DATA SCIENCE PROJECT WITH R

Background and Objective:

Every year thousands of applications are being submitted by international students for admission in colleges of the USA. It becomes an iterative task for the Education Department to know the total number of applications received and then compare that data with the total number of applications successfully accepted and visas processed. Hence to make the entire process easy, the education department in the US analyze the factors that influence the admission of a student into colleges. The objective of this exercise is to analyse the same.

Domain: Education

Dataset Description:

Attribute	**Description**
GRE	Graduate Record Exam Scores
GPA	Grade Point Average
Rank	It refers to the prestige of the undergraduate institution. The variable rank takes on the values 1 through 4. Institutions with a rank of 1 have the highest prestige, while those with a rank of 4 have the lowest.
Admit	It is a response variable; admit/don't admit is a binary variable where 1 indicates that student is admitted and 0 indicates that student is not admitted.
SES	SES refers to socioeconomic status: 1 - low, 2 - medium, 3 - high.
Gender_male	Gender_male (0, 1) = 0 -> Female, 1 -> Male
Race	Race – 1, 2, and 3 represent Hispanic, Asian, and African-American

Analysis Tasks: Analyze the historical data and determine the key drivers for admission.

Predictive:

- Find the missing values. (if any, perform missing value treatment)
- Find outliers (if any, then perform outlier treatment)
- Find the structure of the data set and if required, transform the numeric data type to factor and vice-versa.
- Find whether the data is normally distributed or not. Use the plot to determine the same.
- Normalize the data if not normally distributed.
- Use variable reduction techniques to identify significant variables.
- Run logistic model to determine the factors that influence the admission process of a student (Drop insignificant variables)
- Calculate the accuracy of the model and run validation techniques.
- Try other modelling techniques like decision tree and SVM and select a champion model
- Determine the accuracy rates for each kind of model
- Select the most accurate model
- Identify other Machine learning or statistical techniques

Descriptive:

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

Cross grid for admission variables with GRE Categorization is shown below:

GRE	Categorized
0-440	Low
440-580	Medium
580+	High

Code:

-> Lets upload the College_admission csv file to abstract the data.

```{r}
Data=read.csv("College_admission.csv")
Data

```

admit <int>	gre <int>	gpa <dbl>	ses <int>	Gender_Male <int>	Race <int>	rank <int>
0	380	3.61	1	0	3	3
1	660	3.67	2	0	2	3
1	800	4.00	2	0	2	1
1	640	3.19	1	1	2	4
0	520	2.93	3	1	2	4
1	760	3.00	2	1	1	2
1	560	2.98	2	1	2	1
0	400	3.08	2	0	2	2
1	540	3.39	1	1	1	3
0	700	3.92	1	0	2	2

1-10 of 400 rows Previous 1 2 3 4 5 6 ... 40 Next

-> Head and Tail commands:

```{r}
head(Data)
tail(Data)
```

data.frame
6 x 7

data.frame
6 x 7

	admit	gre	gpa	ses	Gender_Male	Race	rank
	<int>	<int>	<dbl>	<int>	<int>	<int>	<int>
1	0	380	3.61	1	0	3	3
2	1	660	3.67	2	0	2	3
3	1	800	4.00	2	0	2	1
4	1	640	3.19	1	1	2	4
5	0	520	2.93	3	1	2	4
6	1	760	3.00	2	1	1	2

6 rows

data.frame
6 x 7

data.frame
6 x 7

	admit	gre	gpa	ses	Gender_Male	Race	rank
	<int>	<int>	<dbl>	<int>	<int>	<int>	<int>
395	1	460	3.99	3	1	3	3
396	0	620	4.00	2	0	2	2
397	0	560	3.04	2	0	1	3
398	0	460	2.63	3	0	2	2
399	0	700	3.65	1	1	1	2
400	0	600	3.89	2	1	3	3

6 rows

-> Structure of the data.

```{r}
str(Data)
```

```
'data.frame':	400 obs. of  7 variables:
 $ admit      : int  0 1 1 1 0 1 1 0 1 0 ...
 $ gre        : int  380 660 800 640 520 760 560 400 540 700 ...
 $ gpa        : num  3.61 3.67 4 3.19 2.93 3 2.98 3.08 3.39 3.92 ...
 $ ses        : int  1 2 2 1 3 2 2 2 1 1 ...
 $ Gender_Male: int  0 0 0 1 1 1 1 0 1 0 ...
 $ Race       : int  3 2 2 2 2 1 2 2 1 2 ...
 $ rank       : int  3 3 1 4 4 2 1 2 3 2 ...
```

From the above data we can conclude there are 400 datasets with 7 Variables. All the variables are of "INT " Type execpt gpa. gpa variable is off "NUM" type.

-> Lets check for Missing Values.

```{r}
summary(Data)
```

```
     admit             gre             gpa             ses       
 Min.   :0.0000   Min.   :220.0   Min.   :2.260   Min.   :1.000  
 1st Qu.:0.0000   1st Qu.:520.0   1st Qu.:3.130   1st Qu.:1.000  
 Median :0.0000   Median :580.0   Median :3.395   Median :2.000  
 Mean   :0.3175   Mean   :587.7   Mean   :3.390   Mean   :1.992  
 3rd Qu.:1.0000   3rd Qu.:660.0   3rd Qu.:3.670   3rd Qu.:3.000  
 Max.   :1.0000   Max.   :800.0   Max.   :4.000   Max.   :3.000  
  Gender_Male         Race            rank      
 Min.   :0.000   Min.   :1.000   Min.   :1.000  
 1st Qu.:0.000   1st Qu.:1.000   1st Qu.:2.000  
 Median :0.000   Median :2.000   Median :2.000  
 Mean   :0.475   Mean   :1.962   Mean   :2.485  
 3rd Qu.:1.000   3rd Qu.:3.000   3rd Qu.:3.000  
 Max.   :1.000   Max.   :3.000   Max.   :4.000  
```

There are no NA values.

->Lets confirm by using "is.na" Function.

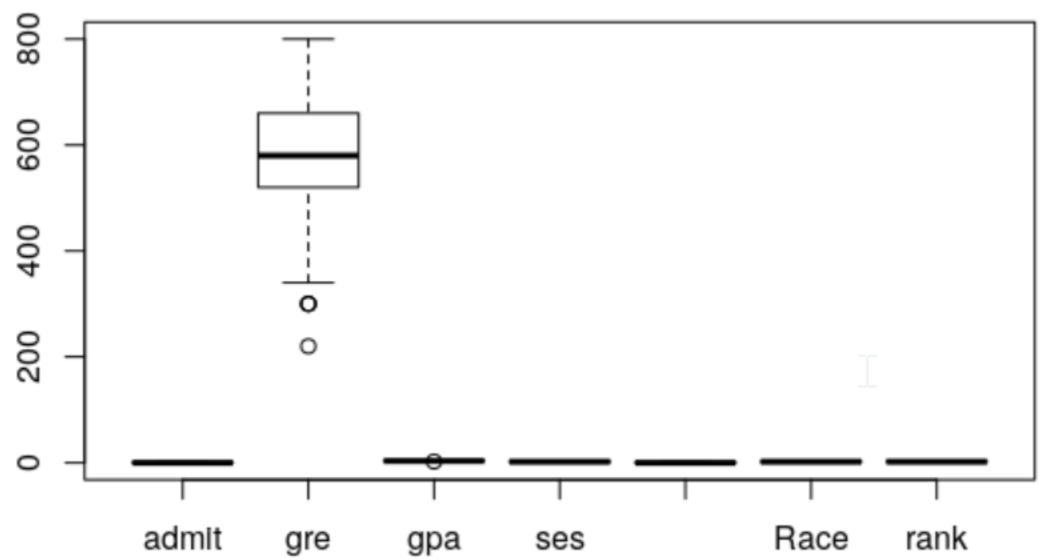
```{r}
sum(is.na(Data))
```

```
[1] 0
```

Therefore, there are 0 missing vaues.

-> Checking for Outliers.

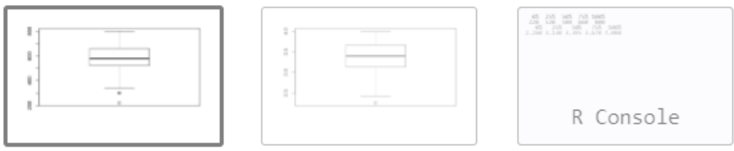
```{r}
boxplot(Data)
```

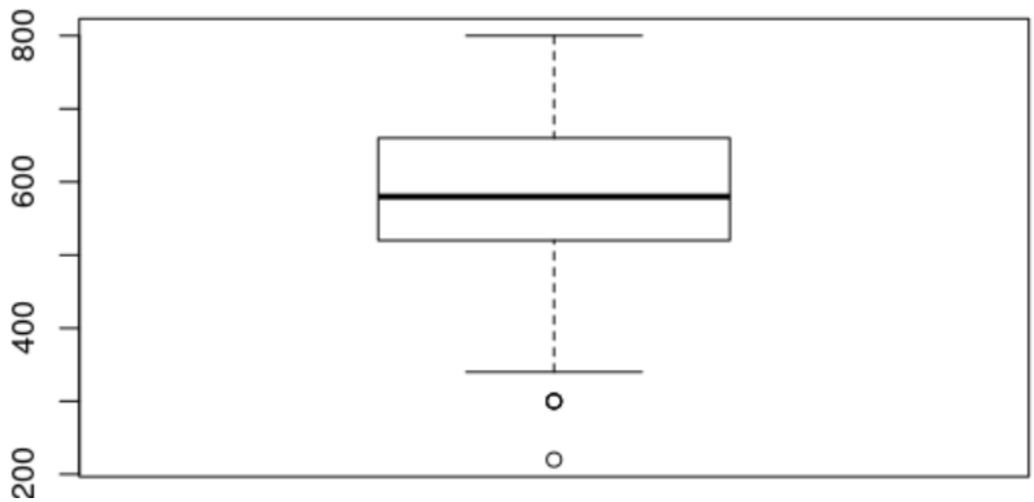


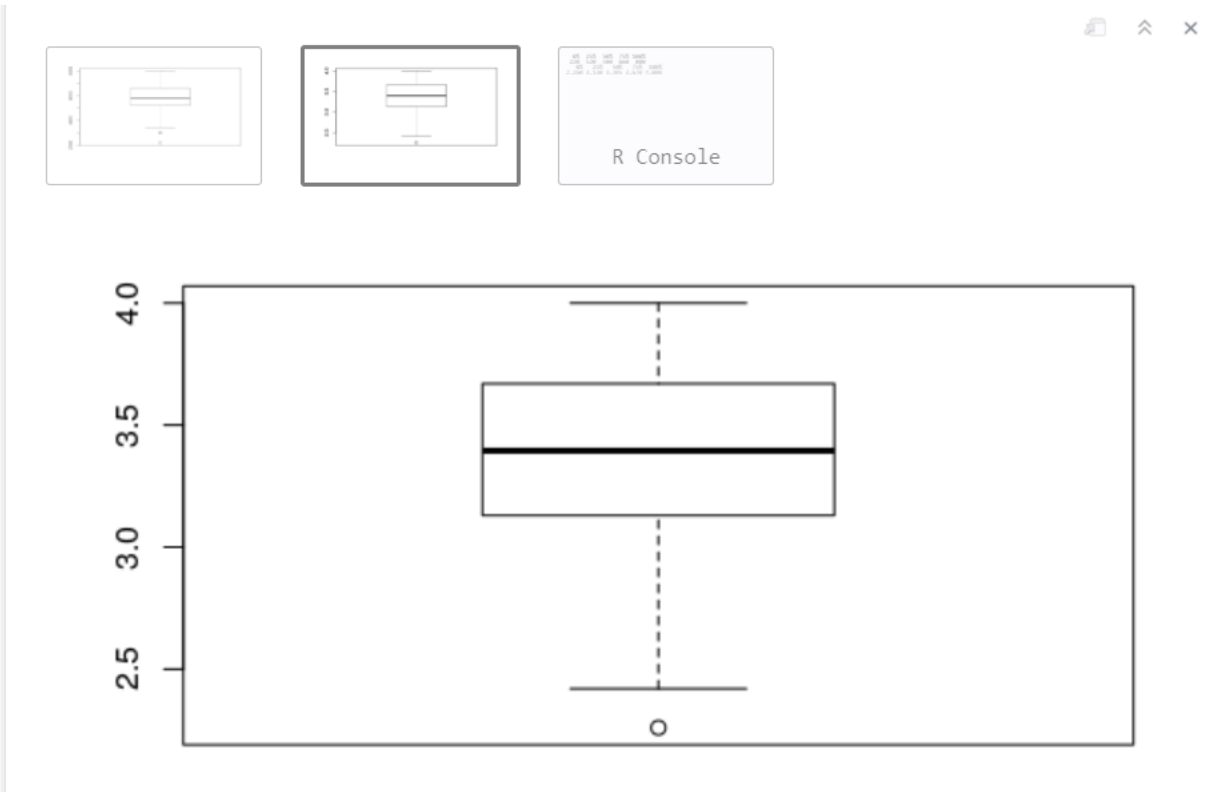
We can see gpa and gre have few outliners.

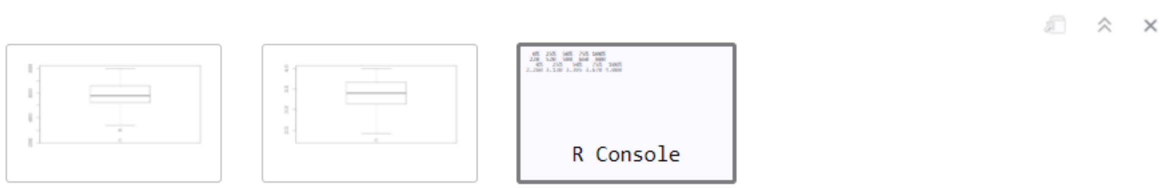
--> Looking gre and gpa

```{r}
boxplot(Data$gre)
boxplot(Data$gpa)
quantile(Data$gre)
quantile(Data$gpa)

```










```
  0%  25%  50%  75% 100% 
 220  520  580  660  800 
   0%    25%    50%    75%   100% 
2.260 3.130 3.395 3.670 4.000 
```

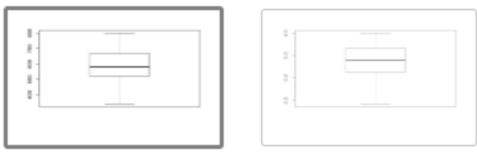
--> let eliminate the outliers

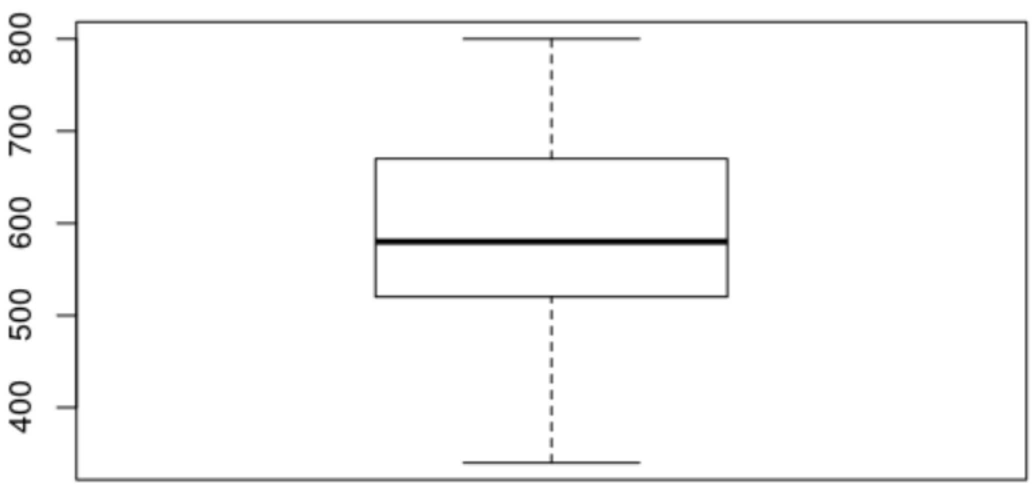
````
```{r}
Data1=subset(Data, gre > 300 & gpa >2.260 )
dim(Data1)
```
````

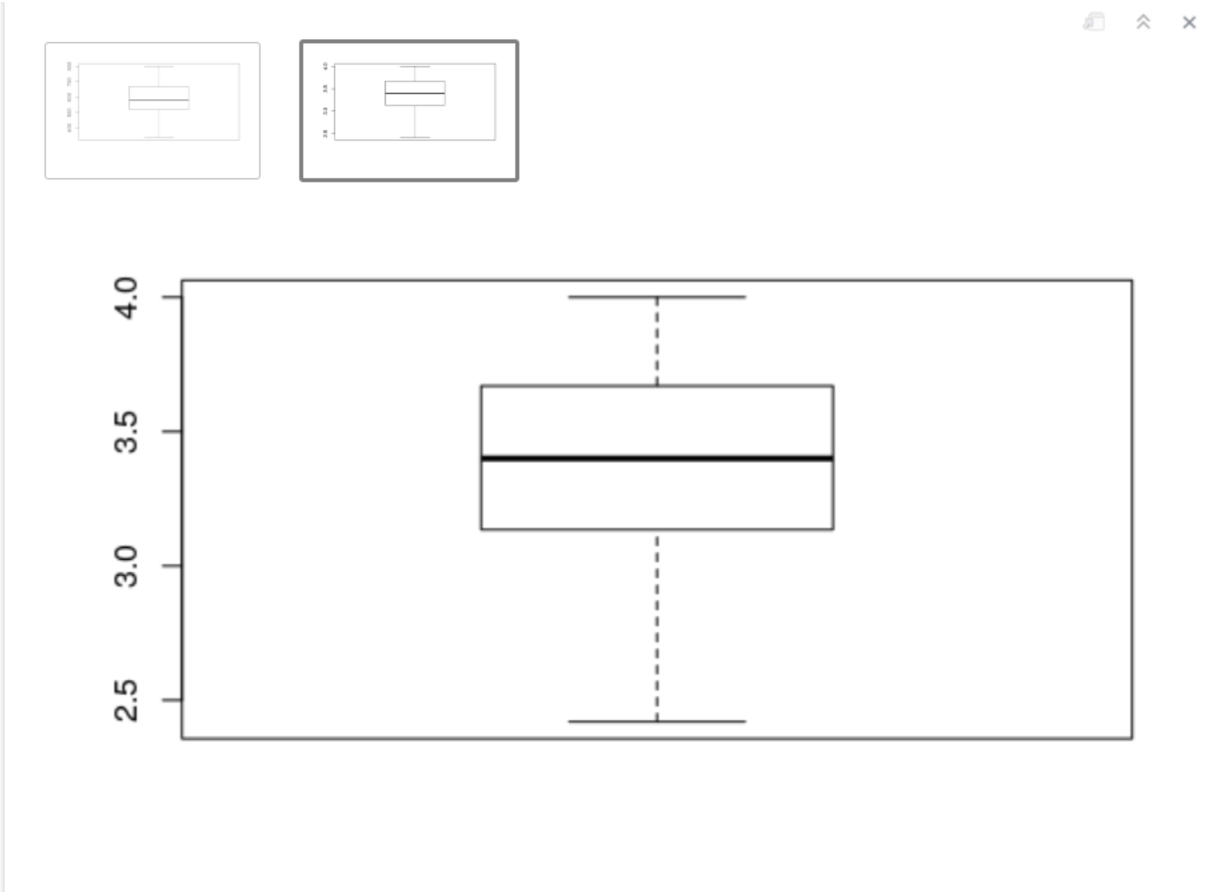
```
[1] 395   7
```

`

````
```{r}
boxplot(Data1$gre)
boxplot(Data1$gpa)
```
````






4.0
3.5
3.0
2.5

We have removed the outliers.

```{r}
Data1
```

	admit <int>	gre <int>	gpa <dbl>	ses <int>	Gender_Male <int>	Race <int>	rank <int>
1	0	380	3.61	1	0	3	3
2	1	660	3.67	2	0	2	3
3	1	800	4.00	2	0	2	1
4	1	640	3.19	1	1	2	4
5	0	520	2.93	3	1	2	4
6	1	760	3.00	2	1	1	2
7	1	560	2.98	2	1	2	1
8	0	400	3.08	2	0	2	2
9	1	540	3.39	1	1	1	3
10	0	700	3.92	1	0	2	2

1-10 of 395 rows Previous 1 2 3 4 5 6 ... 40 Next

Data 1 table has no outliers and the dimensions have reduced by 5 rows.

--> let check the data is normally distributed

--> Lets take the gpa and gre to check the normality as they are dependent variables.

HYPOTHESIS :

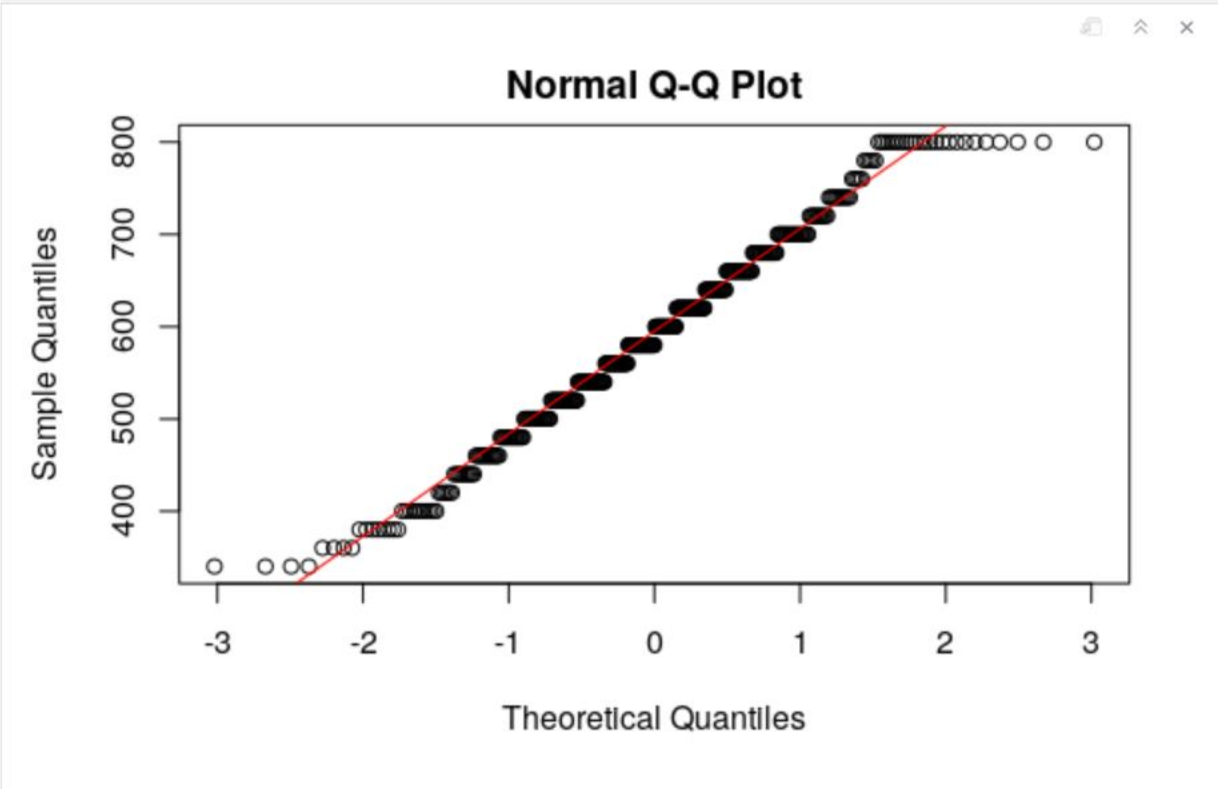
Null Hypothesis:

-> The data is normally distributed .

Alternative Hypothesis: The data is not normally distributed.

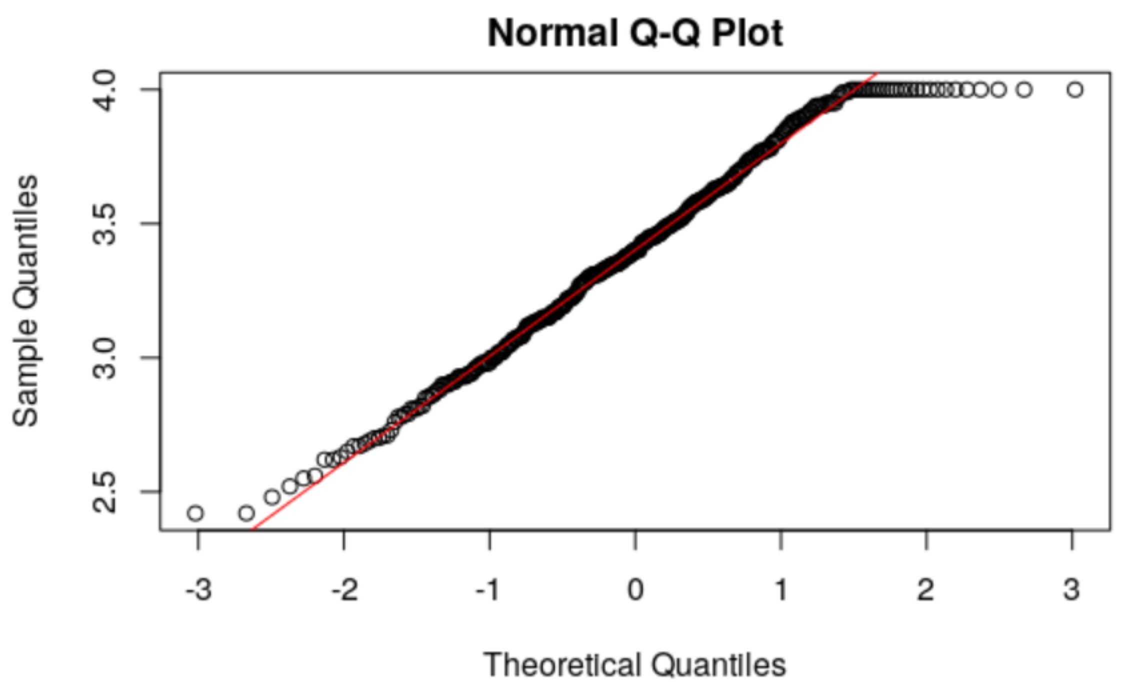
--> Lets use qq plot for normailty test

```{r}
qqnorm(Data1$gre)
qqline(Data1$gre, col="red")
```


Normal Q-Q Plot
Sample Quantiles
400
500
600
700
800
-3
-2
-1
0
1
2
3
Theoretical Quantiles

We can see that most of the data is lies on the ed line and is mostly normally distuributed but not completly.

```{r}
qqnorm(Data1$gpa)
qqline(Data1$gpa,col="red")
```



we see even the data in the gre variable are highly normally distributed but not completely

Lets perform a Normality test to confirm the same.

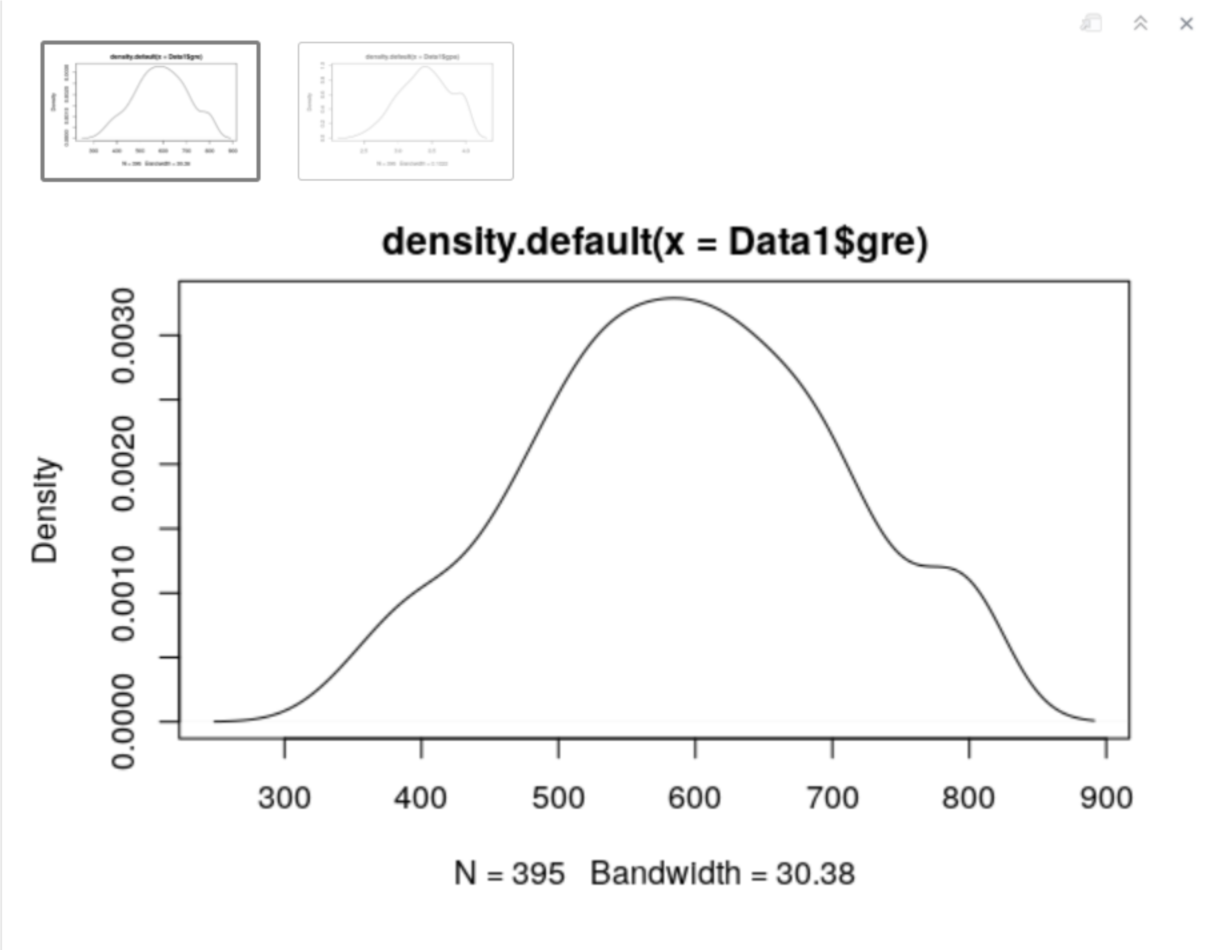
Normality Test

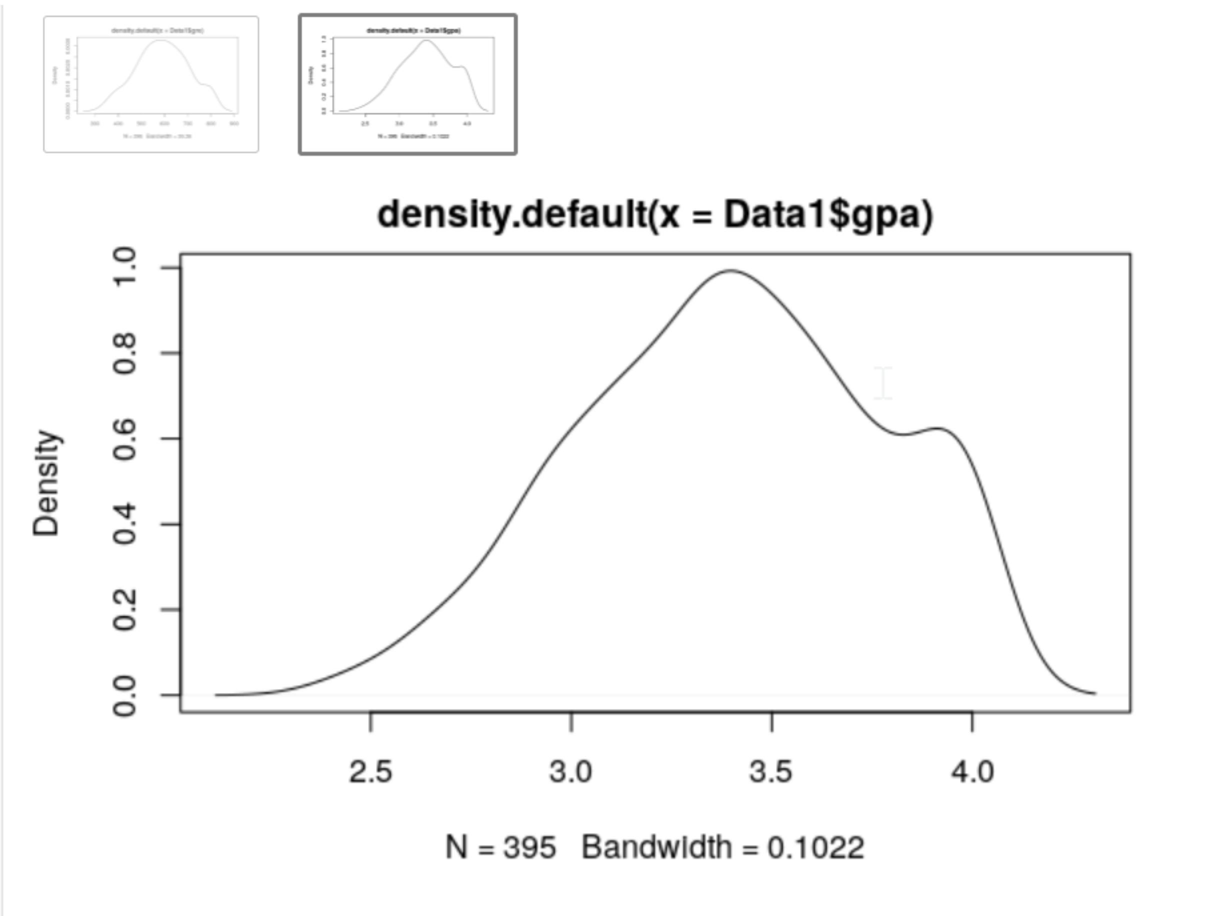
```{r}
shapiro.test(Data1$gpa)
shapiro.test(Data1$gpa)

```

```
        Shapiro-Wilk normality test

data:  Data1$gpa
W = 0.97646, p-value = 5.004e-06


        Shapiro-Wilk normality test

data:  Data1$gpa
W = 0.97646, p-value = 5.004e-06
```

Since the P value is less than 0.05 , we reject the null hypothesis , therefore the data is not normally distributed.

```{r}
plot(density(Data1$gre))
plot(density(Data1$gpa))
```


density.default(x = Data1$gre)
Density
0.0000
0.0010
0.0020
0.0030
300
400
500
600
700
800
900
N = 395   Bandwidth = 30.38


density.default(x = Data1$gpa)
Density
0.0
0.2
0.4
0.6
0.8
1.0
2.5
3.0
3.5
4.0
N = 395   Bandwidth = 0.1022

Lets Normalize the data using Min Max function

What we need to do now is to create a function in R that will normalize the data according to the following formula:

```{r}
normalize <- function(x) {
return ((x - min(x)) / (max(x) - min(x)))
}
```

Let's call our function normalize()

We have just created two new columns with normalized data for "gpa" and "gre" variables

```{r}
Data1$gpa_norm<-normalize(Data1$gpa)
Data1$gre_norm<-normalize(Data1$gre)
```

Take a look at your dataset now:

```{r}
View(Data1)
```

College_admission.Rmd × Data1 ×

Filter

	admit	gre	gpa	ses	Gender_Male	Race	rank	gpa_norm	gre_norm	Categorize
1	0	380	3.61	1	0	3	3	0.75316456	0.08695652	LOW
2	1	660	3.67	2	0	2	3	0.79113924	0.69565217	HIGH
3	1	800	4.00	2	0	2	1	1.00000000	1.00000000	HIGH
4	1	640	3.19	1	1	2	4	0.48734177	0.65217391	HIGH
5	0	520	2.93	3	1	2	4	0.32278481	0.39130435	MEDIUM
6	1	760	3.00	2	1	1	2	0.36708861	0.91304348	HIGH
7	1	560	2.98	2	1	2	1	0.35443038	0.47826087	MEDIUM
8	0	400	3.08	2	0	2	2	0.41772152	0.13043478	LOW
9	1	540	3.39	1	1	1	3	0.61392405	0.43478261	MEDIUM
10	0	700	3.92	1	0	2	2	0.94936709	0.78260870	HIGH
11	0	800	4.00	1	1	1	4	1.00000000	1.00000000	HIGH
12	0	440	3.22	3	0	2	1	0.50632911	0.21739130	LOW
13	1	760	4.00	3	1	2	1	1.00000000	0.91304348	HIGH
14	0	700	3.08	2	0	2	2	0.41772152	0.78260870	HIGH
15	1	700	4.00	2	1	1	1	1.00000000	0.78260870	HIGH
16	0	480	3.44	3	0	1	3	0.64556962	0.30434783	MEDIUM
17	0	780	3.87	2	0	3	4	0.91772152	0.95652174	HIGH

Showing 1 to 21 of 395 entries, 11 total columns

Lets check the summary of the data

```{r}
summary(Data1$gpa)
summary(Data1$gpa_norm)
```

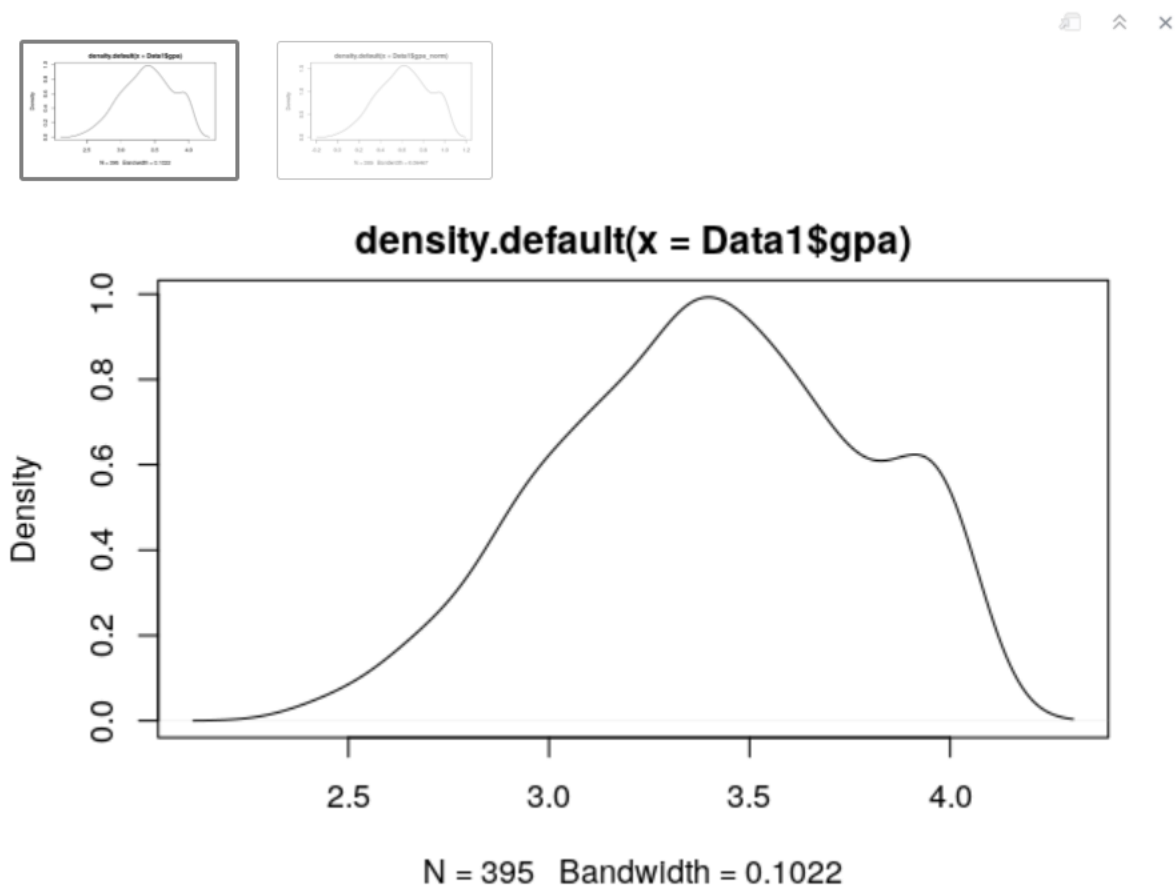
```
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
  2.420   3.135   3.400   3.398   3.670   4.000 
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
 0.0000  0.4525  0.6203  0.6188  0.7911  1.0000 
```

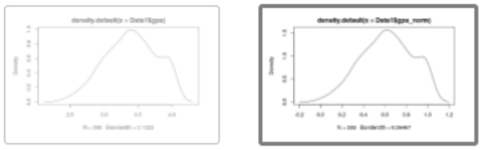
```{r}
summary(Data1$gre)
summary(Data1$gre_norm)
```

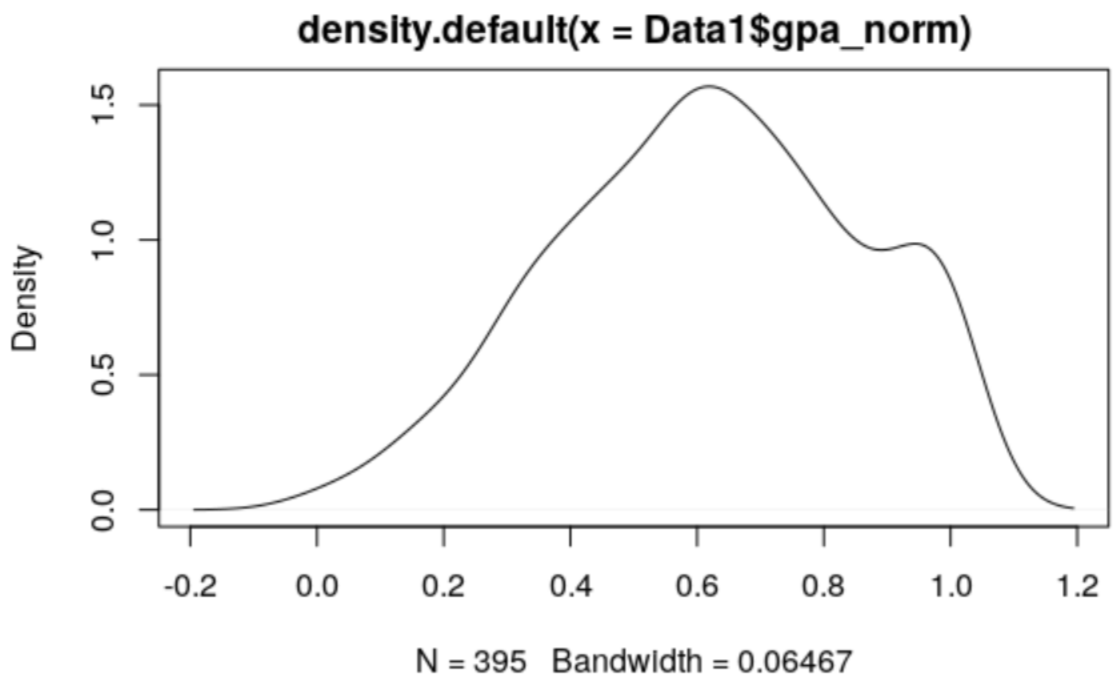
```
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
  340.0   520.0   580.0   591.2   670.0   800.0 
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
 0.0000  0.3913  0.5217  0.5462  0.7174  1.0000 
```

We can see the Mean and median of gre_norm and gpa_norm and closer compared to the gre and gpa variables

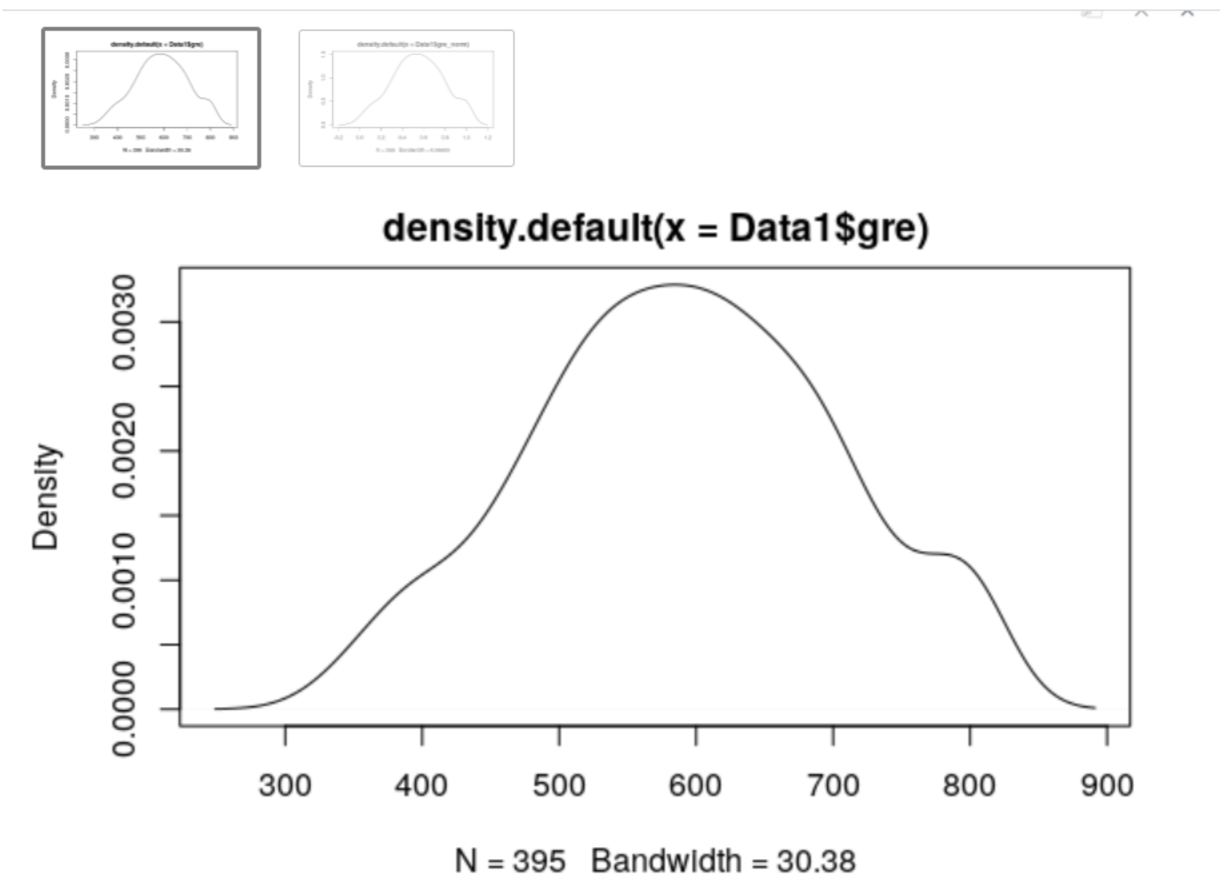
```{r}

plot(density(Data1$gpa))
plot(density(Data1$gpa_norm))
```

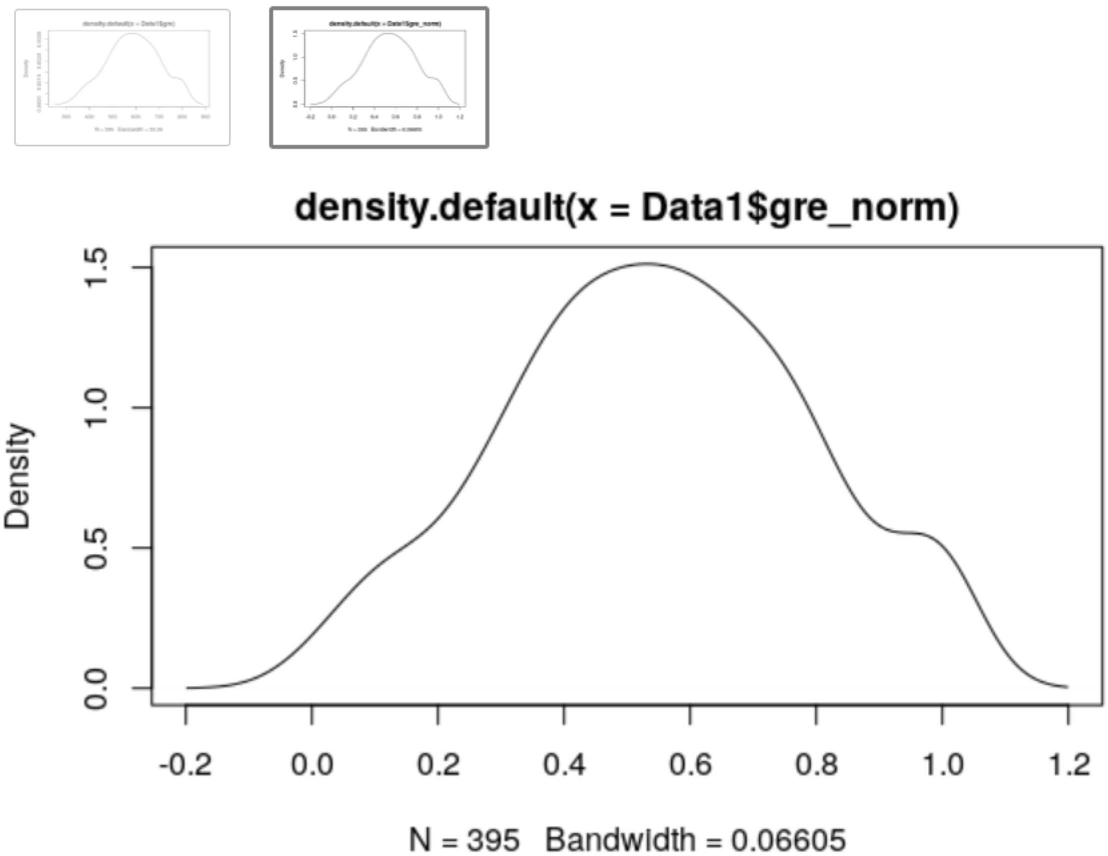

density.default(x = Data1$gpa)
Density
0.0
0.2
0.4
0.6
0.8
1.0
2.5
3.0
3.5
4.0
N = 395   Bandwidth = 0.1022






```{r}
plot(density(Data1$gre))
plot(density(Data1$gre_norm))
```


density.default(x = Data1$gre)
Density
0.0000
0.0010
0.0020
0.0030
300
400
500
600
700
800
900
N = 395   Bandwidth = 30.38



We observe identical density plots even though the X axis is rescaled.

Therefore we show that normalization didn't affect the distribution properties of the rescaled data.

The same hold for the "gre" and "gre_norm".

Converting from Numeric to Factor

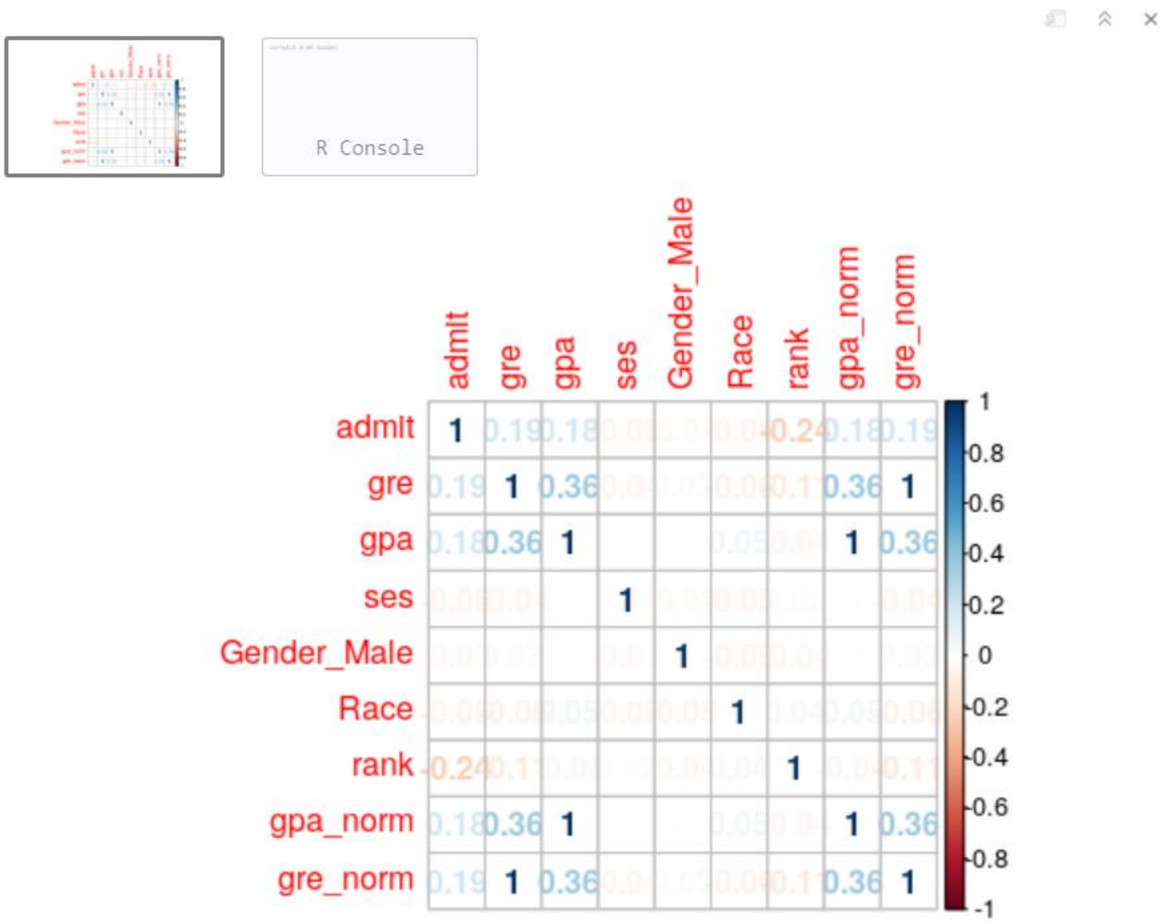
```
```{r}
Data1$gre=as.numeric(Data1$gre)
Data1$admit = as.numeric(Data1$admit)
Data1$ses = as.numeric(Data1$ses)
Data1$Gender_Male = as.numeric(Data1$Gender_Male)
Data1$Race = as.numeric(Data1$Race)
Data1$rank = as.numeric(Data1$rank)

```
```

```
```{r}
str(Data1)
```
```

```
'data.frame':   395 obs. of  9 variables:
 $ admit      : num  0 1 1 1 0 1 1 0 1 0 ...
 $ gre        : num  380 660 800 640 520 760 560 400 540 700 ...
 $ gpa        : num  3.61 3.67 4 3.19 2.93 3 2.98 3.08 3.39 3.92 ...
 $ ses        : num  1 2 2 1 3 2 2 2 1 1 ...
 $ Gender_Male: num  0 0 0 1 1 1 1 0 1 0 ...
 $ Race       : num  3 2 2 2 2 1 2 2 1 2 ...
 $ rank       : num  3 3 1 4 4 2 1 2 3 2 ...
 $ gpa_norm   : num  0.753 0.791 1 0.487 0.323 ...
 $ gre_norm   : num  0.087 0.696 1 0.652 0.391 ...
```

```{r}
summary(Data1)
```

```
     admit             gre             gpa             ses       
 Min.   :0.000   Min.   :340.0   Min.   :2.420   Min.   :1.000  
 1st Qu.:0.000   1st Qu.:520.0   1st Qu.:3.135   1st Qu.:1.000  
 Median :0.000   Median :580.0   Median :3.400   Median :2.000  
 Mean   :0.319   Mean   :591.2   Mean   :3.398   Mean   :1.995  
 3rd Qu.:1.000   3rd Qu.:670.0   3rd Qu.:3.670   3rd Qu.:3.000  
 Max.   :1.000   Max.   :800.0   Max.   :4.000   Max.   :3.000  
  Gender_Male          Race            rank          gpa_norm     
 Min.   :0.0000   Min.   :1.000   Min.   :1.000   Min.   :0.0000  
 1st Qu.:0.0000   1st Qu.:1.000   1st Qu.:2.000   1st Qu.:0.4525  
 Median :0.0000   Median :2.000   Median :2.000   Median :0.6203  
 Mean   :0.4709   Mean   :1.967   Mean   :2.476   Mean   :0.6188  
 3rd Qu.:1.0000   3rd Qu.:3.000   3rd Qu.:3.000   3rd Qu.:0.7911  
 Max.   :1.0000   Max.   :3.000   Max.   :4.000   Max.   :1.0000  
    gre_norm     
 Min.   :0.0000  
 1st Qu.:0.3913  
 Median :0.5217  
 Mean   :0.5462  
 3rd Qu.:0.7174  
 Max.   :1.0000  
```

```{r}
library(corrplot)
Data1cor <-cor(Data1)

corrplot(Data1cor, method="number")
```


R Console
admit
gre
gpa
ses
Gender_Male
Race
rank
gpa_norm
gre_norm
1
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1

Multiple Linear Regression

```{r}
Datalm=lm(admit ~ . , data = Data1)
summary(Datalm)
```

```
Call:
lm(formula = admit ~ ., data = Data1)

Residuals:
    Min      1Q  Median      3Q     Max 
-0.7151 -0.3411 -0.1945  0.5014  0.9578 

Coefficients: (2 not defined because of singularities)
              Estimate Std. Error t value Pr(>|t|)    
(Intercept) -0.1027404  0.2358540  -0.436   0.6634    
gre          0.0004604  0.0002177   2.115   0.0351 *  
gpa          0.1632280  0.0643311   2.537   0.0116 *  
ses         -0.0299036  0.0278312  -1.074   0.2833    
Gender_Male -0.0376464  0.0450381  -0.836   0.4037    
Race        -0.0311635  0.0275326  -1.132   0.2584    
rank        -0.1076063  0.0239679  -4.490 9.42e-06 ***
gpa_norm            NA         NA      NA       NA    
gre_norm            NA         NA      NA       NA    
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.4456 on 388 degrees of freedom
Multiple R-squared:  0.1021,	Adjusted R-squared:  0.08822 
F-statistic: 7.354 on 6 and 388 DF,  p-value: 1.838e-07
```

```
We can see from the above data, Variables gre, gpa and rank are significant
variables that effect the admit of the university.


LOGISTIC REGRESSION :

"To determine the factors that influence the admission process of a student "
```

```{r}
Data2=Data1
Data2

```

	admit <dbl>	gre <dbl>	gpa <dbl>	ses <dbl>	Gender_Male <dbl>	Race <dbl>	rank <dbl>
1	0	380	3.61	1	0	3	3
2	1	660	3.67	2	0	2	3
3	1	800	4.00	2	0	2	1
4	1	640	3.19	1	1	2	4
5	0	520	2.93	3	1	2	4
6	1	760	3.00	2	1	1	2
7	1	560	2.98	2	1	2	1
8	0	400	3.08	2	0	2	2
9	1	540	3.39	1	1	1	3
10	0	700	3.92	1	0	2	2

1-10 of 395 rows | 1-8 of 9 columns Previous 1 2 3 4 5 6 ... 40 Next

```{r}
str(Data2)
```

```
'data.frame':	395 obs. of  9 variables:
 $ admit      : num  0 1 1 1 0 1 1 0 1 0 ...
 $ gre        : num  380 660 800 640 520 760 560 400 540 700 ...
 $ gpa        : num  3.61 3.67 4 3.19 2.93 3 2.98 3.08 3.39 3.92 ...
 $ ses        : num  1 2 2 1 3 2 2 2 1 1 ...
 $ Gender_Male: num  0 0 0 1 1 1 1 0 1 0 ...
 $ Race       : num  3 2 2 2 2 1 2 2 1 2 ...
 $ rank       : num  3 3 1 4 4 2 1 2 3 2 ...
 $ gpa_norm   : num  0.753 0.791 1 0.487 0.323 ...
 $ gre_norm   : num  0.087 0.696 1 0.652 0.391 ...
```

Conditions:
->dependent Varible - Categorical.
->Output of dependent - binary
->Independent Variable - categorical/numerical.

```{r}
Data2$admit=as.factor(Data2$admit)
Data2$ses=as.factor(Data2$ses)
Data2$Gender_Male=as.factor(Data2$Gender_Male)
Data2$Race=as.factor(Data2$Race)
Data2$rank=as.factor(Data2$rank)
```

```{r}
str(Data2)
```

```
'data.frame':   395 obs. of  9 variables:
 $ admit      : Factor w/ 2 levels "0","1": 1 2 2 2 1 2 2 1 2 1 ...
 $ gre        : num  380 660 800 640 520 760 560 400 540 700 ...
 $ gpa        : num  3.61 3.67 4 3.19 2.93 3 2.98 3.08 3.39 3.92 ...
 $ ses        : Factor w/ 3 levels "1","2","3": 1 2 2 1 3 2 2 2 1 1 ...
 $ Gender_Male: Factor w/ 2 levels "0","1": 1 1 1 2 2 2 2 1 2 1 ...
 $ Race       : Factor w/ 3 levels "1","2","3": 3 2 2 2 2 1 2 2 1 2 ...
 $ rank       : Factor w/ 4 levels "1","2","3","4": 3 3 1 4 4 2 1 2 3 2 ...
 $ gpa_norm   : num  0.753 0.791 1 0.487 0.323 ...
 $ gre_norm   : num  0.087 0.696 1 0.652 0.391 ...
```

Split the data into test and train

```{r}
library(caTools)
split <-sample.split(Data2, SplitRatio = 0.8)
split
train <- subset(Data2, split==TRUE)
test  <- subset(Data2, split==FALSE)
print("Train:")
str(train)
print("Test: ")
str(test)
```

```
 [1]  TRUE FALSE  TRUE  TRUE  TRUE  TRUE FALSE  TRUE  TRUE
 [1] "Train:"
'data.frame':   307 obs. of  9 variables:
 $ admit      : Factor w/ 2 levels "0","1": 1 2 2 1 2 1 2 1 1 2 ...
 $ gre        : num  380 800 640 520 760 400 540 700 440 760 ...
 $ gpa        : num  3.61 4 3.19 2.93 3 3.08 3.39 3.92 3.22 4 ...
 $ ses        : Factor w/ 3 levels "1","2","3": 1 2 1 3 2 2 1 1 3 3 ...
 $ Gender_Male: Factor w/ 2 levels "0","1": 1 1 2 2 2 1 2 1 1 2 ...
 $ Race       : Factor w/ 3 levels "1","2","3": 3 2 2 2 1 2 1 2 2 2 ...
 $ rank       : Factor w/ 4 levels "1","2","3","4": 3 1 4 4 2 2 3 2 1 1 ...
 $ gpa_norm   : num  0.753 1 0.487 0.323 0.367 ...
 $ gre_norm   : num  0.087 1 0.652 0.391 0.913 ...
 [1] "Test: "
'data.frame':   88 obs. of  9 variables:
 $ admit      : Factor w/ 2 levels "0","1": 2 2 1 1 2 2 2 2 1 2 ...
 $ gre        : num  660 560 800 480 540 760 780 800 520 600 ...
 $ gpa        : num  3.67 2.98 4 3.44 3.81 3.35 3.22 4 2.9 3.15 ...
 $ ses        : Factor w/ 3 levels "1","2","3": 2 2 1 3 1 2 1 3 2 2 ...
 $ Gender_Male: Factor w/ 2 levels "0","1": 1 2 2 1 1 1 1 1 1 2 ...
 $ Race       : Factor w/ 3 levels "1","2","3": 2 2 1 1 3 2 1 1 2 1 ...
 $ rank       : Factor w/ 4 levels "1","2","3","4": 3 1 4 3 1 2 2 3 3 2 ...
 $ gpa_norm   : num  0.791 0.354 1 0.646 0.88 ...
 $ gre_norm   : num  0.696 0.478 1 0.304 0.435 ...
```

Creating Logistic Regression Model :

```{r}
Data2LR <- glm(admit ~ gre + gpa + ses + Gender_Male + Race + rank, data=train,
family = "binomial")
summary(Data2LR)
```

```
Call:
glm(formula = admit ~ gre + gpa + ses + Gender_Male + Race +
    rank, family = "binomial", data = train)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-1.8298  -0.8189  -0.5601   0.9650   2.2383

Coefficients:
              Estimate Std. Error z value Pr(>|z|)
(Intercept)  -4.416772   1.345471  -3.283 0.001028 **
gre           0.002528   0.001335   1.894 0.058160 .
gpa           1.079493   0.389665   2.770 0.005600 **
ses2         -0.425033   0.333976  -1.273 0.203145
ses3         -0.161081   0.326580  -0.493 0.621845
Gender_Male1 -0.255703   0.273638  -0.934 0.350069
Race2        -0.675610   0.345663  -1.955 0.050639 .
Race3        -0.231135   0.316229  -0.731 0.464834
rank2        -0.709627   0.364143  -1.949 0.051324 .
rank3        -1.628673   0.405938  -4.012 6.02e-05 ***
rank4        -1.901623   0.515677  -3.688 0.000226 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 381.45  on 306  degrees of freedom
Residual deviance: 330.89  on 296  degrees of freedom
AIC: 352.89

Number of Fisher Scoring iterations: 4
```

We can see the the variables that affect the admission of the student are gre, gpa and rank of the institution.

We also see , rank 3 and rank 4 are highly significant .

Gender , Race very sligthly effects and SES dont at all effect the admission of the student.

--> Run the test data.

```{r}
A <-predict(Data2LR, test , type = "response")
A
```

```
         2          7         11         16         20         25 
0.18010596 0.24225690 0.44190435 0.21764265 0.69641340 0.33421426 
        29         34         38         43         47         52 
0.57973596 0.53348336 0.06292557 0.29116860 0.41539947 0.05079896 
        56         61         65         70         75         80 
0.31264157 0.30858066 0.23633222 0.79846415 0.10620842 0.59100996 
        84         89         93         98        102        107 
0.03527439 0.62283555 0.66586193 0.24023541 0.16022994 0.56580689 
       111        116        120        125        129        134 
0.15889304 0.17842877 0.04204435 0.33341932 0.30909800 0.05663793 
       138        143        147        152        156        161 
0.40747713 0.20844913 0.32533792 0.24101852 0.10143510 0.38077207 
       165        170        174        179        184        189 
0.41854284 0.24833689 0.59093094 0.21834002 0.53818402 0.14769204 
       193        198        202        207        211        216 
0.18708818 0.11390869 0.21312101 0.70069043 0.12536871 0.08818142 
       220        225        229        234        238        243 
0.28175373 0.20923774 0.44766127 0.06013147 0.27865190 0.14044531 
       247        252        256        261        265        270 
0.50932680 0.17993710 0.18443226 0.32505733 0.19270242 0.08349285 
       274        279        283        288        293        298 
0.47615576 0.11492770 0.14652546 0.28674304 0.47658586 0.14805330 
       302        308        312        318        322        327 
0.28585837 0.46838420 0.51974619 0.33549916 0.37196399 0.33834396 
       331        336        340        345        349        354 
0.43008149 0.73000537 0.12055007 0.12443491 0.34320607 0.51276031 
       358        363        367        372        376        381 
0.47112615 0.43211952 0.12183278 0.23172819 0.21481145 0.58721588 
       385        390        394        399 
0.14613405 0.40923056 0.35174935 0.58124834 
```

Confusion Matrix

```{r}
con <-table(Actual_Value=test$admit , Predicted_Value = A>0.5)
con
```

```
            Predicted_Value
Actual_Value FALSE TRUE
           0    49    9
           1    22    8
```

Droping the insignificant values :

```{r}
Data3 <- Data2[,c(1,2,3,7)]
str(Data3)
```

```
'data.frame':	395 obs. of  4 variables:
 $ admit: Factor w/ 2 levels "0","1": 1 2 2 2 1 2 2 1 2 1 ...
 $ gre  : num  380 660 800 640 520 760 560 400 540 700 ...
 $ gpa  : num  3.61 3.67 4 3.19 2.93 3 2.98 3.08 3.39 3.92 ...
 $ rank : Factor w/ 4 levels "1","2","3","4": 3 3 1 4 4 2 1 2 3 2 ...
```

The only required columns

Checking for the Accuracy

Accuracy = (True positives + True Negatives)/Total population

```{r}
Acc1= (8+49)/(22+8+49+9) * 100
Acc1
```

```
[1] 64.77273
```

Lets use the Data3 dataset that has all the Significant Variables.

SVM Model:

SVM Model:

```{r}
set.seed(123)
library(caTools)
split1 <-sample.split(Data3, SplitRatio = 0.7)
split1
train1 <- subset(Data3, split==TRUE)
test1  <- subset(Data3, split==FALSE)
print("Train:")
str(train1)
print("Test: ")
str(test1)
```

```
[1]  TRUE FALSE  TRUE FALSE
[1] "Train:"
'data.frame':	307 obs. of  4 variables:
 $ admit: Factor w/ 2 levels "0","1": 1 2 2 1 2 1 2 1 1 2 ...
 $ gre  : num  380 800 640 520 760 400 540 700 440 760 ...
 $ gpa  : num  3.61 4 3.19 2.93 3 3.08 3.39 3.92 3.22 4 ...
 $ rank : Factor w/ 4 levels "1","2","3","4": 3 1 4 4 2 2 3 2 1 1 ...
[1] "Test: "
'data.frame':	88 obs. of  4 variables:
 $ admit: Factor w/ 2 levels "0","1": 2 2 1 1 2 2 2 2 1 2 ...
 $ gre  : num  660 560 800 480 540 760 780 800 520 600 ...
 $ gpa  : num  3.67 2.98 4 3.44 3.81 3.35 3.22 4 2.9 3.15 ...
 $ rank : Factor w/ 4 levels "1","2","3","4": 3 1 4 3 1 2 2 3 3 2 ...
```

```{r}
library(e1071)
Data2_SVM= svm( admit~. ,data=train1 , kernel="linear")
Data2_SVM
```

```
Call:
svm(formula = admit ~ ., data = train1, kernel = "linear")


Parameters:
   SVM-Type:  C-classification
 SVM-Kernel:  linear
       cost:  1

Number of Support Vectors:  195
```

Number of Support Vectors are 195

Lets test the Model on the test data

```{r}
B <-predict(Data2_SVM, test1 , type = "response")
B
```

```
  2   7  11  16  20  25  29  34  38  43  47  52  56  61  65  70  75  80 
  0   1   0   0   1   0   0   0   0   0   0   0   0   0   0   1   0   1 
 84  89  93  98 102 107 111 116 120 125 129 134 138 143 147 152 156 161 
  0   1   0   0   0   1   0   0   0   0   0   0   0   0   0   0   0   0 
165 170 174 179 184 189 193 198 202 207 211 216 220 225 229 234 238 243 
  0   0   0   0   0   0   0   0   0   1   0   0   0   0   0   0   0   0 
247 252 256 261 265 270 274 279 283 288 293 298 302 308 312 318 322 327 
  0   0   0   0   0   0   1   0   0   0   0   0   0   0   0   0   0   0 
331 336 340 345 349 354 358 363 367 372 376 381 385 390 394 399 
  0   1   0   0   0   0   1   0   0   0   0   0   0   0   0   0 
Levels: 0 1
```

Confussion Matrix:

```{r}
con2 <- table(Actual_Value=test1$admit , Predicted_Value = B)
con2
```

```
            Predicted_Value
Actual_Value  0  1
           0 53  5
           1 25  5
```

Accuracy = (True positives + True Negatives)/Total population

```{r}
Acc2= (53+5)/(53+25+5+5) * 100
Acc2
```

```
[1] 65.90909
```

Decision Tree:

We will use the same Train and test data used in SMV Model

Insert the library party

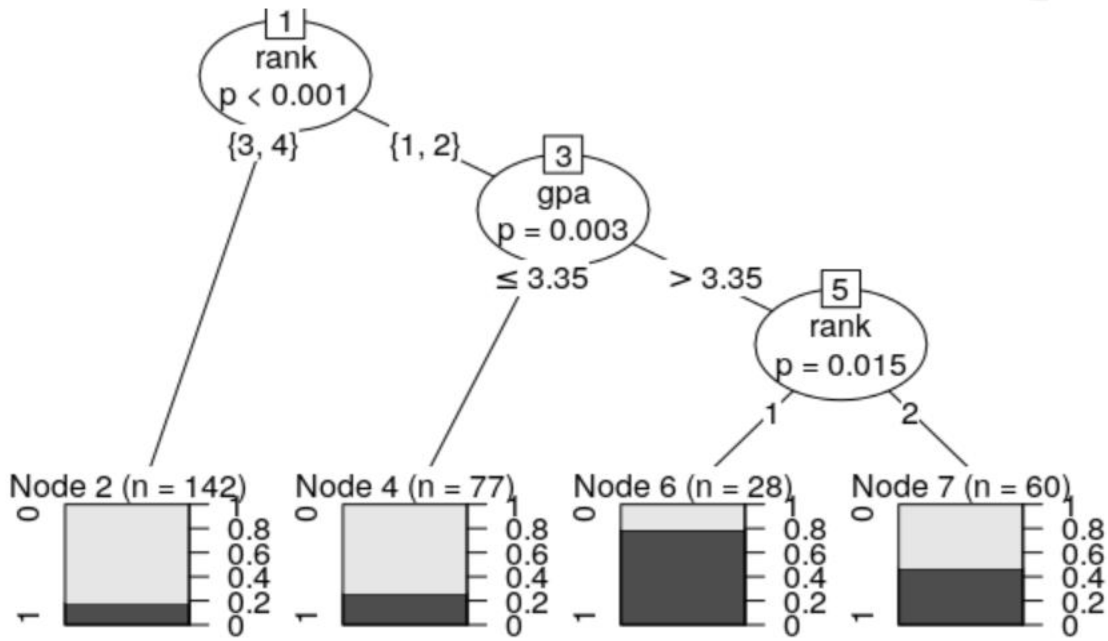
```{r}
library(party)
```

```{r}
tree <- ctree( admit ~ ., data= train1)
tree
```

```
	 Conditional inference tree with 4 terminal nodes

Response:  admit 
Inputs:  gre, gpa, rank 
Number of observations:  307 

1) rank == {3, 4}; criterion = 1, statistic = 26.577
  2)*  weights = 142 
1) rank == {1, 2}
  3) gpa <= 3.35; criterion = 0.997, statistic = 10.962
    4)*  weights = 77 
  3) gpa > 3.35
    5) rank == {1}; criterion = 0.985, statistic = 7.83
      6)*  weights = 28 
    5) rank == {2}
      7)*  weights = 60 
```

For better understanding lets plot the tree:

```{r}
plot(tree)
```



Test the data:

Test the data:

```{r}
C<-predict(tree, test1 , type = "response")
C
```

```
  [1] 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 1 0 1 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0
 [36] 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 [71] 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 Levels: 0 1
```

CONFUSION Matrix:

```{r}
con3 <- table(Actual_Value=test1$admit , Predicted_Value = C)
con3
```

```
             Predicted_Value
 Actual_Value  0  1
            0 56  2
            1 26  4
```

Acurracy Test:

Accuracy = (True positives + True Negatives)/Total population

```{r}
Acc3= (56+4)/(56+26+4+2) * 100
Acc3
```

```
[1] 68.18182
```

The Champion Model between SMV and Decision Tree would be Decision Tree as it gives you better in sights of the data.

The most accurate model is Logisitic Regression Model with 68% .

```{r}
Data1$Categorized[Data1$gre >0 & Data1$gre <441] <- "LOW"
Data1$Categorized[Data1$gre >440 & Data1$gre <581] <- "MEDIUM"
Data1$Categorized[Data1$gre> 580] <- "HIGH"
```

```{r}
str(Data1)
```

```
'data.frame':   395 obs. of  10 variables:
 $ admit      : num  0 1 1 1 0 1 1 0 1 0 ...
 $ gre        : num  380 660 800 640 520 760 560 400 540 700 ...
 $ gpa        : num  3.61 3.67 4 3.19 2.93 3 2.98 3.08 3.39 3.92 ...
 $ ses        : num  1 2 2 1 3 2 2 2 1 1 ...
 $ Gender_Male: num  0 0 0 1 1 1 1 0 1 0 ...
 $ Race       : num  3 2 2 2 2 1 2 2 1 2 ...
 $ rank       : num  3 3 1 4 4 2 1 2 3 2 ...
 $ gpa_norm   : num  0.753 0.791 1 0.487 0.323 ...
 $ gre_norm   : num  0.087 0.696 1 0.652 0.391 ...
 $ Categorized: chr  "LOW" "HIGH" "HIGH" "HIGH" ...
```

Lets categorize the GPA too.

```{r}
tapply(Data1$gpa , INDEX = Data1$Categorized , FUN = mean)

```

```
    HIGH      LOW   MEDIUM 
3.521168 3.165349 3.305290 
```

Lets add the Columns:

```{r}
Data1$Mgpa[Data1$Categorized == "HIGH"] <- 3.52
Data1$Mgpa[Data1$Categorized == "MEDIUM"] <- 3.30
Data1$Mgpa[Data1$Categorized == "LOW"] <- 3.16
```

```{r}
head(Data1)
```

	admit <dbl>	gre <dbl>	gpa <dbl>	ses <dbl>	Gender_Male <dbl>	Race <dbl>	rank <dbl>	gpa_norm <dbl>
1	0	380	3.61	1	0	3	3	0.7531646
2	1	660	3.67	2	0	2	3	0.7911392
3	1	800	4.00	2	0	2	1	1.0000000
4	1	640	3.19	1	1	2	4	0.4873418
5	0	520	2.93	3	1	2	4	0.3227848
6	1	760	3.00	2	1	1	2	0.3670886

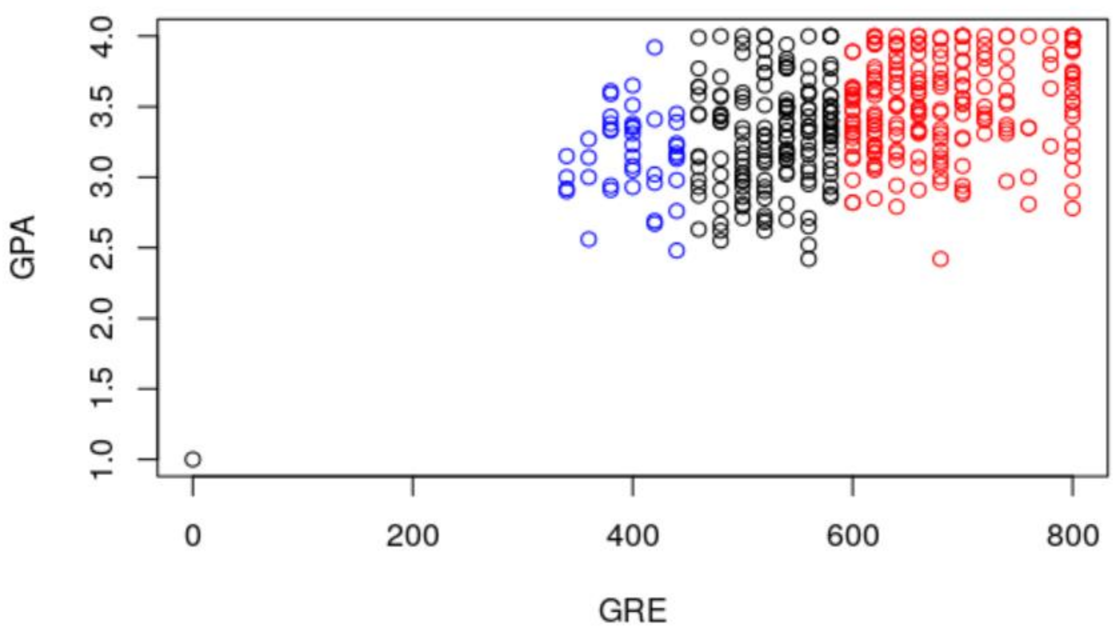
6 rows | 1-9 of 11 columns

Lets plot the point chart

```{r}
X=Data1$gre[Data1$Categorized == "HIGH"]
Y=Data1$gpa[Data1$Categorized == "HIGH"]
X1=Data1$gre[Data1$Categorized == "MEDIUM"]
Y1=Data1$gpa[Data1$Categorized == "MEDIUM"]
X2=Data1$gre[Data1$Categorized == "LOW"]
Y2=Data1$gpa[Data1$Categorized == "LOW"]
```

Create a blank space

```{r}
plot(c(0,800),c(1,4), xlab="GRE", ylab="GPA")
points(X, Y, col = "red")
points(X1,Y1,col="black")
points(X2,Y2,col="blue")
```



Conclusion:

1. The major factors that affect the admission of the student are rank and gpa.
2. Decision tree is the champion model .