DM3

Vibhav Mayekar,Pratik Sharma, Jitesh Patil

2/27/2022

```r
#install.packages('readxl')
#install.packages('readxl')
#install.packages("randomForest")
#install.packages("caret")
#install.packages("ROCR")
#install.packages("corrplot")
library(readxl)
library(tidyverse)
library(tidyr)
library(ggplot2)
library(dplyr)
library(corrplot)
library(apaTables)
library(rpart)
library(rpart.plot)
library(psych)
library(randomForest)
library(caret)
library(ROCR)
library(ISLR)
#install.packages('readxl')
#install.packages("randomForest")
#install.packages("caret")
#install.packages("ROCR")
```

#Selecting the Retention modeling.xlsx file from the directory

```r
f<-file.choose()
Retention_modeling <- read_excel(f)
#Retention_modeling <- read_excel("Retention modeling.xlsx",sheet = 2)
##View(Retention_modeling)
rm1 <- Retention_modeling
ogdata <- Retention_modeling
View(rm1)

rm1 <- rm1[-(2390:2392),,drop=FALSE]
ogdata <- ogdata[-(2390:2392),,drop=FALSE]

rm1$Retained.in.2012. <- as.factor(ifelse(rm1$Retained.in.2012. == 1, "Retained" , "Not Retained"))
```

#For better visualization of the graph following changes were made to the SchoolGradeType Elementary->Elementary", "E->E ;"Middle->Middle", "M->M";"High->High", "H->H rm1$SchoolGradeType <-

#"Undefined->Undefined", "U->U";"Middle->Undefined", "M->U";"Elementary->Middle", "E->M";"Middle->High", "M->H";"Elementary->High", "E->H";"Elementary->Undefined", "E->U

```
replace(rm1$SchoolGradeType,
          rm1$SchoolGradeType == "Elementary->Elementary", "E->E")
rm1$SchoolGradeType <-
  replace(rm1$SchoolGradeType,
          rm1$SchoolGradeType == "Middle->Middle", "M->M")
rm1$SchoolGradeType <-
  replace(rm1$SchoolGradeType,
          rm1$SchoolGradeType == "High->High", "H->H")
rm1$SchoolGradeType <-
  replace(rm1$SchoolGradeType,
          rm1$SchoolGradeType == "Undefined->Undefined", "U->U")
rm1$SchoolGradeType <-
  replace(rm1$SchoolGradeType,
          rm1$SchoolGradeType == "Middle->Undefined", "M->U")
rm1$SchoolGradeType <-
  replace(rm1$SchoolGradeType,
          rm1$SchoolGradeType == "Elementary->Middle", "E->M")
rm1$SchoolGradeType <-
  replace(rm1$SchoolGradeType,
          rm1$SchoolGradeType == "Middle->High", "M->H")
rm1$SchoolGradeType <-
  replace(rm1$SchoolGradeType,
          rm1$SchoolGradeType == "Elementary->High", "E->H")
rm1$SchoolGradeType <-
  replace(rm1$SchoolGradeType,
          rm1$SchoolGradeType == "Elementary->Undefined", "E->U")
```

#The data frame was also modified to Replace the Income.Level (P1,P2,P3,P4) with P this was done to eliminate multiple subcategories for the Income level (A,B,C,D…P)Income Level A,B,C,D,E à Low , Income Level F,G,H,I,J,K,L à Medium ,Income Level M,N,O,P,Q àHigh

```
rm1$Is.Non.Annual. <- as.factor(ifelse(rm1$Is.Non.Annual. == 1, "Yes" , "No")
)
rm1$Income.Level <- replace(rm1$Income.Level, rm1$Income.Level == "P1", "P")
rm1$Income.Level <- replace(rm1$Income.Level, rm1$Income.Level == "P3", "P")
rm1$Income.Level <- replace(rm1$Income.Level, rm1$Income.Level == "P4", "P")
rm1$Income.Level <- replace(rm1$Income.Level, rm1$Income.Level == "P5", "P")

rm1$Income.Level <-
  replace(rm1$Income.Level, rm1$Income.Level == "Z", "Unclassified")
```

```r
rm1$Income.Level <- replace(rm1$Income.Level, rm1$Income.Level == "Q", "High")
rm1$Income.Level <- replace(rm1$Income.Level, rm1$Income.Level == "P", "High")
rm1$Income.Level <- replace(rm1$Income.Level, rm1$Income.Level == "O", "High")
rm1$Income.Level <- replace(rm1$Income.Level, rm1$Income.Level == "N", "High")
rm1$Income.Level <- replace(rm1$Income.Level, rm1$Income.Level == "M", "High")

rm1$Income.Level <- replace(rm1$Income.Level, rm1$Income.Level == "L", "Medium")
rm1$Income.Level <- replace(rm1$Income.Level, rm1$Income.Level == "K", "Medium")
rm1$Income.Level <- replace(rm1$Income.Level, rm1$Income.Level == "J", "Medium")
rm1$Income.Level <- replace(rm1$Income.Level, rm1$Income.Level == "I", "Medium")
rm1$Income.Level <- replace(rm1$Income.Level, rm1$Income.Level == "H", "Medium")
rm1$Income.Level <- replace(rm1$Income.Level, rm1$Income.Level == "G", "Medium")
rm1$Income.Level <- replace(rm1$Income.Level, rm1$Income.Level == "F", "Medium")

rm1$Income.Level <- replace(rm1$Income.Level, rm1$Income.Level == "E", "Low")
rm1$Income.Level <- replace(rm1$Income.Level, rm1$Income.Level == "D", "Low")
rm1$Income.Level <- replace(rm1$Income.Level, rm1$Income.Level == "C", "Low")
rm1$Income.Level <- replace(rm1$Income.Level, rm1$Income.Level == "B", "Low")
rm1$Income.Level <- replace(rm1$Income.Level, rm1$Income.Level == "A", "Low")



#We found that Group.State column was majorly divided into two categories which was similar to the categorical variable Region and hence was removed from the analysis


# Group.State=AK,AL,AR,AZ,Bermuda,CA,CO,CT,FL,IA,ID,IL,IN,KS,MD,MN,MS,NC,NE,NJ,NM,
# NV,NY,OK,OR,PA,TN,TX,UT,VA,WA
#
# Group.State=GA,HI,KY,LA,MA,MI,MO,NH,OH,PR,SC,SD,WI
```

#Data cleaning of the data set all 56 variables

#We found out that the Date Columns did not show significant relation with the Target Variable, and it made more sense to work with the duration column

removing redundant columns with dates and irrelevant info.

```
rm1 <- rm1[,-c(1 ,9 ,10, 11,12 ,17,18,21,37,39,40)]
rm1 <- rm1[,-c(4)]

##data.frame(colnames(rm1)) #Returns column index numbers in table format,df=
DataFrame name
```

Removed ID , Departure.Date , Return.Date , Deposit.Date ,

#Special.Pay , Early.RPL, Latest.RPL

#A MODE function was created to calculate the mode for replacing the NA values for numerical variables and replacing the most occurring value for Categorical values

#Writing functions for mode

```
find_mode <- function(x)
  {
  u <- unique(x)
  tab <- tabulate(match(x, u))
  u[tab == max(tab)]
}


#Checking NA/Null and replacing with with mean/mode - > from.grade

sum((rm1$From.Grade) == 'NA')

## [1] 127

modeFG <- find_mode(rm1$From.Grade)
rm1$From.Grade <- replace(rm1$From.Grade,(rm1$From.Grade) == 'NA',modeFG)


#Checking NA/Null and replacing with with mean/mode - > To grade

sum((rm1$To.Grade) == 'NA')

## [1] 150

modeTG <- find_mode(rm1$To.Grade)
rm1$To.Grade <- replace(rm1$To.Grade,((rm1$To.Grade) == 'NA'),modeTG)
```

```r
###Data Cleaning
#Checking NA/Null and replacing with with mean/mode - > Travel.type

sum((rm1$Travel.Type) == 'N')

## [1] 2

modeTT <- find_mode(rm1$Travel.Type)
rm1$Travel.Type <- replace(rm1$Travel.Type , rm1$Travel.Type == 'N' , modeTT)
#N replaced with mode




sum(is.na(rm1$Poverty.Code))

## [1] 599

modepc <- find_mode(rm1$Poverty.Code)
rm1$Poverty.Code <- replace(rm1$Poverty.Code,is.na(rm1$Poverty.Code)==TRUE,
                            modepc)


sum((rm1$CRM.Segment) == 'NA')

## [1] 4

modecrms <- find_mode(rm1$CRM.Segment)
rm1$CRM.Segment <- replace(rm1$CRM.Segment,(rm1$CRM.Segment) == 'NA',modecrms
)


sum(is.na(rm1$MDR.Low.Grade))

## [1] 68

modemlg <- find_mode(rm1$MDR.Low.Grade)
rm1$MDR.Low.Grade <- replace(rm1$MDR.Low.Grade,is.na(rm1$MDR.Low.Grade)==TRUE
,
                             modemlg)




sum(is.na(rm1$MDR.High.Grade))

## [1] 0

sum((rm1$MDR.High.Grade) == 'NA')
```

```
## [1] 68
```

```
modemhg <- find_mode(rm1$MDR.High.Grade)
rm1$MDR.High.Grade <- replace(rm1$MDR.High.Grade,rm1$MDR.High.Grade== 'NA',
                              modemhg)

sum(is.na(rm1$Total.School.Enrollment))
```

```
## [1] 91
```

```
meantse <- round(mean(rm1$Total.School.Enrollment,na.rm=TRUE))
rm1$Total.School.Enrollment <-
  replace(rm1$Total.School.Enrollment,
          is.na(rm1$Total.School.Enrollment)==TRUE,meantse)


sum(is.na(rm1$Income.Level))
```

```
## [1] 62
```

```
modeil <- find_mode(rm1$Income.Level)
rm1$Income.Level <- replace(rm1$Income.Level,is.na(rm1$Income.Level)==TRUE,mo
deil)


sum((rm1$DifferenceTraveltoFirstMeeting) == 'NA')
```

```
## [1] 337
```

```
meandtfm <-
  round(mean(as.numeric(rm1$DifferenceTraveltoFirstMeeting),na.rm=TRUE))
rm1$DifferenceTraveltoFirstMeeting <-
  replace(rm1$DifferenceTraveltoFirstMeeting,
                            rm1$DifferenceTraveltoFirstMeeting == 'NA',meandt
fm)


sum(is.na(rm1$DifferenceTraveltoLastMeeting))
```

```
## [1] 0
```

```
sum((rm1$DifferenceTraveltoLastMeeting) == 'NA')
```

```
## [1] 337
```

```
meandtlm <- round(mean(as.numeric(rm1$DifferenceTraveltoLastMeeting),na.rm=TR
UE))
rm1$DifferenceTraveltoLastMeeting <- replace(rm1$DifferenceTraveltoLastMeetin
g,
                            rm1$DifferenceTraveltoLastMeeting == 'NA',meandtf
m)
```

```
sum(is.na(rm1$FPP.to.School.enrollment))

## [1] 0

sum((rm1$FPP.to.School.enrollment) == 'NA')

## [1] 91

meanftse <- round(mean(as.numeric(rm1$FPP.to.School.enrollment),na.rm=TRUE))
rm1$FPP.to.School.enrollment <- replace(rm1$FPP.to.School.enrollment,
                            rm1$FPP.to.School.enrollment == 'NA',meanftse)


rm1$SchoolSizeIndicator <-
  sapply(rm1$SchoolSizeIndicator, as.character, na.rm=TRUE)
sum(is.na(rm1$SchoolSizeIndicator))

## [1] 91

modessi <-
  find_mode((rm1$SchoolSizeIndicator))
rm1$SchoolSizeIndicator <-
  replace(rm1$SchoolSizeIndicator,is.na(rm1$SchoolSizeIndicator) == TRUE,mode
ssi)

unique(rm1$SchoolSizeIndicator)

## [1] "L"   "S-M" "M-L" "S"
```

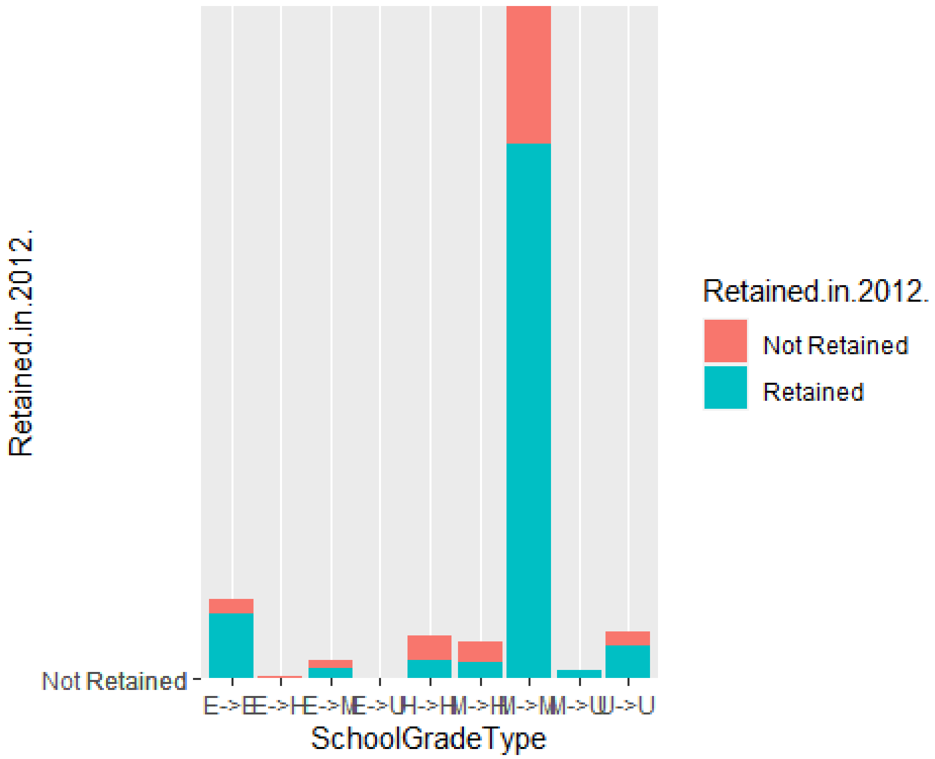
Cpoy of frame for RF

```
rm2 <- rm1

rm1$From.Grade <- as.factor(rm1$From.Grade)
rm1$To.Grade <- as.factor(rm1$To.Grade)
rm1$Travel.Type <- as.factor(rm1$Travel.Type)



rm1$Poverty.Code <- as.factor(rm1$Poverty.Code)
rm1$Region <- as.factor(rm1$Region)
rm1$CRM.Segment <- as.factor(rm1$CRM.Segment)
rm1$School.Type <- as.factor(rm1$School.Type)
rm1$MDR.Low.Grade <- as.factor(rm1$MDR.Low.Grade)
rm1$MDR.High.Grade  <- as.factor(rm1$MDR.High.Grade)
rm1$Income.Level  <- as.factor(rm1$Income.Level)
rm1$SPR.Product.Type  <- as.factor(rm1$SPR.Product.Type)
rm1$SPR.New.Existing  <- as.factor(rm1$SPR.New.Existing)

rm1$DifferenceTraveltoFirstMeeting <- as.numeric(rm1$DifferenceTraveltoFirstM
eeting)
```

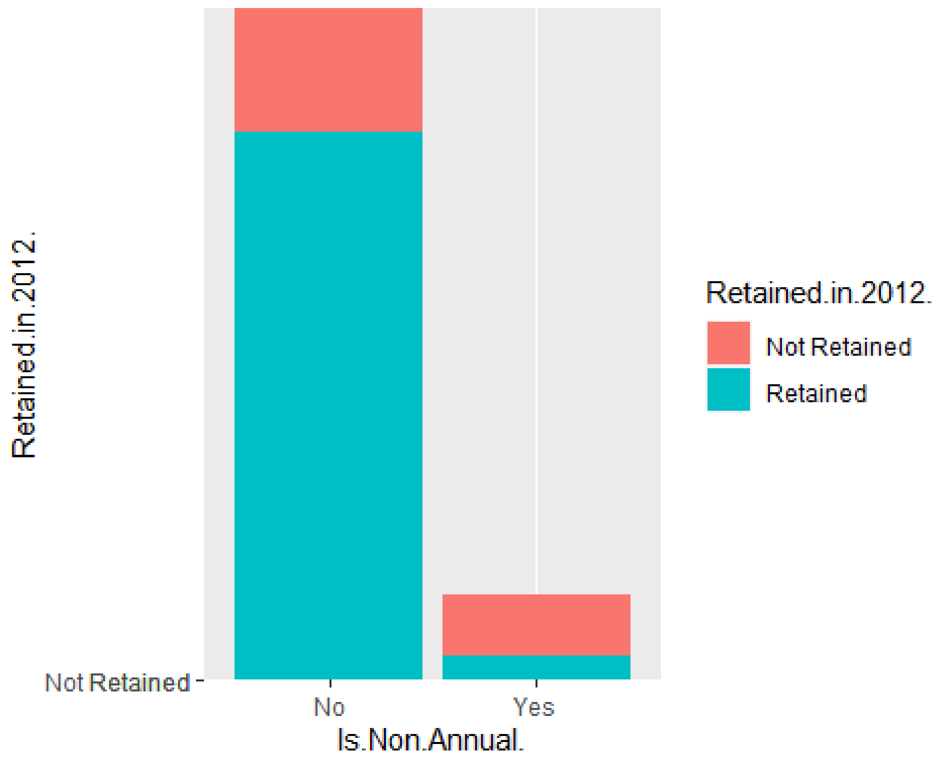
```
rm1$DifferenceTraveltoLastMeeting <- as.numeric(rm1$DifferenceTraveltoLastMee
ting)
rm1$SchoolGradeTypeLow <- as.factor(rm1$SchoolGradeTypeLow)
rm1$SchoolGradeTypeHigh <- as.factor(rm1$SchoolGradeTypeHigh)
rm1$SchoolGradeType <- as.factor(rm1$SchoolGradeType)
rm1$GroupGradeTypeLow <- as.factor(rm1$GroupGradeTypeLow)
rm1$GroupGradeTypeHigh <- as.factor(rm1$GroupGradeTypeHigh)
rm1$GroupGradeType <- as.factor(rm1$GroupGradeType)
rm1$DepartureMonth <- as.factor(rm1$DepartureMonth)
rm1$MajorProgramCode <- as.factor(rm1$MajorProgramCode)
rm1$FPP.to.School.enrollment <- as.numeric(rm1$FPP.to.School.enrollment)
rm1$MajorProgramCode <- as.factor(rm1$MajorProgramCode)
rm1$SchoolSizeIndicator <- as.factor(rm1$SchoolSizeIndicator)
```

#Plotting the graphs of the analysis we performed for various num and categorical variables

```
#plotting schoolgradetype against target
p1<-ggplot(data=rm1,aes(x=SchoolGradeType, y=Retained.in.2012.,
                        fill=Retained.in.2012.))+geom_bar( stat="identity")
p1
```



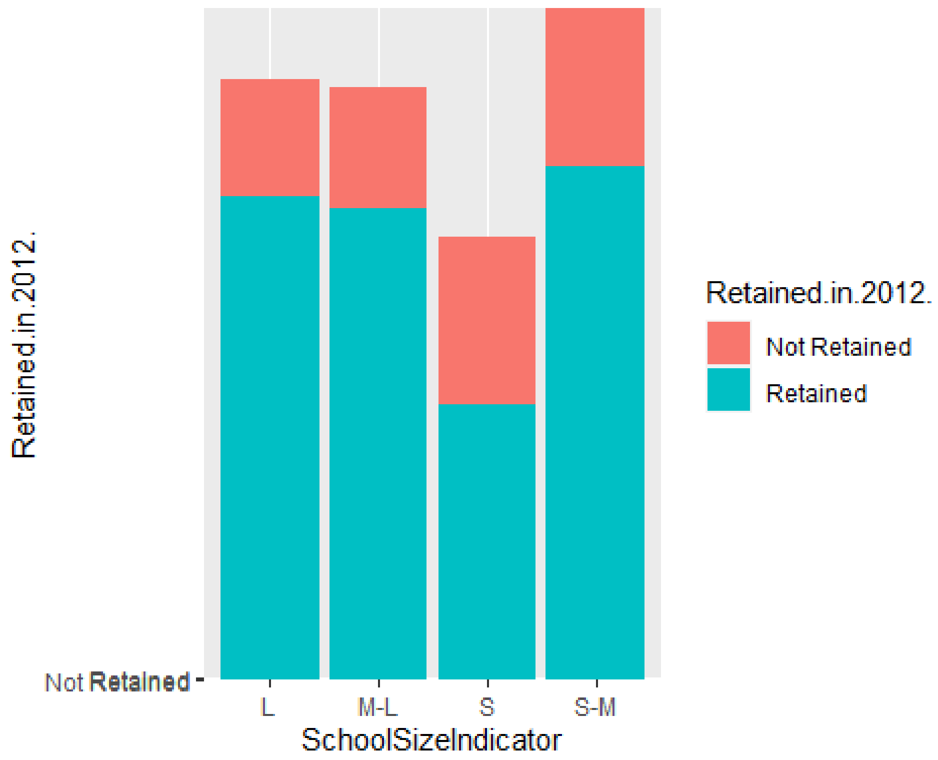
```
#plotting isnonannual against target
p2<-ggplot(data=rm1,
           aes(x=Is.Non.Annual.,y=Retained.in.2012.,
               fill=Retained.in.2012.)) +geom_bar( stat="identity")
p2
```



```
unique(rm1$SchoolSizeIndicator)

## [1] L   S-M M-L S
## Levels: L M-L S S-M

#plotting schoolsize against target
p3<-ggplot(data=rm1, aes(x=SchoolSizeIndicator,
                         y=Retained.in.2012., fill=Retained.in.2012.)) +
  geom_bar( stat="identity")
p3
```

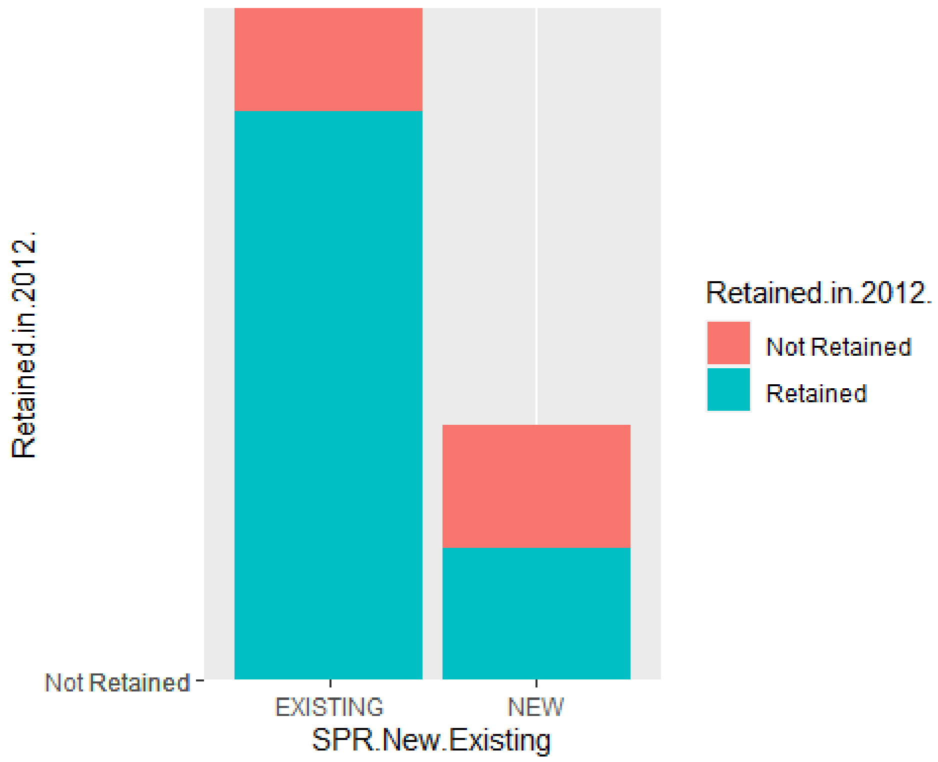
ANALYSIS

- Second Graph shows that for Is.Non.Annual → No i.e. the chances of Retaining STC is more if the program is Annual



```
#plotting spr new existing against target
p4<-ggplot(data=rm1, aes(x=SPR.New.Existing,y=Retained.in.2012.,
                         fill=Retained.in.2012.)) +
  geom_bar( stat="identity")
p4


ANALYSIS:
```

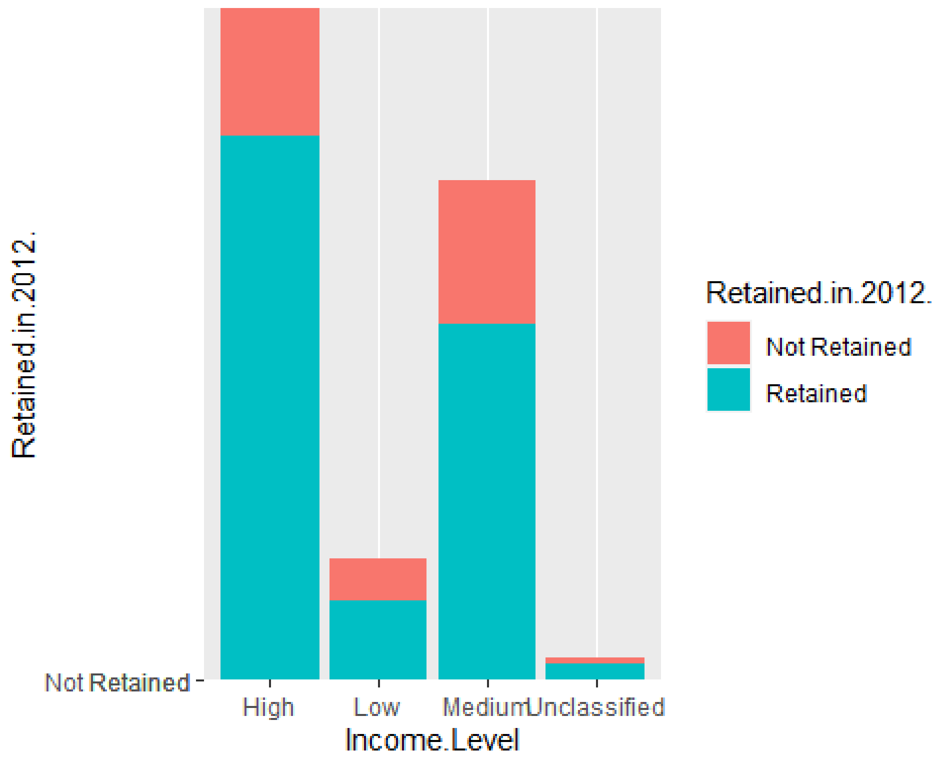
- The non-retained schools are maximum for Small School Size Indicators.



```
p6<-ggplot(data=rm1, aes(x=Income.Level,
                         y=Retained.in.2012., fill=Retained.in.2012.)) +
  geom_bar( stat="identity")
p6
```

ANALYSIS:

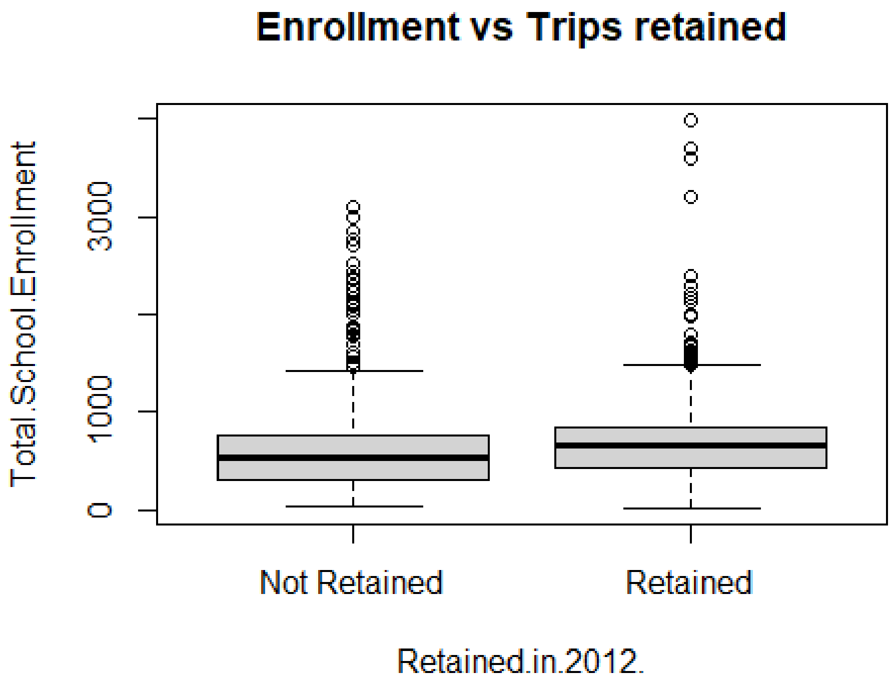
Group that has travelled with STC before has more chances of Retention as compared to the ones which has not travelled



```
## Total School Enrollment vs Retention
input_data <- rm1 [ , c("Total.School.Enrollment" ,"Retained.in.2012.")]


p7<- plot(y=input_data$Total.School.Enrollment , x=input_data$Retained.in.201
2.,
     main="Enrollment vs Trips retained",ylab="Total.School.Enrollment",
     xlab= "Retained.in.2012.")
```

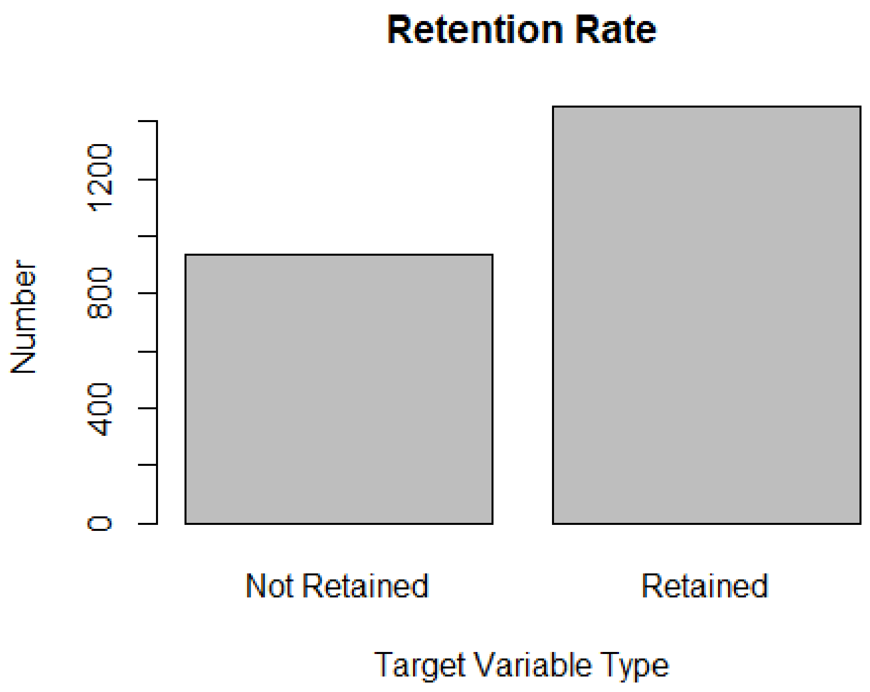
Analysis

Income Level High and Medium have the better retention as compared to Low and Unclassified.



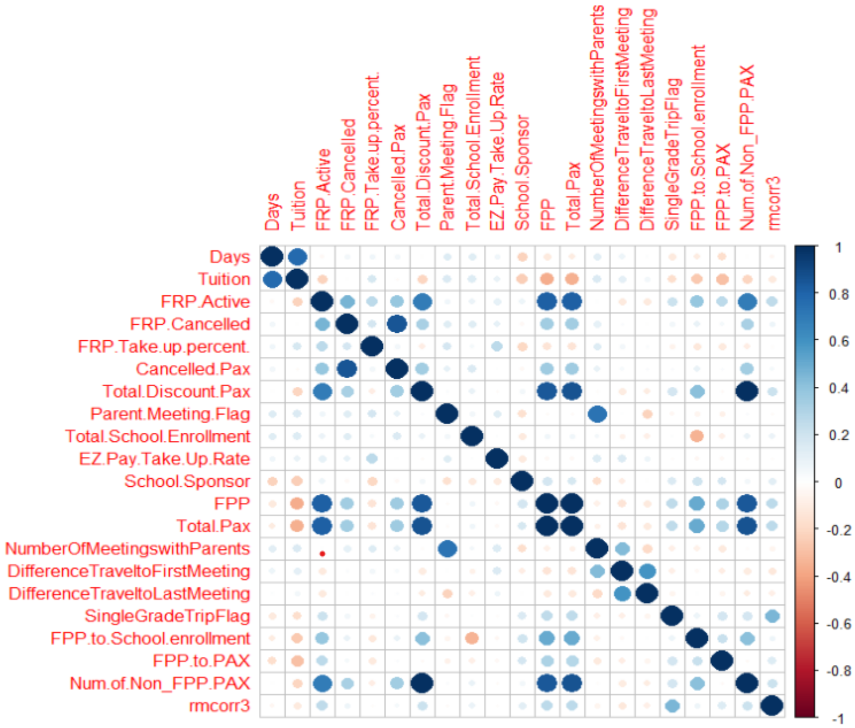
```
p7

## $stats
##      [,1]   [,2]
## [1,]   36   19.0
## [2,]  300  424.5
## [3,]  535  648.0
## [4,]  752  850.0
## [5,] 1425 1486.0
##
## $n
## [1]  938 1451
##
## $conf
##          [,1]     [,2]
## [1,] 511.6818 630.3509
## [2,] 558.3182 665.6491
##
## $out
##  [1] 1688 2159 1500 2778 1853 1602 2098 1606 1470 2850 2050 3100 2300 1559
1558
## [16] 2175 2000 2441 1785 3000 2169 2700 2765 2169 1800 1769 3000 1563 2120
2165
## [31] 2520 2235 2087 2375 2351 2000 1871 1606 1844 1693 2393 1555 1590 1514
1792
## [46] 2127 3600 3200 3200 1693 1600 1538 1611 1554 1625 2200 1700 3990 2000
1500
```

```
## [61] 1500 1974 3600 1494 1700 1528 1587 1500 3700 1497 1500 1558 1712 1672
1500
## [76] 2300 2168
##
## $group
##  [1] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 1 1
## [39] 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
2 2 2
## [77] 2
##
## $names
## [1] "Not Retained" "Retained"

Retained = rm1[which(rm1[,44]=='Retained'),]
Not_Retained = rm1[which(rm1[,31]=="Not Retained"),]

plot(rm1[,44],main="Retention Rate",ylab="Number",
     xlab= "Target Variable Type")
```



```
#Finding the correlation

# plotting correlation between numeric variables
#table(is.na(rmNew))
#is.na(rmNew)

rmcorr2 <- select_if(rm1, is.numeric)
```

```
rmcorr3 <- as.numeric(rm1$Retained.in.2012.)
rmcorr2 <- cbind(rmcorr2,rmcorr3)

df.cor = cor(rmcorr2)
corrplot(df.cor)
```



```
apa.cor.table(rmcorr2,"APA correlation Table3.doc")
```

```
Below are the screenshots from the APA table that was generated for all the n
umerical variables
```

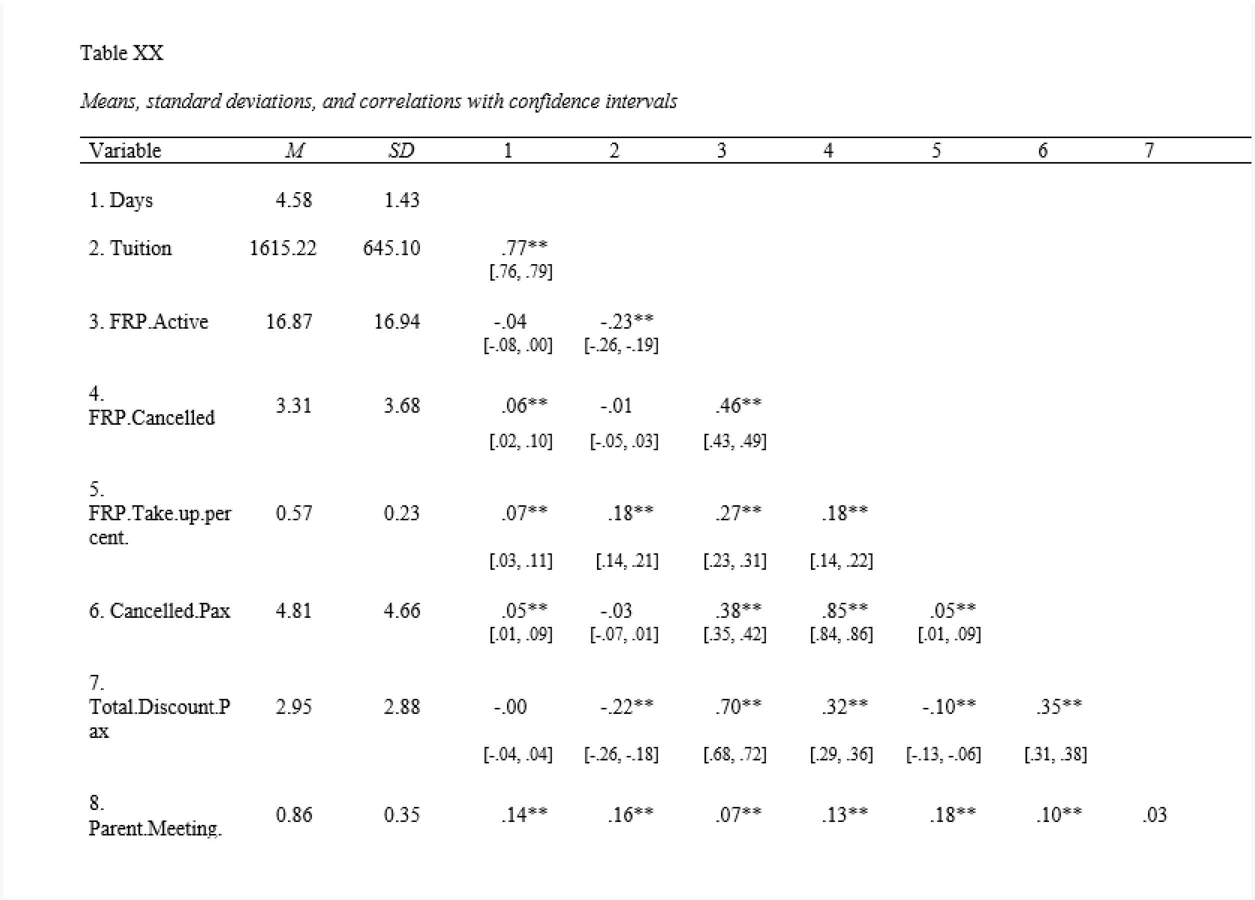
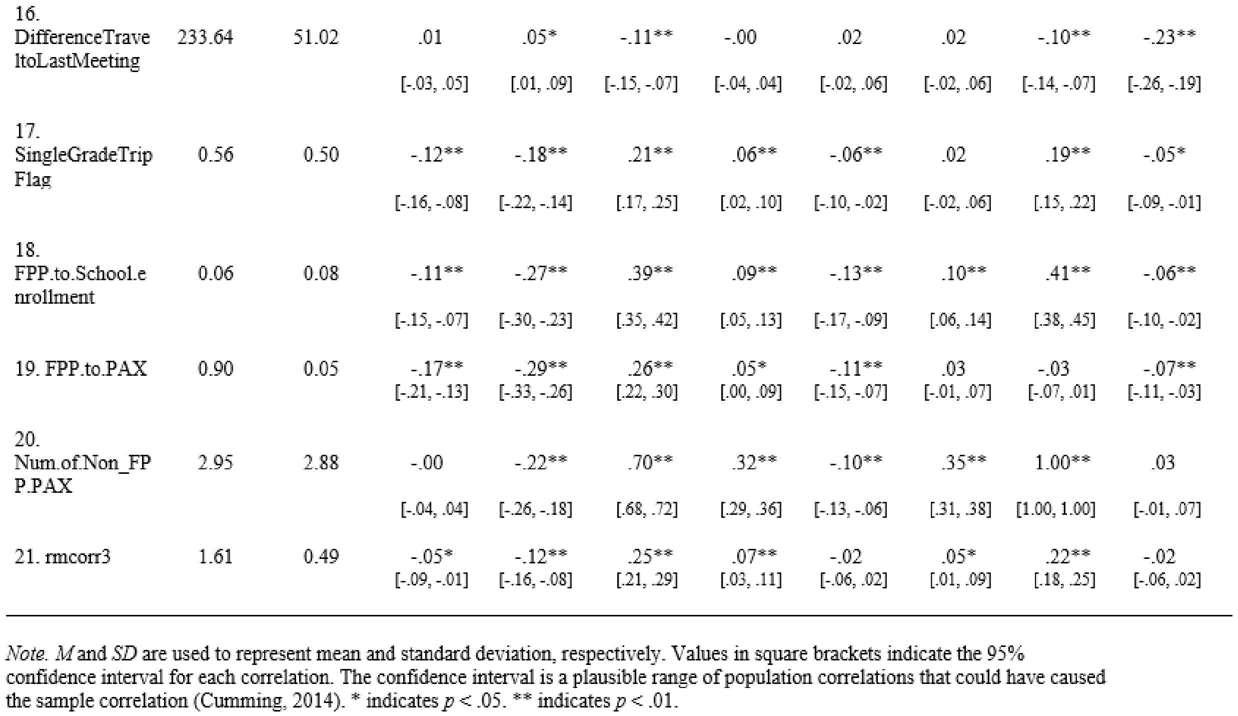


Table XX

Means, standard deviations, and correlations with confidence intervals

Variable	*M*	*SD*	1	2	3	4	5	6	7	
1. Days	4.58	1.43								
2. Tuition	1615.22	645.10	.77** [.76, .79]							
3. FRP.Active	16.87	16.94	-.04 [-.08, .00]	-.23** [-.26, -.19]						
4. FRP.Cancelled	3.31	3.68	.06** [.02, .10]	-.01 [-.05, .03]	.46** [.43, .49]					
5. FRP.Take.up.percent.	0.57	0.23	.07** [.03, .11]	.18** [.14, .21]	.27** [.23, .31]	.18** [.14, .22]				
6. Cancelled.Pax	4.81	4.66	.05** [.01, .09]	-.03 [-.07, .01]	.38** [.35, .42]	.85** [.84, .86]	.05** [.01, .09]			
7. Total.Discount.Pax	2.95	2.88	-.00 [-.04, .04]	-.22** [-.26, -.18]	.70** [.68, .72]	.32** [.29, .36]	-.10** [-.13, -.06]	.35** [.31, .38]		
8. Parent.Meeting.Flag	0.86	0.35	.14** [.10, .18]	.16** [.12, .20]	.07** [.03, .11]	.13** [.09, .17]	.18** [.14, .22]	.10** [.06, .14]	.03 [-.01, .07]	
9. Total.School.Enrollment	648.34	403.81	.13** [.09, .17]	.14** [.10, .18]	.09** [.05, .13]	.13** [.09, .17]	.05* [.01, .09]	.15** [.11, .19]	.10** [.06, .14]	.06** [.02, .10]
10. EZ.Pay.Take.Up.Rate	0.21	0.16	.07** [.03, .11]	.10** [.06, .14]	.11** [.07, .15]	.11** [.07, .15]	.26** [.23, .30]	.00 [-.04, .04]	-.02 [-.06, .02]	.14** [.10, .18]
11. School.Sponsor	0.11	0.31	-.22** [-.26, -.18]	-.25** [-.28, -.21]	.08** [.04, .12]	-.03 [-.07, .01]	-.20** [-.24, -.16]	-.03 [-.07, .01]	.05** [.01, .09]	-.16** [-.20, -.12]
12. FPP	31.30	29.13	-.12** [-.16, -.08]	-.36** [-.40, -.33]	.82** [.80, .83]	.34** [.31, .38]	-.15** [-.19, -.11]	.36** [.32, .39]	.84** [.83, .85]	-.03 [-.07, .01]
13. Total.Pax	34.25	31.59	-.11** [-.15, -.07]	-.36** [-.39, -.32]	.82** [.80, .83]	.34** [.31, .38]	-.15** [-.18, -.11]	.36** [.32, .39]	.87** [.86, .88]	-.02 [-.06, .02]
14. NumberOfMeetingswithParents	1.10	0.61	.13** [.09, .17]	.15** [.11, .19]	.00 [-.04, .04]	.11** [.07, .15]	.14** [.10, .18]	.09** [.05, .13]	-.02 [-.06, .02]	.73** [.71, .75]
15. DifferenceTraveltoFirstMeeting	262.07	73.70	.08** [.04, .12]	.12** [.08, .15]	-.11** [-.15, -.07]	.03 [-.01, .07]	.06** [.01, .09]	.04 [-.00, .08]	-.11** [-.15, -.07]	.00 [-.04, .04]



16. DifferenceTravelToLastMeeting	233.64	51.02	.01 [-.03, .05]	.05* [.01, .09]	-.11** [-.15, -.07]	-.00 [-.04, .04]	.02 [-.02, .06]	.02 [-.02, .06]	-.10** [-.14, -.07]	-.23** [-.26, -.19]
17. SingleGradeTripFlag	0.56	0.50	-.12** [-.16, -.08]	-.18** [-.22, -.14]	.21** [.17, .25]	.06** [.02, .10]	-.06** [-.10, -.02]	.02 [-.02, .06]	.19** [.15, .22]	-.05* [-.09, -.01]
18. FPP.to.School.enrollment	0.06	0.08	-.11** [-.15, -.07]	-.27** [-.30, -.23]	.39** [.35, .42]	.09** [.05, .13]	-.13** [-.17, -.09]	.10** [.06, .14]	.41** [.38, .45]	-.06** [-.10, -.02]
19. FPP.to.PAX	0.90	0.05	-.17** [-.21, -.13]	-.29** [-.33, -.26]	.26** [.22, .30]	.05* [.00, .09]	-.11** [-.15, -.07]	.03 [-.01, .07]	-.03 [-.07, .01]	-.07** [-.11, -.03]
20. Num.of.Non_FPP.PAX	2.95	2.88	-.00 [-.04, .04]	-.22** [-.26, -.18]	.70** [.68, .72]	.32** [.29, .36]	-.10** [-.13, -.06]	.35** [.31, .38]	1.00** [1.00, 1.00]	.03 [-.01, .07]
21. rmcorr3	1.61	0.49	-.05* [-.09, -.01]	-.12** [-.16, -.08]	.25** [.21, .29]	.07** [.03, .11]	-.02 [-.06, .02]	.05* [.01, .09]	.22** [.18, .25]	-.02 [-.06, .02]

Note. *M* and *SD* are used to represent mean and standard deviation, respectively. Values in square brackets indicate the 95% confidence interval for each correlation. The confidence interval is a plausible range of population correlations that could have caused the sample correlation (Cumming, 2014). * indicates $p < .05$. ** indicates $p < .01$.

#Decision Tree

```
#Data partitioning into train and Test data
```

# We have created a function that calculates all the evaluation matrices and gives the result in one go

```
###Functions for Evaluation metrics
EvaluationMatrix <-  function(TP,FP,FN,TN)
{
Accuracy <- (TP+TN)/(TP+TN+FP+FN)
Precision <-  TP/(TP + FP)
Recall <- TP/(TP +FN)
Fscore <- 2*(Recall * Precision) / (Recall + Precision)
FPR <- F/(TN+FP)

EVM <-cbind(Accuracy,Precision,Recall,Fscore,FPR)

}
```

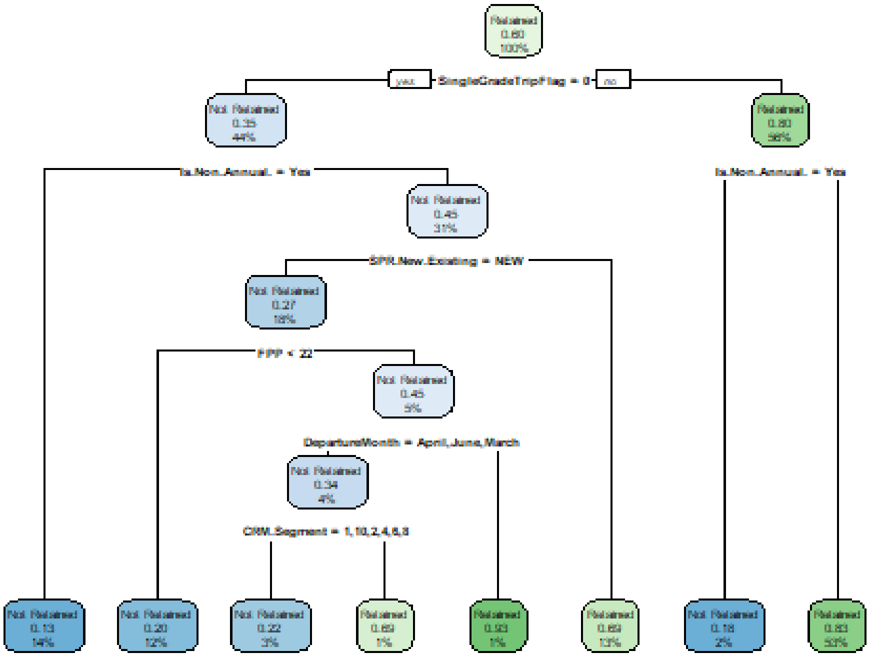
#This is the decision tree plotted for all the other variables vs target variable

```
## Decision Tree 1

set.seed(35)
```

```
indx <-  sample(2, nrow(rm1) , replace = T , prob = c(0.6 , 0.4))
train <- rm1[indx == 1 , ]
test <- rm1[indx == 2 , ]

#nrow(train)/nrow(rm1)
#nrow(test)/nrow(rm1)


myFormula = Retained.in.2012. ~ .
myTree <- rpart(myFormula , data = train)
print(myTree)
```

```
## n= 1398
##
## node), split, n, loss, yval, (yprob)
##       * denotes terminal node
##
##  1) root 1398 555 Retained (0.39699571 0.60300429)
##    2) SingleGradeTripFlag< 0.5 618 216 Not Retained (0.65048544 0.34951456
)
##      4) Is.Non.Annual.=Yes 190  24 Not Retained (0.87368421 0.12631579) *
##      5) Is.Non.Annual.=No 428 192 Not Retained (0.55140187 0.44859813)
##       10) SPR.New.Existing=NEW 249  68 Not Retained (0.72690763 0.27309237
)
##         20) FPP< 21.5 174  34 Not Retained (0.80459770 0.19540230) *
##         21) FPP>=21.5 75  34 Not Retained (0.54666667 0.45333333)
##           42) DepartureMonth=April,June,March 61  21 Not Retained (0.65573
770 0.34426230)
##             84) CRM.Segment=1,10,2,4,6,8 45  10 Not Retained (0.77777778 0
.22222222) *
##             85) CRM.Segment=5,7 16   5 Retained (0.31250000 0.68750000) *
##           43) DepartureMonth=February,May 14   1 Retained (0.07142857 0.92
857143) *
##       11) SPR.New.Existing=EXISTING 179  55 Retained (0.30726257 0.6927374
3) *
##    3) SingleGradeTripFlag>=0.5 780 153 Retained (0.19615385 0.80384615)
##      6) Is.Non.Annual.=Yes 33   6 Not Retained (0.81818182 0.18181818) *
##      7) Is.Non.Annual.=No 747 126 Retained (0.16867470 0.83132530) *
```

```
rpart.plot(myTree)
```



```
# prob of positive and negative class
##predict(myTree, data = train)


# finding the error rate for train data
pred_train1 <- predict(myTree, data = train , type = "class")
mean(train$Retained.in.2012. != pred_train1)

## [1] 0.1866953

##length(pred_train1)
##length(rm1$Retained.in.2012.)

# finding the error rate for test data
pred_test1 <- predict( myTree, data1 = test , type = "class")
mean(test$Retained.in.2012. != pred_test1)

## [1] 0.4484979

##view(pred_test1)
```

```
## Evaluation Matrix
DTMatrix1 <-table(actual=as.factor(train$Retained.in.2012.), pred = pred_test
1)
DTMatrix1

##                pred
## actual          Not Retained Retained
##   Not Retained           368      187
##   Retained                74      769

FinalEVM1 <- EvaluationMatrix(DTMatrix1[1,1], DTMatrix1[2,1],DTMatrix1[1,2],
                              DTMatrix1[2,2])


FinalEVM1

##       Accuracy Precision    Recall    Fscore
## [1,] 0.8133047 0.8325792 0.6630631 0.7382146

summary(myTree)

## Call:
## rpart(formula = myFormula, data = train)
##   n= 1398
## 
##           CP nsplit rel error    xerror       xstd
## 1 0.33513514      0 1.0000000 1.0000000 0.03296201
## 2 0.06216216      1 0.6648649 0.6648649 0.02969438
## 3 0.03783784      3 0.5405405 0.5405405 0.02765765
## 4 0.01081081      4 0.5027027 0.5171171 0.02721143
## 5 0.01000000      7 0.4702703 0.5261261 0.02738561
## 
## Variable importance
## SingleGradeTripFlag          From.Grade      Is.Non.Annual.    SPR.New.Exi
sting 
##                  25                  16                  16 
13 
##      SchoolGradeType SchoolGradeTypeHigh           Total.Pax 
FPP 
##                   9                   8                   2 
2 
##         CRM.Segment        Program.Code      DepartureMonth          FRP.A
ctive 
##                   2                   2                   1 
1 
##  Num.of.Non_FPP.PAX  Total.Discount.Pax 
##                   1                   1 
## 
## Node number 1: 1398 observations,    complexity param=0.3351351
##   predicted class=Retained      expected loss=0.3969957  P(node) =1
##     class counts:   555   843
##    probabilities: 0.397 0.603 
##   left son=2 (618 obs) right son=3 (780 obs)
##   Primary splits:
##       SingleGradeTripFlag < 0.5       to the left,  improve=142.34810, (0 
missing)
##       Is.Non.Annual.      splits as  RL, improve=116.46010, (0 missing)
##       From.Grade          splits as  LLLLRLLLRL, improve=110.41770, (0 mis
sing)
##       FPP                 < 23.5      to the left,  improve= 65.86060, (0 
missing)
##       SPR.New.Existing    splits as  RL, improve= 65.40623, (0 missing)
##   Surrogate splits:
##       From.Grade          splits as  LLRLRLLLRL, agree=0.845, adj=0.650, (
0 split)
##       SchoolGradeType     splits as  RLLLLLRLL,  agree=0.718, adj=0.362, (
0 split)
##       SchoolGradeTypeHigh splits as  RLRL,       agree=0.695, adj=0.309, (
0 split)
##       Is.Non.Annual.      splits as  RL,         agree=0.670, adj=0.254, (
0 split)
##       SPR.New.Existing    splits as  RL,         agree=0.666, adj=0.244, (
0 split)
## 
## Node number 2: 618 observations,    complexity param=0.06216216
##   predicted class=Not Retained  expected loss=0.3495146  P(node) =0.442060
1
##     class counts:   402   216
##    probabilities: 0.650 0.350 
##   left son=4 (190 obs) right son=5 (428 obs)
##   Primary splits:
##       Is.Non.Annual.          splits as  RL, improve=27.33455, (0 missing)
##       MDR.Low.Grade           splits as  -RLLLLRRLLLL, improve=16.20208, (
0 missing)
##       GroupGradeType          splits as  LLLLRLLRRLLLR, improve=16.15899, 
(0 missing)
##       GroupGradeTypeLow       splits as  LLLRLR, improve=14.99374, (0 miss
ing)
##       Total.School.Enrollment < 320.5     to the left,  improve=14.81133, 
(0 missing)
##   Surrogate splits:
##       Program.Code                      splits as  RLL--RRRRRRRR-RRRRRRRRRLR
RR, agree=0.709, adj=0.053, (0 split)
##       DifferenceTraveltoFirstMeeting < 435       to the right, agree=0.709
, adj=0.053, (0 split)
##       School.Sponsor                 < 0.5       to the right, agree=0.707
, adj=0.047, (0 split)
##       MajorProgramCode               splits as  LRRR, agree=0.704, adj=0.0
37, (0 split)
##       FPP.to.School.enrollment       < 0.1981229 to the right, agree=0.699
, adj=0.021, (0 split)
```

```
##
## Node number 3: 780 observations,    complexity param=0.03783784
##   predicted class=Retained      expected loss=0.1961538  P(node) =0.557939
9
##     class counts:   153   627
##    probabilities: 0.196 0.804
##   left son=6 (33 obs) right son=7 (747 obs)
##   Primary splits:
##       Is.Non.Annual.     splits as  RL, improve=26.66477, (0 missing)
##       FPP                < 21.5      to the left,  improve=22.57009, (0 mi
ssing)
##       Total.Pax          < 23.5      to the left,  improve=21.80439, (0 mi
ssing)
##       FRP.Active         < 16.5      to the left,  improve=16.93157, (0 mi
ssing)
##       Total.Discount.Pax < 2.5       to the left,  improve=16.51989, (0 mi
ssing)
##   Surrogate splits:
##       Program.Code splits as  RRLRRRRRR-RRRRRRR-RR---RRRRR, agree=0.959, a
dj=0.03, (0 split)
##
## Node number 4: 190 observations
##   predicted class=Not Retained  expected loss=0.1263158  P(node) =0.135908
4
##     class counts:   166    24
##    probabilities: 0.874 0.126
##
## Node number 5: 428 observations,    complexity param=0.06216216
##   predicted class=Not Retained  expected loss=0.4485981  P(node) =0.306151
6
##     class counts:   236   192
##    probabilities: 0.551 0.449
##   left son=10 (249 obs) right son=11 (179 obs)
##   Primary splits:
##       SPR.New.Existing   splits as  RL, improve=36.67776, (0 missing)
##       FPP                < 21.5      to the left,  improve=21.34238, (0 mi
ssing)
##       Total.Pax          < 23.5      to the left,  improve=21.34238, (0 mi
ssing)
##       FRP.Active         < 19.5      to the left,  improve=19.84926, (0 mi
ssing)
##       Total.Discount.Pax < 3.5       to the left,  improve=16.35588, (0 mi
ssing)
##   Surrogate splits:
##       Program.Code splits as  LLR--RLRRLLLR-LLRRRRLLLL-LLL, agree=0.638, a
dj=0.134, (0 split)
##       CRM.Segment  splits as  LLLLRLRLLLL, agree=0.626, adj=0.106, (0 spli
t)
##       FRP.Active   < 15.5      to the left,  agree=0.624, adj=0.101, (0 sp
lit)
```

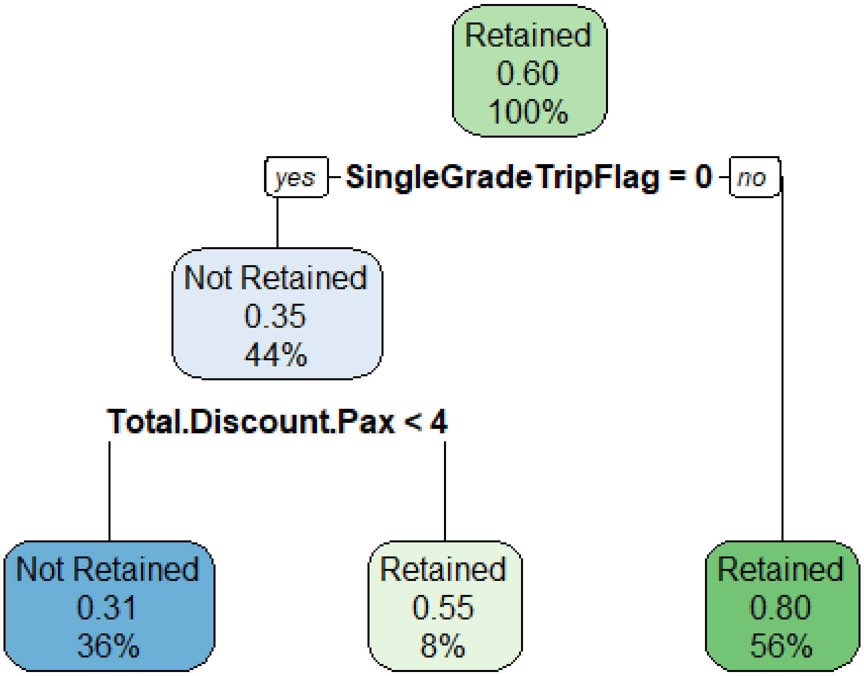
```
##       Total.Pax    < 34.5      to the left,  agree=0.624, adj=0.101, (0 sp
lit)
##       FPP          < 31.5      to the left,  agree=0.621, adj=0.095, (0 sp
lit)
##
## Node number 6: 33 observations
##   predicted class=Not Retained  expected loss=0.1818182  P(node) =0.023605
15
##     class counts:    27     6
##    probabilities: 0.818 0.182
##
## Node number 7: 747 observations
##   predicted class=Retained      expected loss=0.1686747  P(node) =0.534334
8
##     class counts:   126   621
##    probabilities: 0.169 0.831
##
## Node number 10: 249 observations,    complexity param=0.01081081
##   predicted class=Not Retained  expected loss=0.2730924  P(node) =0.178111
6
##     class counts:   181    68
##    probabilities: 0.727 0.273
##   left son=20 (174 obs) right son=21 (75 obs)
##   Primary splits:
##       FPP                            < 21.5      to the left,  improve=6.9
73461, (0 missing)
##       Total.Pax                      < 25.5      to the left,  improve=6.9
40597, (0 missing)
##       FRP.Active                     < 22        to the left,  improve=6.3
69250, (0 missing)
##       DifferenceTraveltoFirstMeeting < 244.5     to the right, improve=6.0
72753, (0 missing)
##       To.Grade                       splits as  RLL--LRRRR, improve=5.9496
07, (0 missing)
##   Surrogate splits:
##       Total.Pax          < 23.5      to the left,  agree=0.996, adj=0.987,
(0 split)
##       FRP.Active         < 12.5      to the left,  agree=0.867, adj=0.560,
(0 split)
##       Total.Discount.Pax < 2.5       to the left,  agree=0.855, adj=0.520,
(0 split)
##       Num.of.Non_FPP.PAX < 2.5       to the left,  agree=0.855, adj=0.520,
(0 split)
##       Travel.Type        splits as  LR-, agree=0.779, adj=0.267, (0 split)
##
## Node number 11: 179 observations
##   predicted class=Retained      expected loss=0.3072626  P(node) =0.128040
1
##     class counts:    55   124
##    probabilities: 0.307 0.693
```

```
##
## Node number 20: 174 observations
##   predicted class=Not Retained  expected loss=0.1954023  P(node) =0.124463
5
##     class counts:   140    34
##    probabilities: 0.805 0.195
##
## Node number 21: 75 observations,    complexity param=0.01081081
##   predicted class=Not Retained  expected loss=0.4533333  P(node) =0.053648
07
##     class counts:    41    34
##    probabilities: 0.547 0.453
##   left son=42 (61 obs) right son=43 (14 obs)
##   Primary splits:
##       DepartureMonth           splits as  LR-LLR, improve=7.775207, (0 mis
sing)
##       Total.School.Enrollment  < 296       to the left,  improve=6.842705,
(0 missing)
##       FPP.to.School.enrollment < 0.1140436 to the right, improve=5.623069,
(0 missing)
##       Program.Code             splits as  LR---RRL--RL--LL------L---RR, im
prove=5.180022, (0 missing)
##       From.Grade               splits as  LL-LLLRLRL, improve=4.425844, (0
missing)
##   Surrogate splits:
##       Poverty.Code   splits as  -LLLLR, agree=0.840, adj=0.143, (0 split)
##       School.Type    splits as  LLRL, agree=0.840, adj=0.143, (0 split)
##       To.Grade       splits as  LLL--L-RLL, agree=0.827, adj=0.071, (0 spl
it)
##       Cancelled.Pax  < 1.5       to the right, agree=0.827, adj=0.071, (0
split)
##       MDR.High.Grade splits as  -L-L---LLLLR, agree=0.827, adj=0.071, (0 s
plit)
##
## Node number 42: 61 observations,    complexity param=0.01081081
##   predicted class=Not Retained  expected loss=0.3442623  P(node) =0.043633
76
##     class counts:    40    21
##    probabilities: 0.656 0.344
##   left son=84 (45 obs) right son=85 (16 obs)
##   Primary splits:
##       CRM.Segment              splits as  LL-L-LRLRL-, improve=5.110428, (
0 missing)
##       Total.School.Enrollment  < 296       to the left,  improve=4.714897,
(0 missing)
##       GroupGradeType           splits as  --LL-LRLRLLLR, improve=4.198992,
(0 missing)
##       FPP.to.School.enrollment < 0.1439773 to the right, improve=3.915984,
(0 missing)
##       FRP.Cancelled            < 5.5       to the left,  improve=3.892203,
(0 missing)
##   Surrogate splits:
##       GroupGradeTypeLow        splits as  LLLLLR, agree=0.836, adj=0.375, 
(0 split)
##       GroupGradeTypeHigh       splits as  LLLR, agree=0.836, adj=0.375, (0 
split)
##       GroupGradeType           splits as  --LL-LLLLLLLR, agree=0.836, adj=
0.375, (0 split)
##       FPP.to.School.enrollment < 0.0040625 to the right, agree=0.836, adj=
0.375, (0 split)
##       Program.Code             splits as  RR---LLR--LL--LL------L---LR, ag
ree=0.820, adj=0.312, (0 split)
## 
## Node number 43: 14 observations
##   predicted class=Retained      expected loss=0.07142857  P(node) =0.01001
431
##     class counts:     1    13
##    probabilities: 0.071 0.929 
## 
## Node number 84: 45 observations
##   predicted class=Not Retained  expected loss=0.2222222  P(node) =0.032188
84
##     class counts:    35    10
##    probabilities: 0.778 0.222 
## 
## Node number 85: 16 observations
##   predicted class=Retained      expected loss=0.3125  P(node) =0.01144492
##     class counts:     5    11
##    probabilities: 0.312 0.688
```

Decision tree with numeric variables

```
set.seed(35)
indx <-  sample(2, nrow(rm1) , replace = T , prob = c(0.6 , 0.4))
train <- rm1[indx == 1 , ]
test <- rm1[indx == 2 , ]

myFormulanum = Retained.in.2012. ~ Tuition + FRP.Active  +School.Sponsor + 
  FPP+ Total.Pax + SingleGradeTripFlag + Total.Discount.Pax

myTreenum <- rpart(myFormulanum , data = train ,method="class")
print(myTreenum)

## n= 1398 
## 
## node), split, n, loss, yval, (yprob)
##       * denotes terminal node
## 
## 1) root 1398 555 Retained (0.3969957 0.6030043)  
##   2) SingleGradeTripFlag< 0.5 618 216 Not Retained (0.6504854 0.3495146)  
##     4) Total.Discount.Pax< 3.5 508 155 Not Retained (0.6948819 0.3051181)
```

```
*
##     5) Total.Discount.Pax>=3.5 110  49 Retained (0.4454545 0.5545455) *
##   3) SingleGradeTripFlag>=0.5 780 153 Retained (0.1961538 0.8038462) *

rpart.plot(myTreenum)
```



```
predict(myTreenum, data = train)

##      Not Retained  Retained
## 1       0.1961538 0.8038462
## 2       0.1961538 0.8038462
## 3       0.4454545 0.5545455
## 4       0.6948819 0.3051181
## 5       0.6948819 0.3051181
## 6       0.1961538 0.8038462
## 7       0.1961538 0.8038462
## 8       0.1961538 0.8038462
## 9       0.1961538 0.8038462
## 10      0.1961538 0.8038462
## 11      0.1961538 0.8038462
## 12      0.1961538 0.8038462
## 13      0.1961538 0.8038462
## 14      0.1961538 0.8038462
## 15      0.1961538 0.8038462
## 16      0.4454545 0.5545455
## 17      0.4454545 0.5545455
## 18      0.1961538 0.8038462
```

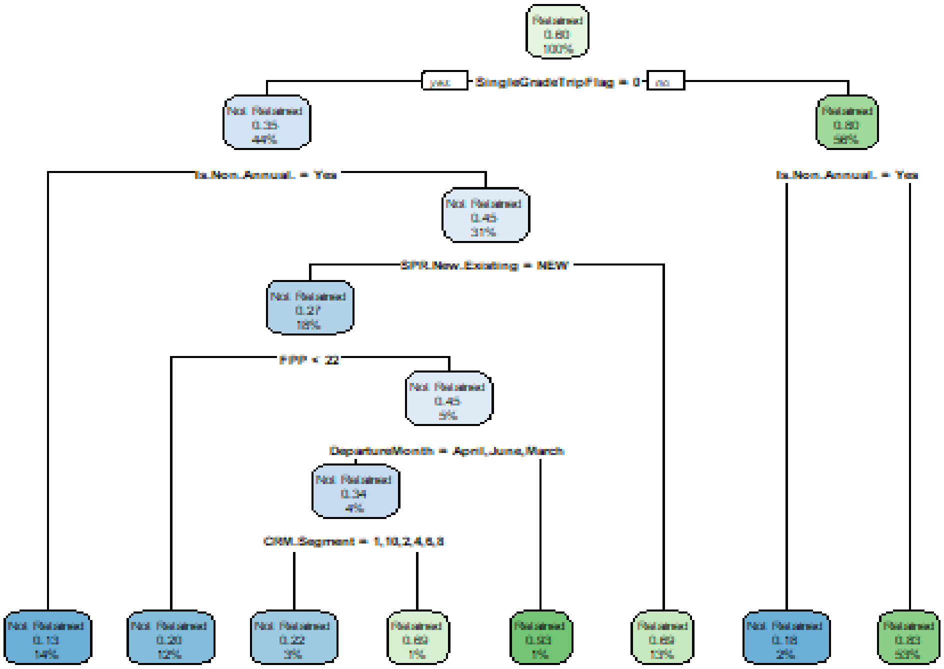
```
##p <- predict(myTreenum,rm1,type="class")
# finding the error rate for train data
pred_trainnum <- predict( myTreenum , data = train , type = "class")
mean(train$Retained.in.2012. != pred_trainnum)

## [1] 0.2553648

# finding the error rate for test data
pred_testnum <- predict( myTreenum , data1 = test , type = "class")
mean(test$Retained.in.2012. != pred_testnum)

## [1] 0.4613734

## Levels: Not Retained Retained

## Evaluation Matrix

DTMatrixnum <- table(actual=train$Retained.in.2012., pred = pred_testnum)
DTMatrixnum

##                pred
## actual         Not Retained Retained
##   Not Retained          353      202
##   Retained              155      688

FinalEVMnum <-
  EvaluationMatrix(DTMatrixnum[1,1],DTMatrixnum[2,1],
                   DTMatrix1[1,2], DTMatrixnum[2,2])
FinalEVMnum

##       Accuracy Precision    Recall    Fscore
## [1,] 0.7527115 0.6948819 0.6537037 0.6736641

summary(myTreenum)

## Call:
## rpart(formula = myFormulanum, data = train, method = "class")
##   n= 1398
##
##           CP nsplit rel error    xerror       xstd
## 1 0.33513514      0 1.0000000 1.0000000 0.03296201
## 2 0.02162162      1 0.6648649 0.6648649 0.02969438
## 3 0.01000000      2 0.6432432 0.6702703 0.02977135
##
## Variable importance
## SingleGradeTripFlag           Total.Pax                 FPP  Total.Discoun
t.Pax
```

```
##                  60                  11                  11                  
8 
##          FRP.Active             Tuition 
##                   7                   3 
## 
## Node number 1: 1398 observations,    complexity param=0.3351351
##   predicted class=Retained      expected loss=0.3969957  P(node) =1
##     class counts:   555   843
##    probabilities: 0.397 0.603 
##   left son=2 (618 obs) right son=3 (780 obs)
##   Primary splits:
##       SingleGradeTripFlag < 0.5    to the left,  improve=142.34810, (0 mis
sing)
##       FPP                 < 23.5   to the left,  improve= 65.86060, (0 mis
sing)
##       Total.Pax           < 25.5   to the left,  improve= 64.89708, (0 mis
sing)
##       FRP.Active          < 16.5   to the left,  improve= 47.74499, (0 mis
sing)
##       Total.Discount.Pax  < 2.5    to the left,  improve= 44.74524, (0 mis
sing)
##   Surrogate splits:
##       Total.Pax          < 17.5   to the left,  agree=0.623, adj=0.147, (0 
split)
##       FPP                < 16.5   to the left,  agree=0.622, adj=0.144, (0 
split)
##       FRP.Active         < 9.5    to the left,  agree=0.602, adj=0.099, (0 
split)
##       Total.Discount.Pax < 1.5    to the left,  agree=0.583, adj=0.057, (0 
split)
##       Tuition            < 2268   to the right, agree=0.581, adj=0.052, (0 
split)
## 
## Node number 2: 618 observations,    complexity param=0.02162162
##   predicted class=Not Retained  expected loss=0.3495146  P(node) =0.442060
1
##     class counts:   402   216
##    probabilities: 0.650 0.350 
##   left son=4 (508 obs) right son=5 (110 obs)
##   Primary splits:
##       Total.Discount.Pax < 3.5    to the left,  improve=11.250870, (0 miss
ing)
##       Total.Pax          < 25.5   to the left,  improve= 9.479772, (0 miss
ing)
##       FPP                < 21.5   to the left,  improve= 8.567690, (0 miss
ing)
##       FRP.Active         < 29.5   to the left,  improve= 8.098361, (0 miss
ing)
##       Tuition            < 1235.5 to the right, improve= 3.526777, (0 miss
ing)
```

```
##   Surrogate splits:
##       Total.Pax  < 42.5   to the left,  agree=0.900, adj=0.436, (0 split)
##       FPP        < 39.5   to the left,  agree=0.892, adj=0.391, (0 split)
##       FRP.Active < 28.5   to the left,  agree=0.869, adj=0.264, (0 split)
##       Tuition    < 371.5  to the right, agree=0.827, adj=0.027, (0 split)
## 
## Node number 3: 780 observations
##   predicted class=Retained      expected loss=0.1961538  P(node) =0.557939
9
##     class counts:   153   627
##    probabilities: 0.196 0.804 
## 
## Node number 4: 508 observations
##   predicted class=Not Retained  expected loss=0.3051181  P(node) =0.363376
3
##     class counts:   353   155
##    probabilities: 0.695 0.305 
## 
## Node number 5: 110 observations
##   predicted class=Retained      expected loss=0.4454545  P(node) =0.078683
83
##     class counts:    49    61
##    probabilities: 0.445 0.555
```

Decision Tree myTreenum2

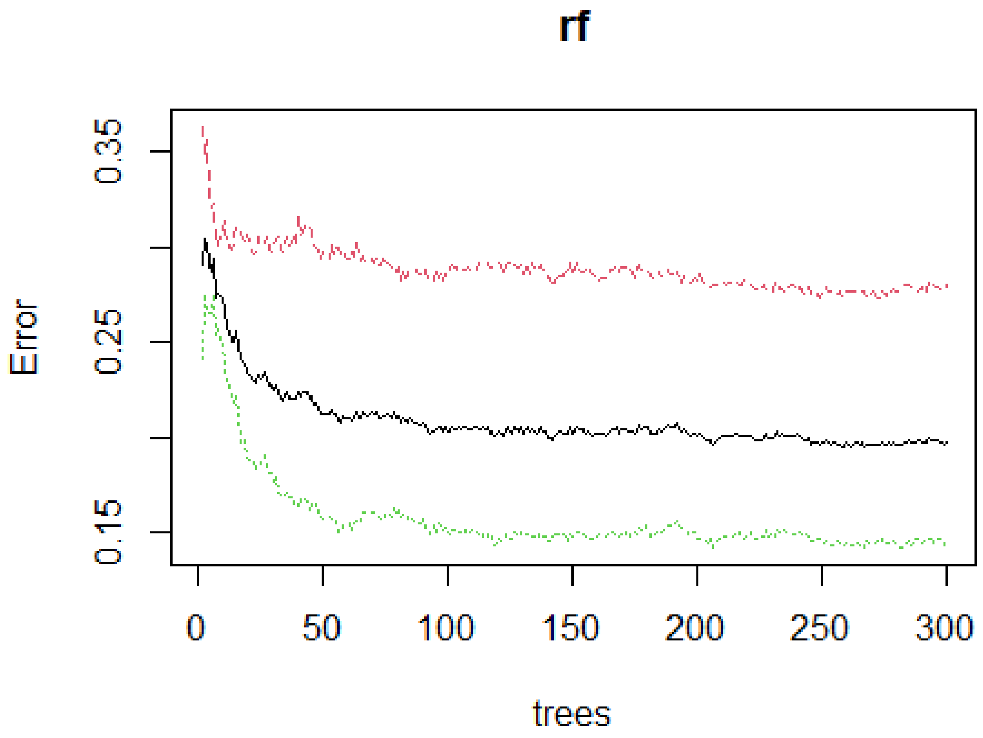
```
# pruning with gini split
myTreenum2 <- rpart(myFormula , data = train  , parms =list (split = "gini")
,
                    control = rpart.control(minsplit = 6 , minbucket = 6 ,
                                            cp = 0.01 ))


rpart.plot(myTreenum2)
```



```
# finding the error rate for train data
pred_trainnum2 <- predict( myTreenum2 , data = train , type = "class")
mean(train$Retained.in.2012. != pred_trainnum2)

## [1] 0.1866953

# finding the error rate for test data
pred_testnum2 <- predict( myTreenum2 , data = test , type = "class")
mean(test$Retained.in.2012. != pred_testnum2)

## [1] 0.4484979

## Evaluation Matrix
DTMatrixnum2 <- table(actual=train$Retained.in.2012., pred = pred_testnum2)
DTMatrixnum2

##               pred
## actual          Not Retained Retained
##   Not Retained           368      187
##   Retained                74      769

FinalEVMnum2 <- EvaluationMatrix(DTMatrixnum2[1,1], DTMatrixnum2[2,1],
                                 DTMatrixnum2[1,2],DTMatrixnum2[2,2])
FinalEVMnum2

##       Accuracy Precision    Recall    Fscore
## [1,] 0.8133047 0.8325792 0.6630631 0.7382146
```

Random Forest

```
library(randomForest)

rf<- randomForest(Retained.in.2012.~.,data = rm2,mtry = sqrt(ncol(rm2)-1),
                   ntree = 300,proximity = T,importance = T)

print(rf)



#We have passed all the variables from the dataset rm2 while applying Random
Forest.

Below are evaluation matrices highlighted from the R output

##
## Call:
##  randomForest(formula = Retained.in.2012. ~ ., data = rm2, mtry = sqrt(nco
l(rm2) -      1), ntree = 300, proximity = T, importance = T)
##                Type of random forest: classification
##                      Number of trees: 300
## No. of variables tried at each split: 7
##
##          OOB estimate of  error rate: 19.76%
## Confusion matrix:
##              Not Retained Retained class.error
## Not Retained          675      263   0.2803838
## Retained              209     1242   0.1440386

## The OOB error rate of our random forest model is .1972

attributes(rf)

## $names
##  [1] "call"            "type"            "predicted"       "err.rate"
##  [5] "confusion"       "votes"           "oob.times"       "classes"
##  [9] "importance"      "importanceSD"    "localImportance" "proximity"
## [13] "ntree"           "mtry"            "forest"          "y"
## [17] "test"            "inbag"           "terms"
##
## $class
## [1] "randomForest.formula" "randomForest"

plot(rf)
```



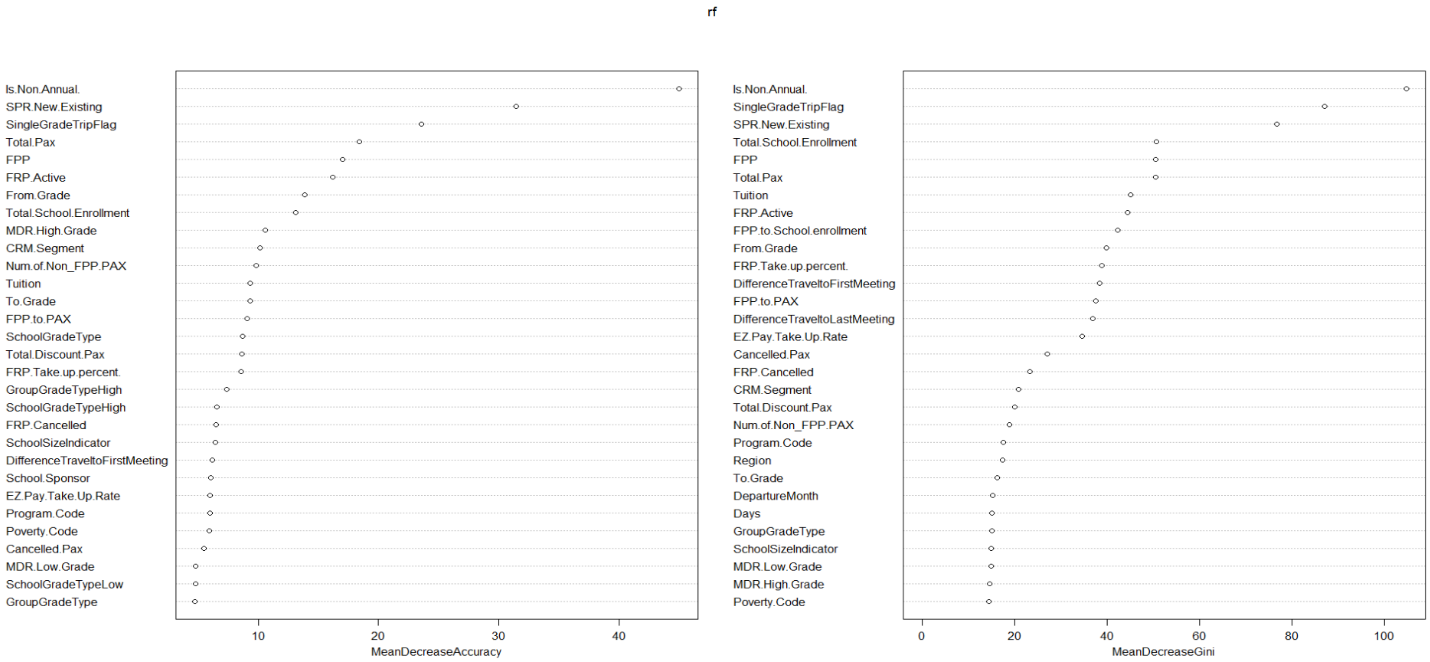
#The red cures is the error rate for the positive class that is **retained**, green curve for the negative class that is **not retained**. And black curve indicates error rate for OOB

```
##rf$err.rate

##  important variables based on MeanDecreaseAccuracy.

library("dplyr")
IMP <- importance(rf, type = 1)
IMP

##                                   MeanDecreaseAccuracy
## Program.Code                                 5.981816
## From.Grade                                  13.853810
## To.Grade                                     9.297925
## Is.Non.Annual.                              44.992248
## Days                                         4.436340
## Travel.Type                                  3.714249
## Tuition                                      9.312343
## FRP.Active                                  16.164282
## FRP.Cancelled                                6.486094
## FRP.Take.up.percent.                         8.548106
## Cancelled.Pax                                5.447301
## Total.Discount.Pax                           8.597315
## Poverty.Code                                 5.885952
## Region                                     4.020956
## CRM.Segment                               10.131317
## School.Type                                4.009359
## Parent.Meeting.Flag                        1.265172
## MDR.Low.Grade                              4.809387
## MDR.High.Grade                            10.605804
## Total.School.Enrollment                   13.129944
## Income.Level                               1.591752
## EZ.Pay.Take.Up.Rate                        6.005964
## School.Sponsor                             6.028242
## SPR.Product.Type                           3.993427
## SPR.New.Existing                          31.426084
## FPP                                       17.030262
## Total.Pax                                 18.369903
## NumberOfMeetingswithParents                1.253938
## DifferenceTraveltoFirstMeeting             6.159915
## DifferenceTraveltoLastMeeting              3.468803
## SchoolGradeTypeLow                         4.751578
## SchoolGradeTypeHigh                        6.539745
## SchoolGradeType                            8.718688
## DepartureMonth                             2.444905
## GroupGradeTypeLow                          4.561715
## GroupGradeTypeHigh                         7.380710
## GroupGradeType                             4.732097
## MajorProgramCode                           1.739418
## SingleGradeTripFlag                       23.550554
## FPP.to.School.enrollment                   2.999440
## FPP.to.PAX                                 9.086139
## Num.of.Non_FPP.PAX                         9.837632
## SchoolSizeIndicator                        6.394895
```

```
#We also tried to see the more important variable by setting a threshold on mean decrease accuracy > 10

subset(IMP, IMP[] >  10)
```

```
##                         MeanDecreaseAccuracy
## From.Grade                          13.85381
## Is.Non.Annual.                      44.99225
## FRP.Active                          16.16428
## CRM.Segment                         10.13132
## MDR.High.Grade                      10.60580
## Total.School.Enrollment             13.12994
## SPR.New.Existing                    31.42608
## FPP                                 17.03026
## Total.Pax                           18.36990
## SingleGradeTripFlag                 23.55055
```

```
##filter(IMP, MeanDecreaseAccuracy >= 10)

varImpPlot(rf)
```



#Below are a few instances printed for proximity calculated for Random Forest

```
rf$proximity

##               1          2          3          4          5          6
## 1    1.00000000 0.00000000 0.02439024 0.00000000 0.00000000 0.00000000
## 2    0.00000000 1.00000000 0.16279070 0.00000000 0.00000000 0.00000000
## 3    0.02439024 0.16279070 1.00000000 0.00000000 0.00000000 0.00000000
## 4    0.00000000 0.00000000 0.00000000 1.00000000 0.00000000 0.00000000
## 5    0.00000000 0.00000000 0.00000000 0.00000000 1.00000000 0.00000000
## 6    0.00000000 0.00000000 0.00000000 0.00000000 0.00000000 1.00000000
## 7    0.00000000 0.00000000 0.00000000 0.02325581 0.02040816 0.04651163
## 8    0.00000000 0.00000000 0.02500000 0.02941176 0.00000000 0.00000000
## 9    0.02777778 0.05405405 0.06818182 0.00000000 0.02272727 0.00000000
## 10   0.05128205 0.03225806 0.12765957 0.00000000 0.06818182 0.00000000
## 11   0.06250000 0.02564103 0.06818182 0.00000000 0.02127660 0.00000000
## 12   0.06976744 0.00000000 0.13953488 0.00000000 0.09090909 0.00000000
## 13   0.00000000 0.16216216 0.08333333 0.00000000 0.00000000 0.00000000
## 14   0.05882353 0.05128205 0.28260870 0.00000000 0.00000000 0.00000000
## 15   0.11428571 0.00000000 0.11428571 0.00000000 0.00000000 0.00000000
## 16   0.00000000 0.03225806 0.12765957 0.00000000 0.01785714 0.00000000
## 17   0.00000000 0.03125000 0.06818182 0.00000000 0.02000000 0.00000000
## 18   0.00000000 0.02702703 0.00000000 0.00000000 0.00000000 0.00000000
## 19   0.05714286 0.00000000 0.14634146 0.00000000 0.02127660 0.00000000
## 20   0.00000000 0.00000000 0.03921569 0.00000000 0.03921569 0.00000000
## 33   0.00000000 0.00000000 0.00000000 0.00000000 0.00000000 0.00000000
## 34   0.00000000 0.00000000 0.00000000 0.00000000 0.00000000 0.00000000
## 35   0.00000000 0.00000000 0.10256410 0.00000000 0.25000000 0.06060606
## 36   0.00000000 0.00000000 0.05128205 0.00000000 0.15909091 0.18181818
## 37   0.00000000 0.00000000 0.10638298 0.00000000 0.12195122 0.10810811
## 38   0.00000000 0.00000000 0.00000000 0.00000000 0.00000000 0.00000000
##
## 19   0.00000000 0.00000000 0.00000000 0.01886792 0.20408163 0.00000000
## 20   0.00000000 0.00000000 0.00000000 0.00000000 0.07142857 0.00000000
## 21   0.00000000 0.00000000 0.02564103 0.00000000 0.21428571 0.00000000
## 22   0.00000000 0.00000000 0.00000000 0.00000000 0.00000000 0.00000000
## 23   0.00000000 0.00000000 0.00000000 0.00000000 0.02702703 0.00000000
## 24   0.00000000 0.00000000 0.00000000 0.00000000 0.19607843 0.00000000
## 25   0.02222222 0.00000000 0.00000000 0.02631579 0.04255319 0.00000000
## 26   0.00000000 0.00000000 0.03030303 0.00000000 0.13513514 0.00000000
## 27   0.00000000 0.00000000 0.00000000 0.07894737 0.00000000 0.00000000
## 28   0.00000000 0.00000000 0.02941176 0.00000000 0.06382979 0.00000000
## 29   0.00000000 0.00000000 0.00000000 0.00000000 0.08510638 0.00000000
## 30   0.00000000 0.00000000 0.00000000 0.00000000 0.06521739 0.00000000
## 31   0.00000000 0.00000000 0.00000000 0.00000000 0.08108108 0.00000000
## 32   0.00000000 0.00000000 0.00000000 0.00000000 0.04651163 0.00000000
## 33   0.03846154 0.00000000 0.00000000 0.02564103 0.00000000 0.00000000
## 34   0.01694915 0.00000000 0.00000000 0.00000000 0.00000000 0.00000000
## 35   0.00000000 0.00000000 0.00000000 0.00000000 0.33333333 0.00000000
## 36   0.00000000 0.00000000 0.00000000 0.00000000 0.13043478 0.00000000
## 37   0.00000000 0.00000000 0.00000000 0.00000000 0.20833333 0.00000000
## 38   0.00000000 0.02857143 0.00000000 0.00000000 0.00000000 0.00000000
## 39   0.00000000 0.00000000 0.00000000 0.00000000 0.10256410 0.00000000
## 40   0.00000000 0.00000000 0.00000000 0.00000000 0.13636364 0.00000000
## 41   0.05405405 0.00000000 0.05405405 0.00000000 0.00000000 0.00000000
##            1843       1844       1845       1846       1847       1848
## 1    0.05263158 0.00000000 0.00000000 0.00000000 0.00000000 0.00000000
## 2    0.00000000 0.00000000 0.00000000 0.00000000 0.00000000 0.00000000
## 3    0.00000000 0.00000000 0.00000000 0.00000000 0.00000000 0.00000000
## 4    0.00000000 0.02941176 0.00000000 0.00000000 0.00000000 0.00000000
## 5    0.00000000 0.00000000 0.00000000 0.00000000 0.00000000 0.00000000
## 6    0.00000000 0.00000000 0.00000000 0.00000000 0.00000000 0.00000000
## 7    0.00000000 0.00000000 0.22222222 0.00000000 0.00000000 0.00000000
## 8    0.00000000 0.00000000 0.00000000 0.00000000 0.00000000 0.02631579
## 9    0.09302326 0.00000000 0.00000000 0.00000000 0.00000000 0.00000000
## 10   0.02272727 0.00000000 0.00000000 0.00000000 0.00000000 0.00000000
## 11   0.08510638 0.00000000 0.00000000 0.00000000 0.00000000 0.00000000
## 12   0.06122449 0.00000000 0.00000000 0.00000000 0.00000000 0.00000000
## 13   0.00000000 0.00000000 0.00000000 0.00000000 0.00000000 0.00000000
## 14   0.05263158 0.00000000 0.00000000 0.00000000 0.00000000 0.00000000
## 15   0.00000000 0.00000000 0.00000000 0.00000000 0.00000000 0.00000000
## 16   0.04081633 0.00000000 0.00000000 0.00000000 0.00000000 0.00000000
## 17   0.06666667 0.00000000 0.00000000 0.00000000 0.00000000 0.00000000
## 18   0.00000000 0.00000000 0.00000000 0.00000000 0.00000000 0.00000000
## 19   0.14035088 0.00000000 0.00000000 0.00000000 0.00000000 0.00000000
## 20   0.04081633 0.00000000 0.00000000 0.00000000 0.00000000 0.00000000
## 21   0.04651163 0.00000000 0.00000000 0.00000000 0.00000000 0.00000000
## 22   0.00000000 0.06250000 0.00000000 0.00000000 0.00000000 0.00000000
##
```

```
## 16   0.00000000 0.05882353 0.00000000 0.00000000 0.00000000 0.00000000
## 17   0.00000000 0.09090909 0.13793103 0.00000000 0.00000000 0.00000000
## 18   0.00000000 0.00000000 0.00000000 0.00000000 0.00000000 0.00000000
## 19   0.00000000 0.15909091 0.07894737 0.00000000 0.00000000 0.00000000
## 20   0.00000000 0.02000000 0.00000000 0.00000000 0.00000000 0.00000000
## 21   0.00000000 0.19512195 0.07142857 0.00000000 0.00000000 0.00000000
## 22   0.08571429 0.00000000 0.00000000 0.02777778 0.05555556 0.00000000
## 23   0.00000000 0.00000000 0.00000000 0.00000000 0.00000000 0.00000000
## 24   0.00000000 0.11627907 0.05000000 0.00000000 0.00000000 0.00000000
## 25   0.00000000 0.00000000 0.02631579 0.00000000 0.00000000 0.00000000
## 26   0.00000000 0.05555556 0.00000000 0.00000000 0.00000000 0.00000000
## 27   0.00000000 0.00000000 0.00000000 0.00000000 0.00000000 0.00000000
## 28   0.00000000 0.02173913 0.02941176 0.00000000 0.00000000 0.00000000
## 29   0.00000000 0.26190476 0.08333333 0.00000000 0.00000000 0.00000000
## 30   0.00000000 0.23404255 0.00000000 0.00000000 0.00000000 0.00000000
## 31   0.00000000 0.13953488 0.02380952 0.00000000 0.00000000 0.00000000
## 32   0.00000000 0.17500000 0.02702703 0.00000000 0.00000000 0.00000000
## 33   0.02631579 0.00000000 0.00000000 0.00000000 0.00000000 0.00000000
## 34   0.00000000 0.00000000 0.00000000 0.00000000 0.00000000 0.04651163
## 35   0.00000000 0.18421053 0.12121212 0.00000000 0.00000000 0.00000000
## 36   0.00000000 0.10526316 0.00000000 0.00000000 0.00000000 0.00000000
## 37   0.00000000 0.13043478 0.08108108 0.00000000 0.00000000 0.00000000
## 38   0.00000000 0.00000000 0.00000000 0.00000000 0.02439024 0.00000000
## 39   0.00000000 0.04545455 0.02564103 0.00000000 0.00000000 0.00000000
## 40   0.00000000 0.17777778 0.02777778 0.00000000 0.00000000 0.00000000
## 41   0.00000000 0.00000000 0.11428571 0.00000000 0.00000000 0.00000000
##            2383       2384       2385       2386       2387       2388
## 1    0.00000000 0.00000000 0.00000000 0.00000000 0.00000000 0.00000000
## 2    0.00000000 0.00000000 0.02857143 0.02857143 0.00000000 0.00000000
## 3    0.00000000 0.00000000 0.07142857 0.02439024 0.00000000 0.00000000
## 4    0.00000000 0.03125000 0.00000000 0.00000000 0.00000000 0.00000000
## 5    0.00000000 0.00000000 0.07142857 0.00000000 0.02702703 0.00000000
## 6    0.17073171 0.00000000 0.00000000 0.00000000 0.10000000 0.00000000
## 7    0.00000000 0.00000000 0.00000000 0.00000000 0.00000000 0.00000000
## 8    0.02500000 0.00000000 0.00000000 0.00000000 0.00000000 0.06896552
## 9    0.00000000 0.00000000 0.00000000 0.02439024 0.00000000 0.00000000
## 10   0.00000000 0.00000000 0.00000000 0.00000000 0.00000000 0.00000000
## 11   0.00000000 0.00000000 0.00000000 0.02083333 0.00000000 0.00000000
## 12   0.00000000 0.00000000 0.00000000 0.00000000 0.02857143 0.00000000
## 13   0.00000000 0.00000000 0.00000000 0.00000000 0.00000000 0.00000000
## 14   0.00000000 0.00000000 0.00000000 0.00000000 0.00000000 0.00000000
## 15   0.00000000 0.00000000 0.00000000 0.00000000 0.03225806 0.00000000
## 16   0.00000000 0.00000000 0.00000000 0.00000000 0.00000000 0.00000000
## 17   0.00000000 0.00000000 0.02941176 0.07317073 0.00000000 0.00000000
## 18   0.00000000 0.00000000 0.00000000 0.02777778 0.00000000 0.00000000
## 19   0.00000000 0.00000000 0.04545455 0.02222222 0.00000000 0.00000000
## 20   0.00000000 0.00000000 0.00000000 0.00000000 0.00000000 0.00000000
## 21   0.00000000 0.00000000 0.02500000 0.00000000 0.00000000 0.00000000
## 22   0.03333333 0.00000000 0.00000000 0.00000000 0.00000000 0.00000000
## 23   0.00000000 0.00000000 0.00000000 0.00000000 0.00000000 0.00000000
## 24   0.00000000 0.00000000 0.02173913 0.00000000 0.00000000 0.00000000
## 25   0.00000000 0.00000000 0.00000000 0.17500000 0.00000000 0.00000000
## 26   0.00000000 0.00000000 0.00000000 0.00000000 0.00000000 0.00000000
## 27   0.00000000 0.00000000 0.00000000 0.00000000 0.00000000 0.00000000
## 28   0.00000000 0.00000000 0.02941176 0.00000000 0.00000000 0.00000000
## 29   0.00000000 0.00000000 0.02325581 0.00000000 0.00000000 0.00000000
## 30   0.00000000 0.00000000 0.05000000 0.00000000 0.04878049 0.00000000
## 31   0.00000000 0.00000000 0.00000000 0.00000000 0.03333333 0.00000000
## 32   0.00000000 0.00000000 0.00000000 0.02631579 0.00000000 0.00000000
## 33   0.00000000 0.00000000 0.00000000 0.00000000 0.00000000 0.00000000
## 34   0.00000000 0.00000000 0.00000000 0.00000000 0.00000000 0.00000000
## 35   0.00000000 0.00000000 0.02564103 0.00000000 0.00000000 0.00000000
## 36   0.00000000 0.00000000 0.00000000 0.00000000 0.00000000 0.00000000
## 37   0.00000000 0.00000000 0.11363636 0.02439024 0.00000000 0.00000000
## 38   0.00000000 0.00000000 0.02857143 0.00000000 0.00000000 0.00000000
## 39   0.00000000 0.00000000 0.00000000 0.00000000 0.00000000 0.00000000
## 40   0.00000000 0.00000000 0.00000000 0.00000000 0.00000000 0.00000000
## 41   0.00000000 0.00000000 0.00000000 0.02325581 0.00000000 0.00000000
##            2389
## 1    0.00000000
## 2    0.05714286
## 3    0.10416667
## 4    0.00000000
## 5    0.02222222
## 6    0.00000000
## 7    0.00000000
## 8    0.00000000
## 9    0.02127660
## 10   0.02777778
## 11   0.04347826
## 12   0.05128205
## 13   0.04347826
## 14   0.04545455
## 15   0.04255319
## 16   0.15217391
## 17   0.00000000
## 18   0.00000000
## 19   0.04545455
## 20   0.00000000
## 21   0.04878049
## 22   0.00000000
## 23   0.00000000
## 24   0.02083333
## 25   0.00000000
## 26   0.00000000
## 27   0.00000000
## 28   0.02777778
## 29   0.02083333
## 30   0.02127660
## 31   0.02631579
## 32   0.06451613
## 33   0.00000000
## 34   0.00000000
## 35   0.02040816
## 36   0.05263158
## 37   0.02127660
## 38   0.00000000
## 39   0.06521739
## 40   0.02325581
## 41   0.02564103
##  [ reached getOption("max.print") -- omitted 2348 rows ]


#We have implemented random forest with the important variable based on the
above analysis

rf3 <- randomForest(Retained.in.2012. ~ From.Grade + To.Grade +
                      Is.Non.Annual.+ FRP.Active+ CRM.Segment+ MDR.High.Grade
+ Total.School.Enrollment + SPR.New.Existing + FPP + Total.Pax +
  SingleGradeTripFlag + FPP.to.PAX, data = rm2,
                   mtry = sqrt(ncol(rm1)-1), ntree = 300,
                   proximity = T, importance = T)

print(rf3)

##
## Call:
##  randomForest(formula = Retained.in.2012. ~ From.Grade + To.Grade +      I
s.Non.Annual. + FRP.Active + CRM.Segment + MDR.High.Grade +      Total.School
.Enrollment + SPR.New.Existing + FPP + Total.Pax +      SingleGradeTripFlag +
FPP.to.PAX, data = rm2, mtry = sqrt(ncol(rm1) -      1), ntree = 300, proximi
ty = T, importance = T)
##                Type of random forest: classification
##                      Number of trees: 300
## No. of variables tried at each split: 7


Below are the evaluation matrices that show the increased error rate

##
##         OOB estimate of  error rate: 20.85%
## Confusion matrix:
##              Not Retained Retained class.error
## Not Retained          667      271   0.2889126
## Retained              227     1224   0.1564438

rf4 <- randomForest(Retained.in.2012. ~ Is.Non.Annual.+SPR.New.Existing + Sin
gleGradeTripFlag, data = rm2,
                   mtry = sqrt(ncol(rm2)-1), ntree = 300,
                   proximity = T, importance = T)
```

```
print(rf4)
```

```
##
## Call:
##  randomForest(formula = Retained.in.2012. ~ Is.Non.Annual. + SPR.New.Exist
ing +      SingleGradeTripFlag, data = rm2, mtry = sqrt(ncol(rm2) -      1),
ntree = 300, proximity = T, importance = T)
##                Type of random forest: classification
##                      Number of trees: 300
## No. of variables tried at each split: 3
```

Based on our recommendations, we see that OOB error rate for RF is this model is 19.84%.

```
##         OOB estimate of  error rate: 19.84%
## Confusion matrix:
##              Not Retained Retained class.error
## Not Retained          629      309   0.3294243
## Retained              165     1286   0.1137147
```

```
head(rf$predicted)
```

```
##            1            2            3            4            5
6
##     Retained     Retained     Retained Not Retained     Retained Not Retai
ned
## Levels: Not Retained Retained
```

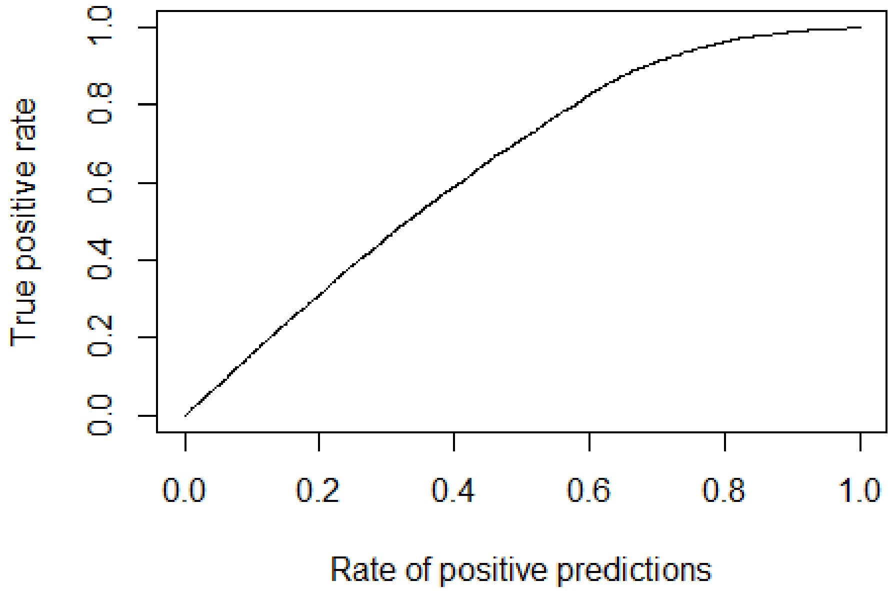
```
head(rf$votes)
```

```
##   Not Retained  Retained
## 1    0.2244898 0.7755102
## 2    0.3750000 0.6250000
## 3    0.2689076 0.7310924
## 4    0.6132075 0.3867925
## 5    0.2601626 0.7398374
## 6    0.8301887 0.1698113
```

```
ind <- sample(2, nrow(rm1), replace = T, prob = c(0.7, 0.3))
trainrf <- rm2[ind == 1, ]
Validation <- rm2[ind == 2, ]
pr.err <- c()
for(mt in seq(1,ncol(trainrf)))
   {
library(randomForest)
rf1 <- randomForest(Retained.in.2012.~.,data = trainrf, ntree = 100,
mtry = ifelse(mt == ncol(trainrf),
mt-1, mt))
predicted <- predict(rf1, newdata = Validation, type = "class")
pr.err <- c(pr.err,mean(Validation$Retained.in.2012. != predicted))
}

bestmtry <- which.min(pr.err)

bestmtry

## [1] 26



# Plotting confusion matrix

rfMAT <- table(rf$predicted, rm2$Retained.in.2012., dnn = c("Predicted", "Actual"))


## Calculating Accuracy , Precision, Recall and Fscore

rfEVALMAT <- EvaluationMatrix(rfMAT[2,2], rfMAT[1,2],rfMAT[2,1],
rfMAT[1,1])
rfEVALMAT

##        Accuracy Precision    Recall    Fscore
## [1,] 0.8024278 0.8559614 0.8252492 0.8403248

# plotting confusion Matrix

confusionMatrix(rf$predicted, rm2$Retained.in.2012., positive = "Not Retained")

## Confusion Matrix and Statistics
##
##                Reference
## Prediction     Not Retained Retained
##   Not Retained          675      209
##   Retained              263     1242
##
##                Accuracy : 0.8024
##                  95% CI : (0.7859, 0.8182)
##     No Information Rate : 0.6074
##     P-Value [Acc > NIR] : < 2e-16
##
##                   Kappa : 0.5815
##
##  Mcnemar's Test P-Value : 0.01471
##
##             Sensitivity : 0.7196
##             Specificity : 0.8560
##          Pos Pred Value : 0.7636
##          Neg Pred Value : 0.8252
##              Prevalence : 0.3926
##          Detection Rate : 0.2825
##    Detection Prevalence : 0.3700
##       Balanced Accuracy : 0.7878
##
##        'Positive' Class : Not Retained
##

score <- rf$votes[, 2]

pred <- prediction(score, rm2$Retained.in.2012.)

### plotting Gain chart
```

Plotting Gain chart

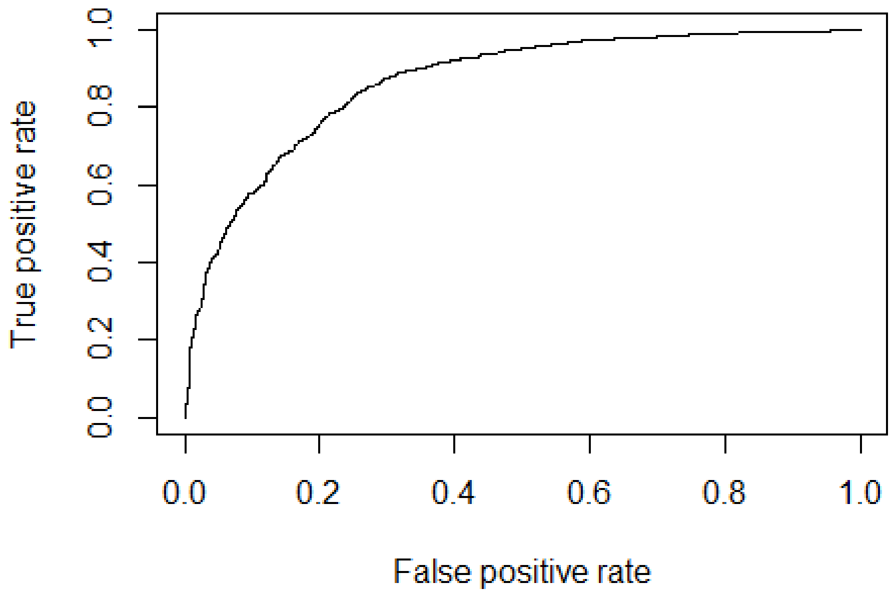
```
perf <- performance(pred, "tpr", "rpp")

plot(perf)
```



#plotting ROC curve

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

```
## A prediction instance
##   with 2389 data points

plot(perf)
```



Finding area under the curve (AUC)

```
auc <- unlist(slot(performance(pred, "auc"), "y.values"))
auc

## [1] 0.8647852
```

- #For the case, we choose “Recall” as the evaluation matrix.

Even though the cost of loss is not defined :

- Because of the loss to miss out on a school that would have actually retained and was not approached because it was falsely marked as not retained is MORE
- as compared to missing out on a school that was actually not retained and was still approached.