Anjuman-I-Islam's

Akbar Peerbhoy College of Commerce and Economics

M. S. Ali Road, Grant Road (East), Mumbai-8

C E R T I F I C A T E

This is to certify that the work entered in this journal is the work of Mr./Ms._____ (Roll No. / Seat No._____)in partial fulfilment for B. Sc. (Data Science) Semester-IV Degree Examination has been found satisfactory in the subject **Big Data** for the Second Year 2023-24 Year B. Sc. (Data Science) - 2024 in the College Laboratory.

Signature Lecturer- In- Charge	Signature External Examiner	Signature Course Coordinator

I N D E X

SR NO.	NAME OF EXPERIMENT	DATE	SIGNATURE
1	Implement of Decision of tree classifier		
2	Implement of SVM Classification technique		
3	Regression Model: Import a data from web storage. Name the dataset and do Logistic Regression to find out relation between variables that are affecting the admission of a student in an institute based on his or her GRE score, GPA obtained and rank of the student. Also check the model is fit or not require (foreign), require (Mass)		
4	MULTIPLE REGRESSION MODEL: Apply multiple regressions, if data have a continuous independent variable. Apply on above dataset.		
5	CLUSTERING MODEL a. Clustering algorithms for unsupervised classification. b. Plot the cluster data using R visualizations		
6	CLASSIFICATION MODEL: a. Install relevant package for classification. b. Choose classifier for classification problem. c. Evaluate the performance of classifier		

Practical No : 1: Implement of Decision of tree classifier

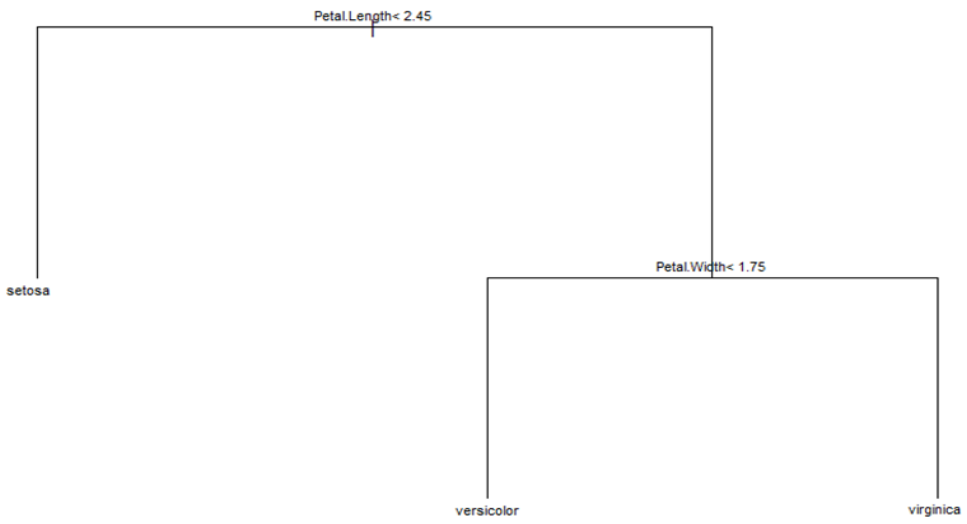
Input :

```
library(rpart)
data("iris")
head(iris)
summary(iris)
tree_model<-rpart(Species ~.,data=iris,method="class")
plot(tree_model)
prediction <-predict(tree_model,newdata=iris,type="class")
confusion_matrix<-table(prediction,iris$Species)
print(confusion_matrix)
text(tree_model,cex=0.7)
accuracy <-sum(diag((confusion_matrix)/sum(confusion_matrix)))
print(accuracy)
```

Output :

```
> library(rpart)
> data("iris")
> head(iris)
  Sepal.Length Sepal.Width Petal.Length Petal.Width Species
1          5.1         3.5          1.4         0.2  setosa
2          4.9         3.0          1.4         0.2  setosa
3          4.7         3.2          1.3         0.2  setosa
4          4.6         3.1          1.5         0.2  setosa
5          5.0         3.6          1.4         0.2  setosa
6          5.4         3.9          1.7         0.4  setosa
> summary(iris)
 Sepal.Length   Sepal.Width      Petal.Length   Petal.Width
 Min.   :4.300   Min.   :2.000   Min.   :1.000   Min.   :0.100
 1st Qu.:5.100   1st Qu.:2.800   1st Qu.:1.600   1st Qu.:0.300
 Median :5.800   Median :3.000   Median :4.350   Median :1.300
 Mean   :5.843   Mean   :3.057   Mean   :3.758   Mean   :1.199
 3rd Qu.:6.400   3rd Qu.:3.300   3rd Qu.:5.100   3rd Qu.:1.800
```

```
 Max.   :7.900   Max.   :4.400   Max.   :6.900   Max.   :2.500
       Species
 setosa   :50
 versicolor:50
 virginica :50
> tree_model<-rpart(Species ~.,data=iris,method="class")
> plot(tree_model)
> prediction <-predict(tree_model,newdata=iris,type="class")
> confusion_matrix<-table(prediction,iris$Species)
> print(confusion_matrix)
prediction   setosa versicolor virginica
  setosa         50         0         0
  versicolor      0        49         5
  virginica       0         1        45
> text(tree_model,cex=0.7)
> accuracy <-sum(diag((confusion_matrix)/sum(confusion_matrix)))
> print(accuracy)
[1] 0.96
```



Practical No 2 : Implement of SVM Classification technique

Input :

```
library(e1071)
data("iris")
head(iris)
summary(iris)
svm_model<-svm(Species ~.,data=iris,kernel="radial")
prediction <-predict(svm_model,iris[,-5])
confusion_matrix<-table(prediction,iris$Species)
print(confusion_matrix)
text(svm_model,cex=0.7)
accuracy <-sum(diag((confusion_matrix)/sum(confusion_matrix)))
print(accuracy)
plot(svm_model,data=iris,Petal.Width~Petal.Length,slice  =list(Sepal.Width=3,Sepal.Length=4))
```

Output :

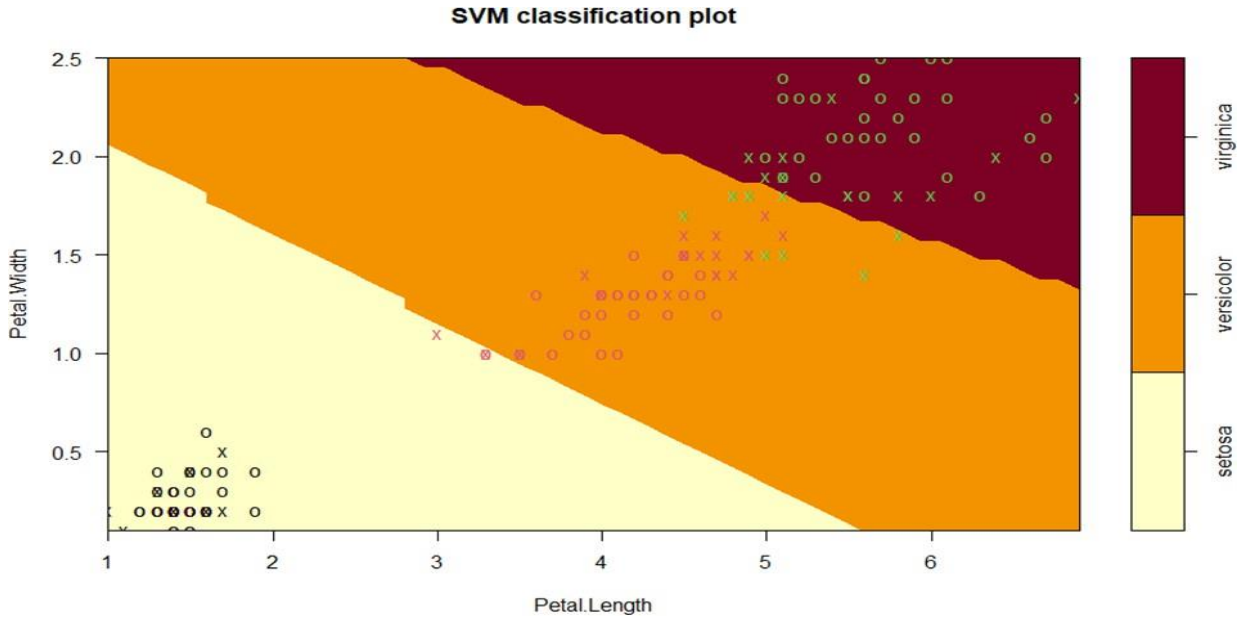
```
> library(e1071)
> data("iris")
> head(iris)
  Sepal.Length Sepal.Width Petal.Length Petal.Width Species
1          5.1         3.5          1.4         0.2  setosa
2          4.9         3.0          1.4         0.2  setosa
3          4.7         3.2          1.3         0.2  setosa
4          4.6         3.1          1.5         0.2  setosa
5          5.0         3.6          1.4         0.2  setosa
6          5.4         3.9          1.7         0.4  setosa
> summary(iris)
  Sepal.Length   Sepal.Width      Petal.Length   Petal.Width
 Min.   :4.300   Min.   :2.000   Min.   :1.000   Min.   :0.100
 1st Qu.:5.100   1st Qu.:2.800   1st Qu.:1.600   1st Qu.:0.300
 Median :5.800   Median :3.000   Median :4.350   Median :1.300
 Mean   :5.843   Mean   :3.057   Mean   :3.758   Mean   :1.199
 3rd Qu.:6.400   3rd Qu.:3.300   3rd Qu.:5.100   3rd Qu.:1.800
```

```
 Max.   :7.900   Max.   :4.400   Max.   :6.900   Max.   :2.500
       Species
 setosa   :50
 versicolor:50
 virginica :50
```

```
> svm_model<-svm(Species ~.,data=iris,kernel="radial")
> prediction  <-predict(svm_model,iris[,-5])
> confusion_matrix<-table(prediction,iris$Species)
> print(confusion_matrix)
```

```
prediction   setosa versicolor virginica
  setosa         50         0         0
  versicolor      0        48         2
  virginica       0         2        48
```

```
> text(svm_model,cex=0.7)
>accuracy <-sum(diag((confusion_matrix)/sum(confusion_matrix)))
> print(accuracy)
[1] 0.9733333
> plot(svm_model,data=iris,Petal.Width~Petal.Length,slice  =list(Sepal.Width=3,Sepal.Length=4))
```



Practical No : 3: Regression Model: Import a data from web storage.

Name the dataset and do Logistic Regression to find out relation between variables that are affecting the admission of a student in an institute based on his or her GRE score, GPA obtained and rank of the student. Also check the model is fit or not require (foreign), require (Mass)

Input :

```
install.packages("MASS")

library(MASS)

admissions <-read.csv("C:/Users/STD/Desktop/Admission.csv")

str(admissions)

log_model <-glm(admit ~gre + gpa + rank, data = admissions,family = binomial)

summary(log_model)

result<-shapiro.test(residuals(log_model))

print(result)

plot(log_model)
```

Output :

```
>library(MASS)

> admissions <-read.csv("C:/Users/STD/Desktop/Admission.csv")

> str(admissions)

'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 …

> log_model <-glm(admit ~gre + gpa + rank, data = admissions,family = binomial)

> summary(log_model)

Call:

glm(formula = admit ~ gre + gpa + rank, family = binomial, data = admissions)

Coefficients:

             Estimate Std. Error z value Pr(>|z|)

(Intercept) -3.449548   1.132846  -3.045  0.00233 **

gre          0.002294   0.001092   2.101  0.03564 *
```

gpa 0.777014 0.327484 2.373 0.01766 *

rank -0.560031 0.127137 -4.405 1.06e-05 ***

Signif. codes: 0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

Null deviance: 499.98 on 399 degrees of freedom

Residual deviance: 459.44 on 396 degrees of freedom

AIC: 467.44

Number of Fisher Scoring iterations: 4

> result<-shapiro.test(residuals(log_model))

> print(result)

Shapiro-Wilk normality test

data: residuals(log_model)

W = 0.79688, p-value < 2.2e-16

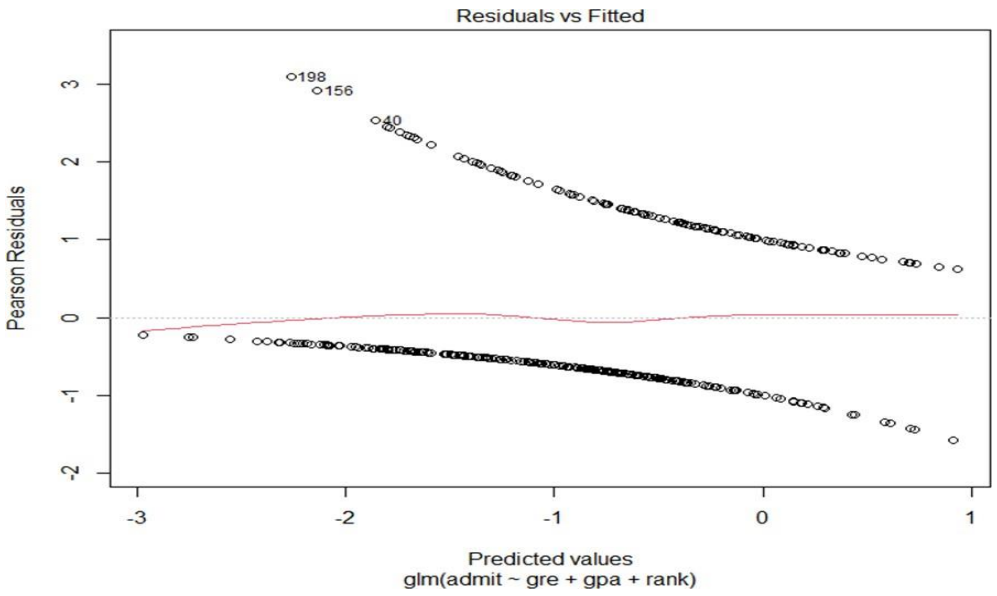
> plot(log_model)

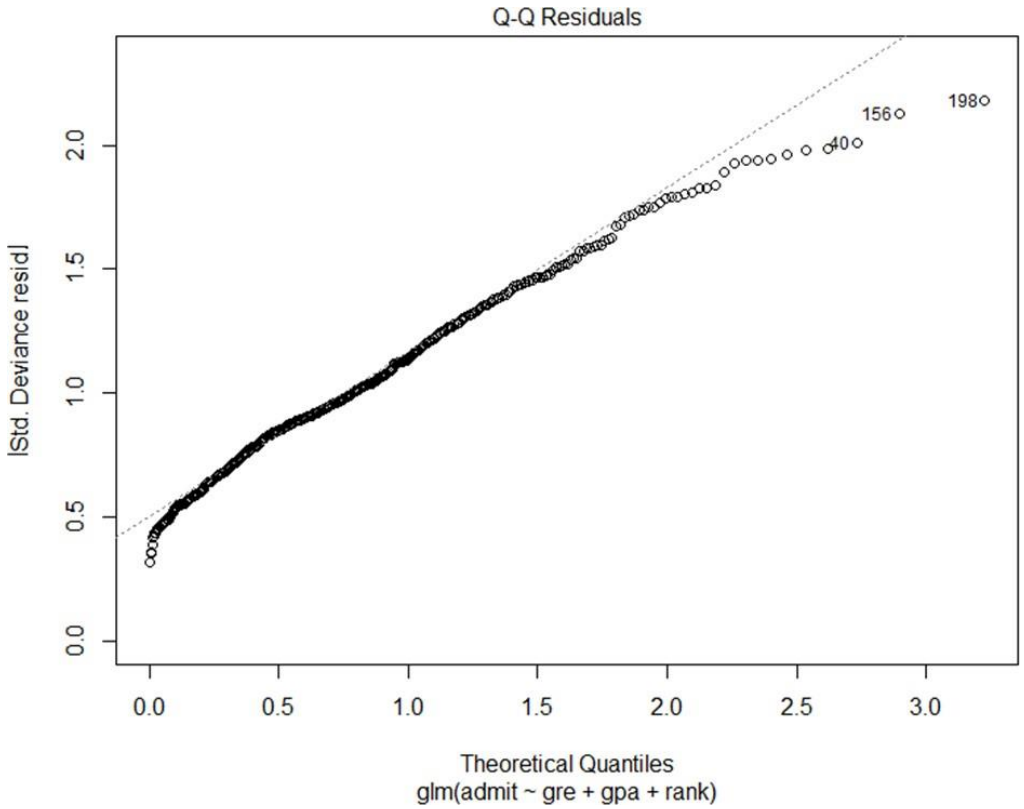
Hit <Return> to see next plot:

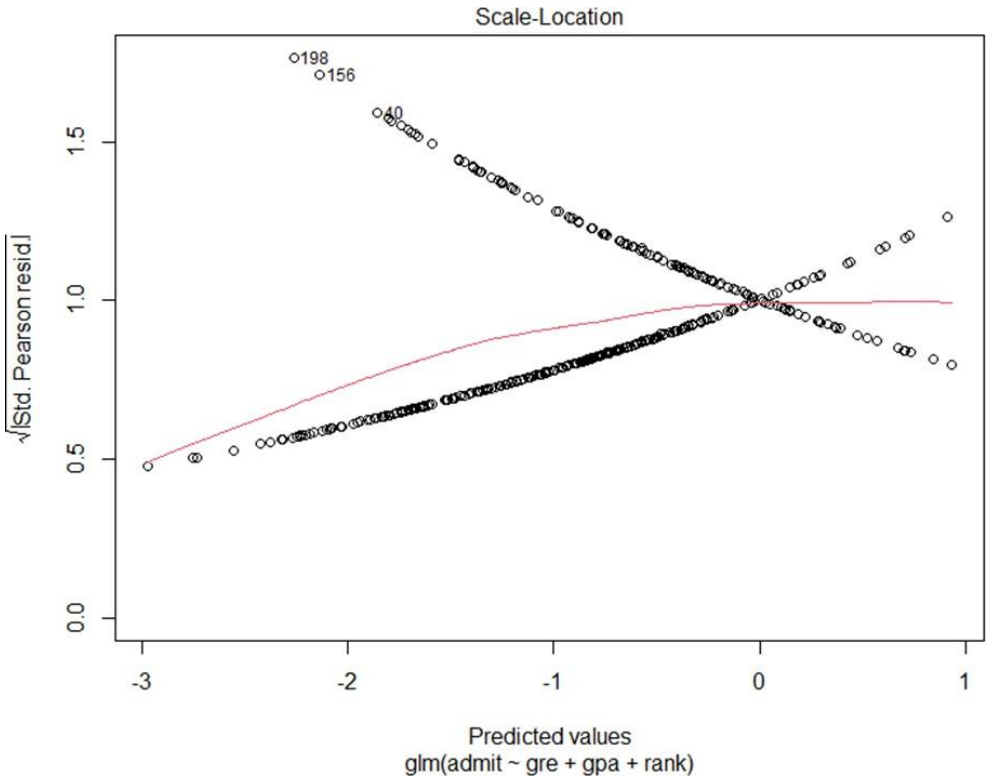
Hit <Return> to see next plot:

Hit <Return> to see next plot:

Hit <Return> to see next plot:







Practical No : 4: MULTIPLE REGRESSION MODEL: Apply multiple regressions, if data have a

continuous independent variable. Apply on above dataset.

Input :

```
admissions <-read.csv("C:/Users/STD/Desktop/Admission.csv")
str(admissions)
mul_model <-lm(admit ~gre + gpa + rank,admissions)
summary(mul_model)
plot(admissions$admit,fitted(mul_model),xlab="Actual",ylab="predicted",main="Actual vs predict")
plot(mul_model)
plot(mul_model$fitted.values)
```

Output :

```
> admissions <-read.csv("C:/Users/STD/Desktop/Admission.csv")
> str(admissions)
'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 …
> mul_model <-lm(admit ~gre + gpa + rank,admissions)
> summary(mul_model)

Call:
lm(formula = admit ~ gre + gpa + rank, data = admissions)
Residuals:
    Min      1Q  Median      3Q     Max
-0.6617 -0.3417 -0.1947  0.5061  0.9556
Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) -0.1824127  0.2169695  -0.841   0.4010
```

gre 0.0004424 0.0002101 2.106 0.0358 *

gpa 0.1510402 0.0633854 2.383 0.0176 *

rank -0.1095019 0.0237617 -4.608 5.48e-06 ***

Signif. codes: 0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.4448 on 396 degrees of freedom

Multiple R-squared: 0.09601, Adjusted R-squared: 0.08916

F-statistic: 14.02 on 3 and 396 DF, p-value: 1.054e-08

> plot(admissions$admit,fitted(mul_model),xlab="Actual",ylab="predicted",main="Actual vs predict")

> plot(mul_model)

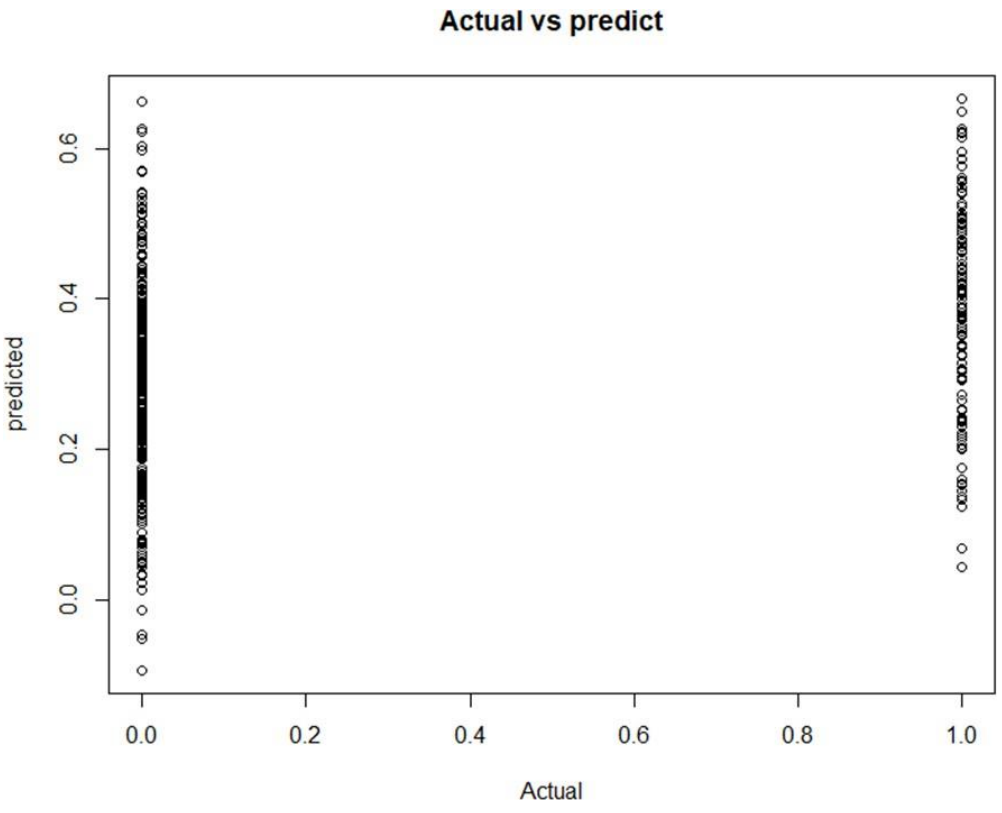
Hit <Return> to see next plot:

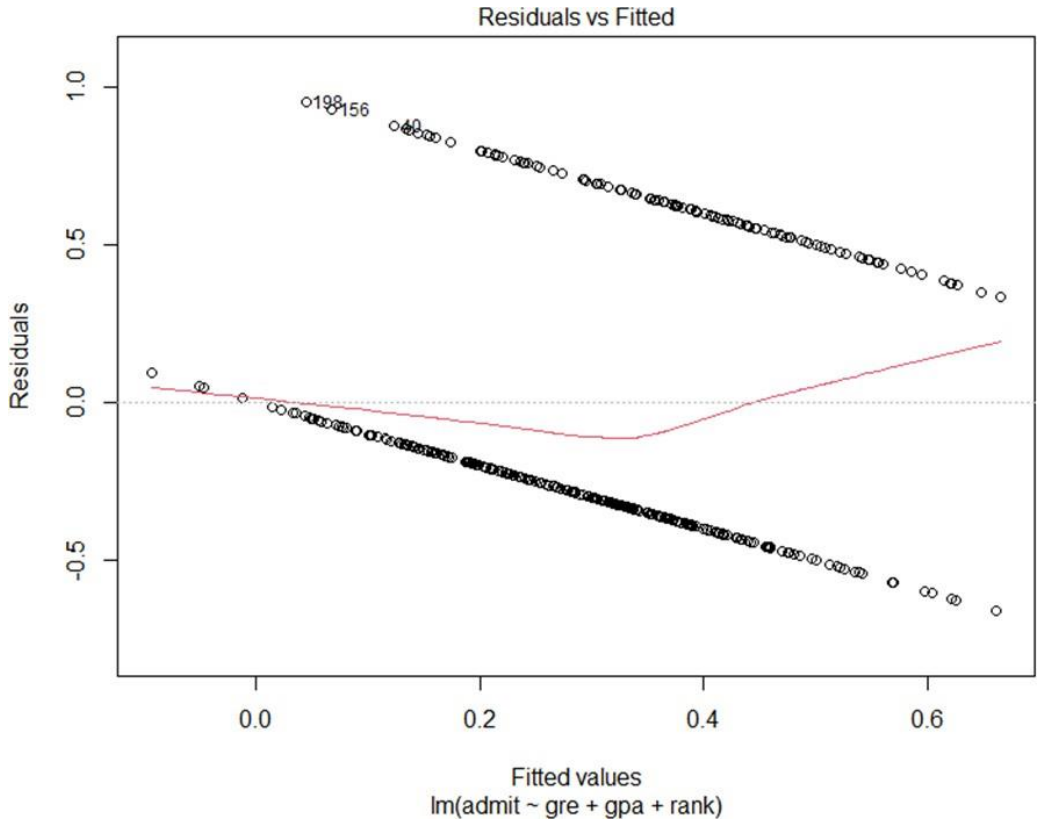
Hit <Return> to see next plot:

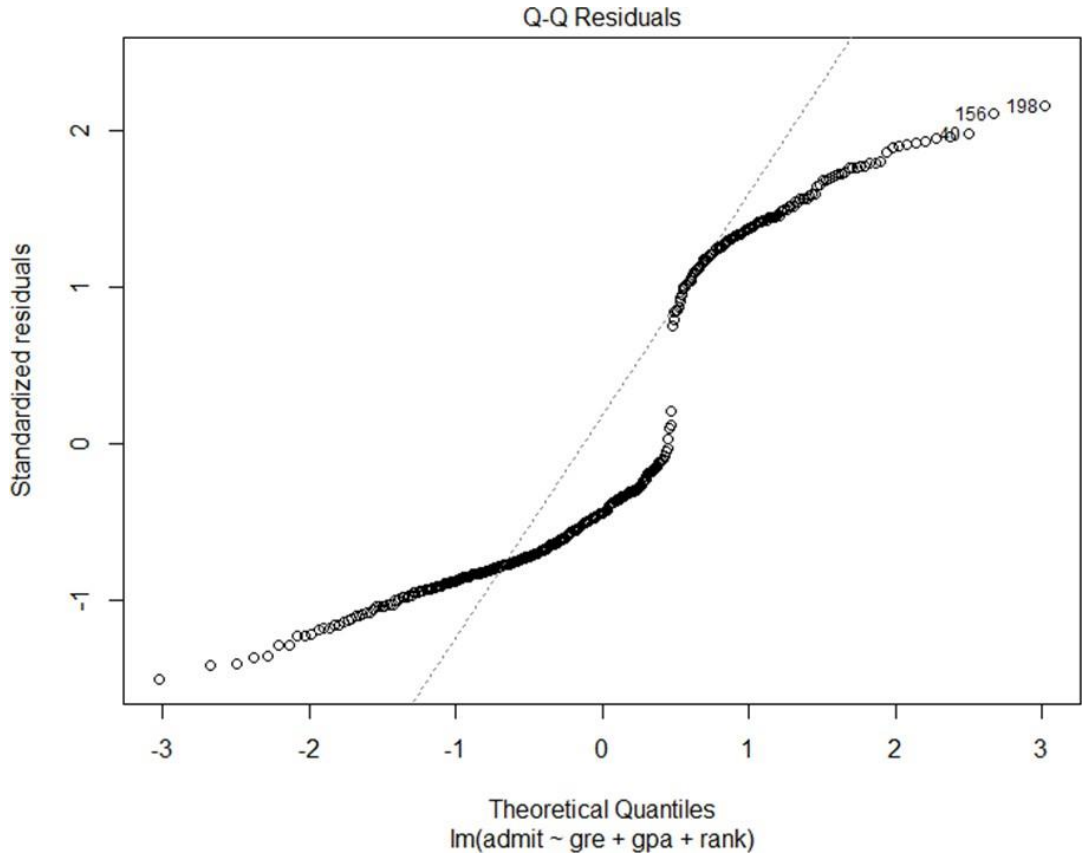
Hit <Return> to see next plot:

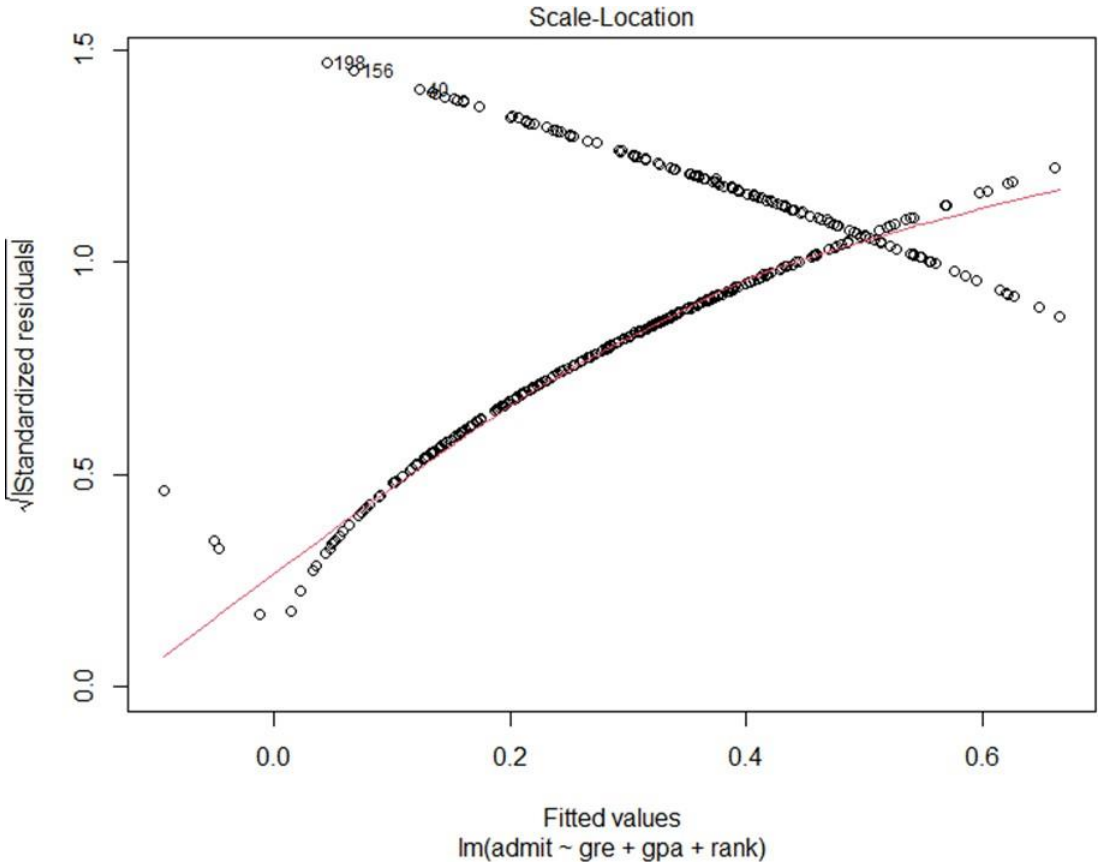
Hit <Return> to see next plot:

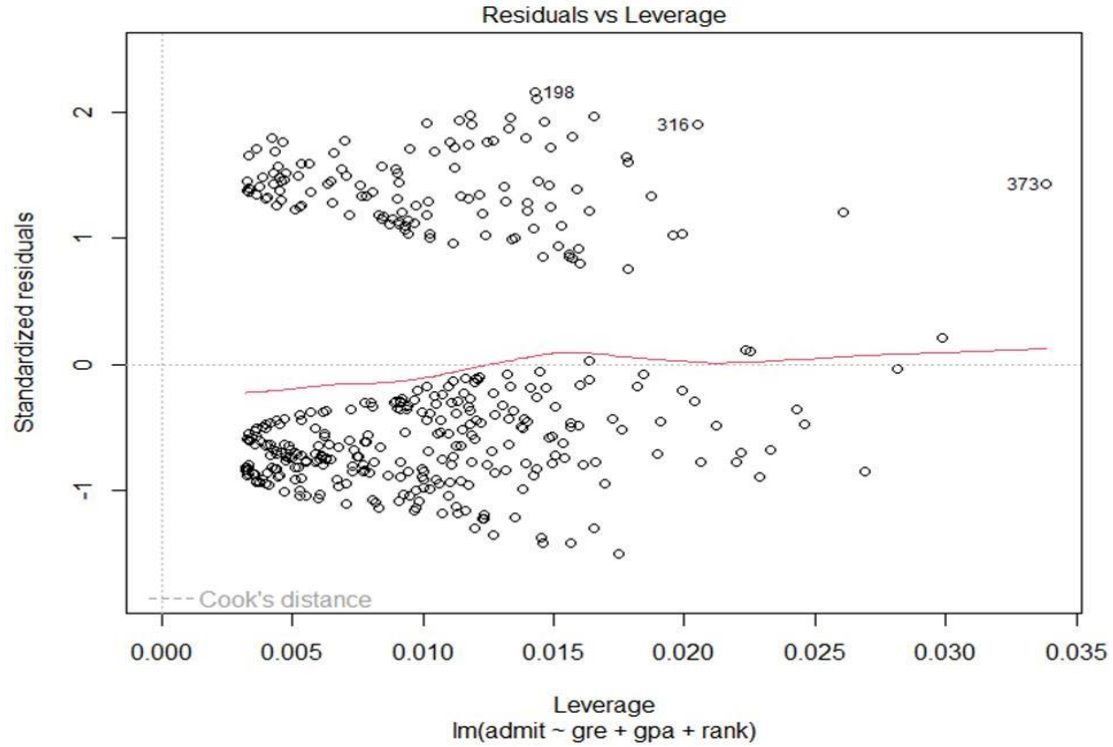
> plot(mul_model$fitted.values)

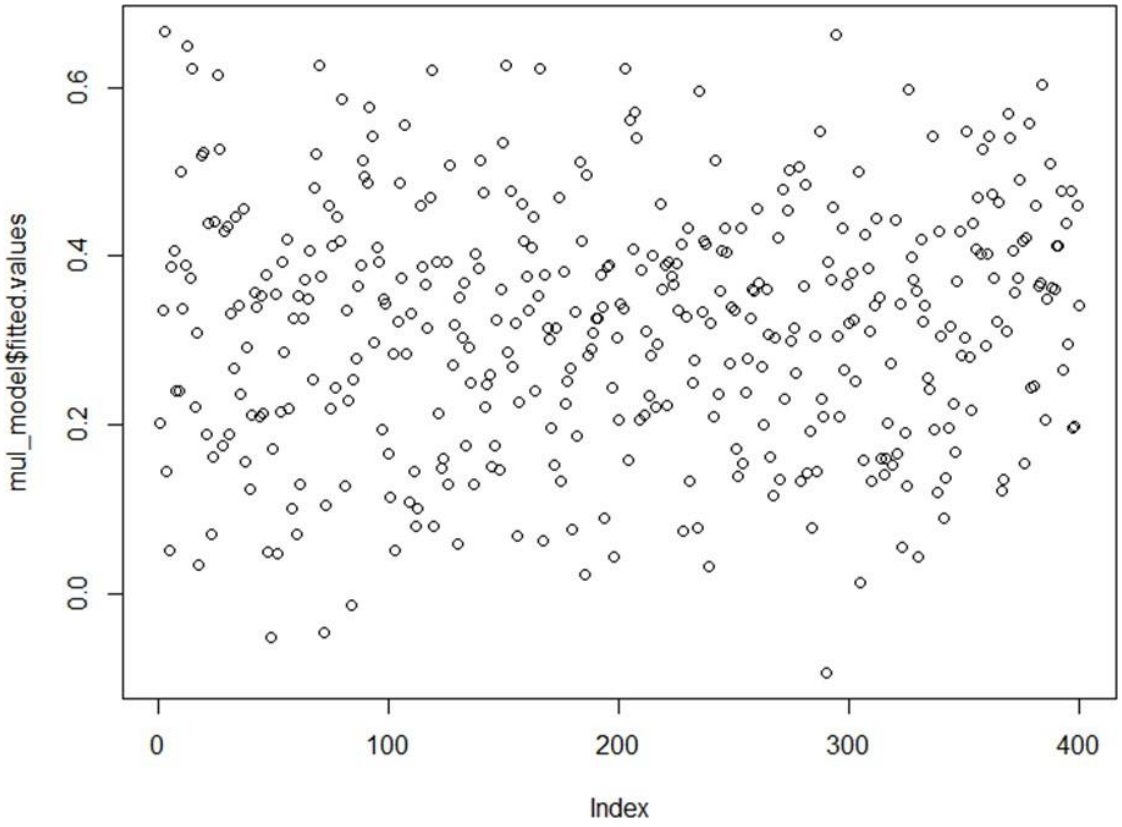



Residuals vs Fitted
Residuals
Fitted values
lm(admit ~ gre + gpa + rank)
198
156
40
1.0
0.5
0.0
-0.5
0.0
0.2
0.4
0.6


Q-Q Residuals
Standardized residuals
Theoretical Quantiles
lm(admit ~ gre + gpa + rank)
156
198
40
2
1
0
-1
-3
-2
-1
0
1
2
3


Scale-Location
198
156
40
√|Standardized residuals|
0.0
0.5
1.0
1.5
0.0
0.2
0.4
0.6
Fitted values
lm(admit ~ gre + gpa + rank)


Residuals vs Leverage
198
316
373
Standardized residuals
2
1
0
-1
Cook's distance
0.000
0.005
0.010
0.015
0.020
0.025
0.030
0.035
Leverage
lm(admit ~ gre + gpa + rank)


mul_model$fitted.values
0.0
0.2
0.4
0.6
0
100
200
300
400
Index

Practical No : 5: CLUSTERING MODEL

Clustering algorithms for unsupervised classification. b. Plot the cluster data using R visualizations

Input :

```
data("iris")
head(iris)
summary(iris)
x=iris[,3:4]
head(x)
model=kmeans(x,3)
library(cluster)
clusplot(x,model$cluster)
clusplot(x,model$cluster,color=T,shade=T)
```

Output :

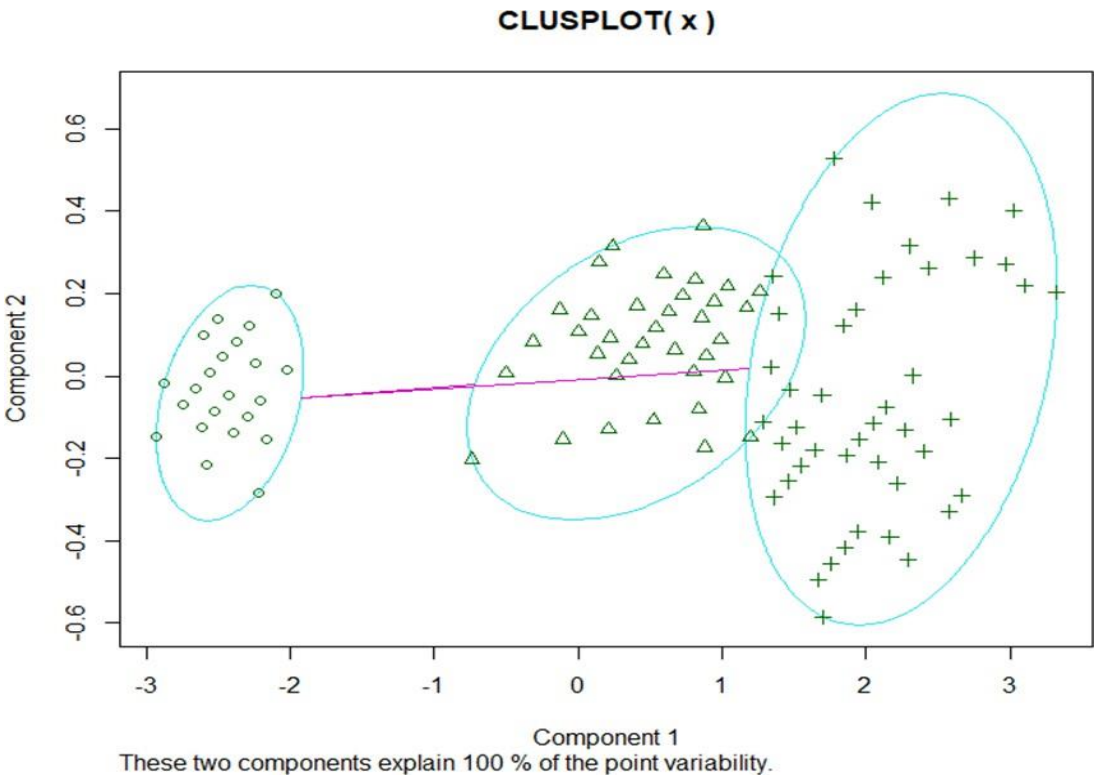
```
>data("iris")
> head(iris)
  Sepal.Length Sepal.Width Petal.Length Petal.Width Species
1          5.1         3.5          1.4         0.2  setosa
2          4.9         3.0          1.4         0.2  setosa
3          4.7         3.2          1.3         0.2  setosa
4          4.6         3.1          1.5         0.2  setosa
5          5.0         3.6          1.4         0.2  setosa
6          5.4         3.9          1.7         0.4  setosa
> summary(iris)
  Sepal.Length   Sepal.Width      Petal.Length   Petal.Width
 Min.  :4.300   Min.  :2.000   Min.  :1.000   Min.  :0.100
 1st Qu.:5.100   1st Qu.:2.800   1st Qu.:1.600   1st Qu.:0.300
 Median :5.800   Median :3.000   Median :4.350   Median :1.300
 Mean  :5.843   Mean  :3.057   Mean  :3.758   Mean  :1.199
 3rd Qu.:6.400  3rd Qu.:3.300  3rd Qu.:5.100  3rd Qu.:1.800
 Max.  :7.900   Max.  :4.400   Max.  :6.900   Max.  :2.500
       Species
```

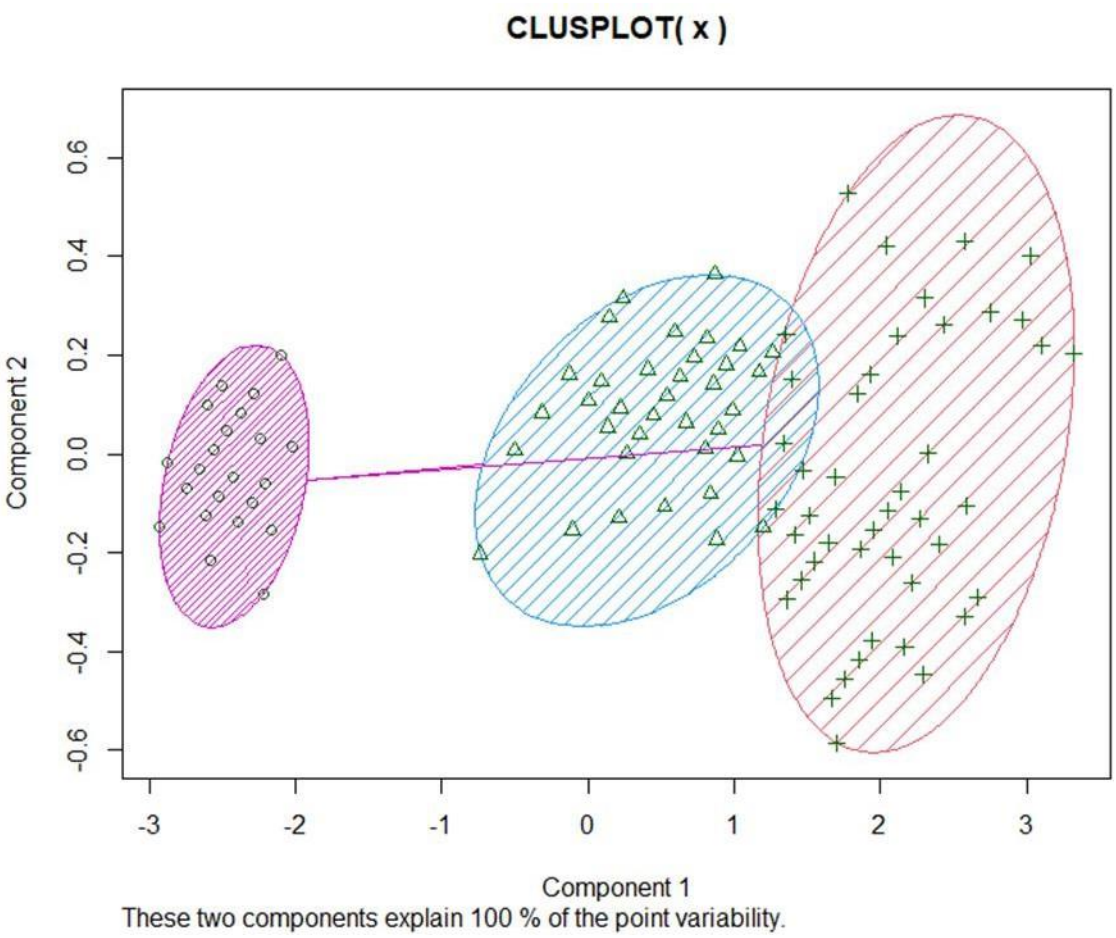
setosa :50

versicolor:50

virginica :50

```
> x=iris[,3:4]
> head(x)
  Petal.Length Petal.Width
1          1.4         0.2
2          1.4         0.2
3          1.3         0.2
4          1.5         0.2
5          1.4         0.2
6          1.7         0.4
> model=kmeans(x,3)
> library(cluster)
> clusplot(x,model$cluster)
> clusplot(x,model$cluster,color=T,shade=T)
```




CLUSPLOT(x)
Component 2
0.6
0.4
0.2
0.0
-0.2
-0.4
-0.6
-3
-2
-1
0
1
2
3
Component 1
These two components explain 100 % of the point variability.

Practical No : 6: CLASSIFICATION MODEL:

a. Install relevant package for classification.

b. Choose classifier for classification problem.

c. Evaluate the performance of classifier

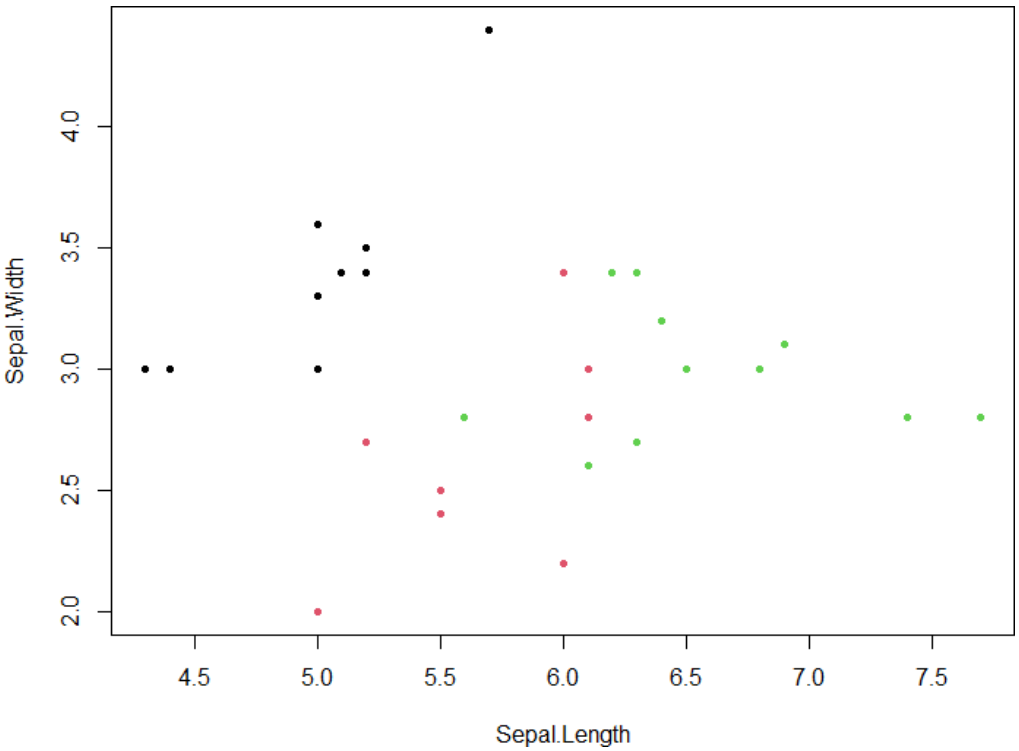
Input :

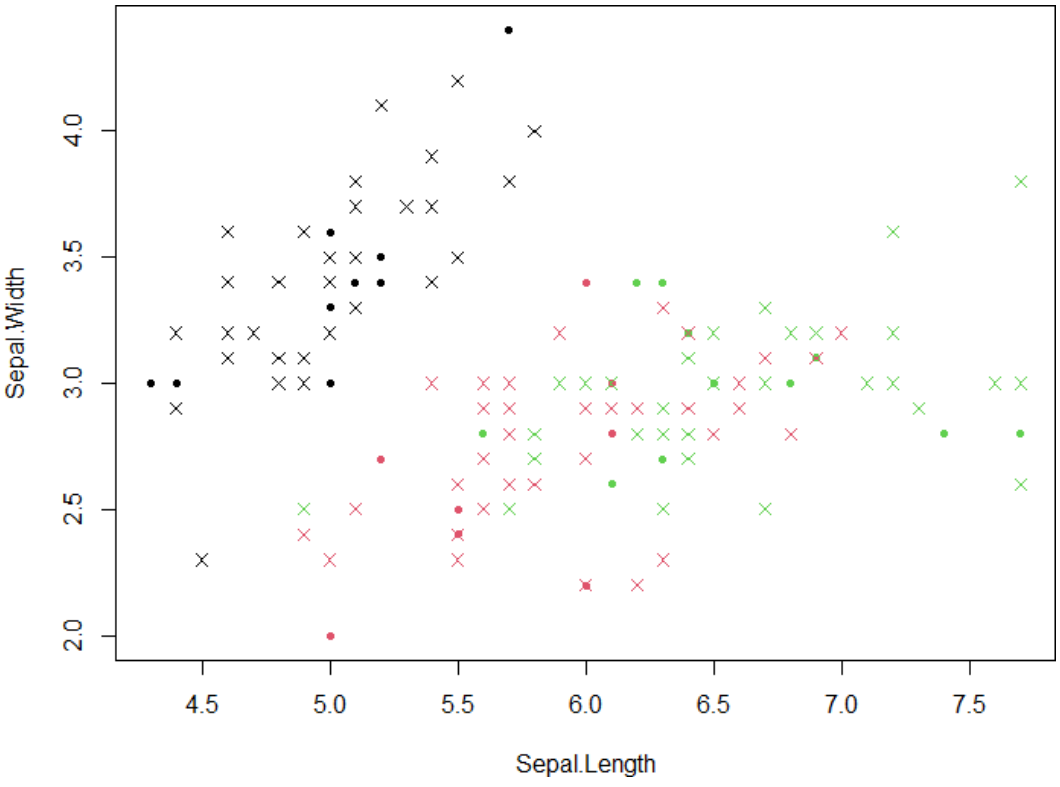
```
library("caret")
library("caTools")
data("iris")
dim(iris)
summary(iris)
head(iris)
set.seed(1234)
random <- runif(150)
iris_random <- iris[order(random),]
head(iris_random)
train <- iris_random[1:120,]
test <- iris_random[121:150,]
train_sp <- iris_random[1:120,5]  # Use 120 samples for training
test_sp <- iris_random[121:150,5]  # Use 30 samples for testing
require(class)

# Convert train_sp and test_sp to factors
train_sp <- as.factor(train_sp)
test_sp <- as.factor(test_sp)
# Train the kNN model
model <- knn(train = train[, -5], test = test[, -5], cl = train_sp, k = 5)  # Use k = 5 for example
print(model)
# Plotting the classification results
plot(test[, c(1, 2)], col = model, pch = 20)
points(train[, c(1, 2)], col = train_sp, pch = 4)
legend("topright", legend = levels(train_sp), col = 1:3, pch = 4)
```

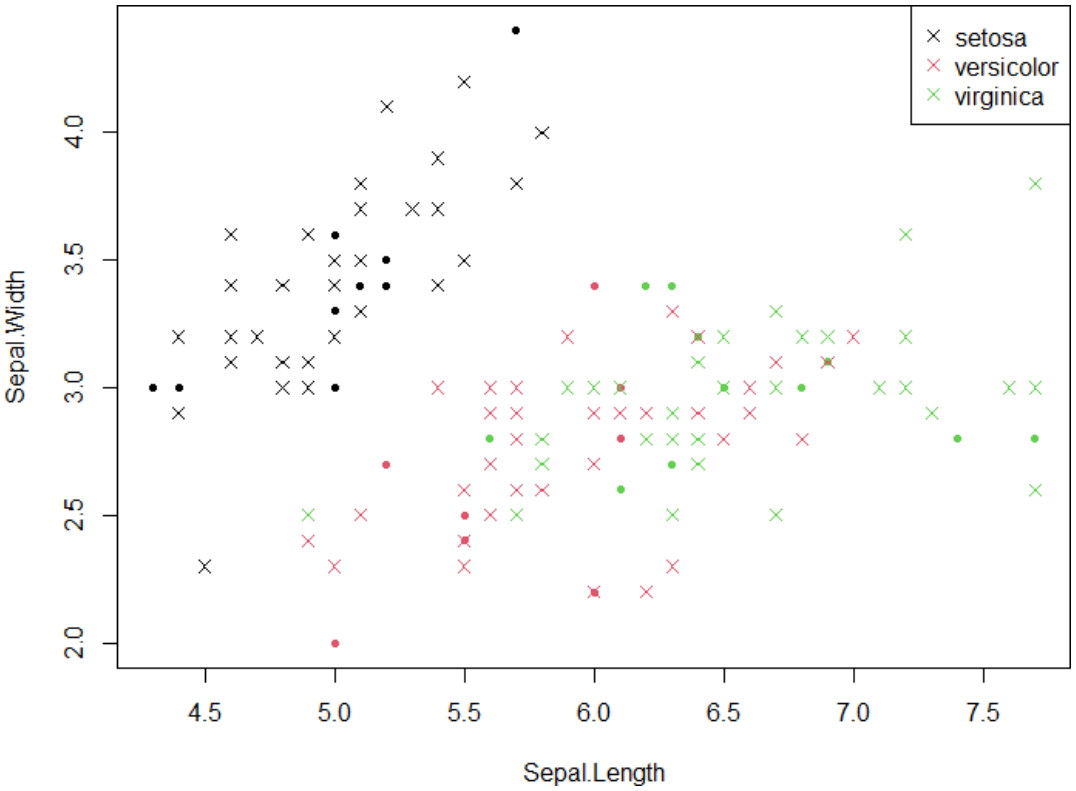
Output :

```
> library("caret")
> library("caTools")
> data("iris")
> dim(iris)
[1] 150   5
> summary(iris)
 Sepal.Length   Sepal.Width      Petal.Length   Petal.Width
 Min.  :4.300   Min.  :2.000   Min.  :1.000   Min.  :0.100
 1st Qu.:5.100   1st Qu.:2.800   1st Qu.:1.600   1st Qu.:0.300
 Median :5.800   Median :3.000   Median :4.350   Median :1.300
 Mean   :5.843   Mean   :3.057   Mean   :3.758   Mean   :1.199
 3rd Qu.:6.400  3rd Qu.:3.300  3rd Qu.:5.100  3rd Qu.:1.800
 Max.  :7.900   Max.  :4.400   Max.  :6.900   Max.  :2.500
      Species
 setosa   :50
 versicolor:50
 virginica :50

> head(iris)
  Sepal.Length Sepal.Width Petal.Length Petal.Width Species
1          5.1         3.5          1.4         0.2  setosa
2          4.9         3.0          1.4         0.2  setosa
3          4.7         3.2          1.3         0.2  setosa
4          4.6         3.1          1.5         0.2  setosa
5          5.0         3.6          1.4         0.2  setosa
6          5.4         3.9          1.7         0.4  setosa
> set.seed(1234)
> random <- runif(150)
> iris_random <- iris[order(random),]
> head(iris_random)
    Sepal.Length Sepal.Width Petal.Length Petal.Width    Species
7            4.6         3.4          1.4         0.3     setosa
64           6.1         2.9          4.7         1.4 versicolor
```

```
73          6.3         2.5         4.9         1.5 versicolor
98          6.2         2.9         4.3         1.3 versicolor
101         6.3         3.3         6.0         2.5  virginica
110         7.2         3.6         6.1         2.5  virginica
> train <- iris_random[1:120,]
> test <- iris_random[121:150,]
> train_sp <- iris_random[1:120,5]  # Use 120 samples for training
> test_sp <- iris_random[121:150,5]  # Use 30 samples for testing
> require(class)
Loading required package: class
> train_sp <- as.factor(train_sp)
> test_sp <- as.factor(test_sp)
> model <- knn(train = train[, -5], test = test[, -5], cl = train_sp, k = 5)  # Use k = 5 for example
> print(model)
 [1] setosa     versicolor versicolor setosa     setosa
 [6] setosa     setosa     versicolor setosa     versicolor
[11] virginica  versicolor versicolor versicolor versicolor
[16] virginica  virginica  virginica  setosa     virginica
[21] setosa     versicolor virginica  virginica  virginica
[26] virginica  virginica  virginica  setosa     virginica
Levels: setosa versicolor virginica
> plot(test[, c(1, 2)], col = model, pch = 20)
> points(train[, c(1, 2)], col = train_sp, pch = 4)
> legend("topright", legend = levels(train_sp), col = 1:3, pch = 4)
```


Sepal.Width
2.0
2.5
3.0
3.5
4.0
4.5
5.0
5.5
6.0
6.5
7.0
7.5
Sepal.Length


Sepal.Width
2.0
2.5
3.0
3.5
4.0
4.5
5.0
5.5
6.0
6.5
7.0
7.5
Sepal.Length


setosa
versicolor
virginica
Sepal.Width
Sepal.Length
2.0
2.5
3.0
3.5
4.0
4.5
5.0
5.5
6.0
6.5
7.0
7.5