :  0.000      Median :0.00000       Median :0.00000
Mean   :  1.155         Mean   :  1.299      Mean   :0.04393       Mean   :0.04529
3rd Qu.:  0.000         3rd Qu.:  0.000      3rd Qu.:0.00000       3rd Qu.:0.00000
Max.   :250.000         Max.   :170.000      Max.   :2.00000       Max.   :2.00000

AVG_AMT_PER_ATM_TXN AVG_AMT_PER_CSH_WDL_TXN AVG_AMT_PER_CHQ_TXN AVG_AMT_PER_NET_TXN
Min.   :    0       Min.   :     0.0        Min.   :     0      Min.   :     0
1st Qu.:    0       1st Qu.:   343.3        1st Qu.:     0      1st Qu.:     0
Median : 5912       Median :145759.0        Median :  8492      Median :     0
Mean   : 7408       Mean   :240425.8        Mean   : 25191      Mean   :177443
3rd Qu.:13600       3rd Qu.:379405.0        3rd Qu.: 28570      3rd Qu.:250758
Max.   :25000       Max.   :999640.0        Max.   :528018      Max.   :999854

AVG_AMT_PER_MOB_TXN FLG_HAS_NOMINEE  FLG_HAS_OLD_LOAN     random
```

```
  Min.   :     0      Min.   :0.0000   Min.   :0.0000   Min.   :0.0001114
  1st Qu.:     0      1st Qu.:1.0000   1st Qu.:0.0000   1st Qu.:0.2486332
  Median :     0      Median :1.0000   Median :0.0000   Median :0.5091718
  Mean   : 20134      Mean   :0.9022   Mean   :0.4927   Mean   :0.5035183
  3rd Qu.:     0      3rd Qu.:1.0000   3rd Qu.:1.0000   3rd Qu.:0.7557175
  Max.   :199120      Max.   :1.0000   Max.   :1.0000   Max.   :0.9999076
```

Table 2 testing data summary

```
> summary(CART.test)
```

```
    CUST_ID         TARGET            AGE         GENDER      BALANCE           OCCUPATION
 C10000 :   1   Min.   :0.0000   Min.   :21.00   F:1657   Min.   :      0   PROF    :1671
 C10007 :   1   1st Qu.:0.0000   1st Qu.:30.00   M:4280   1st Qu.:  65728   SAL     :1789
 C10014 :   1   Median :0.0000   Median :38.00   O:  63   Median : 228505   SELF-EMP:1013
 C1002  :   1   Mean   :0.1278   Mean   :38.28            Mean   : 515844   SENP    :1527
 C10021 :   1   3rd Qu.:0.0000   3rd Qu.:46.00            3rd Qu.: 649902
 C10029 :   1   Max.   :1.0000   Max.   :55.00            Max.   :8360431
 (Other):5994
  AGE_BKT          SCR         HOLDING_PERIOD  ACC_TYPE  LEN_OF_RLTN_IN_MNTH NO_OF_L_CR_TXNS
 <25  : 567   Min.   :100.0   Min.   : 1.00   CA:1246   Min.   : 29         Min.   : 0.00
 >50  : 874   1st Qu.:225.0   1st Qu.: 8.00   SA:4754   1st Qu.: 79         1st Qu.: 6.00
 26-30:1021   Median :360.0   Median :15.00             Median :125         Median :10.00
 31-35:1018   Mean   :437.5   Mean   :14.97             Mean   :125         Mean   :12.37
 36-40: 825   3rd Qu.:641.0   3rd Qu.:22.00             3rd Qu.:171         3rd Qu.:14.00
 41-45: 928   Max.   :999.0   Max.   :31.00             Max.   :221         Max.   :75.00
 46-50: 767
 NO_OF_L_DR_TXNS  TOT_NO_OF_L_TXNS NO_OF_BR_CSH_WDL_DR_TXNS NO_OF_ATM_DR_TXNS NO_OF_NET_DR_TXNS
 Min.   : 0.000   Min.   :  0.00   Min.   : 0.000           Min.   : 0.000    Min.   : 0.000
 1st Qu.: 2.000   1st Qu.:  9.00   1st Qu.: 1.000           1st Qu.: 0.000    1st Qu.: 0.000
 Median : 5.000   Median : 14.00   Median : 1.000           Median : 1.000    Median : 0.000
 Mean   : 6.585   Mean   : 18.95   Mean   : 1.881           Mean   : 1.019    Mean   : 1.154
 3rd Qu.: 7.000   3rd Qu.: 21.00   3rd Qu.: 2.000           3rd Qu.: 1.000    3rd Qu.: 1.000
 Max.   :74.000   Max.   :149.00   Max.   :15.000           Max.   :25.000    Max.   :22.000
```

```
NO_OF_MOB_DR_TXNS NO_OF_CHQ_DR_TXNS   FLG_HAS_CC       AMT_ATM_DR     AMT_BR_CSH_WDL_DR
Min.   : 0.0000   Min.   : 0.000    Min.   :0.0000   Min.   :     0   Min.   :     0
1st Qu.: 0.0000   1st Qu.: 0.000    1st Qu.:0.0000   1st Qu.:     0   1st Qu.:  7980
Median : 0.0000   Median : 2.000    Median :0.0000   Median :  6900   Median :350150
Mean   : 0.3948   Mean   : 2.135    Mean   :0.3007   Mean   : 10873   Mean   :380497
3rd Qu.: 0.0000   3rd Qu.: 4.000    3rd Qu.:1.0000   3rd Qu.: 15700   3rd Qu.:673500
Max.   :25.0000   Max.   :15.000    Max.   :1.0000   Max.   :197000   Max.   :999740

  AMT_CHQ_DR        AMT_NET_DR       AMT_MOB_DR        AMT_L_DR       FLG_HAS_ANY_CHGS
Min.   :      0   Min.   :     0   Min.   :     0   Min.   :      0   Min.   :0.0000
1st Qu.:      0   1st Qu.:     0   1st Qu.:     0   1st Qu.: 253125   1st Qu.:0.0000
Median :  24375   Median :     0   Median :     0   Median : 704192   Median :0.0000
Mean   : 119979   Mean   :240224   Mean   : 22684   Mean   : 774257   Mean   :0.1038
3rd Qu.:  73012   3rd Qu.:482770   3rd Qu.:     0   3rd Qu.:1081062   3rd Qu.:0.0000
Max.   :4891360   Max.   :999854   Max.   :199667   Max.   :6514921   Max.   :1.0000

AMT_OTH_BK_ATM_USG_CHGS AMT_MIN_BAL_NMC_CHGS NO_OF_IW_CHQ_BNC_TXNS NO_OF_OW_CHQ_BNC_TXNS
Min.   :  0.00          Min.   :  0.000      Min.   :0.00          Min.   :0.00000
1st Qu.:  0.00          1st Qu.:  0.000      1st Qu.:0.00          1st Qu.:0.00000
Median :  0.00          Median :  0.000      Median :0.00          Median :0.00000
Mean   :  0.97          Mean   :  1.275      Mean   :0.04          Mean   :0.04233
3rd Qu.:  0.00          3rd Qu.:  0.000      3rd Qu.:0.00          3rd Qu.:0.00000
Max.   :250.00          Max.   :170.000      Max.   :2.00          Max.   :1.00000

AVG_AMT_PER_ATM_TXN AVG_AMT_PER_CSH_WDL_TXN AVG_AMT_PER_CHQ_TXN AVG_AMT_PER_NET_TXN
Min.   :    0       Min.   :     0          Min.   :     0      Min.   :     0
1st Qu.:    0       1st Qu.:  2767          1st Qu.:     0      1st Qu.:     0
Median : 6100       Median :150064          Median :  8952      Median :     0
Mean   : 7410       Mean   :246461          Mean   : 24862      Mean   :182831
3rd Qu.:13400       3rd Qu.:397061          3rd Qu.: 28606      3rd Qu.:272995
Max.   :25000       Max.   :999640          Max.   :537842      Max.   :999854

AVG_AMT_PER_MOB_TXN FLG_HAS_NOMINEE  FLG_HAS_OLD_LOAN     random
Min.   :     0      Min.   :0.0000   Min.   :0.0000   Min.   :0.0000114
1st Qu.:     0      1st Qu.:1.0000   1st Qu.:0.0000   1st Qu.:0.2467119
Median :     0      Median :1.0000   Median :0.0000   Median :0.4988400
Mean   : 20701      Mean   :0.8987   Mean   :0.4935   Mean   :0.4982339
```

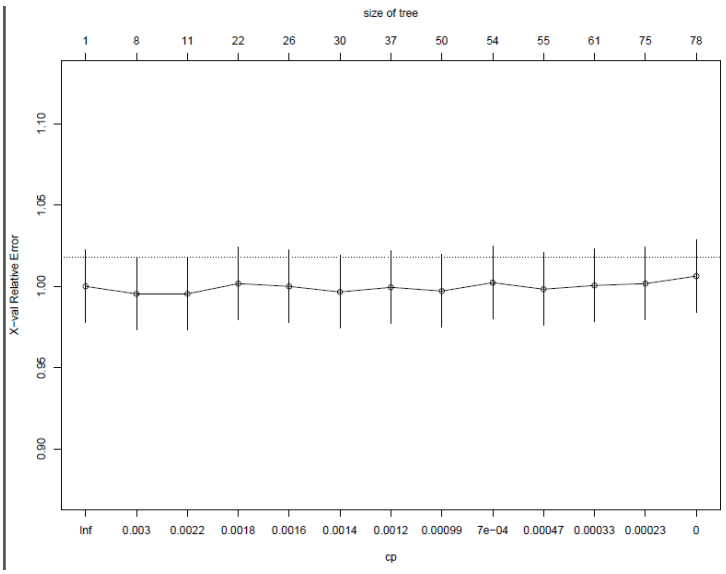
```
 3rd Qu.:     0      3rd Qu.:1.0000   3rd Qu.:1.0000   3rd Qu.:0.7499557
 Max.   :199667      Max.   :1.0000   Max.   :1.0000   Max.   :0.9999471
```

From the above summary table of training and testing data it is clear that both the data are close to equal in all the variables (highlighted above in the table) and hence the training data and testing data are well balanced during the split.

I. Classification & Regression model (CART)

The CART technique was performed in 14000 training and 6000 testing data with following control parameters:

```
> r.ctrl = rpart.control(minsplit=100, minbucket = 10, cp = 0, xval = 10)
> m1 <- rpart(formula = TARGET ~ .,
+             data = CART.train[,-1], method = "class",
+             control = r.ctrl)
> library(rattle)
> library(RColorBrewer)
> fancyRpartPlot(m1)
> printcp(m1)

Classification tree:
rpart(formula = TARGET ~ ., data = CART.train[, -1], method = "class",
    control = r.ctrl)

Variables actually used in tree construction:
 [1] ACC_TYPE                AGE_BKT                 AMT_ATM_DR
 [4] AMT_BR_CSH_WDL_DR       AMT_CHQ_DR              AMT_L_DR
 [7] AMT_NET_DR              AVG_AMT_PER_ATM_TXN     AVG_AMT_PER_CHQ_TXN
[10] AVG_AMT_PER_CSH_WDL_TXN AVG_AMT_PER_MOB_TXN     AVG_AMT_PER_NET_TXN
[13] BALANCE                 FLG_HAS_CC              GENDER
[16] HOLDING_PERIOD          LEN_OF_RLTN_IN_MNTH     NO_OF_ATM_DR_TXNS
[19] NO_OF_L_CR_TXNS         NO_OF_L_DR_TXNS         NO_OF_NET_DR_TXNS
[22] OCCUPATION              SCR                     TOT_NO_OF_L_TXNS

Root node error: 1745/14000 = 0.12464

n= 14000

           CP nsplit rel error  xerror     xstd
1  0.00343840      0   1.00000 1.00000 0.022397
2  0.00267431      7   0.97364 0.99542 0.022353
3  0.00188293     10   0.96562 0.99542 0.022353
4  0.00171920     21   0.94097 1.00172 0.022414
5  0.00143266     25   0.93410 1.00000 0.022397
6  0.00133715     29   0.92837 0.99656 0.022364
7  0.00114613     36   0.91805 0.99943 0.022392
8  0.00085960     49   0.90315 0.99713 0.022370
9  0.00057307     53   0.89971 1.00229 0.022419
10 0.00038204     54   0.89914 0.99828 0.022381
11 0.00028653     60   0.89685 1.00057 0.022403
12 0.00019102     74   0.89284 1.00172 0.022414
13 0.00000000     77   0.89226 1.00630 0.022458
> plotcp(m1)
```



Since the Target 1 is minimum the root node shows the error as 0.12. The pruning was done at cp value 0.0027 which has 7 split.

```
> opt <- m1$cptable[which.min(m1$cptable[,"xerror"]),"CP"]
> ptree<- prune(m1, cp= opt)
> printcp(ptree)

Classification tree:
rpart(formula = TARGET ~ ., data = CART.train[, -1], method = "class",
    control = r.ctrl)

Variables actually used in tree construction:
[1] AVG_AMT_PER_NET_TXN GENDER              NO_OF_L_CR_TXNS
NO_OF_L_DR_TXNS
[5] OCCUPATION          TOT_NO_OF_L_TXNS

Root node error: 1745/14000 = 0.12464

n= 14000

         CP nsplit rel error  xerror     xstd
1 0.0034384      0   1.00000 1.00000 0.022397
2 0.0026743      7   0.97364 0.99542 0.022353
```

Using the prunned the tree the class was predicted.

```
> CART.train$predict.class <- predict(ptree, CART.train, type="class")
> CART.train$predict.score <- predict(ptree, CART.train, type="prob")
> with(CART.train, table(TARGET, predict.class))
      predict.class
TARGET     0     1
     0 12190    65
     1  1634   111
> (65+1634)/14000####Classification Error
[1] 0.1213571
> 12190/(12190+65)###Specificity
[1] 0.994696
> 111/(1634+111)###sesnitivity
[1] 0.06361032
```

Model Performance

- Deciling and Ranking the code

```
> decile <- function(x){
+   deciles <- vector(length=10)
+   for (i in seq(0.1,1,.1)){
+     deciles[i*10] <- quantile(x, i, na.rm=T)
+   }
+   return (
+     ifelse(x<deciles[1], 1,
+            ifelse(x<deciles[2], 2,
+                   ifelse(x<deciles[3], 3,
+                          ifelse(x<deciles[4], 4,
+                                 ifelse(x<deciles[5], 5,
+                                        ifelse(x<deciles[6], 6,
+                                               ifelse(x<deciles[7], 7,
+
ifelse(x<deciles[8], 8,
```

```
+
ifelse(x<deciles[9], 9, 10
+                                                          ))))))))))
+ }
> class(CART.train$predict.score)
[1] "matrix"
> CART.train$deciles <- decile(CART.train$predict.score[,2])
> View(CART.train)
> library(data.table)
> library(scales)
> tmp_DT = data.table(CART.train)
> rank <- tmp_DT[, list(
+   cnt = length(TARGET),
+   cnt_resp = sum(TARGET),
+   cnt_non_resp = sum(TARGET == 0)) ,
+   by=deciles][order(-deciles)]
> rank$rrate <- round(rank$cnt_resp / rank$cnt,4);
> rank$cum_resp <- cumsum(rank$cnt_resp)
> rank$cum_non_resp <- cumsum(rank$cnt_non_resp)
> rank$cum_rel_resp <- round(rank$cum_resp / sum(rank$cnt_resp),4);
> rank$cum_rel_non_resp <- round(rank$cum_non_resp /
sum(rank$cnt_non_resp),4);
> rank$ks <- abs(rank$cum_rel_resp - rank$cum_rel_non_resp) * 100;
> rank$rrate <- percent(rank$rrate)
> rank$cum_rel_resp <- percent(rank$cum_rel_resp)
> rank$cum_rel_non_resp <- percent(rank$cum_rel_non_resp)
> View(rank)
```

	deciles	cnt	cnt_resp	cnt_non_resp	rrate	cum_resp	cum_non_resp	cum_rel_resp	cum_rel_non_resp	ks
1	10	5623	1006	4617	17.89%	1006	4617	57.60%	37.70%	19.98
2	6	8377	739	7638	8.82%	1745	12255	100.00%	100.00%	0

Hence we can see from above table that there are only two deciles and the KS is very low.

```
> library(ROCR)
> library(ineq)
> pred <- prediction(CART.train$predict.score[,2], CART.train$TARGET)
> perf <- performance(pred, "tpr", "fpr")
> plot(perf)
> KS <- max(attr(perf, 'y.values')[[1]]-attr(perf, 'x.values')[[1]])
> auc <- performance(pred,"auc");
> auc <- as.numeric(auc@y.values)
> gini = ineq(CART.train$predict.score[,2], type="Gini")
> auc
[1] 0.6223291
> KS
[1] 0.1997601
> gini
[1] 0.2141633
```

From the above gini coefficient and Ks are low.

Testing on test sample the following were obtained

```
> CART.test$predict.class <- predict(ptree, CART.test, type="class")
> CART.test$predict.score <- predict(ptree, CART.test, type="prob")
> CART.test$deciles <- decile(CART.test$predict.score[,2])
> View(CART.test)
> tmp_DT = data.table(CART.test)
```

```
> h_rank <- tmp_DT[, list(
+   cnt = length(TARGET),
+   cnt_resp = sum(TARGET),
+   cnt_non_resp = sum(TARGET == 0)) ,
+   by=deciles][order(-deciles)]
> h_rank$rrate <- round(h_rank$cnt_resp / h_rank$cnt,4);
> h_rank$cum_resp <- cumsum(h_rank$cnt_resp)
> h_rank$cum_non_resp <- cumsum(h_rank$cnt_non_resp)
> h_rank$cum_rel_resp <- round(h_rank$cum_resp / sum(h_rank$cnt_resp),4);
> h_rank$cum_rel_non_resp <- round(h_rank$cum_non_resp /
sum(h_rank$cnt_non_resp),4);
> h_rank$ks <- abs(h_rank$cum_rel_resp - h_rank$cum_rel_non_resp)*100;
> h_rank$rrate <- percent(h_rank$rrate)
> h_rank$cum_rel_non_resp <- percent(h_rank$cum_rel_non_resp)
> h_rank$cum_rel_resp <- percent(h_rank$cum_rel_resp)
> View(h_rank)
```

Testing data KS_orginal data

	deciles	cnt	cnt_resp	cnt_non_resp	rrate	cum_resp	cum_non_resp	cum_rel_resp	cum_rel_non_resp	ks
1	10	2400	411	1989	17.13%	411	1989	53.60%	38%	15.58
2	6	3600	356	3244	9.89%	767	5233	100.00%	100%	0

```
> pred <- prediction(CART.test$predict.score[,2], CART.test$TARGET)
> perf <- performance(pred, "tpr", "fpr")
> KS <- max(attr(perf, 'y.values')[[1]]-attr(perf, 'x.values')[[1]])
> auc <- performance(pred,"auc");
> auc <- as.numeric(auc@y.values)
> gini = ineq(CART.test$predict.score[,2], type="Gini")
> with(CART.test, table(TARGET, predict.class))
      predict.class
TARGET    0    1
     0 5192   41
     1  716   51
> (41+716)/6000#####Classification error
[1] 0.1261667
> (51)/(51+716)###Sensitivity
[1] 0.06649283
> (5192)/(5192+41)###Specificity
[1] 0.9921651
> auc
[1] 0.5987159
> KS
[1] 0.1557661
> gini
[1] 0.2158261
```

Model performance Techniques	Training	Testing
Classification error	0.1213571	0.1261667
Sensitivity	0.06361032	0.06649283
Specificity	0.994696	0.9921651
KS	0.1997601	0.1557661
Gini	0.2141633	0.2158261
AUC	0.6223291	0.5987159

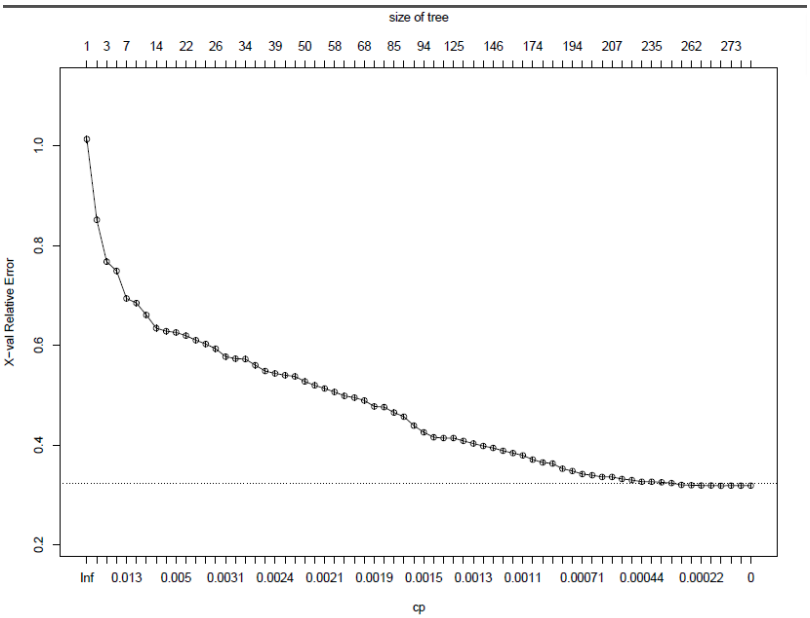
Though there is no much difference seen between training and testing data , the sensitivity, KS and Gini are too low . Hence this model is poor in predicting the target variable 1. Hence oversampling of the Target 1 was done.

Oversampling in CART

The Target 1 was increased such that number of target 1 is equal to number of target 0, using ROSE library in R.

```
> library(ROSE)
Loaded ROSE 0.0-3

Warning message:
package 'ROSE' was built under R version 3.4.2
> ?ROSE
> ?ovun.sample
> table(CART.train$TARGET)

    0     1
12255  1745
> N=12255*2
> N
[1] 24510
> CART.trainover <- ovun.sample(TARGET ~ ., data = CART.train, method =
"over",N = 24510)$data
> table(CART.trainover$TARGET)`

    0     1
12255 12255
```

The tree had 286 split and lowest cp value was taken to prune the tree .



The tree was pruned under lowest cross validation error of cp 0.0026 and it has 34 splits .

```
Classification tree:
rpart(formula = TARGET ~ ., data = CART.trainover[, -1], method = "class",
    control = r.ctrl)

Variables actually used in tree construction:
 [1] AGE_BKT                 AMT_ATM_DR              AMT_BR_CSH_WDL_DR
 [4] AMT_CHQ_DR              AMT_L_DR                AVG_AMT_PER_CHQ_TXN
 [7] AVG_AMT_PER_CSH_WDL_TXN AVG_AMT_PER_NET_TXN     BALANCE
[10] FLG_HAS_CC              HOLDING_PERIOD          LEN_OF_RLTN_IN_MNTH
[13] NO_OF_L_CR_TXNS         NO_OF_L_DR_TXNS         NO_OF_OW_CHQ_BNC_TXNS
[16] OCCUPATION              SCR

Root node error: 12255/24510 = 0.5

n= 24510

          CP nsplit rel error  xerror      xstd
1  0.1485924      0   1.00000 1.01306 0.0063869
2  0.0838841      1   0.85141 0.85141 0.0063166
3  0.0248062      2   0.76752 0.76752 0.0062125
4  0.0172583      3   0.74272 0.74908 0.0061831
5  0.0092207      6   0.68878 0.69384 0.0060807
6  0.0076975      7   0.67956 0.68470 0.0060617
7  0.0061472     10   0.65647 0.66079 0.0060088
8  0.0051816     13   0.63803 0.63435 0.0059452
9  0.0051408     16   0.62203 0.62831 0.0059299
10 0.0047736     18   0.61175 0.62579 0.0059234
11 0.0041616     21   0.59625 0.61934 0.0059066
12 0.0036720     22   0.59208 0.61020 0.0058822
13 0.0035904     23   0.58841 0.60261 0.0058615
14 0.0031824     25   0.58123 0.59306 0.0058347
15 0.0030192     30   0.56459 0.57732 0.0057888
16 0.0029784     31   0.56157 0.57324 0.0057766
17 0.0029376     33   0.55561 0.57234 0.0057739
18 0.0026743     34   0.55267 0.56010 0.0057362
```

Model performance

- The ranking and decile of training of training and testing data.

Oversampling KS Training data

	deciles	cnt	cnt_resp	cnt_non_resp	rrate	cum_resp	cum_non_resp	cum_rel_resp	cum_rel_non_resp	ks
1	10	4289	3275	1014	76.40%	3275	1014	26.70%	8.30%	18.45
2	9	2040	1536	504	75.30%	4811	1518	39.30%	12.40%	26.87
3	8	1104	795	309	72.00%	5606	1827	45.70%	14.90%	30.83
4	7	3475	2351	1124	67.60%	7957	2951	64.90%	24.10%	40.85
5	6	1832	1154	678	63.00%	9111	3629	74.40%	29.60%	44.74
6	5	3839	1376	2463	35.80%	10487	6092	85.60%	49.70%	35.86
7	4	788	275	513	34.90%	10762	6605	87.80%	53.90%	33.92
8	3	2264	699	1565	30.90%	11461	8170	93.50%	66.70%	26.85
9	2	4086	716	3370	17.50%	12177	11540	99.40%	94.20%	5.19

10	1	793	78	715	9.80%	12255	12255	100.00%	100.00%	0
Oversampling KS Testing data										
	deciles	cnt	cnt_resp	cnt_non_resp	rrate	cum_resp	cum_non_resp	cum_rel_resp	cum_rel_non_resp	ks
1	10	911	270	641	29.60%	270	641	35.20%	12.20%	22.95
2	9	345	74	271	21.40%	344	912	44.80%	17.40%	27.42
3	8	837	152	685	18.20%	496	1597	64.70%	30.50%	34.15
4	7	1194	109	1085	9.10%	605	2682	78.90%	51.20%	27.63
5	5	336	33	303	9.80%	638	2985	83.20%	57.00%	26.14
6	4	647	60	587	9.30%	698	3572	91.00%	68.30%	22.74
7	3	1421	55	1366	3.90%	753	4938	98.20%	94.40%	3.81
8	1	309	14	295	4.50%	767	5233	100.00%	100.00%	0

It is seen that the KS have improved and number of deciles in training and testing data have improved compared to non oversampling data. The highest KS in training data is 44 at 5th decile with 74% response rate. The highest KS in testing data is 34.15 at 3rd decile with 64% response rate.

- Other model performance

Training data

```
> with(CART.trainover, table(TARGET, predict.class))
      predict.class
TARGET    0    1
     0 8626 3629
     1 3144 9111
> (3629+3144)/(8626+3629+3144+9111)###classification error
[1] 0.2763362
> (9111)/(9111+3144)###sensitivity
[1] 0.7434517
> (8626)/(8626+3629)####Specificty
[1] 0.703876
> auc
[1] 0.7621891
> KS
[1] 0.4473276
> gini
[1] 0.2621891
```

Testing data

```
> with(CART.test, table(TARGET, predict.class))
      predict.class
TARGET    0    1
     0 3636 1597
     1  271  496
> (1597+271)/(3636+1597+271+496)###classification error
[1] 0.3113333
> 3636/(3636+1597)###specificity
[1] 0.6948213
> 496/(271+491)###Sensitivity
[1] 0.6509186
```

```
> auc
[1] 0.7149457
> KS
[1] 0.3414967
> gini
[1] 0.3053737
```

Model performance Techniques	Training	Testing	Training	Testing
	Original Data		Oversampled data	
Classification error	0.1213571	0.1261667	0.27	0.3113
Sensitivity	0.06361032	0.06649283	0.74	0.65
Specificity	0.994696	0.9921651	0.70	0.69
KS	0.1997601	0.1557661	0.44	0.34
Gini	0.2141633	0.2158261	0.26	0.30
AUC	0.6223291	0.5987159	0.76	0.71

From the above table it is seen that, though the classification error has increased, specificity has decreased; the sensitivity, KS and Gini coefficient have improved. Hence the oversampled data is better when compared to original data. Further it is noted that in the original data, training and testing have no difference but the oversampled training and testing data have little difference but not huge. Hence we can also confirm that the model is not over fit.

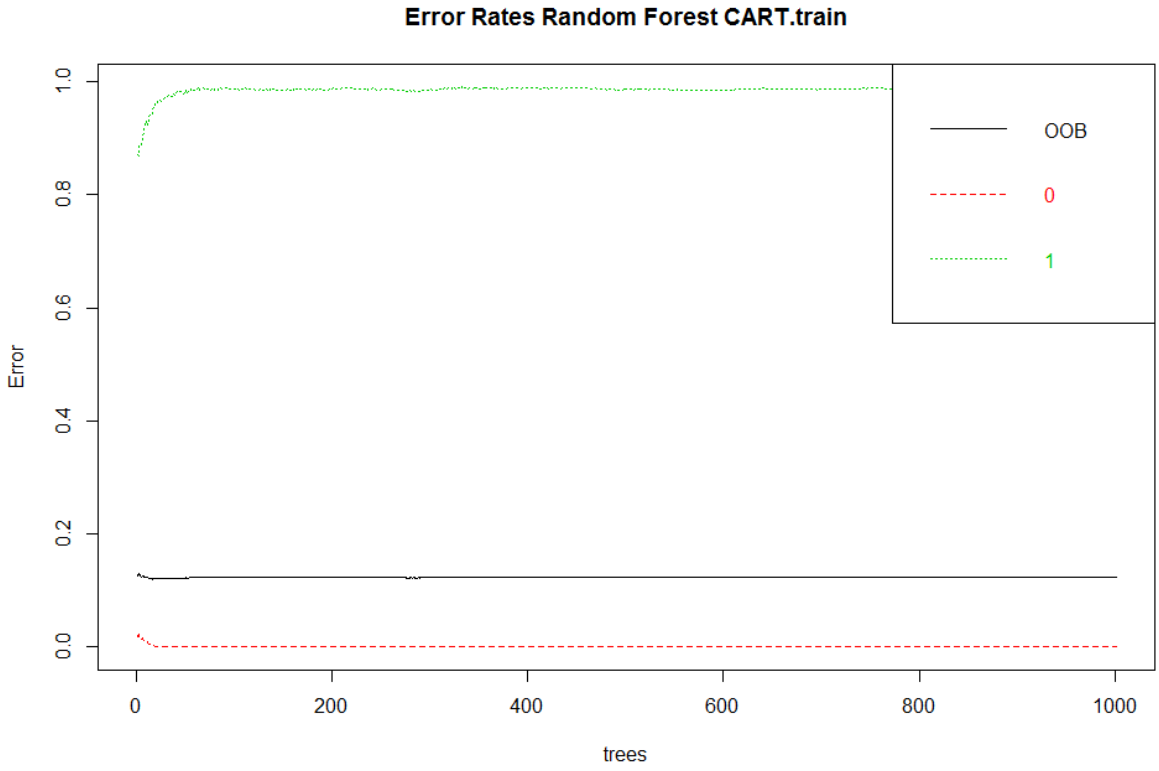
II. Random Forest

TARGET	0	1
Number	12255	1745

Random forest was run initially with ntree=1001, mtry =Sqrt(no of variables)=sqrt(40)=6, nodesize =1. The out of bag error was 12.3%.

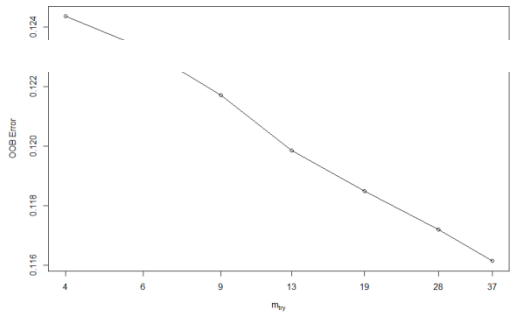
```
> RF <- randomForest(as.factor(TARGET) ~ ., data = CART.train[,-1],
+                    ntree=1001, mtry = 6, nodesize = 100,
+                    importance=TRUE)
> print(RF)

Call:
 randomForest(formula = as.factor(TARGET) ~ ., data = CART.train[,      -
1], ntree = 1001, mtry = 6, nodesize = 100, importance = TRUE)
               Type of random forest: classification
                     Number of trees: 1001
No. of variables tried at each split: 6

        OOB estimate of  error rate: 12.3%
      0  1  class.error
0 12254  1 8.159935e-05
1  1721 24 9.862464e-01
```

The error rate was plotted and seen that at around 200-300 the error is stabilised.



The tree is then tuned with tuneRF with ntreetry=333 (lowest OOB error from the above plot) , mtrystart=6. Following is shown below:

```
> tRF <- tuneRF(x = CART.train[,-c(1,2)],
+               y=as.factor(CART.train$TARGET),
+               mtryStart = 6,
+               ntreeTry=333,
+               stepFactor = 1.5,
+               improve = 0.001,
+               trace=TRUE,
+               plot = TRUE,
+               doBest = TRUE,
+               nodesize = 100,
+               importance=TRUE
+ )
mtry = 6  OOB error = 12.33%
Searching left ...
mtry = 4        OOB error = 12.44%
-0.008690614 0.001
Searching right ...
mtry = 9        OOB error = 12.17%
0.01274623 0.001
mtry = 13       OOB error = 11.99%
0.01525822 0.001
mtry = 19       OOB error = 11.85%
0.011323 0.001
mtry = 28       OOB error = 11.72%
0.01084991 0.001
mtry = 37       OOB error = 11.61%
0.009140768 0.001
```



From the above plot lowest OOB was obtained at mtry 37. The important variable in the tree which has highest mean decrease Gini are Generic Marketing Score, Average Monthly Balance, No. of Debit Transactions, Ability to hold money in the account (Range 0 - 31), No. of Credit Transactions, Total No. of Transaction, occupation and Total Amount Debited (highlighted below).

	0	1	MeanDecreaseAccuracy	MeanDecreaseGini
AGE	6.658257e-04	3.934676e-03	1.073915e-03	19.0452750
GENDER	1.125503e-03	3.439278e-03	1.413349e-03	14.1318771
BALANCE	2.496783e-03	2.864571e-02	5.749077e-03	76.2440064
OCCUPATION	4.427686e-03	3.130026e-02	7.772320e-03	47.0247853
AGE_BKT	1.450280e-03	1.154798e-02	2.711462e-03	35.6778546
SCR	3.447348e-03	3.004499e-02	6.761601e-03	78.5295317
HOLDING_PERIOD	1.626577e-02	1.492595e-02	1.609560e-02	54.6498116
ACC_TYPE	1.269248e-03	1.891748e-04	1.137317e-03	4.2664769
LEN_OF_RLTN_IN_MNTH	1.419039e-03	8.229859e-03	2.267661e-03	44.9835254
NO_OF_L_CR_TXNS	2.901721e-02	-1.056218e-02	2.407279e-02	52.3038073
NO_OF_L_DR_TXNS	9.775766e-02	-1.872772e-02	8.322142e-02	59.8608255
TOT_NO_OF_L_TXNS	4.021954e-02	-2.497089e-02	3.209089e-02	49.5305506
NO_OF_BR_CSH_WDL_DR_TXNS	2.035034e-03	1.767660e-03	2.002427e-03	12.8910016
NO_OF_ATM_DR_TXNS	2.516733e-02	-1.319192e-02	2.038791e-02	20.1918318
NO_OF_NET_DR_TXNS	3.634095e-04	-4.083712e-05	3.123702e-04	1.6639898
NO_OF_MOB_DR_TXNS	2.879447e-04	8.884124e-06	2.525729e-04	1.3217554
NO_OF_CHQ_DR_TXNS	4.113370e-03	-7.068011e-04	3.511524e-03	8.8277528
FLG_HAS_CC	2.750757e-03	2.384519e-02	5.377710e-03	24.3069631
AMT_ATM_DR	1.189529e-02	1.057973e-03	1.054446e-02	30.6500925
AMT_BR_CSH_WDL_DR	7.945726e-03	1.661749e-03	7.157440e-03	30.1087656
AMT_CHQ_DR	7.821507e-03	6.000132e-04	6.922630e-03	23.6060497
AMT_NET_DR	1.673148e-03	1.778030e-03	1.687744e-03	16.1515655
AMT_MOB_DR	3.100137e-03	5.016416e-04	2.781297e-03	12.1500712
AMT_L_DR	2.861276e-02	-1.397859e-02	2.330521e-02	46.5039436
FLG_HAS_ANY_CHGS	9.359105e-05	4.184617e-04	1.335680e-04	3.0938591
AMT_OTH_BK_ATM_USG_CHGS	-3.119093e-06	2.779036e-05	7.684078e-07	0.1369732
AMT_MIN_BAL_NMC_CHGS	6.172011e-06	9.005171e-05	1.666809e-05	0.3953571
NO_OF_IW_CHQ_BNC_TXNS	1.024331e-04	7.402694e-04	1.817369e-04	4.3080113
NO_OF_OW_CHQ_BNC_TXNS	3.872931e-05	8.619321e-04	1.416036e-04	4.7110435
AVG_AMT_PER_ATM_TXN	7.249133e-03	9.482363e-04	6.465185e-03	29.6824424
AVG_AMT_PER_CSH_WDL_TXN	8.980700e-03	1.498015e-03	8.045803e-03	32.0460535
AVG_AMT_PER_CHQ_TXN	9.185759e-03	-2.518257e-03	7.727599e-03	18.8526264
AVG_AMT_PER_NET_TXN	3.023841e-03	2.663614e-03	2.977918e-03	24.7186852
AVG_AMT_PER_MOB_TXN	2.271877e-03	1.604403e-03	2.188695e-03	15.1146877
FLG_HAS_NOMINEE	5.633723e-05	2.480677e-04	8.000466e-05	1.7459695
FLG_HAS_OLD_LOAN	9.515809e-05	3.855383e-04	1.313327e-04	1.9338694
random	-4.971442e-05	-1.833244e-05	-4.500242e-05	18.1091333

Model performance of training and testing data

- The ranking and decile of training and testing data are as follows:
  Training data

	deciles	cnt	cnt_resp	cnt_non_resp	rrate	cum_resp	cum_non_resp	cum_rel_resp	cum_rel_non_resp	ks
1	10	1418	1007	411	71%	1007	411	58%	3%	0.55
2	9	1418	468	950	33%	1475	1361	85%	11%	0.74
3	8	1373	167	1206	12%	1642	2567	94%	21%	0.73
4	7	1426	62	1364	4%	1704	3931	98%	32%	0.66
5	6	1531	26	1505	2%	1730	5436	99%	44%	0.55
6	5	1709	11	1698	1%	1741	7134	100%	58%	0.42
7	4	985	2	983	0%	1743	8117	100%	66%	0.34
8	3	4140	2	4138	0%	1745	12255	100%	100%	0

Testing data

	deciles	cnt	cnt_resp	cnt_non_resp	rrate	cum_resp	cum_non_resp	cum_rel_resp	cum_rel_non_resp	ks
1	10	610	329	281	54%	329	281	43%	5%	0.38
2	9	600	187	413	31%	516	694	67%	13%	0.54
3	8	593	110	483	19%	626	1177	82%	22%	0.6
4	7	613	46	567	8%	672	1744	88%	33%	0.55
5	6	704	43	661	6%	715	2405	93%	46%	0.47
6	5	498	21	477	4%	736	2882	96%	55%	0.41
7	4	941	25	916	3%	761	3798	99%	73%	0.26
8	3	491	2	489	0%	763	4287	99%	82%	0.17
9	2	950	4	946	0%	767	5233	100%	100%	0

From the above two tables the KS is high as 0.74 in training at 2nd decile with cum response rate of 85%and 0.60 in testing at 3rd decile with 82% of cumulative response rate. When you take first three deciles response, it is 71%, 33% and 12% in training data & 54%, 31% and 19% in testing data.

- Confusion Matrix and Statistics

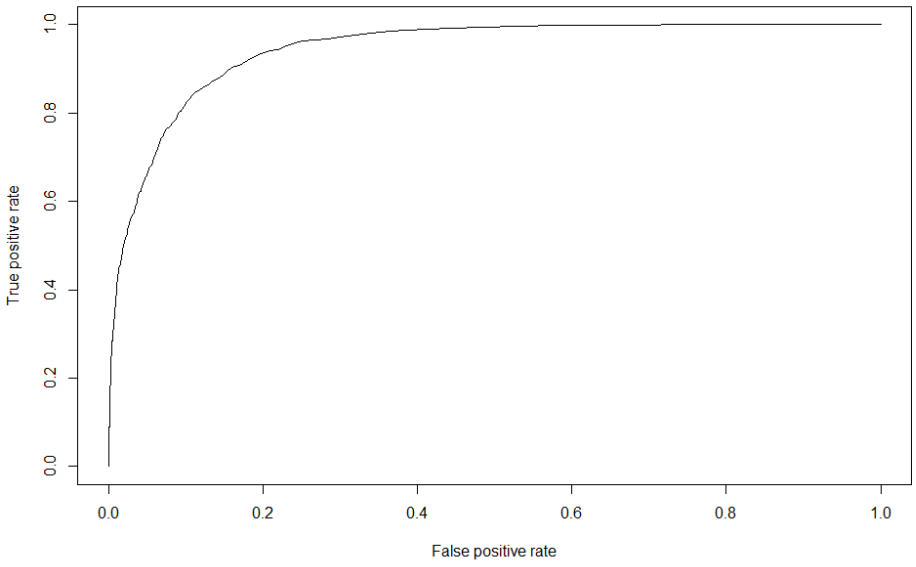
```
Training data:

          Reference
Prediction     0     1
         0 12243  1545
         1    12   200

               Accuracy : 0.8888
                 95% CI : (0.8835, 0.8939)
```

```
    No Information Rate : 0.8754
    P-Value [Acc > NIR] : 5.536e-07

                  Kappa : 0.1823
 Mcnemar's Test P-Value : < 2.2e-16

            Sensitivity : 0.11461
            Specificity : 0.99902
         Pos Pred Value : 0.94340
         Neg Pred Value : 0.88795
             Prevalence : 0.12464
         Detection Rate : 0.01429
   Detection Prevalence : 0.01514
      Balanced Accuracy : 0.55682

       'Positive' Class : 1

   Classification error : 0.11
```

Testing Data:

```
          Reference
Prediction    0    1
         0 5218   15
         1  699   68

               Accuracy : 0.881
                 95% CI : (0.8725, 0.8891)
    No Information Rate : 0.9862
    P-Value [Acc > NIR] : 1

                  Kappa : 0.1385
 Mcnemar's Test P-Value : <2e-16

            Sensitivity : 0.81928
            Specificity : 0.88187
         Pos Pred Value : 0.08866
         Neg Pred Value : 0.99713
             Prevalence : 0.01383
         Detection Rate : 0.01133
   Detection Prevalence : 0.12783
      Balanced Accuracy : 0.85057
       'Positive' Class : 1
   Classification error : 0.11
```

Model performance plot



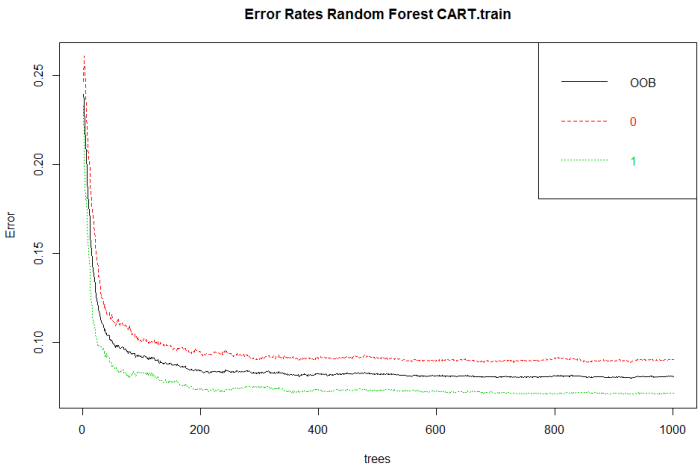
Training	Testing
> auc [1] 0.9439395 > gini [1] 0.7585535 > KS [1] 0.7420654	> auc [1] 0.8659382 > gini [1] 0.725691 > KS [1] 0.593464

The training and testing data has classification error of 0.11. But it is noted that the sensitivity of training data (0.11) is very low when compared to testing data set(0.81). The specificity of training data is 0.99 and for testing 0.88. Further it is seen that the model is different for training and testing data in KS . Hence the model does not hold good and over fitting was observed.

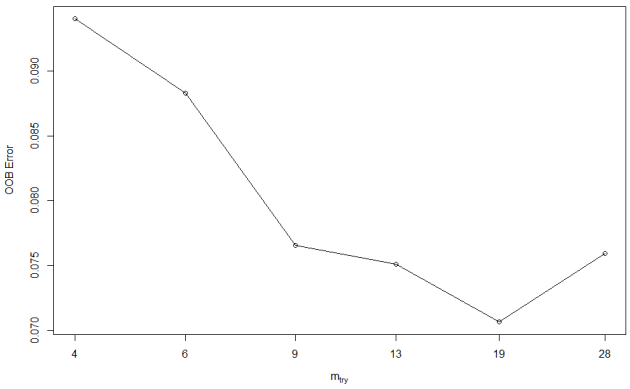
Oversampling the data for RF

Many techniques tried “over” (with different N) , and “both” (both over and under with total N as 20000) , none of the model came perfect. Hence presenting the “over” with N =24510 where Target 1= 12255 and Target 0= 12255 .This model was considered better than the other models tried and used to compare it with the original model above without oversampling.

Initial Random Forest was done with mtry=1001, mtry = 6, nodesize = 100. The OOB rate obtained was 8.09% which is much lesser than the previous model. The error rates were plotted and lowest was obtained again around 200-300.



TuneRF with ntreetry at 301 got mtry-19 with lowest OOB of 7.06%.



The important variable used in the tree which has highest Mean gini decrease are Ability to hold money in the account (Range 0 - 31), Average Monthly Balance, Generic Marketing Score, Age bucket, Length of Relationship in Months, No. of Debit Transactions , No. of Credit Transactions, Total No. of Transaction, occupation, Age, Has Credit Card - 1: Yes, 0: No , Amount Withdrawn from ATM and Amount cash withdrawn from Branch.

	0	1	MeanDecAccuracy	MeanDecreaseGini
AGE	3.447412e-03	0.0470820851	2.526877e-02	257.1859698
GENDER	2.050607e-03	0.0121467511	7.097908e-03	73.7729840
BALANCE	1.136447e-02	0.1305506819	7.095287e-02	509.1933214
OCCUPATION	1.555076e-02	0.1383216846	7.693412e-02	347.9107193
AGE_BKT	6.936799e-03	0.0906631540	4.880296e-02	410.1441480
SCR	1.221363e-02	0.1301416764	7.117784e-02	502.0861435
HOLDING_PERIOD	2.751684e-02	0.1649678875	9.624352e-02	607.3066640
ACC_TYPE	1.728721e-03	0.0094588266	5.594786e-03	41.1115035
LEN_OF_RLTN_IN_MNTH	5.429960e-03	0.0653820071	3.540325e-02	375.5609053
NO_OF_L_CR_TXNS	1.937408e-02	0.0865440679	5.295741e-02	275.9203007
NO_OF_L_DR_TXNS	2.268040e-02	0.1653661724	9.402244e-02	371.6708688
TOT_NO_OF_L_TXNS	1.894117e-02	0.0911420170	5.503285e-02	291.8714413

NO_OF_BR_CSH_WDL_DR_TXNS	1.414941e-03	0.0128254312	7.124216e-03	56.6768626
NO_OF_ATM_DR_TXNS	2.054859e-02	0.0089008252	1.472680e-02	73.3041414
NO_OF_NET_DR_TXNS	6.355344e-04	0.0032111387	1.925061e-03	16.3260752
NO_OF_MOB_DR_TXNS	6.905783e-04	0.0015866488	1.137362e-03	10.5186650
NO_OF_CHQ_DR_TXNS	2.271394e-03	0.0170631051	9.667476e-03	59.5182350
FLG_HAS_CC	1.400803e-02	0.1202726543	6.715428e-02	242.7663474
AMT_ATM_DR	4.840069e-03	0.0986382311	5.175022e-02	259.0180465
AMT_BR_CSH_WDL_DR	4.566667e-03	0.0411197127	2.284392e-02	202.8190071
AMT_CHQ_DR	4.721766e-03	0.0390753301	2.189255e-02	154.8933669
AMT_NET_DR	2.520566e-03	0.0224277900	1.246893e-02	113.2193127
AMT_MOB_DR	2.579552e-03	0.0165111070	9.539651e-03	67.9349935
AMT_L_DR	9.947097e-03	0.0804149486	4.518696e-02	361.3935035
FLG_HAS_ANY_CHGS	8.951478e-04	0.0048076941	2.852127e-03	31.9000201
AMT_OTH_BK_ATM_USG_CHGS	3.502692e-05	0.0000260255	3.056799e-05	0.9729309
AMT_MIN_BAL_NMC_CHGS	5.539509e-05	0.0001693293	1.123808e-04	3.4630959
NO_OF_IW_CHQ_BNC_TXNS	2.029933e-04	0.0020018388	1.102687e-03	14.1720536
NO_OF_OW_CHQ_BNC_TXNS	3.944555e-04	0.0022239867	1.309075e-03	18.7759296
AVG_AMT_PER_ATM_TXN	4.661013e-03	0.0505056317	2.758999e-02	193.4740913
AVG_AMT_PER_CSH_WDL_TXN	3.516118e-03	0.0420339974	2.277898e-02	220.2456260
AVG_AMT_PER_CHQ_TXN	4.391449e-03	0.0302565658	1.732406e-02	144.0801415
AVG_AMT_PER_NET_TXN	2.137359e-03	0.0193331644	1.073775e-02	101.2327579
AVG_AMT_PER_MOB_TXN	3.003593e-03	0.0158430869	9.422215e-03	78.2223418
FLG_HAS_NOMINEE	1.833352e-04	0.0025117695	1.347497e-03	17.8653974
FLG_HAS_OLD_LOAN	3.895129e-04	0.0057096487	3.048968e-03	27.6493053
random	8.889186e-05	0.0092912833	4.689124e-03	87.3789367

Model performance of oversampled data

- The ranking and decile of training and testing data are as follows:

Oversampling- Training data

	deciles	cnt	cnt_resp	cnt_non_resp	rrate	cum_resp	cum_non_resp	cum_rel_resp	cum_rel_non_resp	ks
1	10	2458	2426	32	99%	2426	32	20%	0%	0.2
2	9	2464	2374	90	96%	4800	122	39%	1%	0.38
3	8	2440	2164	276	89%	6964	398	57%	3%	0.54
4	7	2504	2083	421	83%	9047	819	74%	7%	0.67
5	6	2419	1605	814	66%	10652	1633	87%	13%	0.74
6	5	2448	1121	1327	46%	11773	2960	96%	24%	0.72
7	4	2464	364	2100	15%	12137	5060	99%	41%	0.58
8	3	3152	95	3057	3%	12232	8117	100%	66%	0.34
9	2	4161	23	4138	1%	12255	12255	100%	100%	0

Oversampling- Testing data										
	deciles	cnt	cnt_resp	cnt_non_resp	rrate	cum_resp	cum_non_resp	cum_rel_resp	cum_rel_non_resp	ks
1	10	602	476	126	79%	476	126	62%	2%	0.6
2	9	600	122	478	20%	598	604	78%	12%	0.66
3	8	608	57	551	9%	655	1155	85%	22%	0.63
4	7	604	32	572	5%	687	1727	90%	33%	0.57
5	6	594	21	573	4%	708	2300	92%	44%	0.48
6	5	606	24	582	4%	732	2882	95%	55%	0.4
7	4	587	20	567	3%	752	3449	98%	66%	0.32
8	3	616	8	608	1%	760	4057	99%	78%	0.21
9	2	591	2	589	0%	762	4646	99%	89%	0.1
10	1	592	5	587	1%	767	5233	100%	100%	0

From the above two tables the KS is high as 0.74 at 5th decile in training data and 0.66 second decile in testing data having response rate of 87% and 78% respectively. When you take the first three deciles, the response rate is 99%, 96% and 89% in training data & 70%20% and 9% in testing data.

- Confusion matrix of training and testing data

Training data

```
Confusion Matrix and Statistics

          Reference
Prediction     0     1
         0 12243 10837
         1    12  1418

               Accuracy : 0.5574
                 95% CI : (0.5511, 0.5636)
    No Information Rate : 0.5
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.1147
 Mcnemar's Test P-Value : < 2.2e-16

            Sensitivity : 0.11571
            Specificity : 0.99902
         Pos Pred Value : 0.99161
         Neg Pred Value : 0.53046
             Prevalence : 0.50000
         Detection Rate : 0.05785
   Detection Prevalence : 0.05834
      Balanced Accuracy : 0.55736

       'Positive' Class : 1
   Classification error : 0.4426
```

Testing data

```
          Reference
Prediction    0    1
         0 4849  384
         1  205  562

               Accuracy : 0.9018
                 95% CI : (0.894, 0.9092)
    No Information Rate : 0.8423
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.5996
 Mcnemar's Test P-Value : 2.228e-13

            Sensitivity : 0.59408
            Specificity : 0.95944
         Pos Pred Value : 0.73272
         Neg Pred Value : 0.92662
             Prevalence : 0.15767
         Detection Rate : 0.09367
   Detection Prevalence : 0.12783
      Balanced Accuracy : 0.77676

       'Positive' Class : 1
  Classification error :0.09
```

- Other model performance

Training

```
> KS
[1] 0.7427989
> auc
[1] 0.943655
> gini
[1] 0.6317231
```

Testing

```
> auc
[1] 0.9092192
> gini
[1] 0.4218219
> KS
[1] 0.6700497
```

Model performance Techniques	Training	Testing	Training	Testing
	Original Data		**Oversampled data**	
Classification error	0.11	0.11	0.4426	0.09
Sensitivity	0.11	0.81	0.11571	0.59408
Specificity	0.99	0.88	0.99902	0.95944
KS	0.74	0.60	0.7427989	0.6700497
Gini	0.75	0.72	0.6317231	0.4218219
AUC	0.94	0.86	0.943655	0.9092192

From the above table it is seen that classification error in original data seem to be same between training and testing data; but different in oversample data. The sensitivity in both training and testing data are different in both the data. Specificity is more different in original data compared to oversampled data. KS in both original and oversampled data shows an over fit model with bigger difference in original data compared to oversampled data. Similarly gini coefficient also shows a mild over fit in original data and in oversampled data the difference is higher. When the deciles of original data and over sampled data were analysed previously it was seen that original data was less over fit when compared to oversampled data. Hence we can conclude that the original data model is better than oversample model.

III.Neural Network

The training data had following TARGET

TARGET 0=12248, TARGET 1= 1752

The response rate of training and target as follows

```
> sum(NN.train$TARGET) / nrow(NN.train)
[1] 0.1251429
> sum(NN.test$TARGET) / nrow(NN.test)
[1] 0.1266667
```

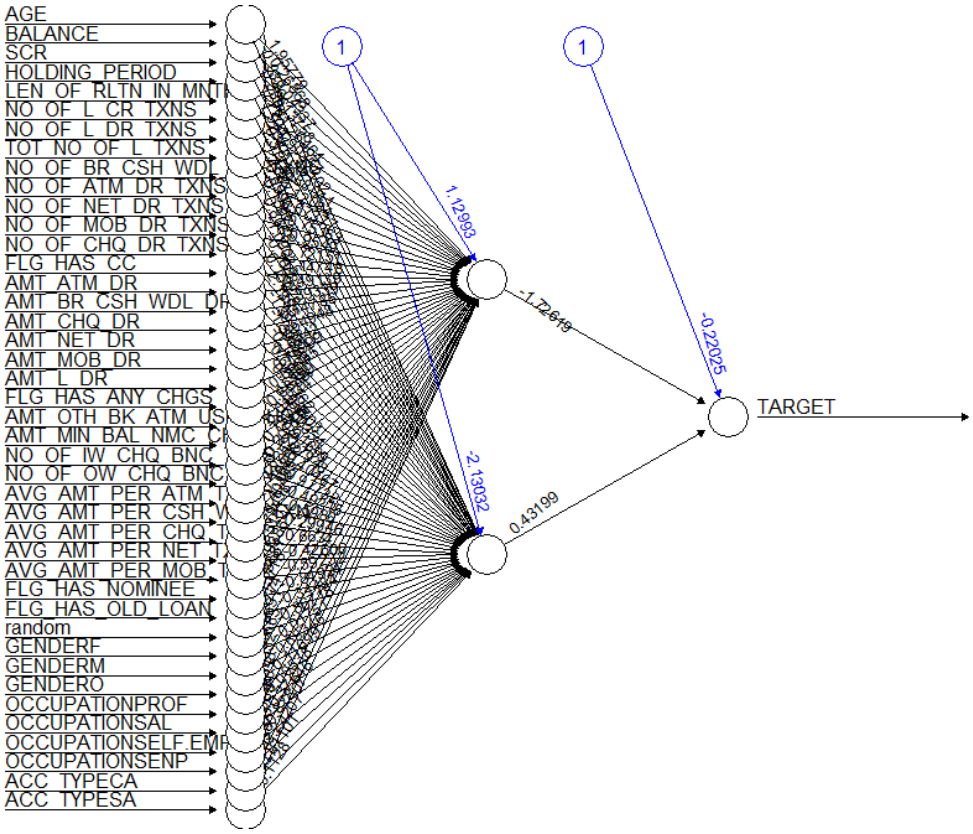
Separate dummy variables are created to convert the categorical variables. The structure of data is as follows:

```
> str(NN.train)
'data.frame':  14000 obs. of  44 variables:
 $ CUST_ID                : Factor w/ 20000 levels "C1","C10","C100",..:
8400 19424 14508 15721 1371 8169 1934 10595 13829 1464 ...
 $ TARGET                 : int  0 0 0 0 0 0 0 0 0 0 ...
 $ AGE                    : int  39 48 35 28 47 44 22 54 54 27 ...
 $ BALANCE                : num  1076252 1195703 3183 176375 347194 ...
 $ SCR                    : int  479 297 199 667 505 258 642 145 811 144
...
```

```
 $ HOLDING_PERIOD          : int  1 27 22 20 10 15 12 7 24 26 ...
 $ LEN_OF_RLTN_IN_MNTH     : int  107 131 60 97 132 168 78 118 44 191 ...
 $ NO_OF_L_CR_TXNS         : int  8 6 32 35 13 13 16 19 7 5 ...
 $ NO_OF_L_DR_TXNS         : int  6 3 4 21 7 7 15 15 1 1 ...
 $ TOT_NO_OF_L_TXNS        : int  14 9 36 56 20 20 31 34 8 6 ...
 $ NO_OF_BR_CSH_WDL_DR_TXNS: int  1 3 0 2 1 3 3 5 1 0 ...
 $ NO_OF_ATM_DR_TXNS       : int  1 0 4 2 1 1 2 2 0 1 ...
 $ NO_OF_NET_DR_TXNS       : int  0 0 0 6 1 1 4 3 0 0 ...
 $ NO_OF_MOB_DR_TXNS       : int  0 0 0 1 0 0 1 1 0 0 ...
 $ NO_OF_CHQ_DR_TXNS       : int  4 0 0 10 4 2 5 4 0 0 ...
 $ FLG_HAS_CC              : int  1 0 0 0 0 1 0 1 1 0 ...
 $ AMT_ATM_DR              : int  19700 0 88700 39300 13900 13600 2900
4900 0 14500 ...
 $ AMT_BR_CSH_WDL_DR       : int  230880 361460 0 823120 511570 854720
389680 397610 682900 0 ...
 $ AMT_CHQ_DR              : int  88810 0 0 3428390 20290 11610 944230
32140 0 0 ...
 $ AMT_NET_DR              : num  0 0 0 639448 941982 ...
 $ AMT_MOB_DR              : int  0 0 0 101468 0 0 74504 16590 0 0 ...
 $ AMT_L_DR                : num  339390 361460 88700 5031726 1487742 ...
 $ FLG_HAS_ANY_CHGS        : int  0 1 0 0 0 0 1 1 0 0 ...
 $ AMT_OTH_BK_ATM_USG_CHGS : int  0 0 0 0 0 0 0 0 0 0 ...
 $ AMT_MIN_BAL_NMC_CHGS    : int  0 0 0 0 0 0 0 0 0 0 ...
 $ NO_OF_IW_CHQ_BNC_TXNS   : int  0 0 0 0 0 0 0 1 0 0 ...
 $ NO_OF_OW_CHQ_BNC_TXNS   : int  0 0 0 0 0 0 1 0 0 0 ...
 $ AVG_AMT_PER_ATM_TXN     : num  19700 0 22175 19650 13900 ...
 $ AVG_AMT_PER_CSH_WDL_TXN : num  230880 120487 0 411560 511570 ...
 $ AVG_AMT_PER_CHQ_TXN     : num  22203 0 0 342839 5072 ...
 $ AVG_AMT_PER_NET_TXN     : num  0 0 0 106575 941982 ...
 $ AVG_AMT_PER_MOB_TXN     : num  0 0 0 101468 0 ...
 $ FLG_HAS_NOMINEE         : int  1 1 1 1 1 1 1 1 0 1 ...
 $ FLG_HAS_OLD_LOAN        : int  0 0 0 0 1 1 0 0 0 0 ...
 $ random                  : num  0.54 0.98 0.916 0.207 0.242 ...
 $ GENDERF                 : num  1 1 0 0 0 0 1 1 0 0 ...
 $ GENDERM                 : num  0 0 1 1 1 1 0 0 1 1 ...
 $ GENDERO                 : num  0 0 0 0 0 0 0 0 0 0 ...
 $ OCCUPATIONPROF          : num  0 0 0 0 0 0 0 1 0 0 ...
 $ OCCUPATIONSAL           : num  0 1 0 0 0 1 0 0 0 1 ...
 $ OCCUPATIONSELF.EMP      : num  0 0 1 0 0 0 1 0 1 0 ...
 $ OCCUPATIONSENP          : num  1 0 0 1 1 0 0 0 0 0 ...
 $ ACC_TYPECA              : num  0 0 1 1 0 0 0 0 0 0 ...
 $ ACC_TYPESA              : num  1 1 0 0 1 1 1 1 1 1 ...
```

```
> nn1 <- neuralnet(formula = TARGET ~ AGE+BALANCE+SCR+HOLDING_PERIOD+
LEN_OF_RLTN_IN_MNTH +NO_OF_L_CR_TXNS+NO_OF_L_DR_TXNS +TOT_NO_OF_L_TXNS +
NO_OF_BR_CSH_WDL_DR_TXNS+NO_OF_ATM_DR_TXNS+NO_OF_NET_DR_TXNS +
NO_OF_MOB_DR_TXNS+NO_OF_CHQ_DR_TXNS+FLG_HAS_CC+AMT_ATM_DR+AMT_BR_CSH_WDL_D
R+AMT_CHQ_DR+AMT_NET_DR+AMT_MOB_DR+AMT_L_DR+FLG_HAS_ANY_CHGS+AMT_OTH_BK_AT
M_USG_CHGS+AMT_MIN_BAL_NMC_CHGS+NO_OF_IW_CHQ_BNC_TXNS+NO_OF_OW_CHQ_BNC_TXN
S+AVG_AMT_PER_ATM_TXN+AVG_AMT_PER_CSH_WDL_TXN+AVG_AMT_PER_CHQ_TXN+AVG_AMT_
PER_NET_TXN+AVG_AMT_PER_MOB_TXN+FLG_HAS_NOMINEE+FLG_HAS_OLD_LOAN+random+GE
NDERF+GENDERM+GENDERO+OCCUPATIONPROF+OCCUPATIONSAL+OCCUPATIONSELF.EMP+OCCU
PATIONSENP+ACC_TYPECA +ACC_TYPESA,
+                  data = NN.train,
+                  hidden = 2,
```

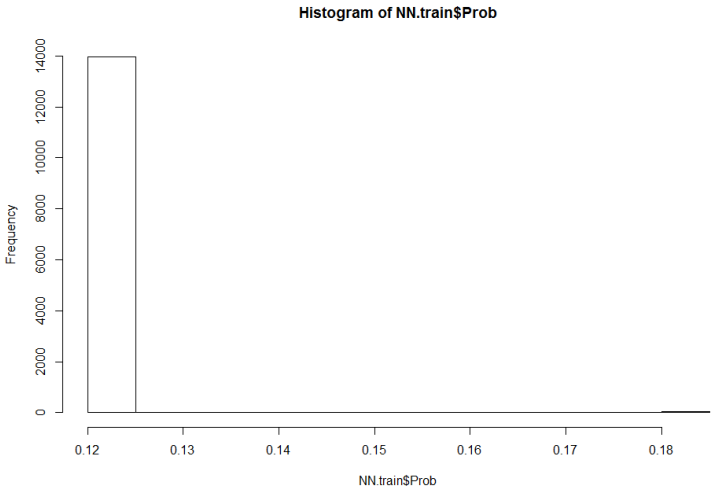
```
+                   err.fct = "sse",
+                   linear.output = FALSE,
+                   lifesign = "full",
+                   lifesign.step = 10,
+                   threshold = 0.01,
+                   stepmax = 2000
+
+ )
```



The distribution of the estimated probabilities

```
          0%           1%           5%          10%          25%
50%          75%
0.1249419862 0.1249419862 0.1249419862 0.1249419862 0.1249419862
0.1249419862 0.1249419862
         90%          95%          98%          99%         100%
0.1249419862 0.1249419862 0.1249419862 0.1249419862 0.1802803574
```

The histogram of the probability is shown below. It is clear from the histogram that the probability does not have a clear distribution.

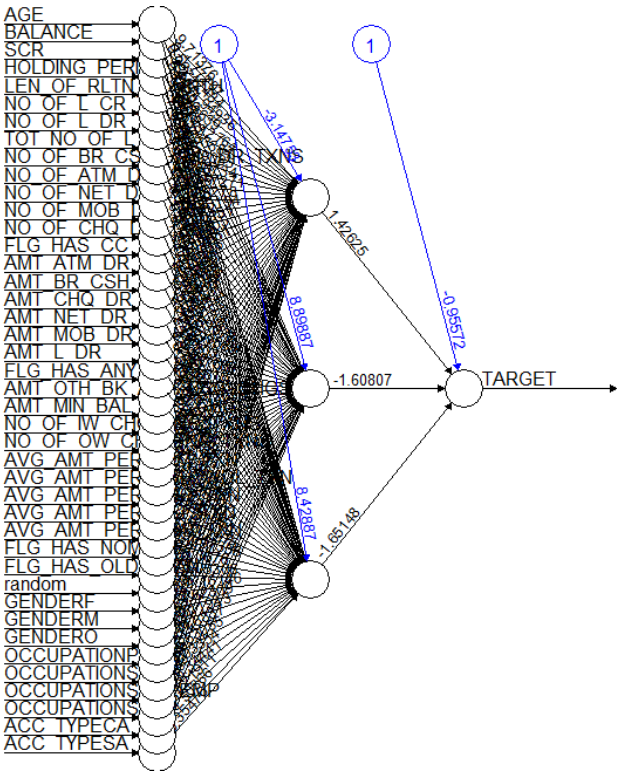


The model performance was seen for the above model but the KS value was 0 with only 1 decile (table below).

	deciles	cnt	cnt_resp	cnt_non_resp	rrate	cum_resp	cum_non_resp	cum_rel_resp	cum_rel_non_resp	ks
1	10	14000	1752	12248	13%	1752	12248	100%	100%	0

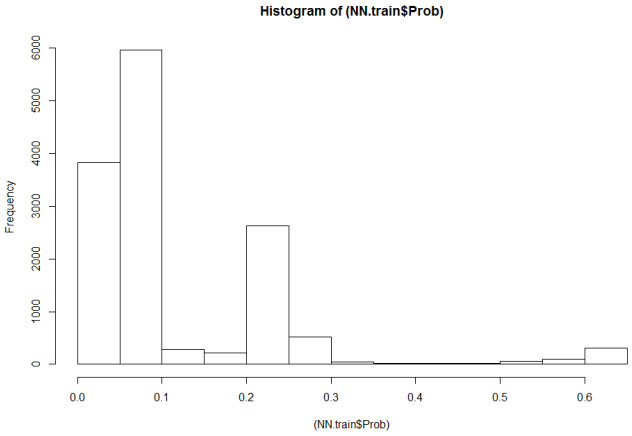
Hence **SCALING OF THE VARIABLES** was done to get better model.

The neural network obtained in the scaling model is shown below with three 1 hidden layer with 3 neurons:



The probability distribution of above network is given below:

```
           0%            1%            5%           10%           25%
50%
0.01455334666 0.01455334666 0.01455334676 0.01455343698 0.03139187359
0.06867617297
          75%           90%           95%           99%          100%
0.22517007027 0.24277218475 0.27773548375 0.61547874804 0.61550889934
```



Model performance of training data

- The ranking and decile of training data are as follows:

	deciles	cnt	cnt_resp	cnt_non_resp	rrate	cum_resp	cum_non_resp	cum_rel_resp	cum_rel_non_resp	ks
1	10	1636	583	1053	36%	583	1053	33%	9%	0.24
2	9	1164	261	903	22%	844	1956	48%	16%	0.32
3	8	1400	272	1128	19%	1116	3084	64%	25%	0.39
4	7	1400	84	1316	6%	1200	4400	68%	36%	0.32
5	6	1400	88	1312	6%	1288	5712	74%	47%	0.27
6	5	1400	123	1277	9%	1411	6989	81%	57%	0.24
7	4	1400	127	1273	9%	1538	8262	88%	67%	0.21
8	3	1400	69	1331	5%	1607	9593	92%	78%	0.14
9	2	1400	86	1314	6%	1693	10907	97%	89%	0.08
10	1	1400	59	1341	4%	1752	12248	100%	100%	0

The third decile has the highest KS with cumulative response of almost 64%.Though there is a crack seen in the bottom of the decile, there are 10 decile with 0.39 as highest KS.

- Confusion Matrix

```
Confusion Matrix and Statistics

          Reference
Prediction    0    1
         0 9637 2611
         1  711 1041
```

```
               Accuracy : 0.7627143
                 95% CI : (0.7555774, 0.7697418)
    No Information Rate : 0.7391429
    P-Value [Acc > NIR] : 0.00000000007504467

                  Kappa : 0.2601265
 Mcnemar's Test P-Value : < 0.00000000000000022204

            Sensitivity : 0.28504929
            Specificity : 0.93129107
         Pos Pred Value : 0.59417808
         Neg Pred Value : 0.78682234
             Prevalence : 0.26085714
         Detection Rate : 0.07435714
   Detection Prevalence : 0.12514286
      Balanced Accuracy : 0.60817018

       'Positive' Class : 1
    Classification error: 0.237
```

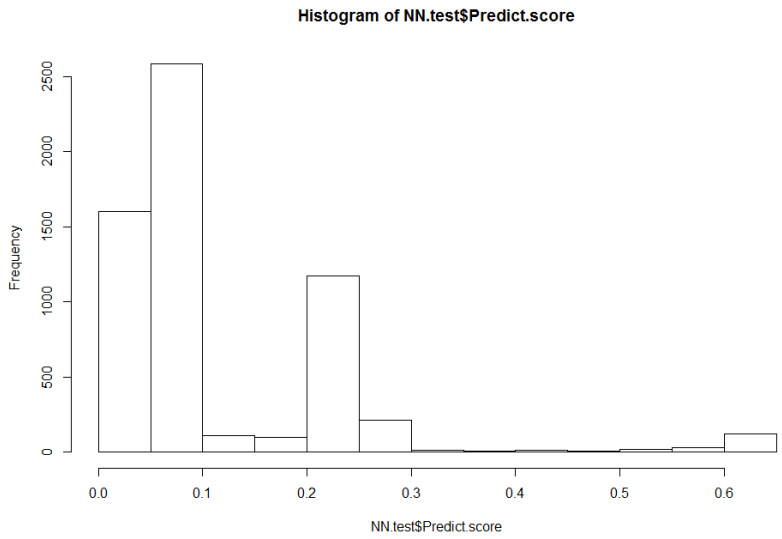
- Other model performance were:

```
> auc
[1] 0.7115791346
> KS
[1] 0.3938753363
> gini
[1] 0.5102136791
```

SCORING THE ABOVE MODEL IN SCALED TESTING DATA SET

The probability distribution is as follows:

```
    0%            1%            5%           10%           25%
50%
0.01455334666 0.01455334666 0.01455334686 0.01455343121 0.03593868340
0.06867618094
          75%           90%           95%           99%          100%
0.22665727920 0.24277218475 0.27773536553 0.61532877172 0.61550889934
```



Model Performance of testing data

- The ranking and decile of testing data are as follows:

	deciles	cnt	cnt_resp	cnt_non_resp	rrate	cum_resp	cum_non_resp	cum_rel_resp	cum_rel_non_resp	ks
1	10	679	219	460	32%	219	460	29%	9%	0.2
2	9	521	119	402	23%	338	862	44%	16%	0.28
3	8	600	108	492	18%	446	1354	59%	26%	0.33
4	7	600	49	551	8%	495	1905	65%	36%	0.29
5	6	600	48	552	8%	543	2457	71%	47%	0.24
6	5	600	60	540	10%	603	2997	79%	57%	0.22
7	4	600	60	540	10%	663	3537	87%	68%	0.19
8	3	600	43	557	7%	706	4094	93%	78%	0.15
9	2	600	35	565	6%	741	4659	98%	89%	0.09
10	1	600	19	581	3%	760	5240	100%	100%	0

The third decile has the highest KS of 0.33 with cumulative response of almost 60%.

- Confusion Matrix

```
Confusion Matrix and Statistics

          Reference
Prediction    0    1
         0 4084 1156
         1  341  419

               Accuracy : 0.7505
                 95% CI : (0.7393473, 0.7614087)
    No Information Rate : 0.7375
    P-Value [Acc > NIR] : 0.01118808

                  Kappa : 0.2267562
 Mcnemar's Test P-Value : < 0.000000000000000222

            Sensitivity : 0.26603175
            Specificity : 0.92293785
         Pos Pred Value : 0.55131579
         Neg Pred Value : 0.77938931
             Prevalence : 0.26250000
         Detection Rate : 0.06983333
   Detection Prevalence : 0.12666667
      Balanced Accuracy : 0.59448480

       'Positive' Class : 1
   Classification error : 0.2495
```

- Other model measurement

```
> auc
[1] 0.7115791346
> KS
[1] 0.3238753363
> gini
[1] 0.5027532078
```

Comparing training and testing model

Model performance Techniques	Training	Testing
Classification error	0.237	0.2495
Sensitivity	0.28504929	0.26603175
Specificity	0.93129107	0.92293785
KS	0.3938	0.3238
Gini	0.5102	] 0.5027
AUC	0.7115	0.7115

From the above table it is clear that there is no difference between training and testing data. There is no over fitting. But it is also seen that the sensitivity of the model is low.

IV. Comparison of three model performance

The table below shows the different model performance of CART, Random Forest and Neural Network. In CART the oversampling data was considered to compare the performance and in Random Forest , the original data without oversampling was considered. From the table it is clear that none of the techniques has given a complete perfect model. In the table, green highlights are indicated to the best values in the table of training and testing data in each model . Thereby we can see that neural network seem to have a better model compare to CART and Random Forest. But it is also observed that neural network sensitivity is low. The sensitivity was better seen in oversampling technique in CART. Hence, the second best model in terms of KS and sensitivity would be CART. Though the specificity is lower compared to Neural network and Random forest in CART , the similarities between the model performance figures are very close between training and testing data . Random Forest gave a overfit data which makes the model not good compared to other two models.

Model performance Techniques	Training	Testing	Training	Testing	Training	Testing
	CART		Random forest		Neural Network	
Classification error	0.27	0.3113	0.11	0.11	0.237	0.2495
Sensitivity	0.74	0.65	0.11	0.81	0.28505	0.26603
Specificity	0.7	0.69	0.99	0.88	0.93129	0.92294
KS	0.44	0.34	0.74	0.593	0.3938	0.3238
Achieved cummuative response rate of 50% at Decile	4th	3rd	1st	2nd	3rd	3rd
Gini	0.26	0.3	0.75	0.72	0.5102	0.5027
AUC	0.76	0.71	0.94	0.86	0.7115	0.7115

Conclusion

We can never generalize the order of predictive power among a CART , Random forest , neural networking, or rather any predictive algorithm. The reason being every model has its own strength. Random forest generally tends to have a very high accuracy on the training data, because it uses many different characteristics to make a prediction. But, because of the same reason, it sometimes over fits the model on the data which could be the reason we did not get a good model in Random Forest for this data. Moreover, all the predicitive alogirithm used in this report need improvement especially in terms of accuracy especially in predicting the positive class.