**Programming Codes:**

#Reading Comcast Data and loading libraries

rm(list=ls())

library(dplyr)

library(ggplot2)

library(lubridate)

library(plyr)

library(tidyverse)

library(caret)

library(rattle)

library(party) # For decision tree

library(rpart) # for Rpart

library(rpart.plot) #for Rpart plot

library(lattice) # Used for Data Visualization

library(randomForest)# FOr Random Forest

library(pROC)

library(e1071) # For SVM

library(naivebayes) # For Naive Bayes

setwd("E:\\Simplilearn\\Data Science with R\\Project\\Education")

ca<- read.csv("College\_Admission.csv")

head(ca)

# Missing Data

sum(is.na(ca))

anyNA(ca)

#Checking outliers

attach(ca)

par(mfrow=c(1,2))

boxplot(ca$gre, main="Boxplot of GRE")

boxplot(ca$gpa, main="Boxplot of GPA")

#Performing Outlier treatment

#Removing outliers for gre

Q\_gre <- quantile(ca$gre, probs = c(0.25, 0.75)) # 25% is 520 and 75% is 660

iqr\_gre <- IQR(ca$gre) # Interquantile range is 140

uper\_gre <- Q\_gre[2]+1.5\*iqr\_gre # uper limit is 870

lower\_gre <- Q\_gre[1]-1.5\*iqr\_gre # lower limit is 310

ca <- subset(ca, ca$gre > lower\_gre & ca$gre < uper\_gre)

#Removing outliers for gpa on already removed dataset

Q\_gpa <- quantile(ca$gpa, probs = c(0.25, 0.75)) # 25% is 3.13 and 75% is 3.67

iqr\_gpa <- IQR(ca$gpa) # Interquantile range is 0.54

uper\_gpa <- Q\_gpa[2]+1.5\*iqr\_gpa # uper limit is 4.48

lower\_gpa <- Q\_gpa[1]-1.5\*iqr\_gpa # lower limit is 2.32

ca <- subset(ca, ca$gpa > lower\_gpa & ca$gpa < uper\_gpa)

par(mfrow=c(1,2))

boxplot(ca$gre, main="Boxplot of GRE")

boxplot(ca$gpa, main="Boxplot of GPA")

#Factoring the categorical data

str(ca)

ca$admit = factor(ca$admit, levels = c("0","1"), labels = c("Accepted","Rejected"))

ca$ses = factor(ca$ses, levels = c("1","2",'3'), labels = c("Low","Medium",'High'))

ca$Gender\_Male = factor(ca$Gender\_Male, levels = c("0","1"), labels = c("Female","Male"))

ca$Race = factor(ca$Race, levels = c("1","2",'3'), labels = c("Hispanic","Asian",'African-American'))

ca$rank <- factor(ca$rank, order = TRUE)

#Categorising GRE Marks to Category

ca = mutate(ca,GRE\_category = ifelse(gre <= 440,"Low",

ifelse(gre<=580,"Medium","High")))

Freq= table(ca$GRE\_category)

Freq

#Checking if normally distributed

summary(ca)

# Density plot

par(mfrow=c(1,2))

d <- density(ca$gpa)

plot(d, main="Kernel Density of GPA")

polygon(d, col="red", border="blue")

d1 <- density(ca$gre)

plot(d1, main="Kernel Density of GRE")

polygon(d1, col="red", border="blue")

hist(ca$gpa, freq = FALSE)

x <- seq(0, 4, length.out=100)

y <- with(ca, dnorm(x, mean(gpa), sd(gpa)))

lines(x, y, col = "red")

hist(ca$gpa, freq = FALSE)

x <- seq(0, 4, length.out=100)

y <- with(ca, dnorm(x, mean(gpa), sd(gpa)))

lines(x, y, col = "red")

#Normalise the data

ca$gpa1 <- scale(ca$gpa)

ca$gre1 <- scale(ca$gre)

hist(ca$gpa1, freq = FALSE)

x <- seq(-3, 3, length.out=100)

y <- with(ca, dnorm(x, mean(gpa1), sd(gpa1)))

lines(x, y, col = "blue")

hist(ca$gre1, freq = FALSE)

x <- seq(-3, 3, length.out=100)

y <- with(ca, dnorm(x, mean(gpa1), sd(gpa1)))

lines(x, y, col = "blue")

#variable reduction techniques to identify significant variables

model <- glm(admit~ ., family = binomial(link = 'logit'), data = ca)

summary(model)

anova(model, test = 'Chisq')

# Logistic regression model with significance independent variable

set.seed(123)

splitIndex <- createDataPartition(ca$admit, p = .70,list = FALSE, times = 1)

train <- ca[ splitIndex,]

test <- ca[-splitIndex,]

model1 <- glm(admit~ gpa1+gre1+rank , data = train,family=binomial(link = "logit"))

summary(model1)

#accuracy of the model and run validation techniques

#Predict on Test through Model

pred = predict(model1,test, type="response")

pred = ifelse(pred>0.5,1,0)

pred = factor(pred, levels = c("0","1"), labels = c("Accepted","Rejected"))

####Validate the model - Confusion Matrix##

act <- test$admit

# Accuracy

table(pred, act)

a=confusionMatrix(pred, act)

a

#Model generation using other ML techniques

#1. Decision tree

model\_dt = rpart(admit~ gpa1+gre1+rank, data = train,method = "class",

control = rpart.control(minsplit = 30,cp = 0.01))

par(mfrow=c(1,1))

fancyRpartPlot(model\_dt)

pred\_dt = predict(model\_dt,test, type="class")

table(pred\_dt, act)

a\_dt=confusionMatrix(pred\_dt,act)

a\_dt

#2. SVM

svmfit =svm(admit~ gpa1+gre1+rank, data = train, kernel="linear",

scale = T)

pred\_svm = predict(svmfit,test, type="response")

table(pred\_svm, act)

a\_svm=confusionMatrix(pred\_svm,act)

a\_svm

#3. Random Forest

fit\_rf = randomForest(admit~ gpa1+gre1+rank, data = train, do.trace=F)

pred\_rf = predict(fit\_rf,test)

table(pred\_rf, act)

a\_rf=confusionMatrix(pred\_rf,act)

a\_rf

#4. Naive Bayes

fit\_nb = naive\_bayes(admit~ gpa1+gre1+rank, data = train)

pred\_nb = predict(fit\_rf,test)

table(pred\_nb, act)

a\_nb=confusionMatrix(pred\_nb,act)

a\_nb

#Categorize the average of grade point into High, Medium,

#and Low (with admission probability percentages) and plot it on a point chart.

df <- cut(test$gpa,breaks = c(2,2.7,3.4,4),labels = c("LOW","MEDIUM","HIGH"))

tail(df)

prob <- predict(model1,test,type = "response")

pl <- ggplot(test,aes(df,prob )) + geom\_point(col="green")

pl + xlab("GRADE POINT AVERAGE") + ylab("Probability Percentage") + scale\_y\_continuous()

-------------------------------------------------------------------The End-------------------------------------------------------------------