```
##Gives the path details.##
getwd()

##Write the path details for data to read.##
setwd("//home//labsuser")

##Data is been uploaded to read
data<-read.csv("College_admission.csv")
data
head(data)
data1<-data
nrow(data)

##Find the missing values. (if any, perform missing value treatment)##
sum(is.na(data))

##Find outliers (if any, then perform outlier treatment)##
#For Gre
IQR1<-IQR(data$gre)
IQR1
quantile(data$gre,na.rm = TRUE)

max1<-660+1.5*IQR1
max1

min1<-520-1.5*IQR1
min1

##The upper limit
print(which(data$gre>max1))

##The lower limit
print(which(data$gre<min1))

##For gpa
IQR2<-IQR(data$gpa)
IQR2
quantile(data$gpa,na.rm = TRUE)

max2<-3.670+1.5*IQR2
max2

min2<-3.130-1.5*IQR2
min2

##The upper limit
print(which(data$gpa>max2))

##The lower limit
print(which(data$gpa<min2))


##Treatment for outliers
data<-data[-c(72,180,290,305,316),]
data
nrow(data)


##Find the structure of the data set and if required, transform the numeric data type to factor
and vice-versa.##
str(data)

drn<-factor(data)
data

as.numeric(drn)
data

##Find whether the data is normally distributed or not. Use the plot to determine the same.##
```

```
library("dplyr")

shapiro.test(data$gre)
shapiro.test(data$gpa)

plot(data)

##Normalize the data if not normally distributed.##

#Data is normally distributed as p value is less then 0.05

##Use variable reduction techniques to identify significant variables.##

library(caTools)

set.seed(12)
split = sample.split(data$rank, SplitRatio = 0.75)
split

dataTrain = subset(data, split == TRUE)
dataTest = subset(data, split == FALSE)

dataLog = glm(admit ~., dataTrain, family='binomial')
summary(dataLog)

##so here gre,ses,gendermale,race are  insignificant
##removing insignificant variable

##Run logistic model to determine the factors that influence the admission process of a student
(Drop insignificant variables)##
##building new new model with gpa and rank

dataLog1 = glm(admit ~ gpa + rank, dataTrain , family='binomial')
summary(dataLog1)

##Calculate the accuracy of the model and run validation techniques.##

predicted_val1 <- predict(dataLog,dataTest,type = "response")
dataTest$pred_admit1 <- ifelse(predicted_val1>0.5,1,0)

#confusion matrix
conf_mat1<-table(predicted=dataTest$pred_admit1,actual=dataTest$admit)
conf_mat1

#accuracy
accuracy1<-sum(diag(conf_mat1))/sum(conf_mat1)
accuracy1

##Try other modelling techniques like decision tree and SVM and select a champion model ##

#decision tree

library("rpart")
library("rpart.plot")
nrow(dataTrain)
0.03*nrow(dataTrain)
0.03*nrow(dataTrain)*3
r.cntrl<-rpart.control(minsplit = 15,minbucket = 3,xval = 2)
dec_clf<-rpart(admit~.,control = r.cntrl, data = dataTrain)
rpart.plot(dec_clf)
summary(dec_clf)

#svm

library(e1071)
data(iris)
nrow(iris)
set.seed(75)
```

```
svm1 <- svm(admit~., data=dataTrain,
            method="C-classification", kernal="radial",
            gamma=0.1, cost=15)
svm1$SV

plot(svm1, dataTrain, gre, gpa, ses, Gender_Male, Race, rank)


prediction <- predict(svm1, dataTest)
xtab <- table(dataTest$admit, prediction)
xtab

#accuracy of svm------
predicted_val2 <- predict(svm1,dataTest[-1])
predicted_val2

#Confusion matrix
conf_mat2<-table(predicted=predicted_val2,actual=dataTest$admit)
conf_mat2

#accuracy
accuracy2<-sum(diag(conf_mat2))/sum(conf_mat2)
accuracy2

##accuracy of decision tree------

predicted_val3 <- predict(dec_clf,dataTest[-1])
predicted_val3

#Confusion matrix
conf_mat3<-table(predicted=predicted_val3,actual=dataTest$admit)
conf_mat3

#accuracy
accuracy3<-sum(diag(conf_mat3))/sum(conf_mat3)
accuracy3


##Determine the accuracy rates for each kind of model and Select the most accurate model ##

#accuracy3=12 of decision tree, accuracy2=01 of svm, accuracy1=76 of of logistic regression
#so most accurate is logistic regression



##Categorize the average of grade point into High, Medium, and Low (with admission probability
percentages) and plot it on a point chart.##

descriptive = transform(data1,grelevels=ifelse(gre<440,"Low",ifelse(gre<580,"Medium","High")))
View(descriptive)
sum_decs=aggregate(admit~grelevels,descriptive,FUN = sum)
length_desc=aggregate(admit~grelevels,descriptive,FUN = length)
Probability_Table = cbind(sum_decs,resc=length_desc[,2])
Probability_Table_final = transform(Probability_Table,Probability_Admission = admit/resc)

Probability_Table_final

library("ggplot2")

ggplot(Probability_Table_final,aes(x=grelevels,y=Probability_Admission))+geom_point()

table(descriptive$admit,descriptive$grelevels)
```