SCREENSHOTS

Dataset:

	admit	gre	gpa	ses	Gender_Male	Race	rank
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
11	0	800	4.00	1	1	1	4
12	0	440	3.22	3	0	2	1
13	1	760	4.00	3	1	2	1
14	0	700	3.08	2	0	2	2
15	1	700	4.00	2	1	1	1
16	0	480	3.44	3	0	1	3
17	0	780	3.87	2	0	3	4
18	0	360	2.56	3	1	3	3
19	0	800	3.75	1	1	3	2
20	1	540	3.81	1	0	3	1
21	0	500	3.17	3	0	2	3

Head ()

```
> head(data_file)
  admit gre  gpa ses Gender_Male Race rank
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
>
```

Tail()

```
> tail(data_file)
    admit gre  gpa ses Gender_Male Race rank
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
```

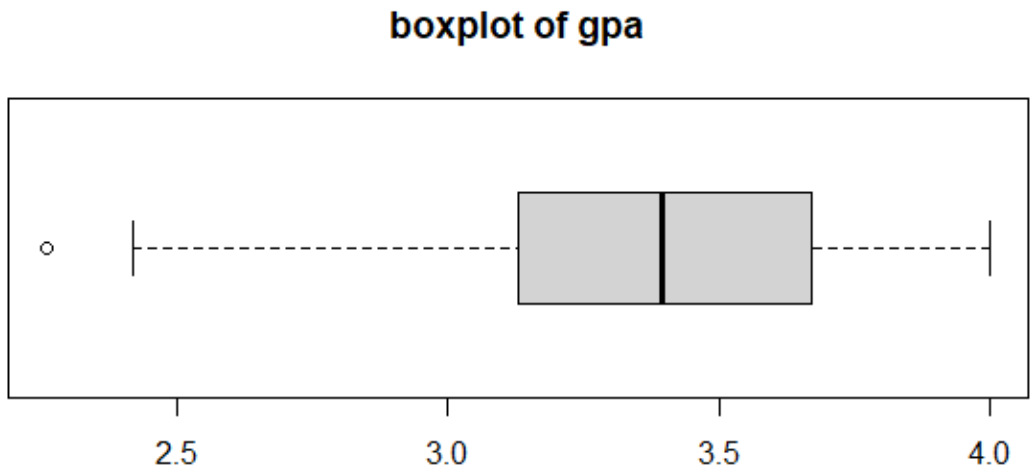
Str()

```
> str(data_file)
'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 ...
>
```

Dim(), class(), summary()

```
> class(data_file)
[1] "data.frame"
> dim(data_file)
[1] 400   7
> summary(data_file)
     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
>
```

Outlier detection using box plot for gre and gpa


boxplot of gpa
2.5
3.0
3.5
4.0


boxplot of gre
200
300
400
500
600
700
800

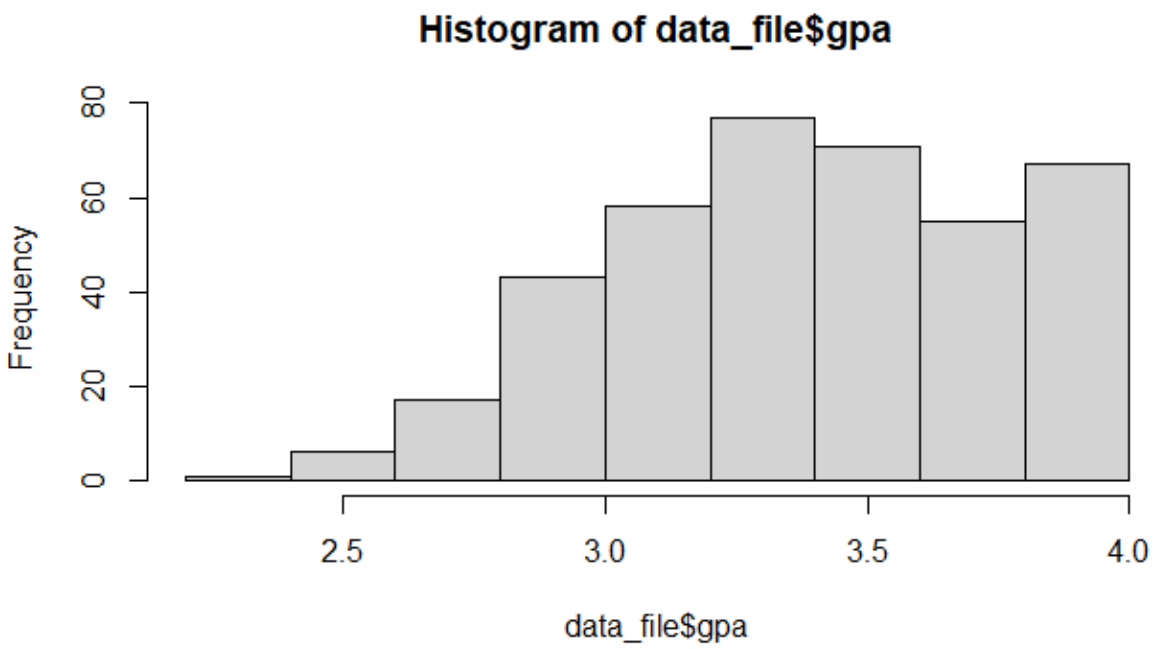
Outlier removal

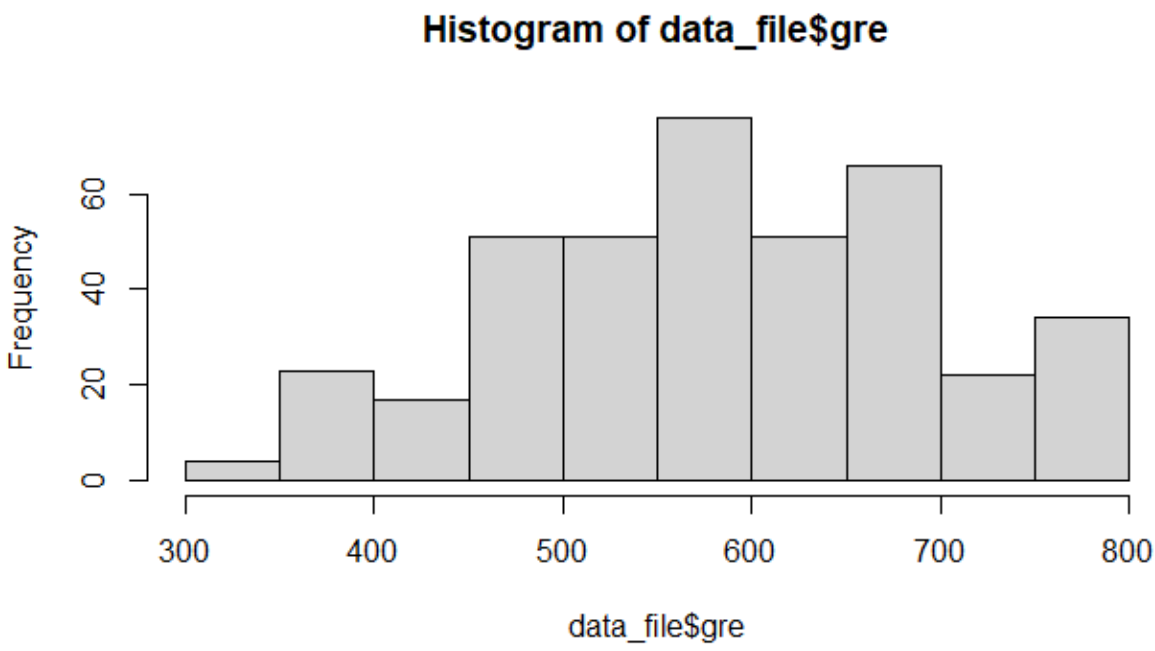
```
> data_file[out_ind,]
    admit gre  gpa ses Gender_Male Race rank
72      0 300 2.92   1           1    1    4
180     0 300 3.01   2           0    1    3
305     0 220 2.83   1           1    3    3
316     1 300 2.84   3           1    1    2
> data_file <- data_file[-c(72,180,305,316),] #dropping the rows with outliers
> dim(data_file) #now we have 396 records
[1] 396   7
> View(data_file)
> #performing similar operation for gpa
> out2 <- boxplot.stats(data_file$gpa)$out
> out_ind2 <- which(data_file$gpa %in% c(out2))
> out_ind2
[1] 288
> data_file[out_ind2,]
    admit gre  gpa ses Gender_Male Race rank
290     0 420 2.26   2           1    2    4
> data_file <- data_file[-c(290),]
> dim(data_file)
[1] 395   7
>
```

Variable datatype conversion

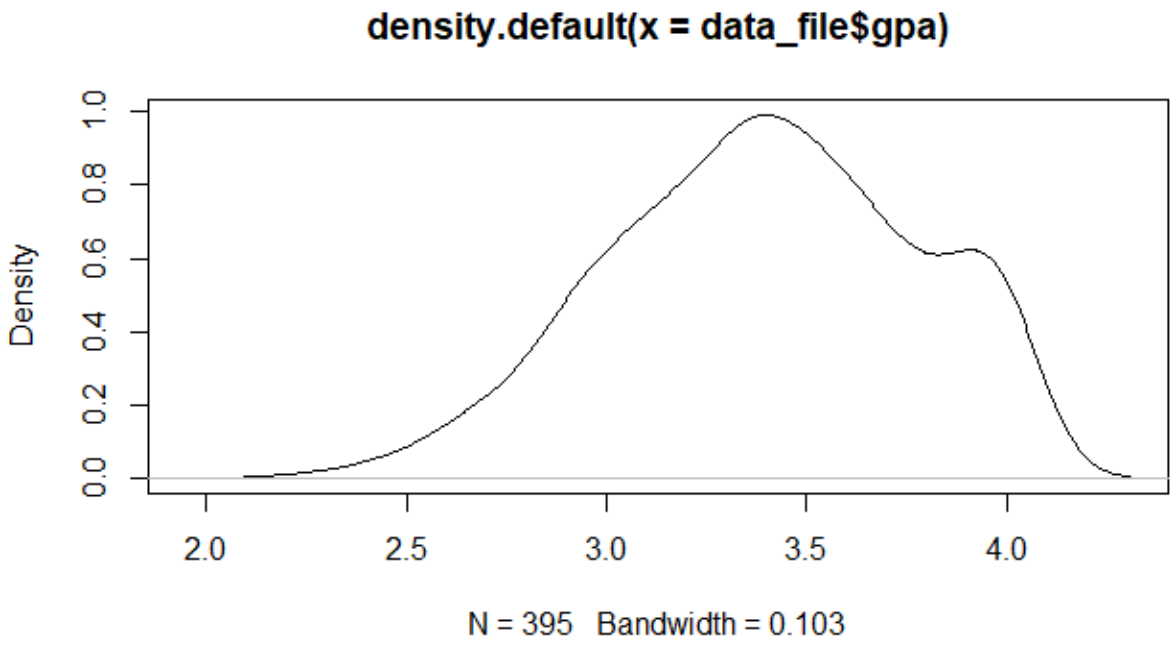
```
> str(data_file)
'data.frame':   395 obs. of  7 variables:
 $ admit      : Factor w/ 2 levels "0","1": 1 2 2 2 1 2 2 1 2 1 ...
 $ gre        : int  380 660 800 640 520 760 560 400 540 700 ...
 $ gpa        : num  3.61 3.67 4 3.19 2.93 3 2.98 3.08 3.39 3.92 ...
 $ ses        : Factor w/ 3 levels "1","2","3": 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 ...
>
```

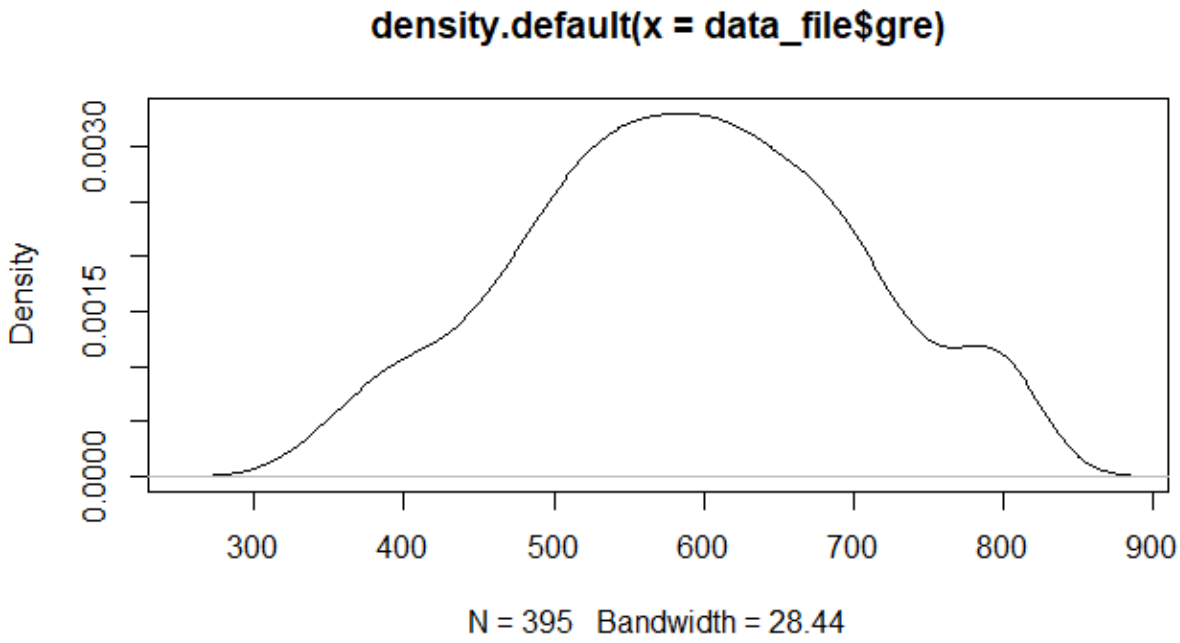
Using histogram to check if data is normally distributed in gre and gpa


Histogram of data_file$gpa
Frequency
0
20
40
60
80
2.5
3.0
3.5
4.0
data_file$gpa


Histogram of data_file$gre
Frequency
0
20
40
60
300
400
500
600
700
800
data_file$gre

Using density plot to check the normalilty of gre and gpa

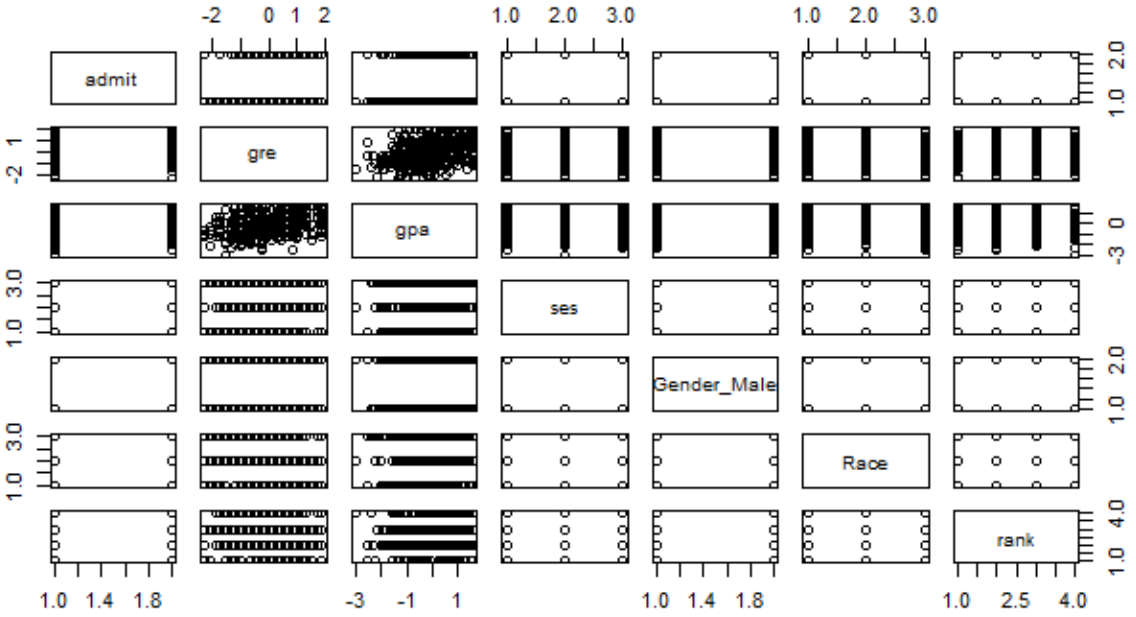




Scaling gre and gpa

```
> data_file2 <- data_file
> data_file$gre <- scale(data_file$gre, center = T, scale = T)
> data_file$gpa <- scale(data_file$gpa, center = T, scale = T)
> head(data_file)
  admit        gre        gpa ses Gender_Male Race rank
1     0 -1.8869146  0.5644567   1           0    3    3
2     1  0.6256397  0.7230157   2           0    2    3
3     1  1.8819168  1.5950902   2           0    2    1
4     1  0.4461715 -0.5454564   1           1    2    4
5     0 -0.6306375 -1.2325454   3           1    2    4
6     1  1.5229805 -1.0475599   2           1    1    2
> |
```

Plotting datafile



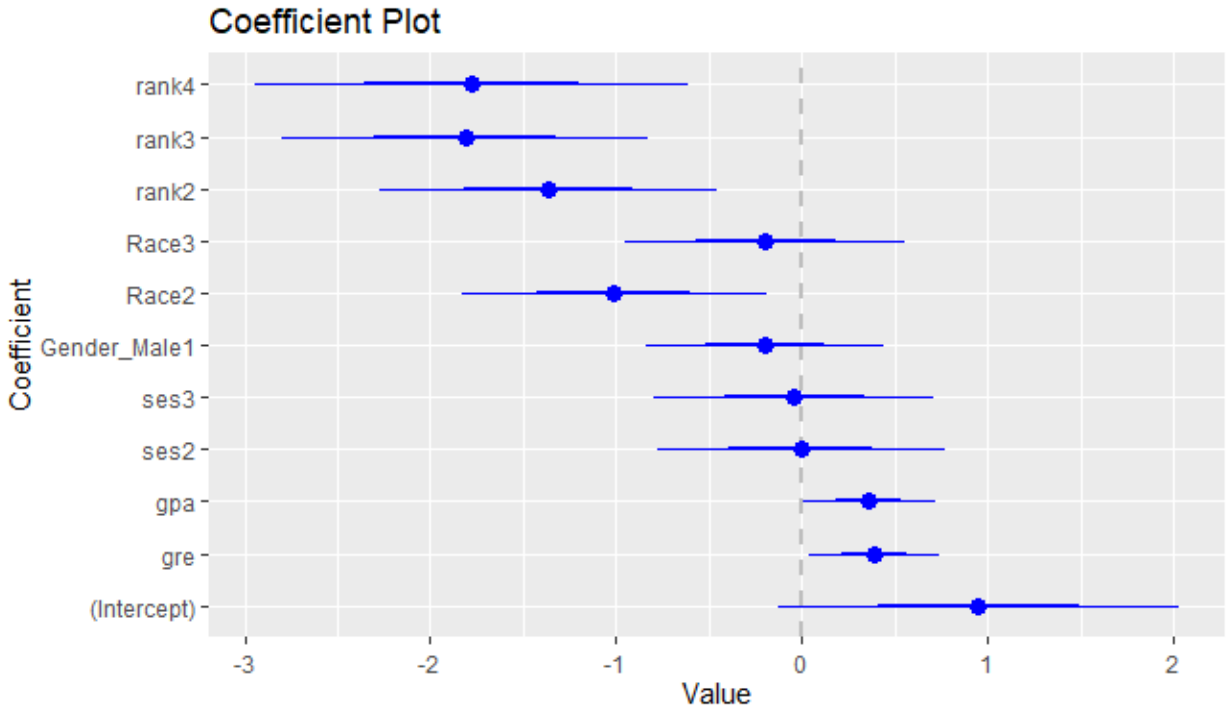
Splitting data into train and test set

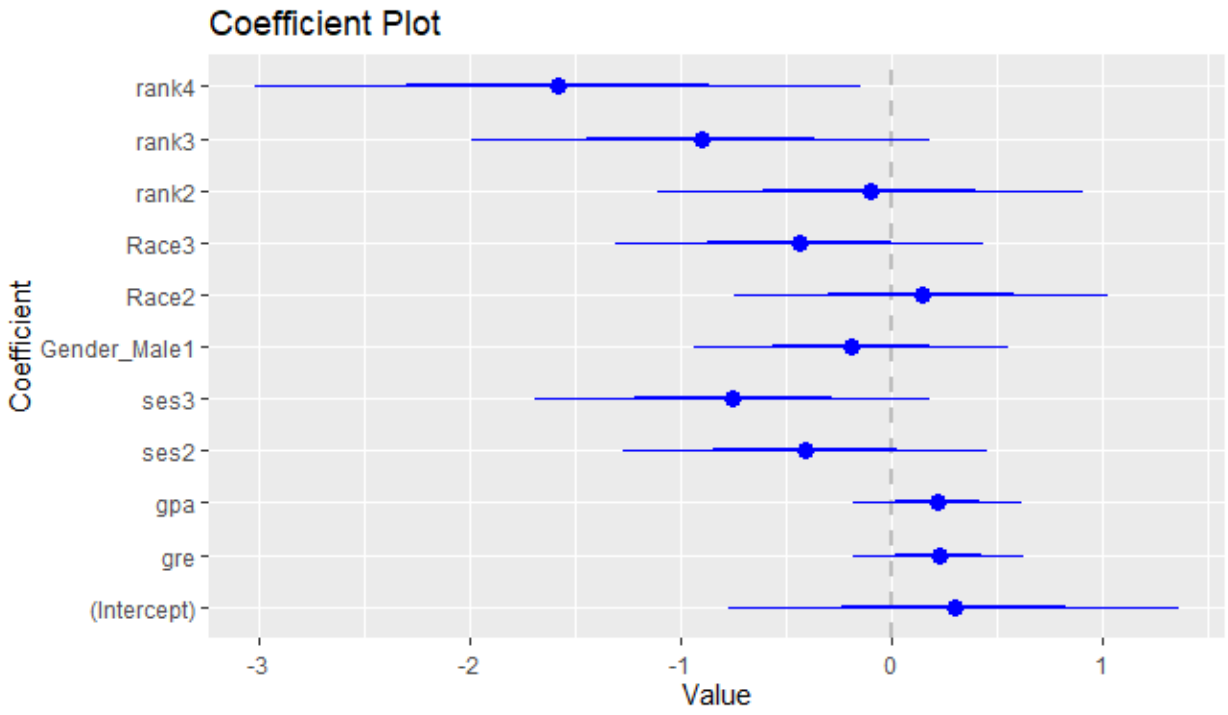
```
> ###splitting data set ###
> set.seed(123)
> indices <- sample.split(data_file, SplitRatio = 0.7)
> train <- data_file[indices == T,]
> test <- data_file[indices == F,]
> View(train)
> View(test)
> dim(train)
[1] 226   7
> dim(test)
[1] 169   7
> |
```

Logistic regression summary

```
> ### building model ###
> rmodel <- glm(admit~ ., data = train, family = "binomial")
> summary(rmodel)

Call:
glm(formula = admit ~ ., family = "binomial", data = train)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-1.9828  -0.8515  -0.5780   1.0027   2.0542

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept)   0.95236    0.54008   1.763 0.077839 .
gre           0.39361    0.17635   2.232 0.025613 *
gpa           0.36140    0.17779   2.033 0.042081 *
ses2         -0.00305    0.38806  -0.008 0.993728
ses3         -0.04002    0.37740  -0.106 0.915555
Gender_Male1 -0.19684    0.31918  -0.617 0.537426
Race2        -1.01149    0.41030  -2.465 0.013691 *
Race3        -0.19340    0.37693  -0.513 0.607879
rank2        -1.36131    0.45522  -2.990 0.002786 **
rank3        -1.81060    0.49255  -3.676 0.000237 ***
rank4        -1.77824    0.58183  -3.056 0.002241 **
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 284.36  on 225  degrees of freedom
Residual deviance: 245.29  on 215  degrees of freedom
AIC: 267.29

Number of Fisher Scoring iterations: 4

> |
```

Using coefplot() to visualize regression model





Confusion matrix of logistic regression model

```
> confusionMatrix(comp,positive = "0")
Confusion Matrix and Statistics

   preds_train
     0  1
  0 76  9
  1 21 13

               Accuracy : 0.7479
                 95% CI : (0.6601, 0.823)
    No Information Rate : 0.8151
    P-Value [Acc > NIR] : 0.97411

                  Kappa : 0.3092

 Mcnemar's Test P-Value : 0.04461

            Sensitivity : 0.7835
            Specificity : 0.5909
         Pos Pred Value : 0.8941
         Neg Pred Value : 0.3824
             Prevalence : 0.8151
         Detection Rate : 0.6387
   Detection Prevalence : 0.7143
      Balanced Accuracy : 0.6872

       'Positive' Class : 0

>
```

Support vector machine summary

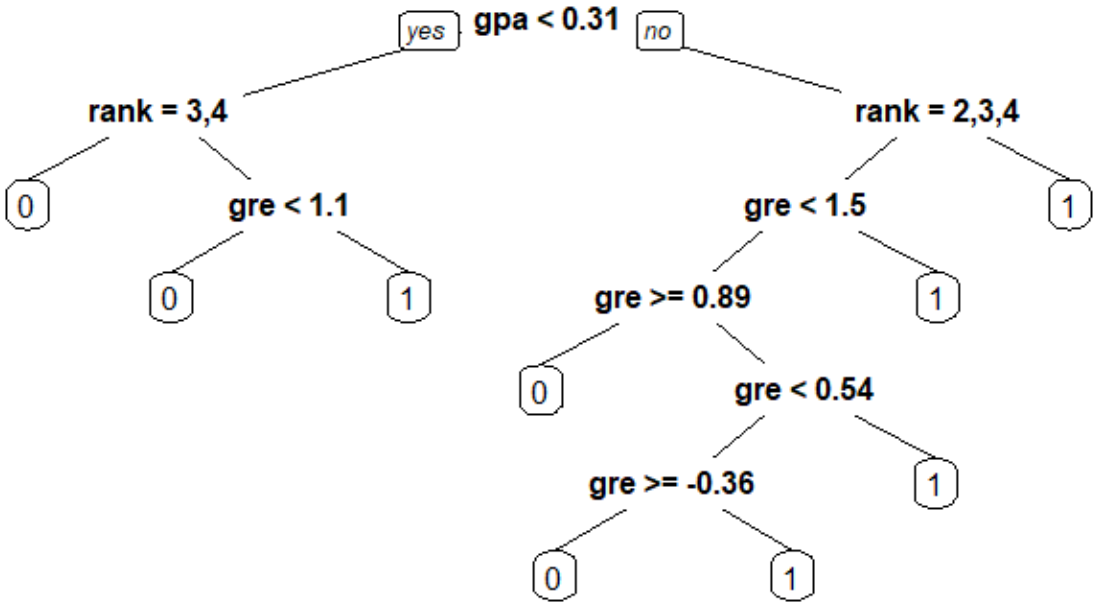
```
> model_svm = svm(admit~.,Train,kernel = "linear")
> summary(model_svm)

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


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

Number of Support Vectors:  192

 ( 101 91 )


Number of Classes:  2

Levels:
 0 1
```

Confusion matrix of SVM

```
> confusionMatrix(predictn,Test$admit)
Confusion Matrix and Statistics

          Reference
Prediction  0  1
         0 75 22
         1 10 12

               Accuracy : 0.7311
                 95% CI : (0.6421, 0.8082)
    No Information Rate : 0.7143
    P-Value [Acc > NIR] : 0.38556

                  Kappa : 0.2632

 Mcnemar's Test P-Value : 0.05183

            Sensitivity : 0.8824
            Specificity : 0.3529
         Pos Pred Value : 0.7732
         Neg Pred Value : 0.5455
             Prevalence : 0.7143
         Detection Rate : 0.6303
   Detection Prevalence : 0.8151
      Balanced Accuracy : 0.6176

       'Positive' Class : 0

>
```

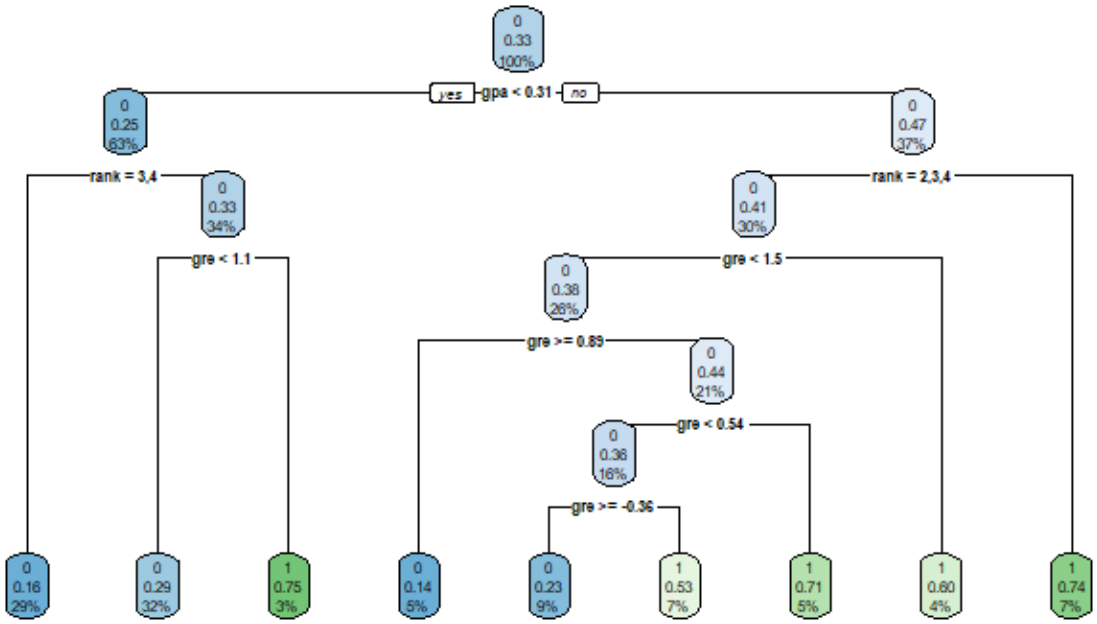
Decision tree

Using prp()



```
> confusionMatrix(pred_class, Test$admit, positive = '0')
Confusion Matrix and Statistics

          Reference
Prediction  0  1
         0 68 18
         1 17 16

               Accuracy : 0.7059
                 95% CI : (0.6154, 0.7858)
    No Information Rate : 0.7143
    P-Value [Acc > NIR] : 0.6245

                  Kappa : 0.273

 Mcnemar's Test P-Value : 1.0000

            Sensitivity : 0.8000
            Specificity : 0.4706
         Pos Pred Value : 0.7907
         Neg Pred Value : 0.4848
             Prevalence : 0.7143
         Detection Rate : 0.5714
   Detection Prevalence : 0.7227
      Balanced Accuracy : 0.6353

       'Positive' Class : 0
```

Using rpart()



Randomforest

```
               Type of random forest: classification
                     Number of trees: 30
No. of variables tried at each split: 2

        OOB estimate of  error rate: 30.8%
Confusion matrix:
    0  1 class.error
0 152 32    0.173913
1  53 39    0.576087
> tpred <- predict(rand_f,newdata = Test)
> confusionMatrix(tpred,Test$admit)
Confusion Matrix and Statistics

          Reference
Prediction  0  1
         0 70 26
         1 15  8

               Accuracy : 0.6555
                 95% CI : (0.5628, 0.7402)
    No Information Rate : 0.7143
    P-Value [Acc > NIR] : 0.9339

                  Kappa : 0.0651

 Mcnemar's Test P-Value : 0.1183

            Sensitivity : 0.8235
            Specificity : 0.2353
         Pos Pred Value : 0.7292
         Neg Pred Value : 0.3478
             Prevalence : 0.7143
         Detection Rate : 0.5882
   Detection Prevalence : 0.8067
      Balanced Accuracy : 0.5294

       'Positive' Class : 0
```