Student Retention

Sandhya Suryanarayana

Student Dropout Prediction

Introduction

Student retention and graduation are paramount to the mission and sustainability of institutions. Here we are using predictive analytics to identify students at risk of dropping out, and using that knowledge to take steps to improve retention and graduation rates.

To improve the efficiency/accuracy of prediction, 4 different models are tried with different features and different sampling technique. Even though the Bagging classification tree gave the highest prediction accuracy, the Simple decision tree being the simplest model won the contest with just 2% lower accuracy.

Data Wrangling

The Data sets provided had 5 set of folders with different number of Train and Test files. 1) Student Progress Data: Contains student's academic progress over time from Fall 2011 to Summer 2017. These data is collected every term. 2)Student Static Data: Contains static data like Demographics, age, Gender etc, once for each cohert . 3) Student Financial Aid Data: This was a single file with all finance related information from Fall 2011 to Summer 2017.This provided information about Scholarships, work/Study funds, Loans etc. 4) Dropout Train Label: This CSV file contained StudentID's and if that particular student has dropped out or not. This is used to train the models. 5) TestIDs : This file has StudentID's for which we are predicting the Dropouts.

```
library(Hmisc) #Datacleaning
library(caret)
library(data.table)
library(dplyr)
library(ggplot2)
library(ggthemes)
library(corrplot)
library(stringr)
library(leaps)
library(e1071)
library(rattle)
library(rpart)
library(rpart.plot)
library(RColorBrewer)
```

Data Merging

The Student Progress Data contains duplicate ID's as they are collected every term. First extract all the information for the latest Academic year as well as latest term to get unique ID's using MySql. The Column names in Financial Aid file have been changed for better understanding.

Import all the files in R and change the data types of each column to the required format and merge Progress, Static and Fiancial data using Primary key: StudentID.

```
Financial_Aid <- read.csv('Financial_Aid.csv',na.strings = "")
Progress <- read.csv('Progress.csv',na.strings = "")
static <- read.csv('Static.csv',na.strings = "")

names(Financial_Aid)[1] <- "StudentID"
names(Financial_Aid)[3] <-"cohortterm"
names(Financial_Aid)[4] <- "MaritalStatus"
names(Financial_Aid)[5] <- "AdjustedGrossIncome"
names(Financial_Aid)[6] <- "ParentAdjustedGrossIncome"
names(Financial_Aid)[7] <- "FatherHighestGradeLevel"
names(Financial_Aid)[8] <- "MotherHighestGradeLevel"
names(Financial_Aid)[10] <- "Loan2012"
names(Financial_Aid)[11] <- "Scholarship2012"
names(Financial_Aid)[12] <- "WorkStudy2012"
names(Financial_Aid)[13] <- "Grant2012"
names(Financial_Aid)[14] <- "Loan2013"
names(Financial_Aid)[15] <- "Scholarship2013"
names(Financial_Aid)[16] <- "WorkStudy2013"
names(Financial_Aid)[17] <- "Grant2013"
names(Financial_Aid)[18] <- "Loan2014"
names(Financial_Aid)[19] <- "Scholarship2014"
names(Financial_Aid)[20] <- "WorkStudy2014"
names(Financial_Aid)[21] <- "Grant2014"
names(Financial_Aid)[22] <- "Loan2015"
names(Financial_Aid)[23] <- "Scholarship2015"
names(Financial_Aid)[24] <- "WorkStudy2015"
names(Financial_Aid)[25] <- "Grant2015"
names(Financial_Aid)[26] <- "Loan2016"
names(Financial_Aid)[27] <- "Scholarship2016"
names(Financial_Aid)[28] <- "WorkStudy2016"
names(Financial_Aid)[29] <- "Grant2016"
names(Financial_Aid)[30] <- "Loan2017"
names(Financial_Aid)[31] <- "Scholarship2017"
names(Financial_Aid)[32] <- "WorkStudy2017"
names(Financial_Aid)[33] <- "Grant2017"

##----Data Manipulation----
Merge1 <- merge(Progress,static,by='StudentID')
Merge2 <- merge(Merge1,Financial_Aid,by='StudentID')
```

```
Merge2$StudentID<- as.numeric(Merge2$StudentID)

Merge2$Term <- as.factor(Merge2$Term)

Merge2$CompleteDevMath <- as.factor(Merge2$CompleteDevMath)

Merge2$CompleteDevEnglish <- as.factor(Merge2$CompleteDevEnglish)

Merge2$Complete1 <- as.factor(Merge2$Complete1)

Merge2$Gender <- as.factor(Merge2$Gender)

Merge2$HsDip <- as.factor(Merge2$HsDip)

Merge2$Enrollmentstatus<- as.factor(Merge2$Enrollmentstatus)

Merge2$HighDeg <- as.factor(Merge2$HighDeg)

Merge2$MathPlacement <- as.factor(Merge2$MathPlacement)

Merge2$EngPlacement <- as.factor(Merge2$EngPlacement)

Merge2$GatewayMathStatus <-as.factor(Merge2$GatewayMathStatus)

Merge2$GatewayEnglishStatus<-as.factor(Merge2$GatewayEnglishStatus)

Merge2$MaritalStatus <- as.factor(Merge2$MaritalStatus)
```

Data Cleaning

For data cleaning each of the columns are studied to understand its importance after which we can decide if it should be omitted or replaced with 0/Unknown or some other values.

To get the data in the required format the following data cleaning was done on the merged data

1) BirthYear: the only Null value id replaced with the average BirthYear.
2) A new Race column was created to specify a categorical value for each the race and -1 if the race is not present or if it is Unknown.
3) A MaritalStatusNew column was created which replaces all the Null value with the most occuring class(i.e Single)
4) The NULL values in FatherHighestGradeLevel and MotherHighestGradLevel is replaced with 'Unknown'.
5) All the NULL values related to Financial columns like Grant, Scholarships are imputed with 0 assuming if the information is not provided the student might not have gotten the scholarship/Funds/Loan.

6) A new column Funds is created for each year to repesent if the particular student has been involved in any kind of Loan or fund activity.(1 if YES: 0 if NO)

```
Merge2$BirthYear <- as.numeric(Hmisc::impute(year(as.Date(as.character(Merge2
$BirthYear),"%Y")),1989))

Merge2$HSDipYr<-ifelse(Merge2$HSDipYr != -1,year(as.Date(as.character(Merge2$
HSDipYr),"%Y")),-1)

Merge2$Race <- ifelse(Merge2$Hispanic==1,1,
                      ifelse(Merge2$AmericanIndian==1,2,
                      ifelse(Merge2$Asian==1,3,
                      ifelse(Merge2$Black==1,4,
                      ifelse(Merge2$NativeHawaiian==1,5,
                      ifelse(Merge2$White==1,6,
                      ifelse(Merge2$TwoOrMoreRace==1,7
                             ))))))
Merge2$Race <- replace(Merge2$Race,is.na(Merge2$Race),-1)

Merge2$Race <- as.factor(Merge2$Race)

Merge2$State <- replace(Merge2$State,is.na(Merge2$State),'NJ')

Merge2$MaritalStatus <- as.factor(Merge2$MaritalStatus)

Merge2$MaritalStatusNew <- Hmisc::impute(Merge2$MaritalStatus,'Single')

Merge2$MaritalStatusNew <- as.factor(Merge2$MaritalStatusNew)

Merge2$FatherHighestGradeLevel <- Hmisc::impute(Merge2$FatherHighestGradeLeve
l,'Unknown')
Merge2$MotherHighestGradeLevel <- Hmisc::impute(Merge2$MotherHighestGradeLeve
l,'Unknown')

Merge2$AdjustedGrossIncome <- Hmisc::impute(Merge2$AdjustedGrossIncome,0)
Merge2$ParentAdjustedGrossIncome <- Hmisc::impute(Merge2$ParentAdjustedGrossI
ncome,0)

Merge2$Loan2012 <- Hmisc::impute(Merge2$Loan2012,0)
Merge2$Scholarship2012 <- Hmisc::impute(Merge2$Scholarship2012,0)
Merge2$WorkStudy2012 <- Hmisc::impute(Merge2$WorkStudy2012,0)
Merge2$Grant2012 <- Hmisc::impute(Merge2$Grant2012,0)

Merge2$Loan2013 <- Hmisc::impute(Merge2$Loan2013,0)
Merge2$Scholarship2013 <- Hmisc::impute(Merge2$Scholarship2013,0)
Merge2$WorkStudy2013 <- Hmisc::impute(Merge2$WorkStudy2013,0)
Merge2$Grant2013 <- Hmisc::impute(Merge2$Grant2013,0)
```

```
Merge2$Loan2014 <- Hmisc::impute(Merge2$Loan2014,0)
Merge2$Scholarship2014 <- Hmisc::impute(Merge2$Scholarship2014,0)
Merge2$WorkStudy2014 <- Hmisc::impute(Merge2$WorkStudy2014,0)
Merge2$Grant2014 <- Hmisc::impute(Merge2$Grant2014,0)



Merge2$Loan2015 <- Hmisc::impute(Merge2$Loan2015,0)
Merge2$Scholarship2015 <- Hmisc::impute(Merge2$Scholarship2015,0)
Merge2$WorkStudy2015 <- Hmisc::impute(Merge2$WorkStudy2015,0)
Merge2$Grant2015 <- Hmisc::impute(Merge2$Grant2015,0)

Merge2$Loan2016 <- Hmisc::impute(Merge2$Loan2016,0)
Merge2$Scholarship2016 <- Hmisc::impute(Merge2$Scholarship2016,0)
Merge2$WorkStudy2016 <- Hmisc::impute(Merge2$WorkStudy2016,0)
Merge2$Grant2016 <- Hmisc::impute(Merge2$Grant2016,0)

Merge2$Loan2017 <- Hmisc::impute(Merge2$Loan2017,0)
Merge2$Scholarship2017 <- Hmisc::impute(Merge2$Scholarship2017,0)
Merge2$WorkStudy2017 <- Hmisc::impute(Merge2$WorkStudy2017,0)
Merge2$Grant2017 <- Hmisc::impute(Merge2$Grant2017,0)

Merge2$Funds12 <- ifelse(Merge2$Scholarship2012 > 0 | Merge2$WorkStudy2012 >
0 |
                                Merge2$Grant2012 >0 | Merge2$Loan2012>0,1,0 )
Merge2$Funds13 <- ifelse(Merge2$Scholarship2013 > 0 | Merge2$WorkStudy2013 >
0 |
                                Merge2$Grant2013 >0|Merge2$Loan2013 > 0,1,0 )

Merge2$Funds14 <- ifelse(Merge2$Scholarship2014 >0 | Merge2$WorkStudy2014 > 0
|
                                Merge2$Grant2014 >0 | Merge2$Loan2014>0,1,0 )
Merge2$Funds15 <- ifelse(Merge2$Scholarship2015 >0 | Merge2$WorkStudy2015 > 0
|
                                Merge2$Grant2015 >0 | Merge2$Loan2015>0,1,0 )
Merge2$Funds16 <- ifelse(Merge2$Scholarship2016 >0 | Merge2$WorkStudy2016 > 0
|
                                Merge2$Grant2016 >0 | Merge2$Loan2016>0,1,0 )
Merge2$Funds17 <- ifelse(Merge2$Scholarship2017 >0 | Merge2$WorkStudy2017 > 0
|
                                Merge2$Grant2017 >0 | Merge2$Loan2017>0,1,0 )



Merge2$Funds12<- as.factor(Merge2$Funds12)
Merge2$Funds13  <- as.factor(Merge2$Funds13)
Merge2$Funds14 <- as.factor(Merge2$Funds14)
Merge2$Funds15 <- as.factor(Merge2$Funds15)
Merge2$Funds16 <- as.factor(Merge2$Funds16)
Merge2$Funds17 <- as.factor(Merge2$Funds17)
```

Columns that has ALL Null or constant value is removed from the data set before merging it with Train and the Test Dataset.

1. *Static Data*

   Campus

   Address2

   FirstGen

   DualSummerEnroll

2. *Progress Data*

   Complete2

   DegreeTypeSought

Train Data

Once the required format of the data is obtained, merge the DropoutTainLabel file with other datasets to get TrainData.

```
TrainLabel <- read.csv('DropoutTrainLabels.csv')
TrainLabel$StudentID <- as.numeric(TrainLabel$StudentID)
TrainLabel$Dropout <- as.factor(TrainLabel$Dropout)
TrainData <- merge(Merge2,TrainLabel,by='StudentID')
```

Test Data

Merge TestIDs file with the merged Progress, Static and Financial dataset using Unique StudentID column to form the final TestData.

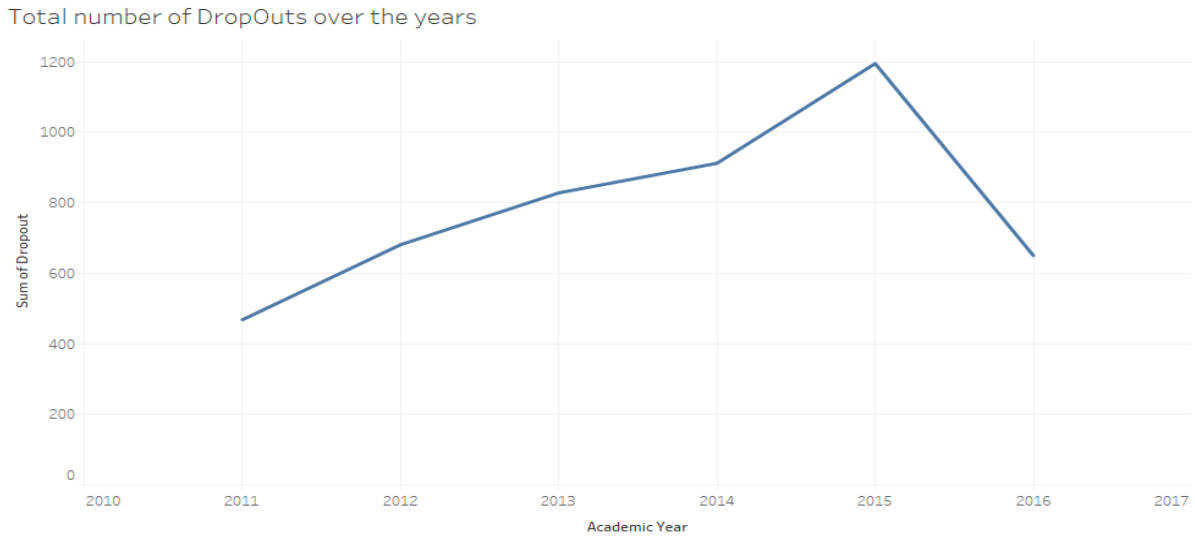
```
TestID <- read.csv('TestIDs.csv')
TestID$StudentID <- as.numeric(TestID$StudentID)
TestData <- merge(Merge2,TestID,by='StudentID')
```

Exploratory Data Analysis

As we know, In Data Science 80% of the time is spent in data preparation. Let's do exploratory data analysis to have a better understanding of the data and helps us investigate the relationship between the variables.

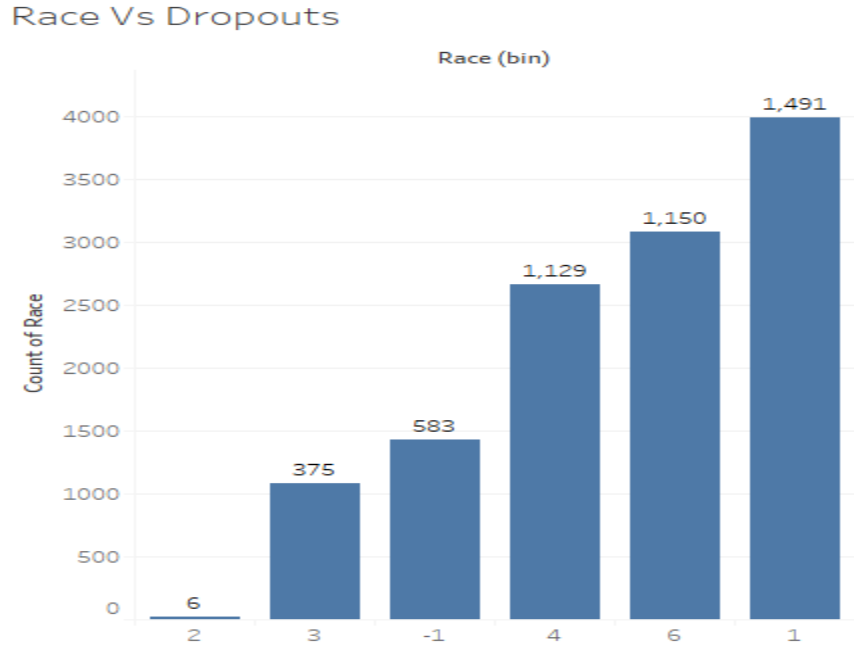
1) Let's understand the total number of dropouts out of entire student population.

We can see in the below graph that the number of students dropping out is increasing every year and reached its highest in 2015. Something happened in 2015-16 academic year that made lot of students to dropout.



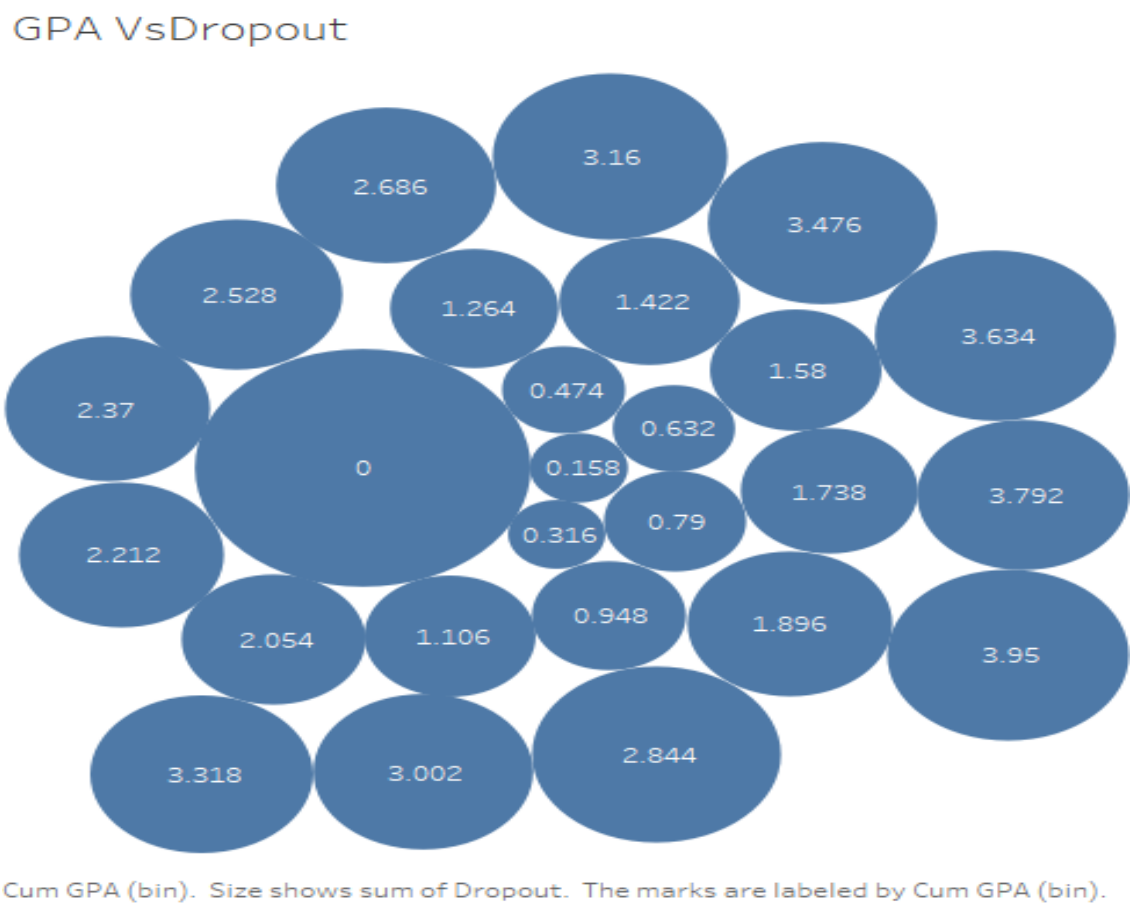
The trend of sum of Dropout for Academic Year.

2) The below graph of Number of DropOuts Vs race shows that the trend of dropouts is higher for Race 1(Hispanic) followed by White race.

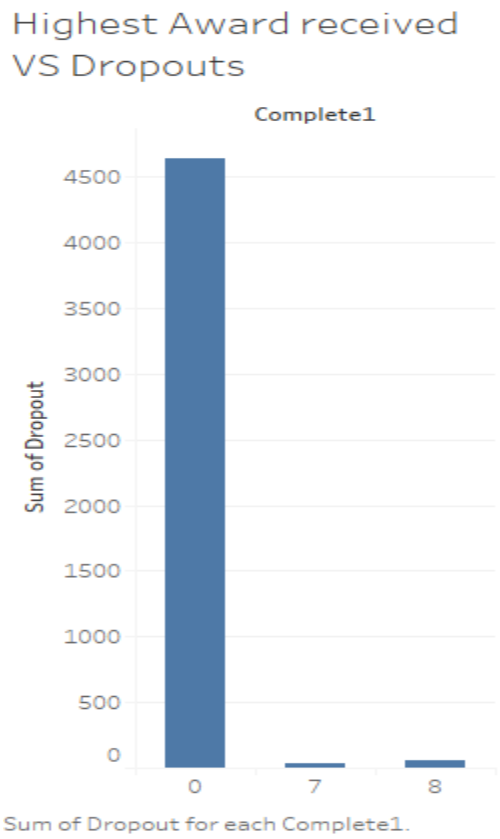


Count of Race for each Race (bin). The marks are labeled by sum of Dropout.

3)Now, lets have a look at how the Cumulative GPA affects the dropouts. As we can see, though the number of Dropouts are highest with GPA = 0. there are also lot of dropouts for students with more than 3.5



4)Here in the below graph we can see that, the students who have received any kind of certificate/graduate awards are the least dropouts. Out of 3500 stdents who have received awards, only 100 students have dropped out. It can be because students who get certified or gets rewarded could be more motivated or be more interested in higher education.



Correlation Matrix

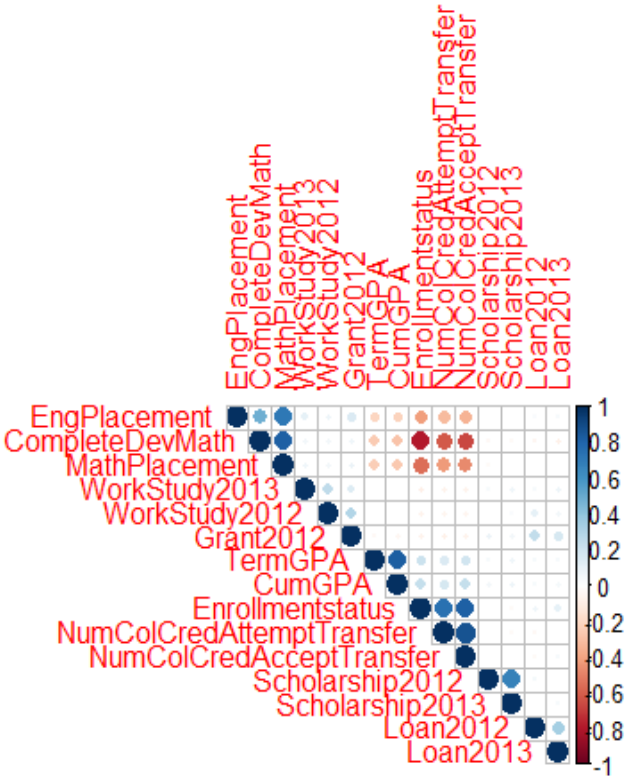
```
Feature_co_2r <- TrainData[c(5,28,29,10,11,24,25,26)]
correlation_2 <- as.matrix(Feature_co_2r,use="pairwise.complete.obs")
cor_mat<-rcorr(correlation_2)
cor_mat$r

corrplot(cor_mat$r, type="upper", order="hclust",

         p.mat = cor_mat$P, sig.level = 0.01, insig = "blank")
```

The Correlation plot is observed and if the correlation between any pair is > 0.75, one of the variables can be removed to reduce linear dependency

These are the few variables that was removed to reduce linear dependency

1. MathPlacement
2. TermGPA
3. NumColCredAcceptTransfer
4. NumColCredAttemptTransfer

Correlation Plot



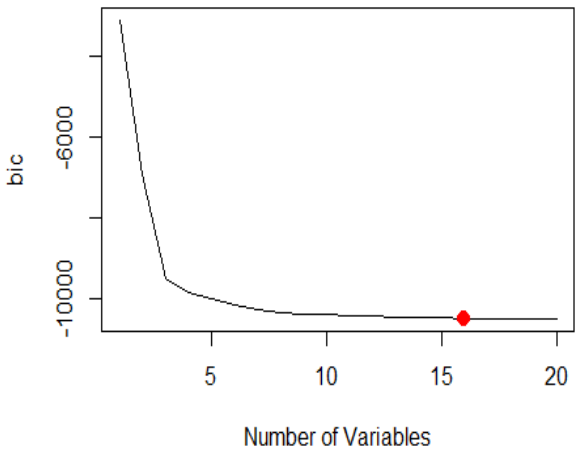
Feature Selection

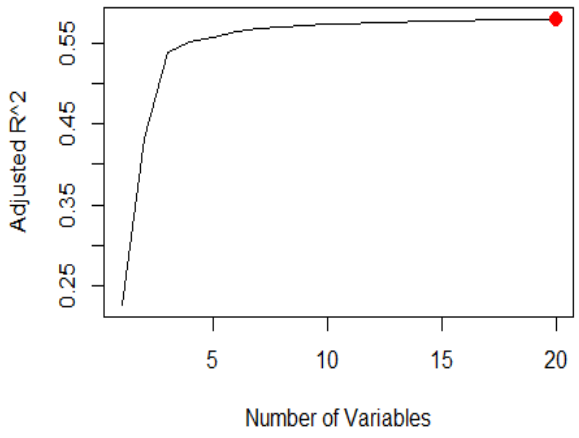
When Best subset selection method is used on all the variables to select the feature variables, the adjusted R^2 gives 20 variables and BIC gives 16 feature variable.

First lets use 16 variable to build the model.

```
Feature_subset <- Feature_1[c("StudentID","Term","CompleteDevMath",
                              "Major1","Complete1","CumGPA","Gender","BirthYear",
                            "HsDip","Enrollmentstatus","FatherHighestGradeLevel",
                                 "MotherHighestGradeLevel",
                                 "Race","Funds12" ,"Funds13","Funds14","Funds15"
,"Funds16","Funds17", "MaritalStatusNew","Dropout")]
bstsubset <- regsubsets(Dropout ~.-StudentID, data=Feature_subset,nvmax=20)
reg.summary <-summary(bstsubset)
```

```
reg.summary$adjr2#20
reg.summary$bic#16
which.max(reg.summary$adjr2)
which.min(reg.summary$bic)

plot(reg.summary$bic,xlab="Number of Variables",ylab="bic",type="l")
points(16,reg.summary$bic[16], col="red",cex=2,pch=20)
coef(bstsubset,16)

plot(reg.summary$adjr2,xlab="Number of Variables",ylab="Adjusted R^2",type="l
")
points(20,reg.summary$adjr2[20], col="red",cex=2,pch=20)
coef(bstsubset,20)
```





Model Building

Model 1: Classification tree

Lets build a simple decision tree with a K fold cross validation with 10 fold with all the features selected by our Best Subset selection method.

```
trctrl <- trainControl(method = "cv", number = 10)

tree_fit <- train(Dropout ~ Term+CompleteDevMath+Major1+Complete1+ CumGPA+Bir
thYear
                  +HSGPAUnwtd+Enrollmentstatus+NumColCredAttemptTransfer+High
Deg
                  +GatewayEnglishStatus+MaritalStatusNew+
                    Funds17+Funds13+Funds15,
                  data = TrainData, method = "rpart",
                  trControl=trctrl)
```

```
tree_fit$finalModel
```

```
## n= 12261
##
## node), split, n, loss, yval, (yprob)
##       * denotes terminal node
##
##  1) root 12261 4734 0 (0.61389772 0.38610228)
##    2) Funds171>=0.5 4643  418 0 (0.90997200 0.09002800) *
##    3) Funds171< 0.5 7618 3302 1 (0.43344710 0.56655290)
##      6) Complete18>=0.5 1874   28 0 (0.98505870 0.01494130) *
##      7) Complete18< 0.5 5744 1456 1 (0.25348189 0.74651811)
##       14) Complete17>=0.5 772    9 0 (0.98834197 0.01165803) *
##       15) Complete17< 0.5 4972  693 1 (0.13938053 0.86061947) *
```

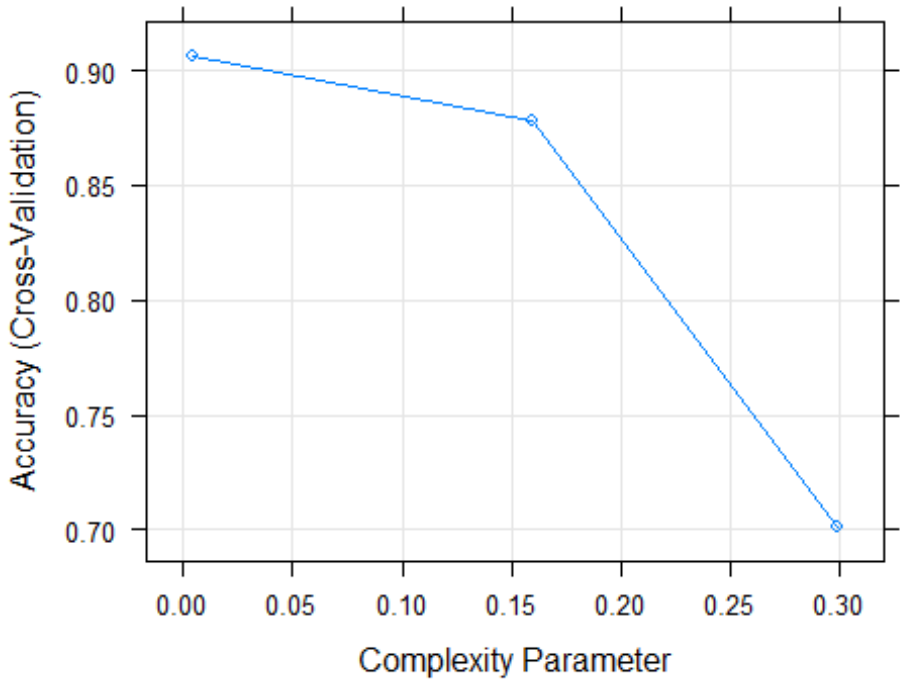
```
tree_fit$bestTune
```

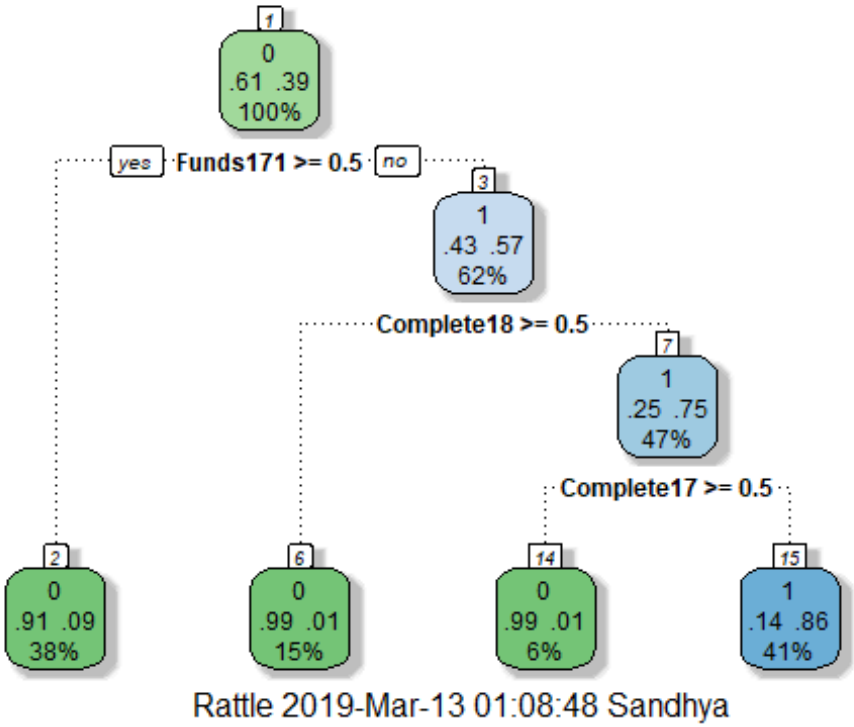
```
##            cp
## 1 0.004013519
```

```
tree_fit$results
```

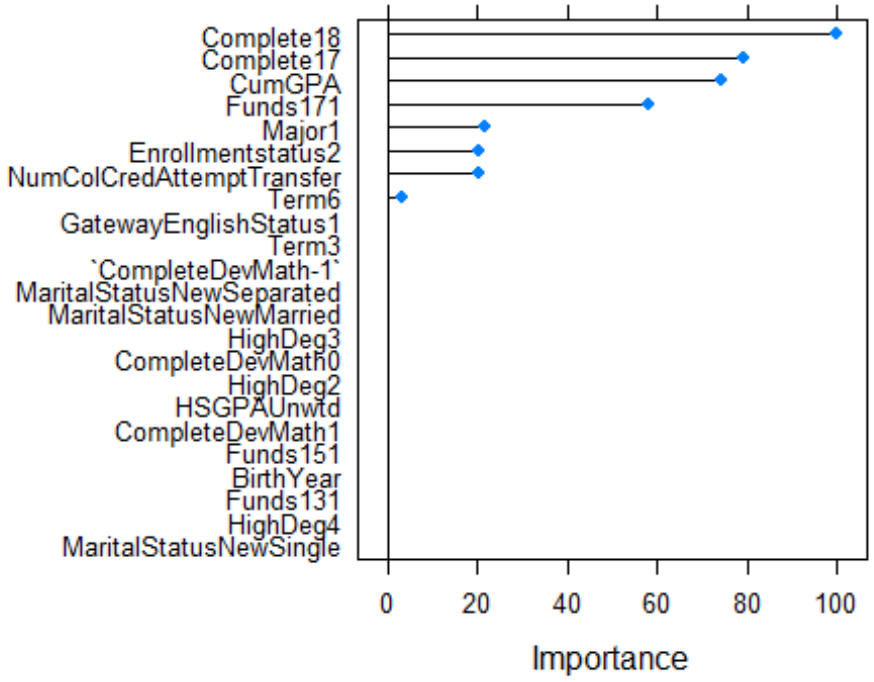
```
##            cp  Accuracy     Kappa  AccuracySD    KappaSD
## 1 0.004013519 0.9066946 0.8050635 0.009558781 0.02015624
## 2 0.159273342 0.8782336 0.7501693 0.037045651 0.07121491
## 3 0.299112801 0.7008472 0.2634777 0.112169210 0.34017414
```

```
plot(tree_fit)
fancyRpartPlot(tree_fit$finalModel)
predict_tree <- predict(tree_fit,TestData)
predict_df <- cbind.data.frame(TestData$StudentID,predict_tree)
#names(predict_df) <- c('StudentID','DropOuts')
treeImp <- varImp(tree_fit, scale = TRUE)
plot(treeImp)
```





Rattle 2019-Mar-13 01:08:48 Sandhya



We can see from the Variable importance plot that for this model 'complete1' plays the most important role for the model followed by CumGPA, Funds17, Major1, EnrollmentStatus and NumColCredAttemptTransfer. Rest of the variable has 0% importance which can be removed to make the model more efficient. The Decision tree result: Accuracy = 90.7% cp = 0.03 Kappa = 80.7%

Model 2: Support Vector Machine: Radial Kernel

A small variation was made to the feature variable was made based on Best Subset collection with 20 variables. Adding Race, Funds12 and Funds16

```
## Support Vector Machines with Radial Basis Function Kernel
##
## 12261 samples
##    12 predictor
##     2 classes: '0', '1'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
## Resampling results across tuning parameters:
##
##   C     Accuracy   Kappa
##   0.25  0.9070216  0.8057544
##   0.50  0.9080004  0.8079337
##   1.00  0.9108542  0.8136518
```

```
##
## Number of Support Vectors : 2854
##
## Objective Function Value : -2321.256
## Training error : 0.085964

##        sigma C
## 3 0.03425687 1

##        sigma    C  Accuracy     Kappa  AccuracySD    KappaSD
## 1 0.03425687 0.25 0.9070216 0.8057544 0.008534611 0.01738189
## 2 0.03425687 0.50 0.9080004 0.8079337 0.008995336 0.01835718
## 3 0.03425687 1.00 0.9108542 0.8136518 0.009129979 0.01890911
```

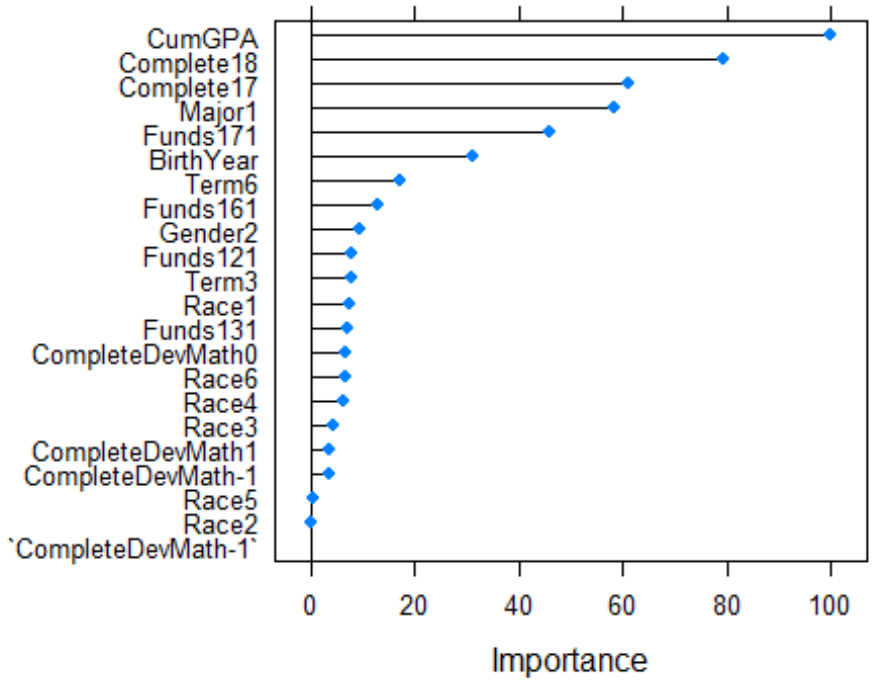
The accuracy here is increased by 1% than Decision tree model.

Accuracy : 91.3% Kappa: 80.55% Sigma: 0.03 Cost : 1

Model 3: Ensemble model(Bagging)

```
## Bagged CART
##
## 12261 samples
##    12 predictor
##     2 classes: '0', '1'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 11034, 11036, 11034, 11036, 11035, 11034, ...
## Resampling results:
##
##   Accuracy  Kappa
##   0.913792  0.8181575

## treebag variable importance
##
##   only 20 most important variables shown (out of 22)
##
##                    Overall
## CumGPA            100.0000
## Complete18         79.2281
## Complete17         60.9038
## Major1             58.4085
## Funds171           45.8494
## BirthYear          31.2933
## Term6              17.1675
## Funds161           12.8500
## Gender2             9.3406
## Funds121            7.9954
## Term3               7.9424
## Race1               7.6251
## Funds131            7.2195
```

```
## CompleteDevMath0    6.7153
## Race6               6.6339
## Race4               6.3129
## Race3               4.5556
## CompleteDevMath1    3.7666
## CompleteDevMath-1   3.4365
## Race5               0.7025
```



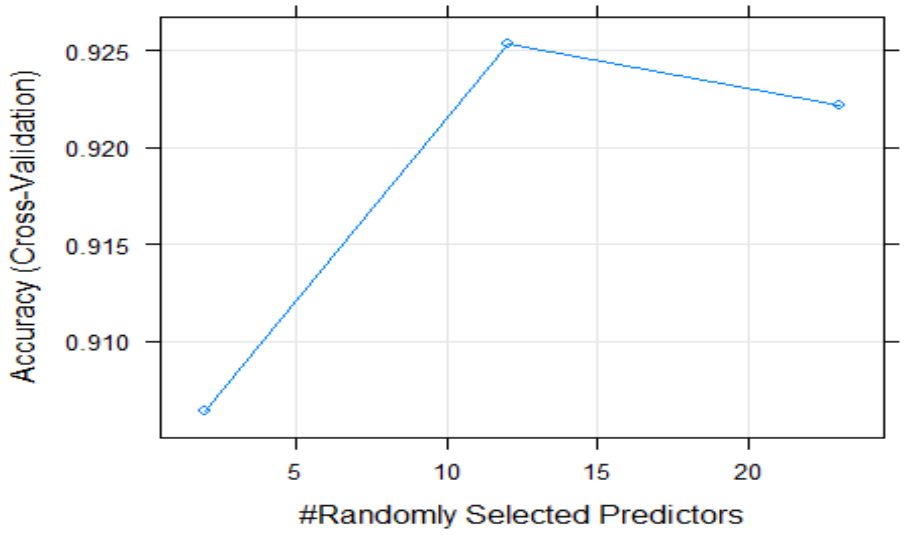
We can see in the variable importance plot that almost all of the variables place an important role while building the model.

For Bagging model we get: Accuracy: 91.7% Kappa : 81.39%

Model 4: Random forest

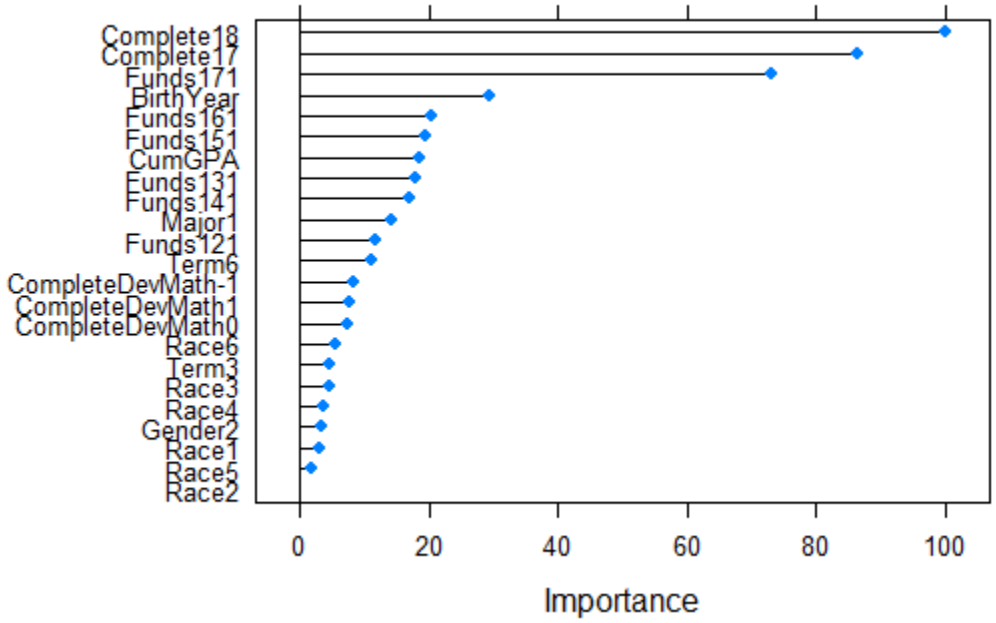
```
## Random Forest
##
## 12261 samples
##    14 predictor
##     2 classes: '0', '1'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 11034, 11035, 11036, 11036, 11035, 11035, ...
## Resampling results across tuning parameters:
##
##   mtry  Accuracy   Kappa
##    2    0.9063680  0.8009332
```

```
##   12    0.9253740  0.8431180
##   23    0.9221926  0.8364503
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was mtry = 12.
##
##         OOB estimate of  error rate: 7.36%
## Confusion matrix:
##      0    1 class.error
## 0 7045  482  0.06403614
## 1  420 4314  0.08871990
```



The complexity parameter tuning plot shows the highest accuracy at 12th random predictor value.(That is at mtry=12)

Variable Importance plot:The Complete1 variable has the most important variable.



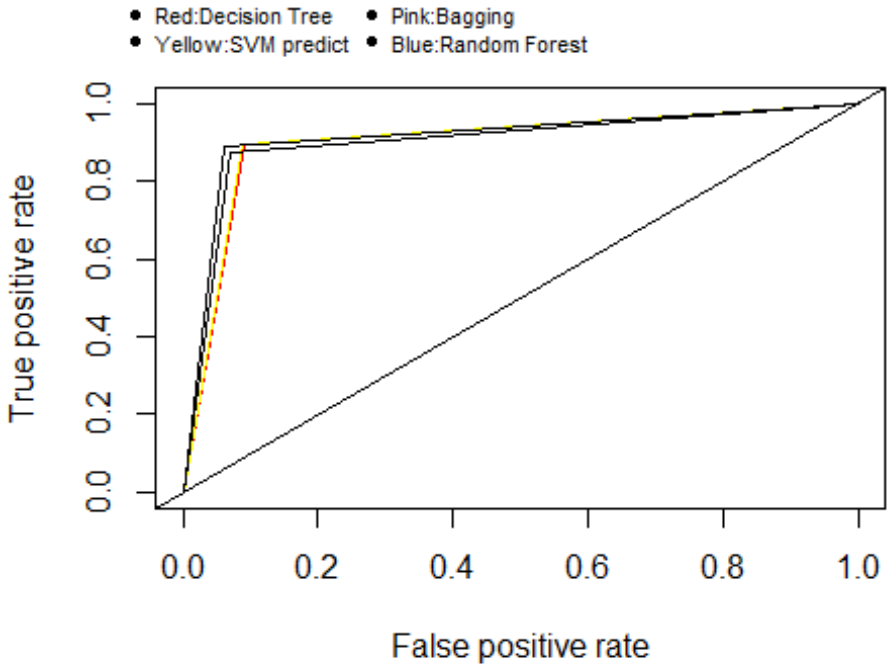
This model gives the best accuracy of 92.65% making with highest accuracy among the 4 models. The number of randomly selected predictors(mtry):12 Out of Bag error estimate: 7.36%

Receiver Operating Characteristic curve(ROC curve):

ROC is a plot of the false positive rate versus the true positive rate for a number of different candidate threshold values between 0.0 and 1.0. A model with perfect skill is represented at a higher point.

```
library(ROCR)
ROCRpred1 <- prediction(as.numeric(predict_tree_Test), as.numeric(test1$Dropo
ut))
ROCRpred2 <- prediction(as.numeric(SVMpredict_Test), as.numeric(test1$Dropout
))
ROCRpred3 <- prediction(as.numeric(BegFitPredict_Test), as.numeric(test1$Drop
out))
ROCRpred4 <- prediction(as.numeric(RfTrainPredict_Test), as.numeric(test1$Dro
pout))
#ROC Curve
ROCRperf1 <- performance(ROCRpred1, 'tpr','fpr')
ROCRperf2 <- performance(ROCRpred2, 'tpr','fpr')
ROCRperf3 <- performance(ROCRpred3, 'tpr','fpr')
```

```
ROCRperf4 <- performance(ROCRpred4, 'tpr','fpr')
plot(ROCRperf1,col = 'Red')
abline(0, 1)
plot(ROCRperf2, add = TRUE, col = 'Yellow')
plot(ROCRperf3, add = TRUE, col='black')
plot(ROCRperf4, add = TRUE, Col='Orange')
legend(x=-0.1,y=1.3,legend=paste(rep(c("Red","Yellow","Orange","Blue")),
                                 rep(c("Decision Tree","SVM predict","Bagging",
"Random Forest"))
                                 ,sep=":"),pch=rep(c(16,18),each=4),bty="n",
                                  ncol=2,cex=0.7,pt.cex=0.7,xpd=TRUE)## Warning
: package 'gplots' was built under R version 3.5.2
```



The Random forest has the highest AUC and hence can be selected as the best model.

Conclusion:

1. Data Wrangling and Data Manipulation plays a vital role in getting higher accuracy which also takes more than 70% of your time.
2. The random forest model gives the best Presiction accuracy of 93.3%.
3. The Area Under the curve for the ROC is also highest for Random forest with AUC=0.8.
4. The simple Decision Tree being the simplest model gives an accuracy of 90%