

Project – Data Mining

EDA OF THE DATA AVAILABLE. SHOWCASE THE RESULTS USING APPROPRIATE GRAPHS

EDA was performed on the dataset after it being imported in R environment. We discovered the following in the Dataset.

- Family has 18 rows containing NA
- Experience has got negative values like -1, -2, -3
- Around 70% of the rows in mortgage has got value 0
- Experience, Education, Personal Loans, Securities Account, CD Account, Online, Credit Card are as numeric variables in the Data set. We need to convert them into factor variables.

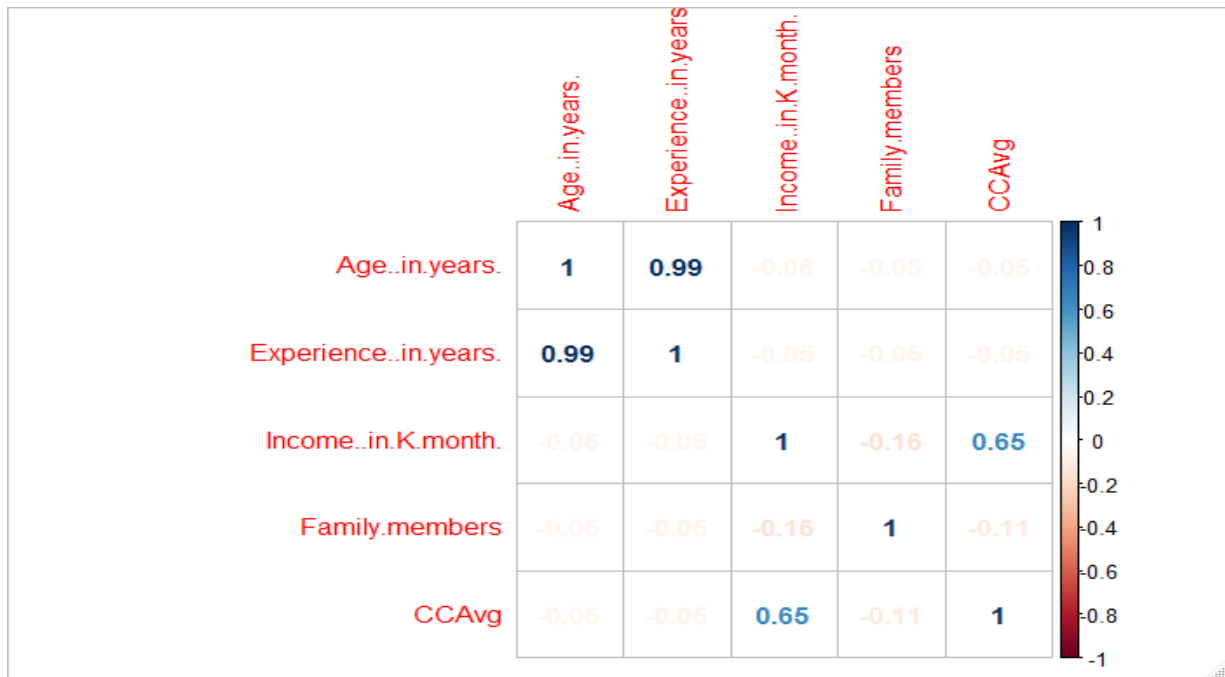
The following remediation were taken due to the above finding:

- The 18 NA rows in Family were omitted.
- We converted the negative values in Experience to 0, and then converted it to a factor variable.
- The rest 30 % of the rows in mortgage having value greater than 0 were converted to 1, now the entire mortgage column was converted to a factor variable
- Experience, Education, Personal Loans, Securities Account, CD Account, Online, Credit Card are converted to factors.

We have a plot of histograms for the Numeric Variables.



We also found out the correlation between the numeric variables of the dataset.



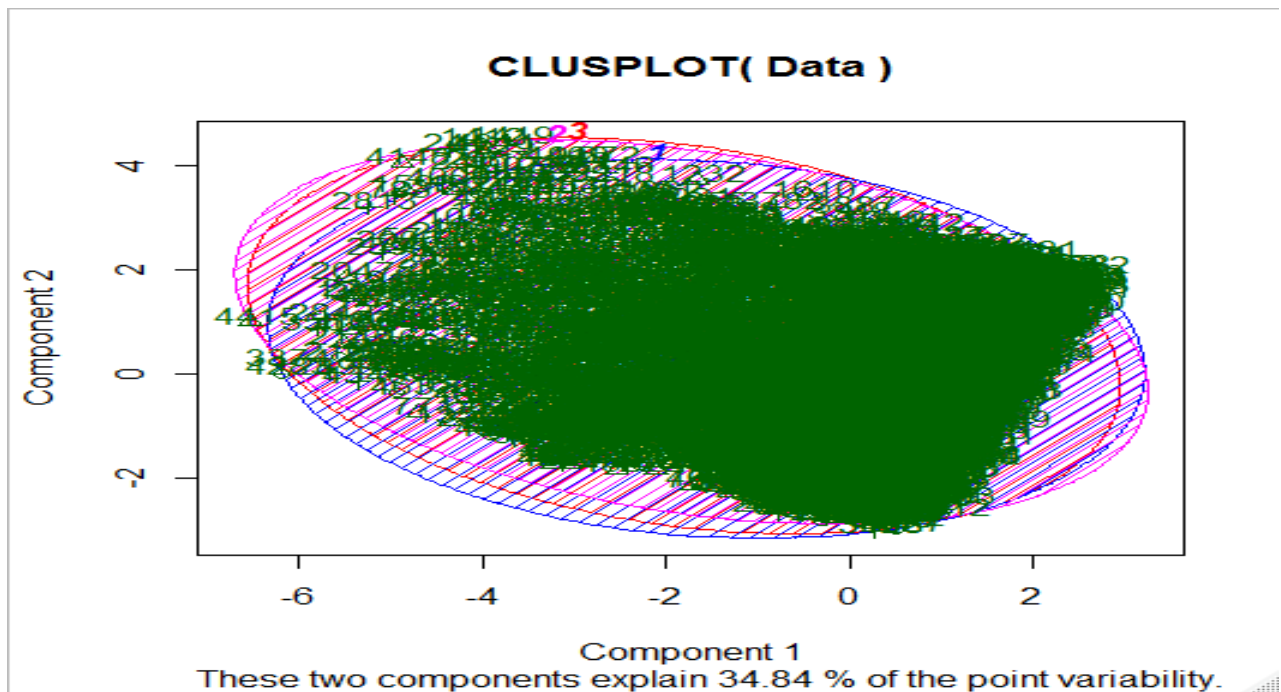
We can see that Age has high correlation with experience, which is obvious.

We also found out that credit card average has got something (high correlation) to do with a person's salary.

APPLY APPROPRIATE CLUSTERING ON THE DATA AND INTERPRET THE OUTPUT

Since our dataset has both Numeric and Factor variables. So, we used Gower method for calculating the distances. Then we used the pam algorithm for clustering. We found that 3 clusters.

We have removed ZIP code during clustering.



After clustering, we find out the customer profile according to each cluster as shown below.

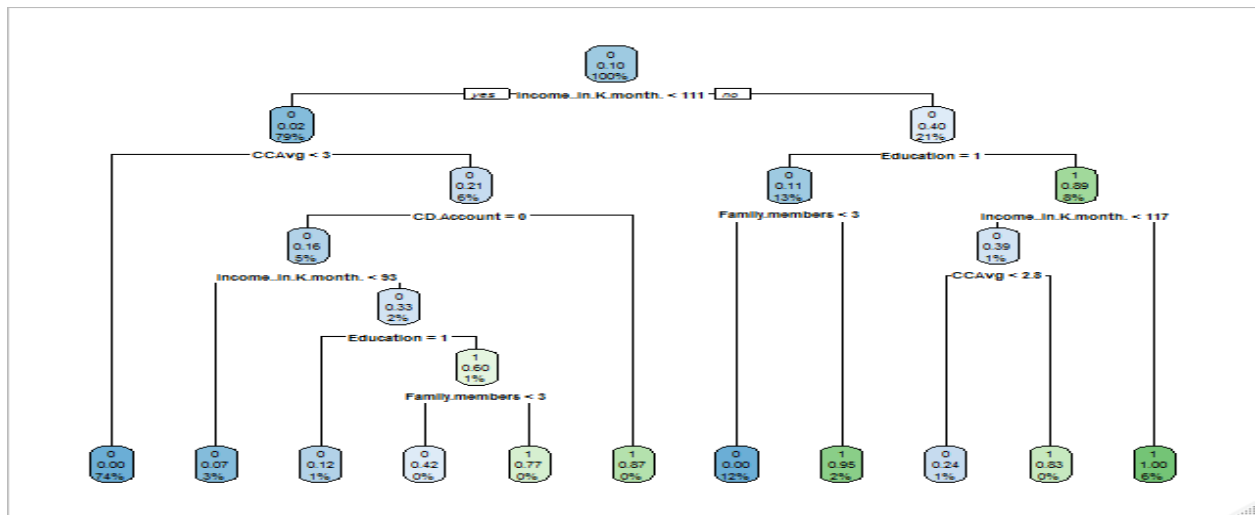
	Group.1	Experience.in.years	Income.in.K.month	Family.members	Education	Mortgage	Personal.Loan	Securities.Account	CD.Account	Online	CreditCard	cluster
1	1	20	65	2.9	2.4	30	0.127	0.11	0.024	0.17	0.29	1
2	2	20	79	2.1	1.6	0	0.069	0.10	0.071	0.83	0.30	2
3	3	20	76	2.3	1.7	187	0.103	0.11	0.087	0.73	0.30	3

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)

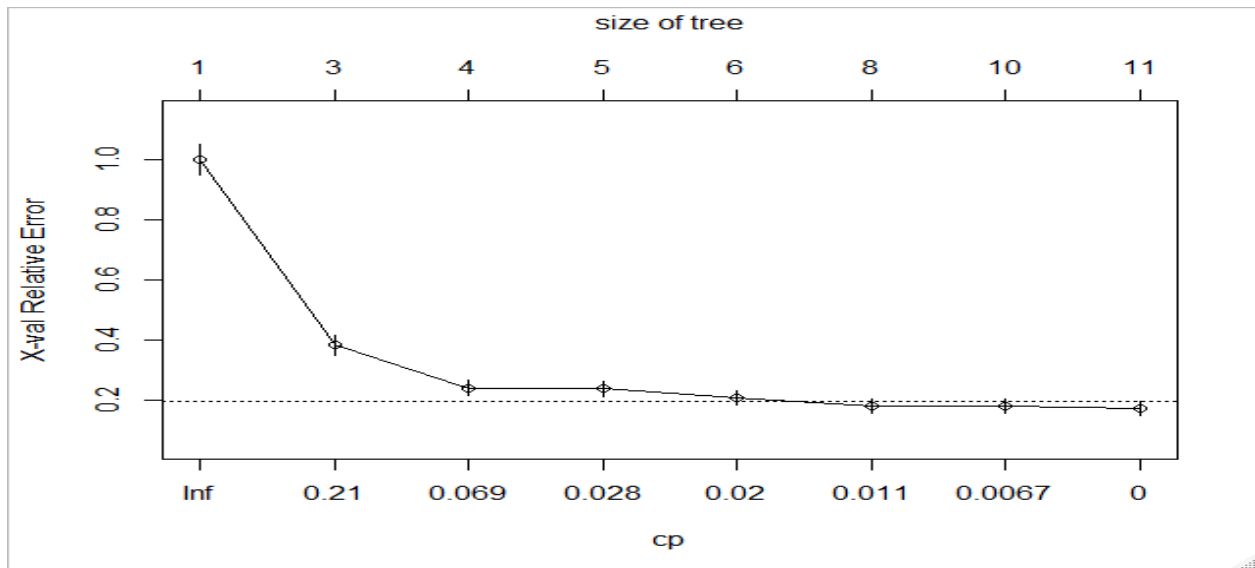
CART

We have split the data set into train & test containing 70 % and 30% of the data respectively.

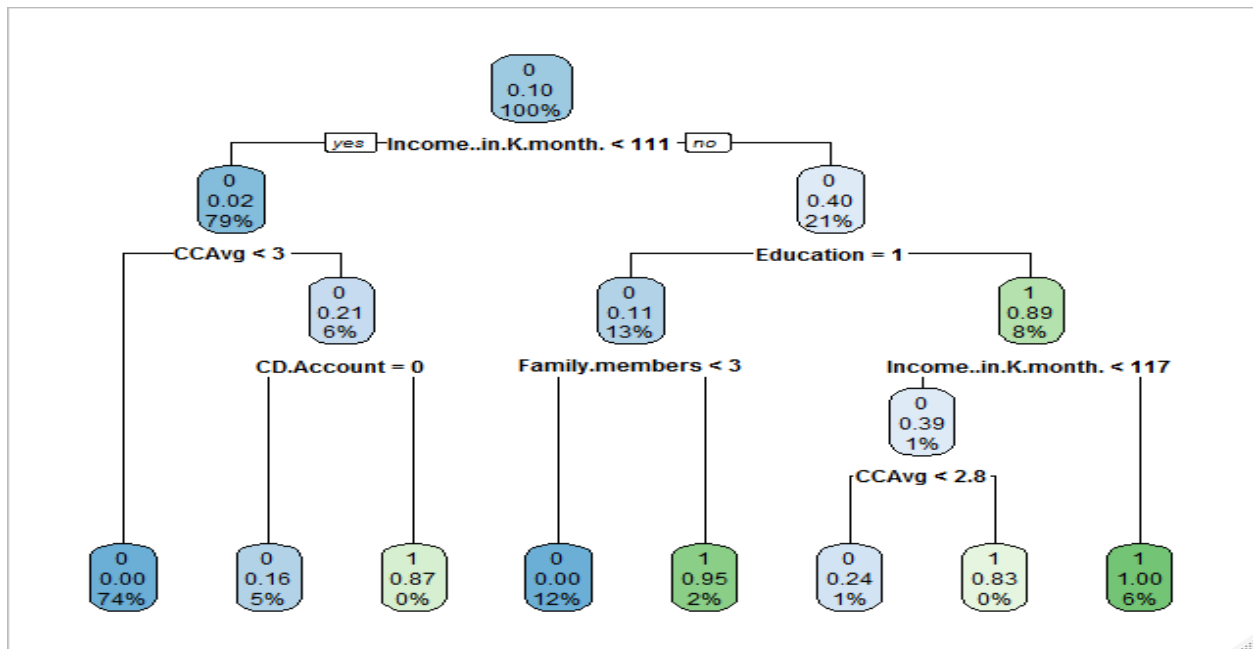
Then we form a tree with $cp=0$ i.e. with no pruning.



From the below graph, we find $cp=0.0085$



Now we'll prune the tree with $cp=0.0085$

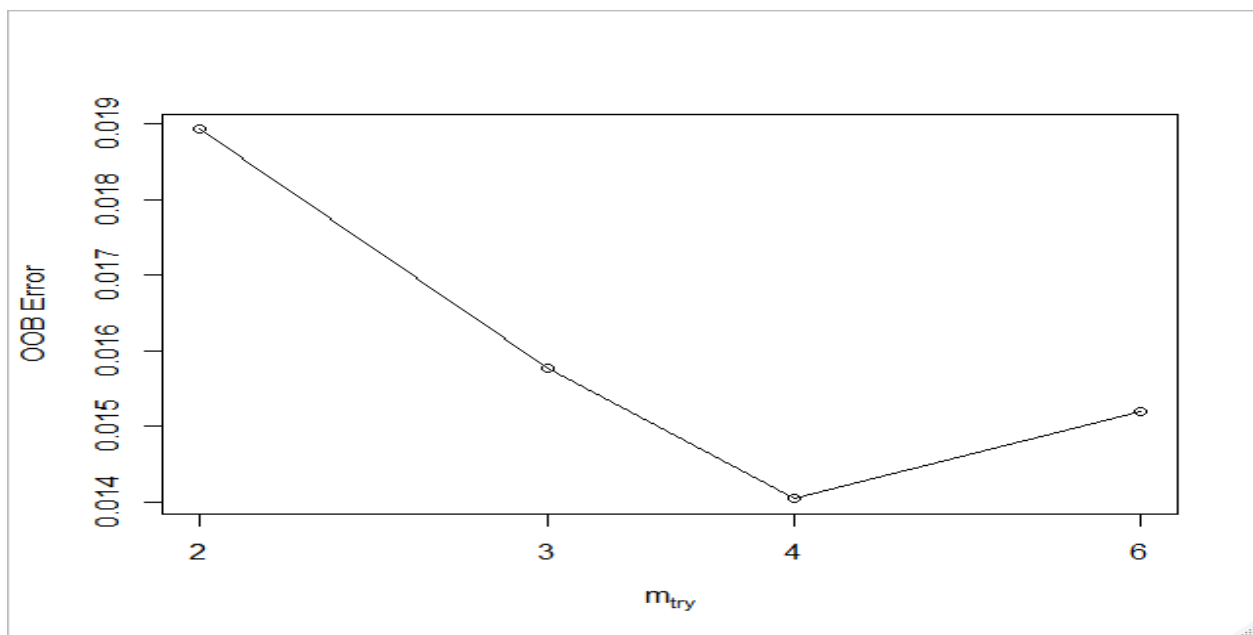


The above is the tree we have got after pruning.

Random Forest:

Here we first find the random forest using a value of m , then we tune this random forest to find the optimal value of m , where m is the no of variables we use for each tree.

After tuning the random forest, we get $m=4$ as optimal.



```

> importance(trnFor,type=1)
                                MeanDecreaseAccuracy
Age..in.years.                   13.64
Experience..in.years.            13.53
Income..in.K.month.             140.35
ZIP.Code                        0.16
Family.members                  81.69
CCAvg                          34.49
Education                      123.00
Mortgage                       1.18
Securities.Account              3.08
CD.Account                     16.58
online                         3.84
CreditCard                     7.61
> |

```

The above shows the importance of the various variables in the dataset in the formation of the random forest. Income, education and Family members play the most important role in classifying customers having a higher probability of buying a personal loan.

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.

CART

Confusion Matrix:

Train Data set:

	0	1
0	3146	7
1	40	295

Accuracy=0.98

Sensitivity=0.98

Specificity=0.99

Test Data Set:

	0	1
0	1345	6
1	23	120

Accuracy = 0.98

Sensitivity= 0.95

Specificity= 0.98

Rank Table:

Train Dataset

	deciles	cnt	cnt_tar1	cnt_tar0	rrate	cum_resp	cum_non_resp	cum_rel_resp	cum_rel_non_resp	ks
1:	[0.157,1]	517	332	185	64.22	332	185	99	5.9	93
2:	[0.00117,0.157)	2564	3	2561	0.12	335	2746	100	87.1	13
3:	[0,0.00117)	407	0	407	0.00	335	3153	100	100.0	0

KS=0.93

AUC=0.99

Gini = 0.89

Test Dataset

	deciles	cnt	cnt_tar1	cnt_tar0	rrate	cum_resp	cum_non_resp	cum_rel_resp	cum_rel_non_resp	ks
1:	[0.157,1]	218	141	77	64.68	141	77	99	5.7	93
2:	[0.00117,0.157)	1084	2	1082	0.18	143	1159	100	85.8	14
3:	[0,0.00117)	192	0	192	0.00	143	1351	100	100.0	0

KS=0.93

AUC=0.99

Gini = 0.89

Random Forest

Train Data Set

	0	1
0	3153	0
1	26	309

Accuracy=0.99

Sensitivity=1

Specificity=0.99

Test Data Set

	0	1
0	1347	4
1	12	131

Accuracy=0.99

Sensitivity=1

Specificity=0.99

Rank Table:

Train Data Set

	deciles	cnt	cnt_tar1	cnt_tar0	rrate	cum_resp	cum_non_resp	cum_rel_resp	cum_rel_non_resp	ks
1:	[0.242,1]	349	334	15	95.70	334	15	100	0.48	99
2:	[0.018,0.242)	350	1	349	0.29	335	364	100	11.54	88
3:	[0.004,0.018)	468	0	468	0.00	335	832	100	26.39	74
4:	[0.002,0.004)	338	0	338	0.00	335	1170	100	37.11	63
5:	[0,0.002)	1983	0	1983	0.00	335	3153	100	100.00	0
>										

KS=0.99

AUC=1

Gini=0.89

Test Data Set

	deciles	cnt	cnt_tar1	cnt_tar0	rrate	cum_resp	cum_non_resp	cum_rel_resp	cum_rel_non_resp	ks
1:	[0.345,0.998]	150	140	10	93.3	140	10	98	0.74	97
2:	[0.024,0.345)	155	3	152	1.9	143	162	100	11.99	88
3:	[0.008,0.024)	155	0	155	0.0	143	317	100	23.46	77
4:	[0.002,0.008)	312	0	312	0.0	143	629	100	46.56	53
5:	[0,0.002)	722	0	722	0.0	143	1351	100	100.00	0

KS=0.97

AUC=1

Gini=0.88

By the Model performance parameters, we can understand that by using the random forest model, we can better predict the probability of a customer buying a Loan from There Bank since the Sensitivity of the Random Forest model is 100% and accuracy is 99%. AUC is 100% while the first decile has a KS of 99% for Train and 97% for Test. So, we can safely say, Random Forest Model performs better.


```
1 setwd("C:\\Users\\Saptarshi Datta\\Desktop\\R PROGRAMMING")
2 Data=read.csv("Thera Bank.csv", header=TRUE)
3
4
5 Data=Data[,-c(1,5)]
6
7 attach(Data)
8
9 Data$Mortgage=ifelse(Data$Mortgage>0,1,0)
10 Data$Experience..in.years.=ifelse(Data$Experience..in.years.<0,0,Data$Experience..in.years.)
11 Data$Education=as.factor(Data$Education)
12 Data$Personal.Loan=as.factor(Data$Personal.Loan)
13 Data$Securities.Account=as.factor(Data$Securities.Account)
14 Data$CD.Account=as.factor(Data$CD.Account)
15 Data$Online=as.factor(Data$Online)
16 Data$CreditCard=as.factor(Data$CreditCard)
17 ##Data$ZIP.Code=as.factor(Data$ZIP.Code)
18 Data$Mortgage=as.factor(Data$Mortgage)
19
20 summary(Data)
21 str(Data)
22 is.na(Data)
23 Data1=na.omit(Data)
24 Data=Data1
25
26
27 library(funModeling)
28 library(tidyverse)
29 library(Hmisc)
30
31 basic_eda <- function(data)
32 {
33   glimpse(data)
34   df_status(data)
35   freq(data)
36   profiling_num(data)
37   plot_num(data)
38   describe(data)
39 }
40
41 basic_eda(Data)
42
43 boxplot(Data)
44 x=cor(Data[,c(1,2,3,4,5)])
45 library(corrplot)
46 corrplot(x,method="number")
47
48 library(cluster)
49 seed=10000
50 set.seed(seed)
51 gower_matrix <-daisy(Data,metric="gower")
52 dist <-gower_matrix
53 pamxx <-pam(dist,k=3)
54 sil1 = silhouette(pamxx$clustering,dist)
55 plot(sil1,col=c("red","blue"))
56 summary(sil1)
57 print(pamxx$silinfo$avg.width)
58 clusplot(Data,pamxx$clustering,color=TRUE,shade=TRUE,label= 2, lines = 1)
59 Data$cluster=pamxx$clustering
60 custprofile=aggregate(Data,list(Data$cluster),FUN ="mean")
61
62
63
```

```

61 set.seed(seed)
62 a=rep(0,10)
63 for(j in 1:100)
64 {
65   pamxx <- pam(dist,j)
66   sil = silhouette(pamxx$clustering,dist)
67   print(j)
68   print(pamxx$silinfo$avg.width)
69 }
70 plot(c(1:10),a$avg.width)
71

```

```

28
29 library(caTools)
30
31 split = sample.split(Data$Personal.Loan, SplitRatio = .70)
32 train = subset(Data, split == TRUE)
33 test = subset(Data, split == FALSE)
34
35 table(train$Personal.Loan)
36
37 library(rpart)
38 library(rpart.plot)
39
40 tree = rpart(formula = Personal.Loan ~ .,data=train,method="class",cp=0)
41 rpart.plot(tree)
42 printcp(tree)
43 plotcp(tree)
44
45 ptree = prune(tree,cp=0.0085,"cp")
46
47 ##check the updated tree
48 ##plot tree
49 rpart.plot(ptree)
50
51 ##print cp value
52 printcp(ptree)
53
54 train$CART.Pred = predict(ptree,data=train,type="class")
55 train$CART.Score = predict(ptree,data=train,type="prob")[,"1"]
56 table(train$Personal.Loan,train$CART.Pred)
57
58
59 test$CART.Pred = predict(ptree,test,type="class")
60 test$CART.Score = predict(ptree,test,type="prob")[,"1"]
61 table(test$Personal.Loan,test$CART.Pred)
62

```

```

63 #TRAIN data set
64 probs=seq(0,1,length=11)
65 qs=quantile(train$CART.Score, probs)
66 print(qs)
67 train$deciles=cut(train$CART.Score, unique(qs),include.lowest = TRUE,right=FALSE)
68 table(train$deciles)
69
70 library(data.table)
71 trainDT=data.table(train)
72 rankTbl=trainDT[, list(
73   cnt = length(Personal.Loan),
74   cnt_tar1 = sum(as.numeric(Personal.Loan ==1)),
75   cnt_tar0 = sum(as.numeric(Personal.Loan == 0))), by=deciles][order(-deciles)]
76
77 rankTbl$rrate = round(rankTbl$cnt_tar1 / rankTbl$cnt,4)*100;
78 rankTbl$cum_resp = cumsum(rankTbl$cnt_tar1)
79 rankTbl$cum_non_resp = cumsum(rankTbl$cnt_tar0)
80 rankTbl$cum_rel_resp = round(rankTbl$cum_resp / sum(rankTbl$cnt_tar1),4)*100;
81 rankTbl$cum_rel_non_resp = round(rankTbl$cum_non_resp / sum(rankTbl$cnt_tar0),4)*100;
82 rankTbl$ks = abs(rankTbl$cum_rel_resp - rankTbl$cum_rel_non_resp);
83
84 print(rankTbl)
85
86 library(ROCR)
87 library(ineq)
88 predobj = prediction(train$CART.Score, train$Personal.Loan)
89 perf = performance(predobj, "tpr", "fpr")
90 plot(perf)
91 KS = max(perf@y.values[[1]]-perf@x.values[[1]])
92 print(KS)
93 auc = performance(predobj, "auc");
94 auc = as.numeric(auc@y.values)
95 print(auc)
96 gini = ineq(train$CART.Score, type="Gini")
97 print(gini)
98

```

```

99 #test DATA SET
100 probs=seq(0,1,length=11)
101 qs=quantile(test$CART.Score, probs)
102 print(qs)
103 test$deciles=cut(test$CART.Score, unique(qs),include.lowest = TRUE,right=FALSE)
104 table(test$deciles)
105
106 library(data.table)
107 testDT=data.table(test)
108 rankTbl=testDT[, list(
109   cnt = length(Personal.Loan),
110   cnt_tar1 = sum(as.numeric(Personal.Loan ==1)),
111   cnt_tar0 = sum(as.numeric(Personal.Loan == 0))), by=deciles][order(-deciles)]
112
113 rankTbl$rrate = round(rankTbl$cnt_tar1 / rankTbl$cnt,4)*100;
114 rankTbl$cum_resp = cumsum(rankTbl$cnt_tar1)
115 rankTbl$cum_non_resp = cumsum(rankTbl$cnt_tar0)
116 rankTbl$cum_rel_resp = round(rankTbl$cum_resp / sum(rankTbl$cnt_tar1),4)*100;
117 rankTbl$cum_rel_non_resp = round(rankTbl$cum_non_resp / sum(rankTbl$cnt_tar0),4)*100;
118 rankTbl$ks = abs(rankTbl$cum_rel_resp - rankTbl$cum_rel_non_resp);
119
120 print(rankTbl)
121
122 library(ROCR)
123 library(ineq)
124 predobj = prediction(test$CART.Score, test$Personal.Loan)
125 perf = performance(predobj, "tpr", "fpr")
126 plot(perf)
127 KS = max(perf@y.values[[1]]-perf@x.values[[1]])
128 print(KS)
129 auc = performance(predobj, "auc");
130 auc = as.numeric(auc@y.values)
131 print(auc)
132 gini = ineq(test$CART.Score, type="Gini")
133 print(gini)
134

```

```

139 library(caTools)
140 split = sample.split(Data$Personal.Loan, SplitRatio = .70)
141 train = subset(Data, split == TRUE)
142 test = subset(Data, split == FALSE)
143
144
145 library(randomForest)
146 seed=1000
147 set.seed(seed)
148 rndFor = randomForest(Personal.Loan ~ ., data = train,
149                       ntree=501, mtry = 5, nodesize = 10,
150                       importance=TRUE)
151
152 print(rndFor)
153 importance(rndFor,type=1)
154
155
156 set.seed(seed)
157 tRndFor = tuneRF(x = train[,c(-9)],
158                 y=train$Personal.Loan,
159                 mtryStart = 3,
160                 ntreeTry = 501,
161                 stepFactor = 1.5,
162                 improve = 0.0001,
163                 trace=TRUE,
164                 plot = TRUE,
165                 doBest = TRUE,
166                 nodesize = 10,
167                 importance=TRUE)
168 )
169 importance(tRndFor,type=1)
170
171 train$predict.class = predict(tRndFor, train, type="class")
172 train$prob1 = predict(tRndFor, train, type="prob")[, "1"]
173 tbl=table(train$Personal.Loan, train$predict.class)
174 print(tbl)

```

```

177 test$predict.class = predict(tRndFor, test, type="class")
178 test$prob1 = predict(tRndFor, test, type="prob")[, "1"]
179 tbl=table(test$Personal.Loan, test$predict.class)
180 print(tbl)
181
182 #TRAIN data set
183 probs=seq(0,1,length=11)
184 qs=quantile(train$prob1, probs)
185 print(qs)
186 train$deciles=cut(train$prob1, unique(qs),include.lowest = TRUE,right=FALSE)
187 table(train$deciles)
188
189 library(data.table)
190 trainDT=data.table(train)
191 rankTbl=trainDT[, list(
192   cnt = length(Personal.Loan),
193   cnt_tar1 = sum(as.numeric(Personal.Loan ==1)),
194   cnt_tar0 = sum(as.numeric(Personal.Loan == 0))), by=deciles][order(-deciles)]
195
196 rankTbl$rrate = round(rankTbl$cnt_tar1 / rankTbl$cnt,4)*100;
197 rankTbl$cum_resp = cumsum(rankTbl$cnt_tar1)
198 rankTbl$cum_non_resp = cumsum(rankTbl$cnt_tar0)
199 rankTbl$cum_rel_resp = round(rankTbl$cum_resp / sum(rankTbl$cnt_tar1),4)*100;
200 rankTbl$cum_rel_non_resp = round(rankTbl$cum_non_resp / sum(rankTbl$cnt_tar0),4)*100;
201 rankTbl$ks = abs(rankTbl$cum_rel_resp - rankTbl$cum_rel_non_resp);
202
203 print(rankTbl)
204
205 library(ROCR)
206 library(ineq)
207 predobj = prediction(train$prob1, train$Personal.Loan)
208 perf = performance(predobj, "tpr", "fpr")
209 plot(perf)
210 KS = max(perf@y.values[[1]]-perf@x.values[[1]])
211 print(KS)
212 auc = performance(predobj,"auc");

```

```

213 auc = as.numeric(auc@y.values)
214 print(auc)
215 gini = ineq(train$prob1, type="Gini")
216 print(gini)
217
218 #test DATA SET
219 probs=seq(0,1,length=11)
220 qs=quantile(test$pr0b1, probs)
221 print(qs)
222 test$deciles=cut(test$pr0b1, unique(qs),include.lowest = TRUE,right=FALSE)
223 table(test$deciles)
224
225 library(data.table)
226 testDT=data.table(test)
227 rankTbl=testDT[, list(
228   cnt = length(Personal.Loan),
229   cnt_tar1 = sum(as.numeric(Personal.Loan ==1)),
230   cnt_tar0 = sum(as.numeric(Personal.Loan == 0))), by=deciles][order(-deciles)]
231
232 rankTbl$rrate = round(rankTbl$cnt_tar1 / rankTbl$cnt,4)*100;
233 rankTbl$cum_resp = cumsum(rankTbl$cnt_tar1)
234 rankTbl$cum_non_resp = cumsum(rankTbl$cnt_tar0)
235 rankTbl$cum_rel_resp = round(rankTbl$cum_resp / sum(rankTbl$cnt_tar1),4)*100;
236 rankTbl$cum_rel_non_resp = round(rankTbl$cum_non_resp / sum(rankTbl$cnt_tar0),4)*100;
237 rankTbl$ks = abs(rankTbl$cum_rel_resp - rankTbl$cum_rel_non_resp);
238
239 print(rankTbl)
240
241 library(ROCR)
242 library(ineq)
243 predObj = prediction(test$pr0b1, test$Personal.Loan)
244 perf = performance(predObj, "tpr", "fpr")
245 plot(perf)
246 KS = max(perf@y.values[[1]]-perf@x.values[[1]])
247 print(KS)
248 auc = performance(predObj,"auc");

```

```

249 auc = as.numeric(auc@y.values)
250 print(auc)
251 gini = ineq(test$pr0b1, type="Gini")
252 print(gini)
253
254

```