Practical No:02

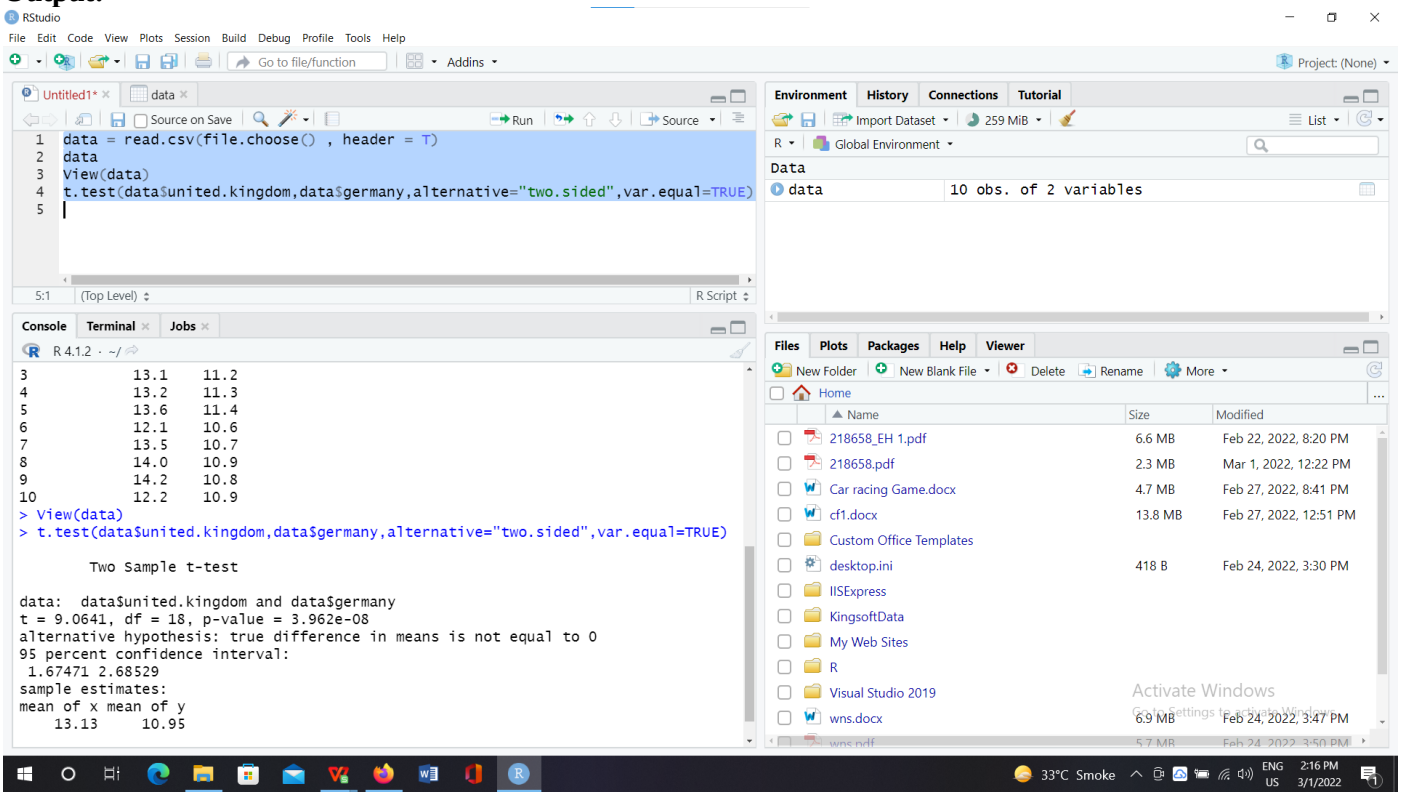
Aim: Practical Of Hypothesis Testing

1) Independent 't' Test

A researcher studied education in the united kingdom and germany wanted to compare how many years of on average women in each country spent in school .The Researched obtained the random sample from both the countries. Test whether the average number of years spent in the school by women in 2 countries are equal or not.

united kingdom	germany
12.8	10.8
12.6	10.9
13.1	11.2
13.2	11.3
13.6	11.4
12.1	10.6
13.5	10.7
14	10.9
14.2	10.8
12.2	10.9

Output:



Conclusion:

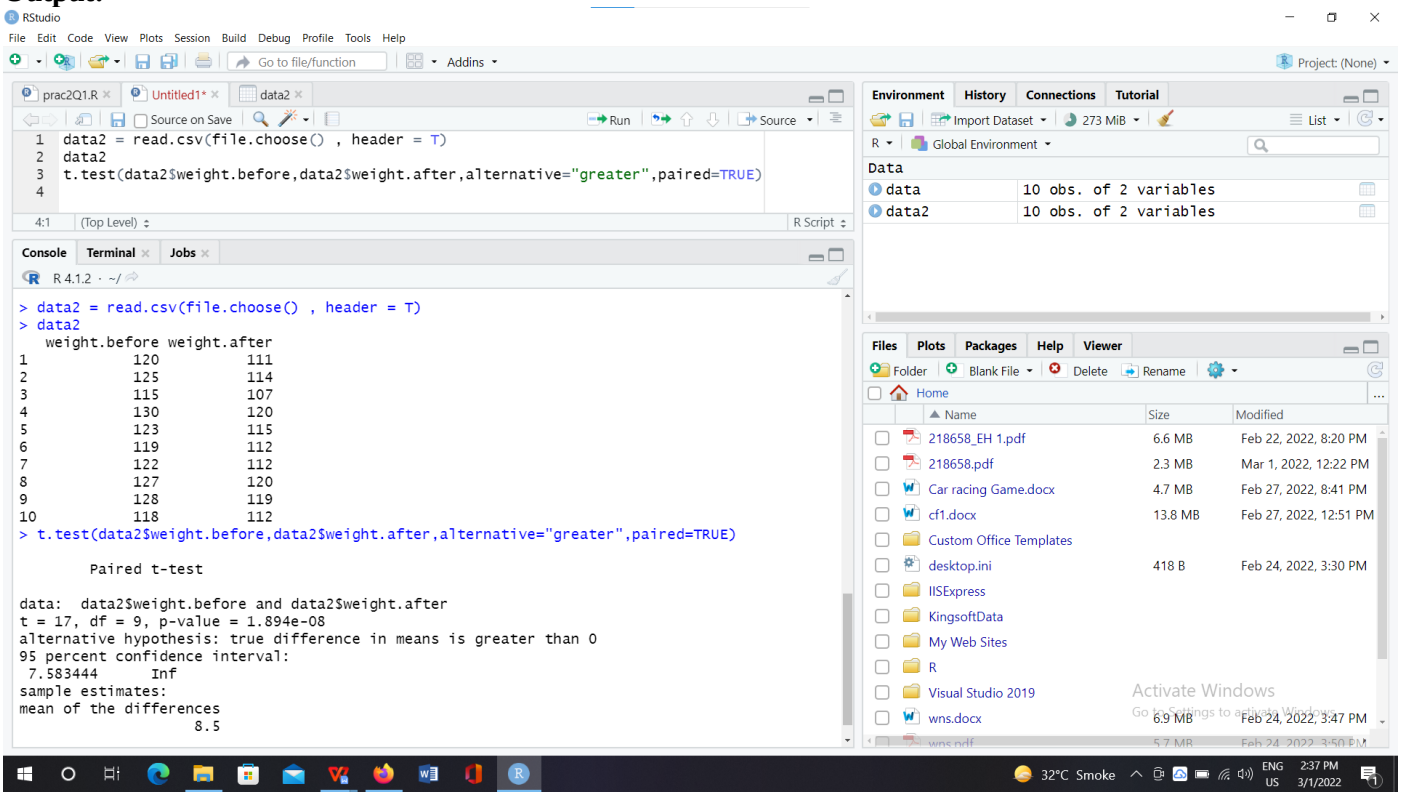
When alternative = 'two.sided',
The p-value = 3.962e-08 which is less than 0.05 therefore **we reject the hypothesis H0**.
Therefore Average number of years spend in school by women in two contries are not equal 0.

2) Paired 't' Test

A health club advertised a weight reducing program and claimed that on an average a participant in an program loses weight in 6 months.A person wanted to, verify the claim ,the club allowed him to select randomly the records of 10 participants about their weights before and after the program.

weight before	weight after
120	111
125	114
115	107
130	120
123	115
119	112
122	112
127	120
128	119
118	112

Output:



Conclusion:

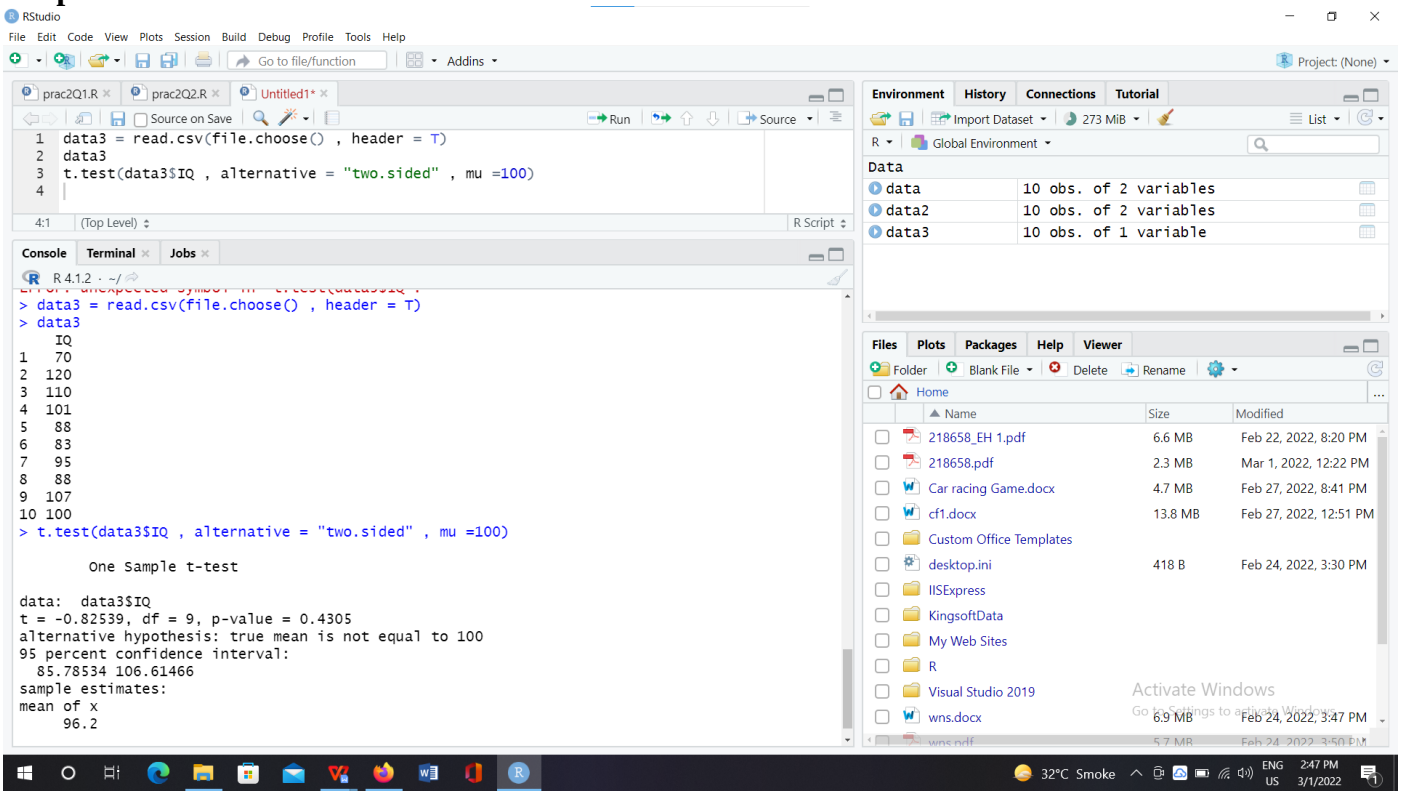
When alternative = 'greater',
The p-value=1.894e-08 which is less than 0.05 therefore **we reject the hypothesis H0**.

3) One Sample 't' Test

A random sample of 10 Students has the following IQ:

IQ
70
120
110
101
88
83
95
88
107
100

Output:



Conclusion:

When alternative = two.sided,
The p-value=0.4305 which is less than 0.05 therefore **we reject the hypothesis H0**.Therefore population IQ is not equal to 100.

Practical No:03

Aim:Practical of Analysis of Variance

1. F-test(practical of analysis of data):

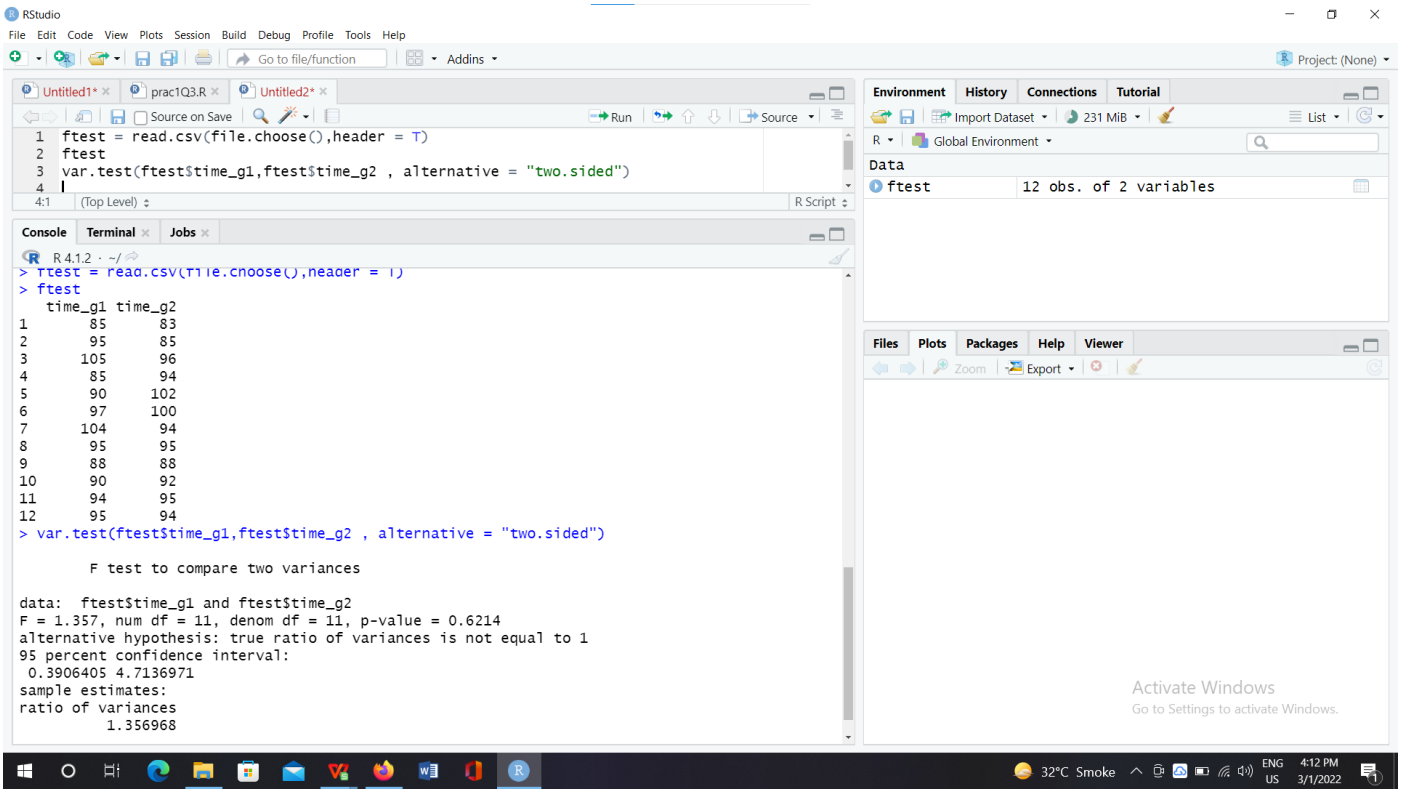
Two samples are drawn from two normal populations. From the following data test whether the two samples have the same variance.

WPS Office | Document1

Menu | Cut | Copy | Format Painter | Paste | Calibri

B2

	A	B	C
1	time_g1	time_g2	
2	85	83	
3	95	85	
4	105	96	
5	85	94	
6	90	102	
7	97	100	
8	104	94	
9	95	95	
10	88	88	
11	90	92	
12	94	95	
13	95	94	
14			



Conclusion:

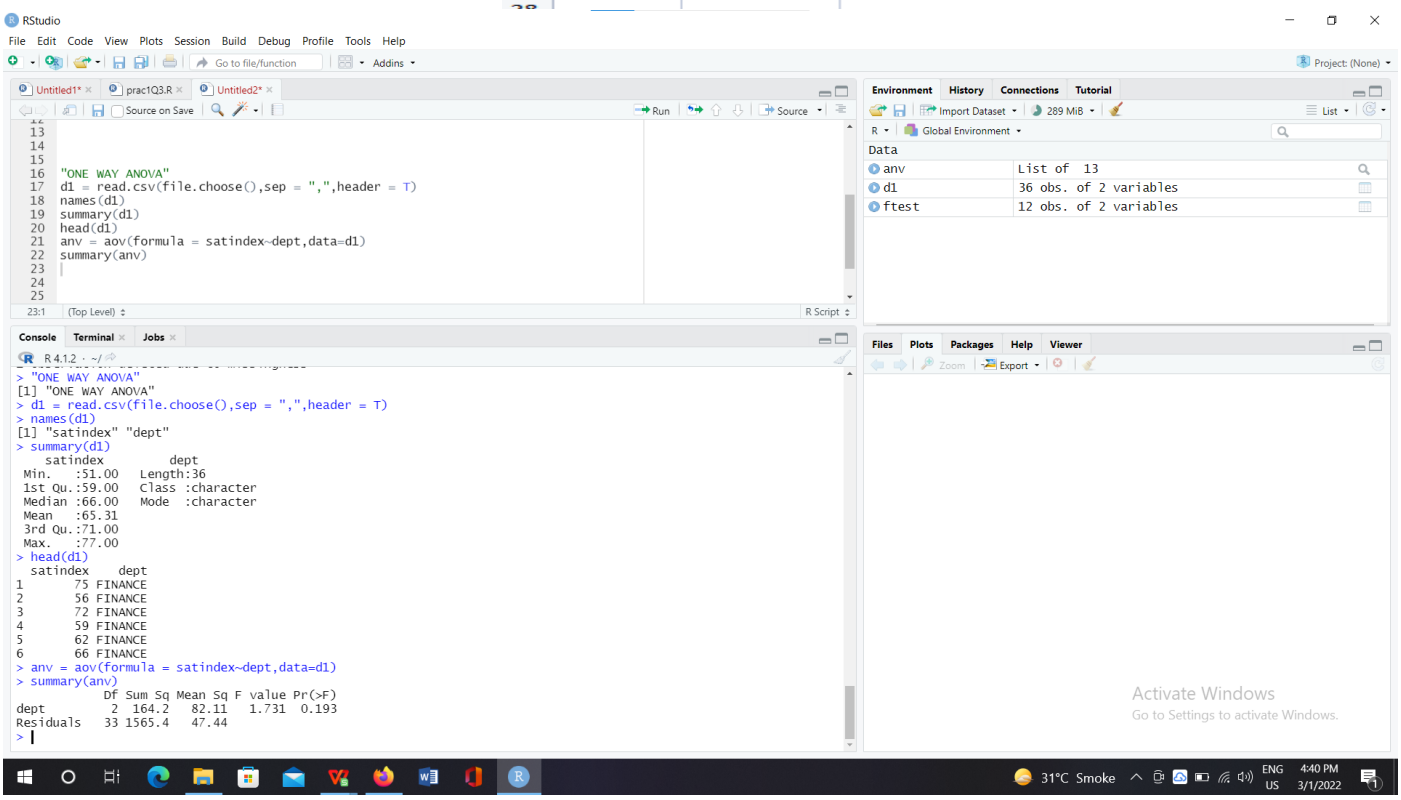
When alternative = two.sided ,
Here p = 0.4524, is greater than 0.05 so, **accept the hypothesis H0**.Therefore no significant difference in variance of two group.

2. One-way Anova:

A company is accessing the difference in time to complete the task between three groups of employees.
State the hypothesis and do the variance test for given dataset: (One way ANOVA)
Group-1: Fianance
Group-2: Marketing
Group-3: CS

I12

	A	B
1	satindex	dept
2	75	FINANCE
3	56	FINANCE
4	72	FINANCE
5	59	FINANCE
6	62	FINANCE
7	66	FINANCE
8	67	FINANCE
9	71	FINANCE
10	59	FINANCE
11	62	FINANCE
12	66	FINANCE
13	58	FINANCE
14	58	MARKETING
15	63	MARKETING
16	53	MARKETING
17	74	MARKETING
18	77	MARKETING
19	69	MARKETING
20	57	MARKETING
21	70	MARKETING
22	68	MARKETING
23	51	MARKETING
24	64	MARKETING
25	55	MARKETING
26	72	CS
27	69	CS
28	77	CS
29	71	CS
30	59	CS
31	70	CS
32	67	CS
33	73	CS
34	74	CS
35	60	CS
36	62	CS
37	65	CS



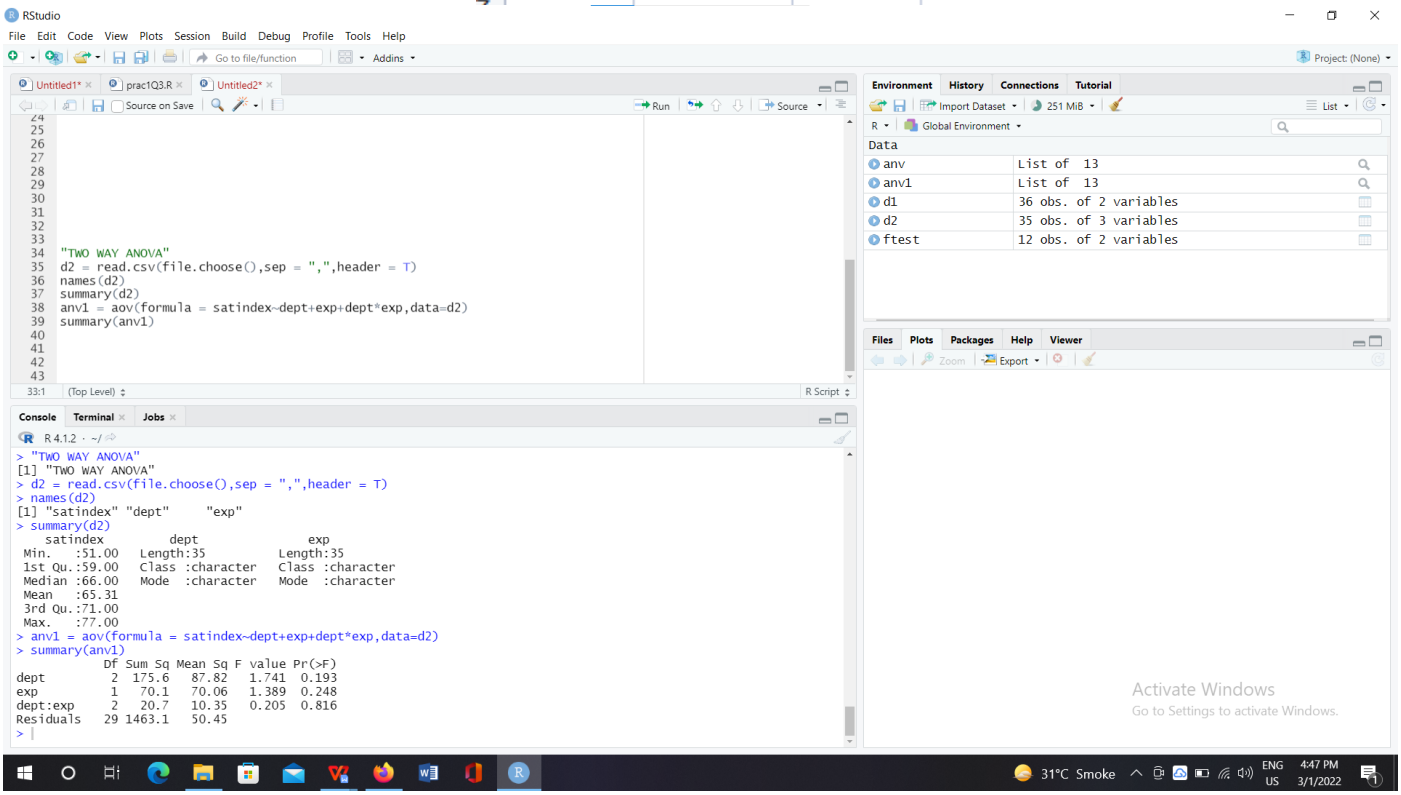
Conclusion:

Here p = 0.193 which is greater than 0.05, **therefore accept the hypothesis H0**.
Therefore, the three group of employees require same amount of time to perform the task.

3. Two-way Anova:

Test whether the given factors have a significant impact on the dataset.

	A	B	C
1	satindex	dept	exp
2	75	FINANCE	lt5
3	56	FINANCE	lt5
4	72	FINANCE	lt5
5	59	FINANCE	lt5
6	62	FINANCE	lt5
7	66	FINANCE	lt5
8	67	FINANCE	gt5
9	71	FINANCE	gt5
10	59	FINANCE	gt5
11	62	FINANCE	gt5
12	66	FINANCE	gt5
13	58	FINANCE	gt5
14	58	MARKETING	lt5
15	63	MARKETING	lt5
16	53	MARKETING	lt5
17	74	MARKETING	lt5
18	77	MARKETING	lt5
19	69	MARKETING	lt5
20	57	MARKETING	gt5
21	70	MARKETING	gt5
22	68	MARKETING	gt5
23	51	MARKETING	gt5
24	64	MARKETING	gt5
25	55	MARKETING	gt5
26	72	CS	lt5
27	69	CS	lt5
28	77	CS	lt5
29	71	CS	lt5
30	59	CS	lt5
31	70	CS	lt5
32	67	CS	gt5
33	73	CS	gt5
34	74	CS	gt5
35	60	CS	gt5
36	62	CS	gt5



Conclusion:

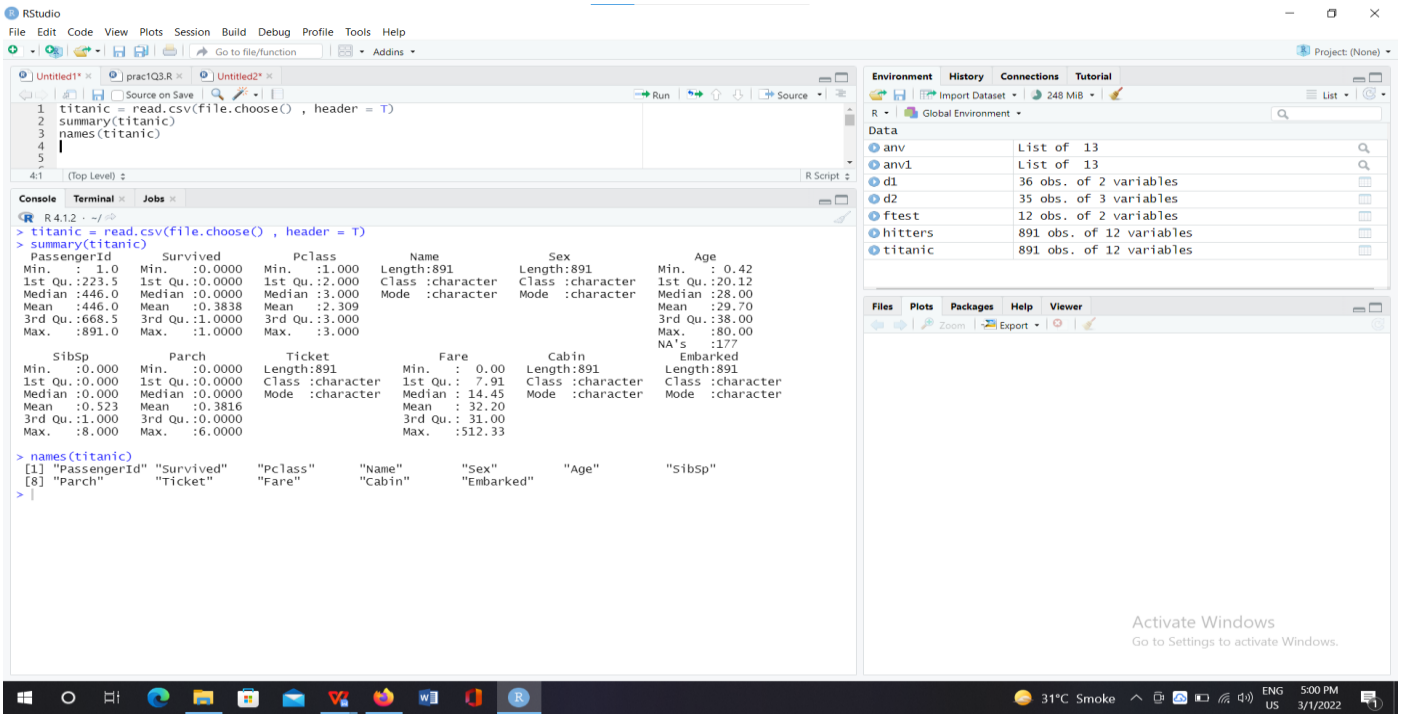
The p-value for both the factors is greater than 0.05 **we accept the hypothesis H0**.
Therefore, both the factors viz. dept and exp have a statistically significant impact on the dataset.

Practical No:04

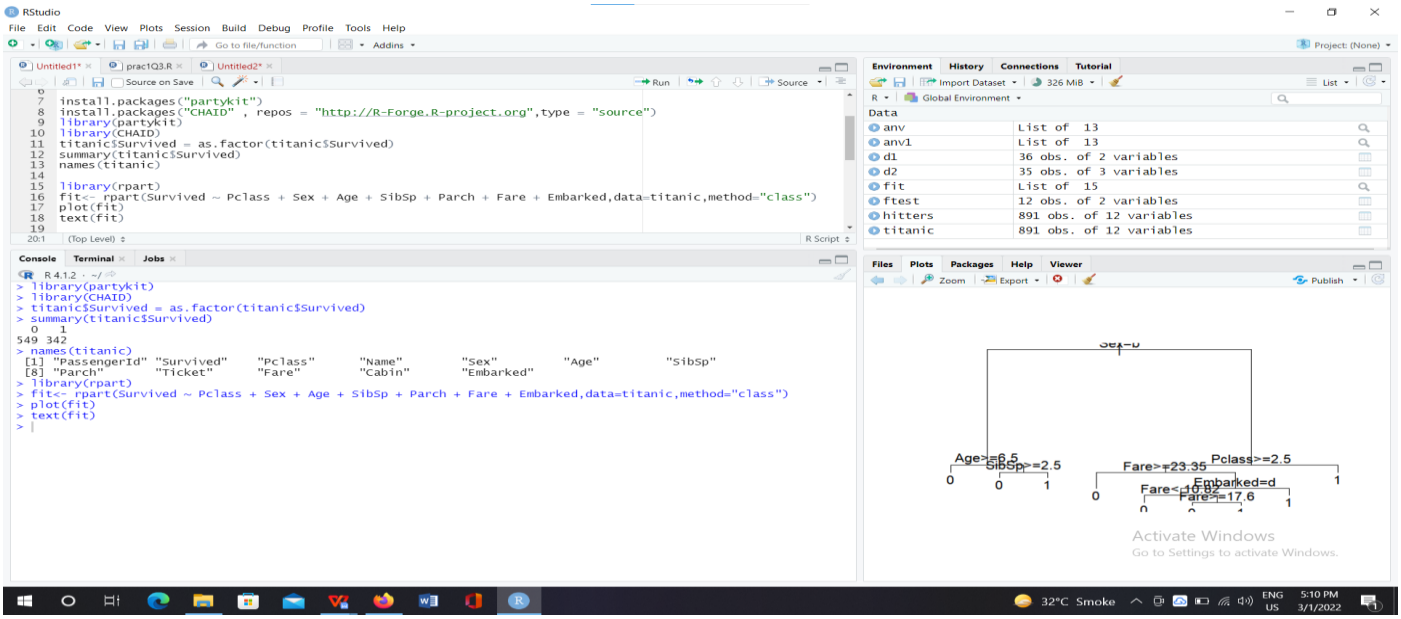
Aim:Practical of Decision Tree

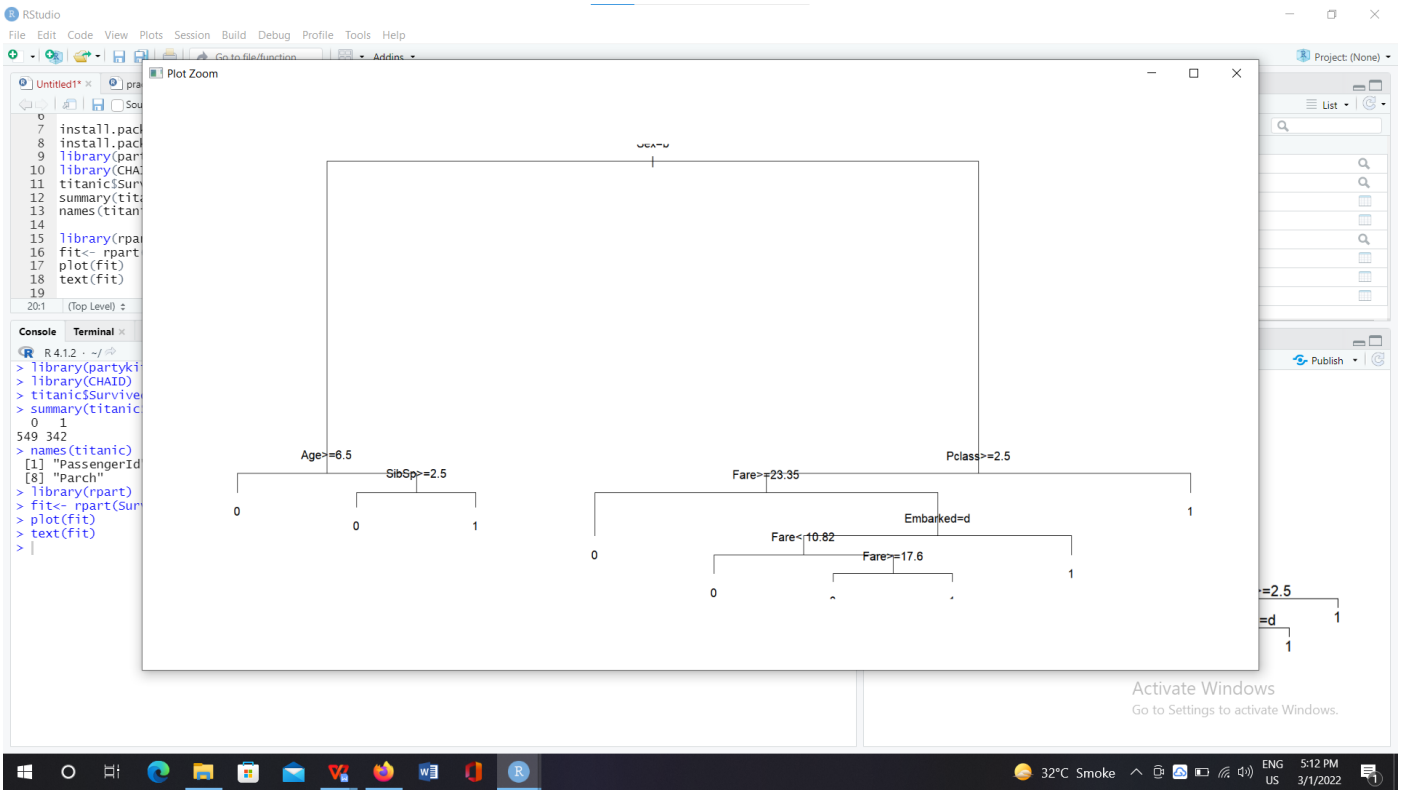
"PassengerId:type should be integers
Survived:Survived or Not
Pclass:Class of Travel
Name:Name of Passenger
Sex:Gender
Age:Age of Passengers
SibSp:Number of Sibling/Spouse aboard
Parch:Number of Parent/Child aboard
Ticket , Fare , Cabin
Embarked:The port in which a passenger has embarked. C - Cherbourg, S - Southampton, Q= Queenstown"

1) Exploring the dataset. The purpose of this dataset is to predict which people are more likely to survive after the collision with the iceberg. The dataset contains 12 variables and 891 observations.

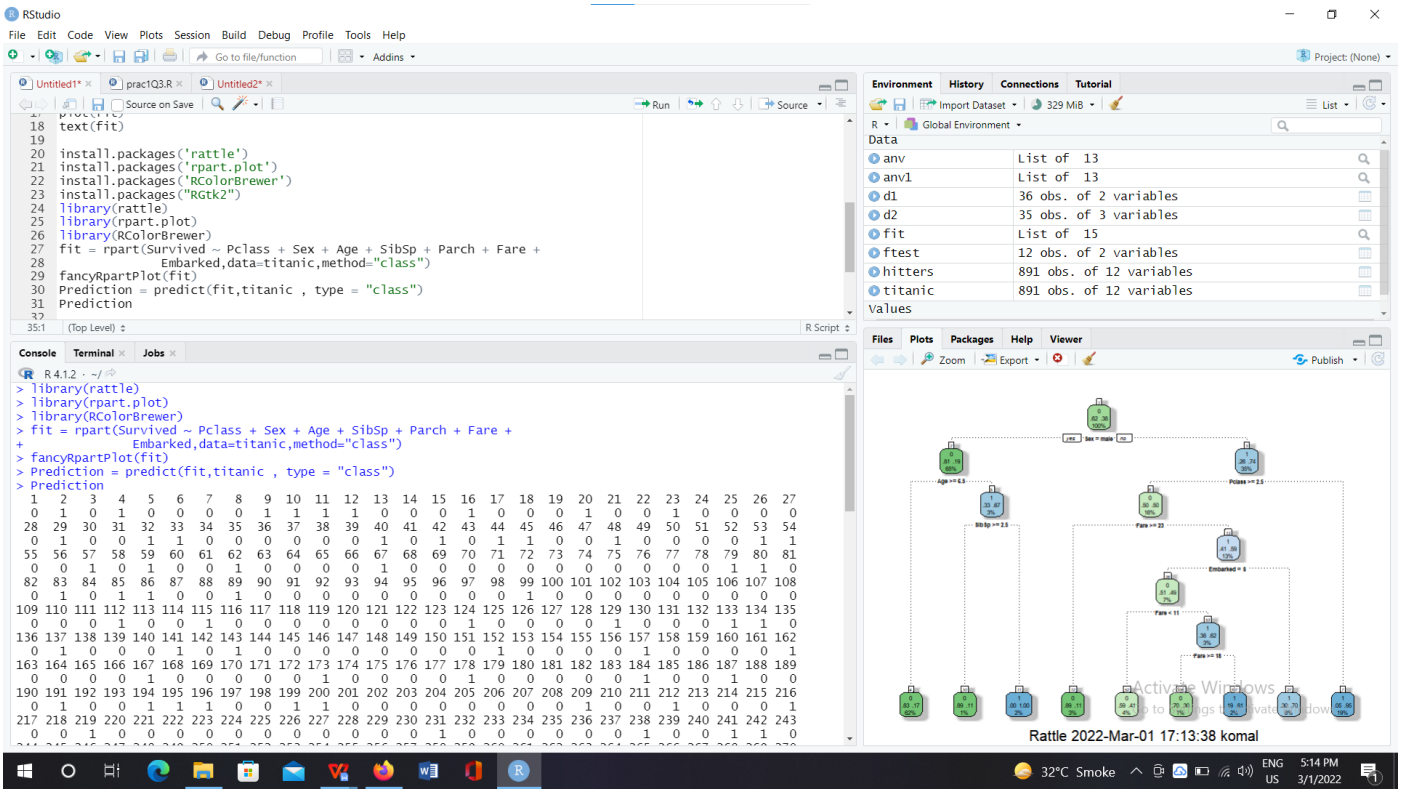


2) From the following summary we can say that almost 549 passengers are dead and 342 have survived.

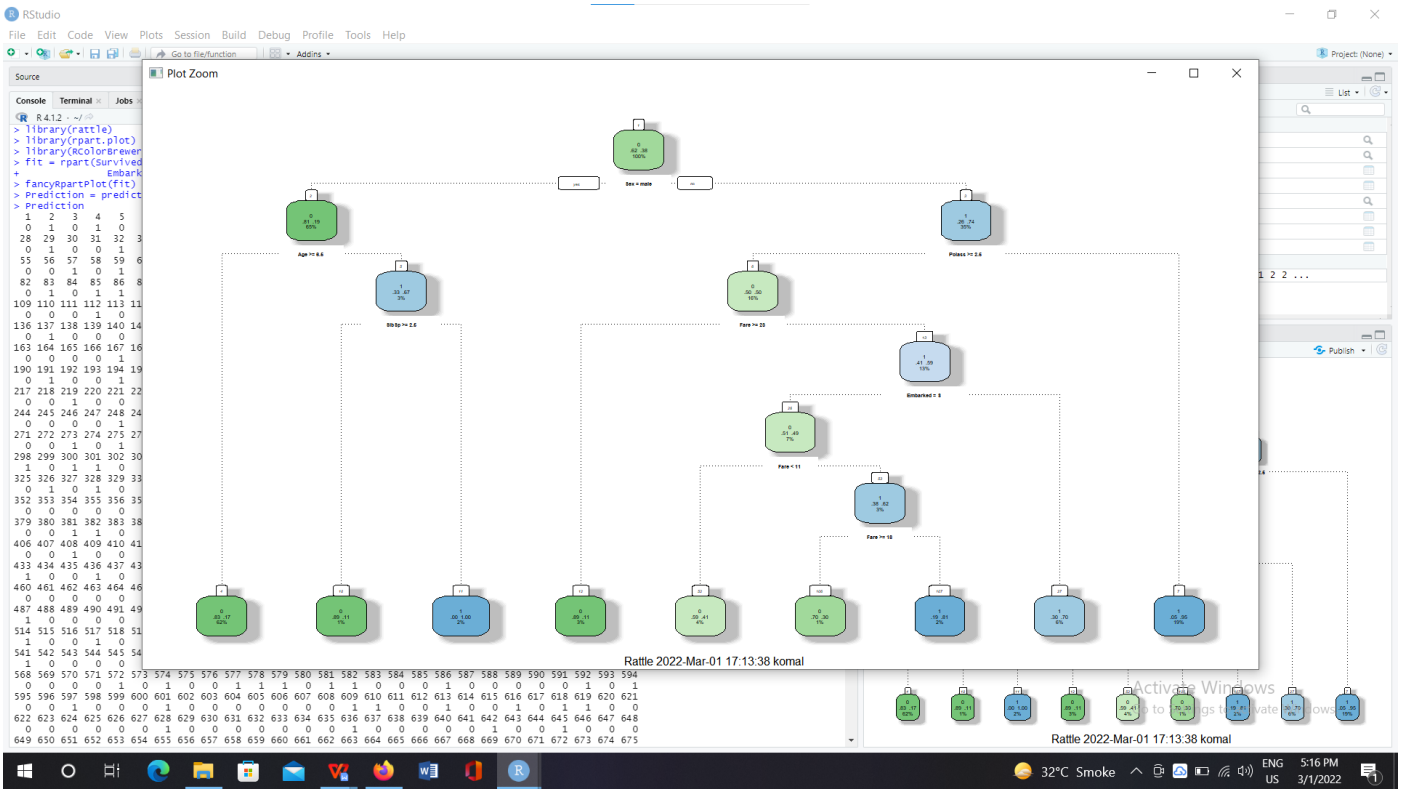




3) In the following prediction we can see the passenger Id with the value of 0 and 1. Where 0 means passenger with that passenger Id is dead and 1 means passenger with that passenger Id survived.



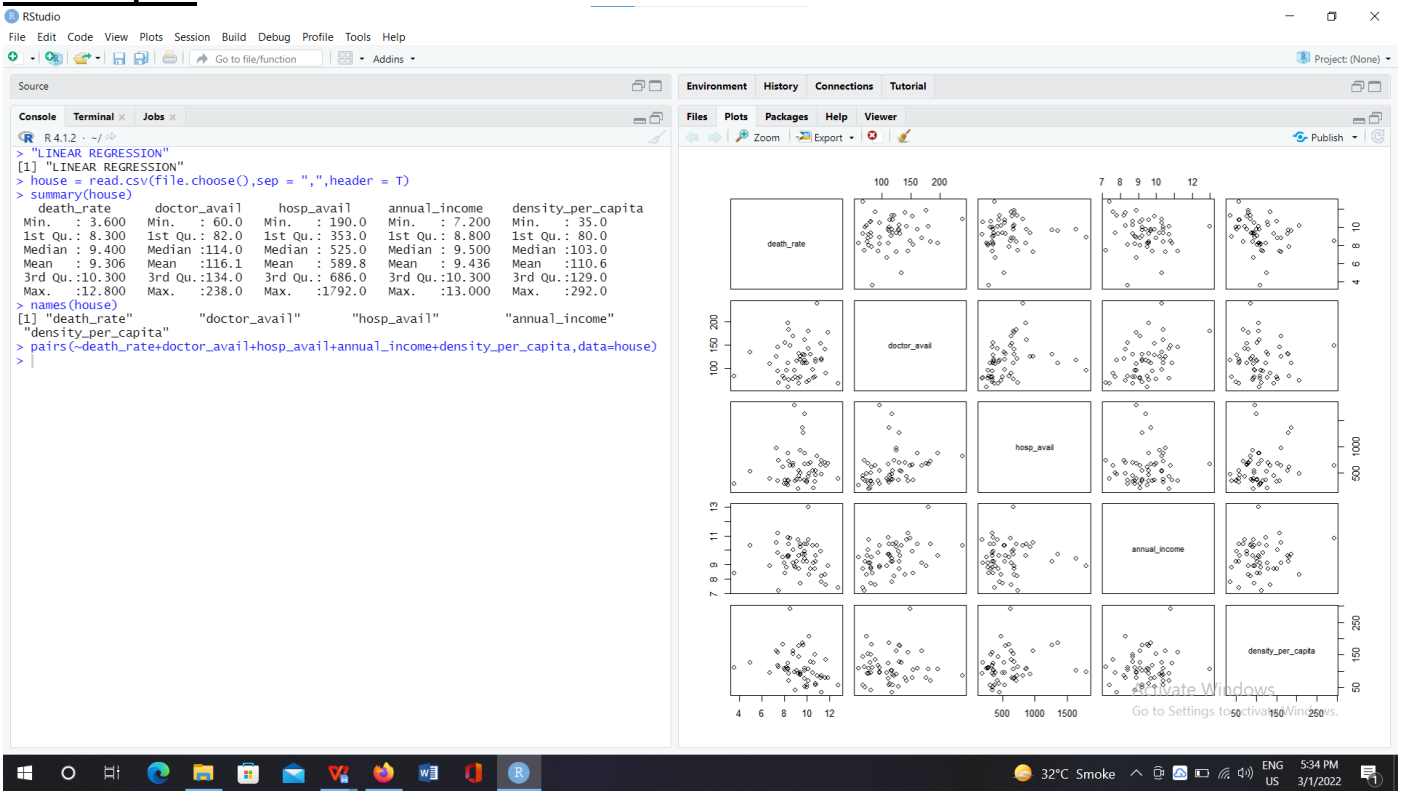
4) At the top, it is the overall probability of survival. It shows the proportion of passengers that survived the crash.
Node-0: 38% of passengers survived and 62% of passengers are not.
Node-1: This node asks whether the gender of the passenger is male.
If yes, then you go down to the root's left child node. 65% are males with a survival probability of 19%. Node-2: In the second node, you ask if the male passenger is above 6.5 years old. If yes, then the chance of survival is 17 percent.
This way we can calculate the likelihood of survival.



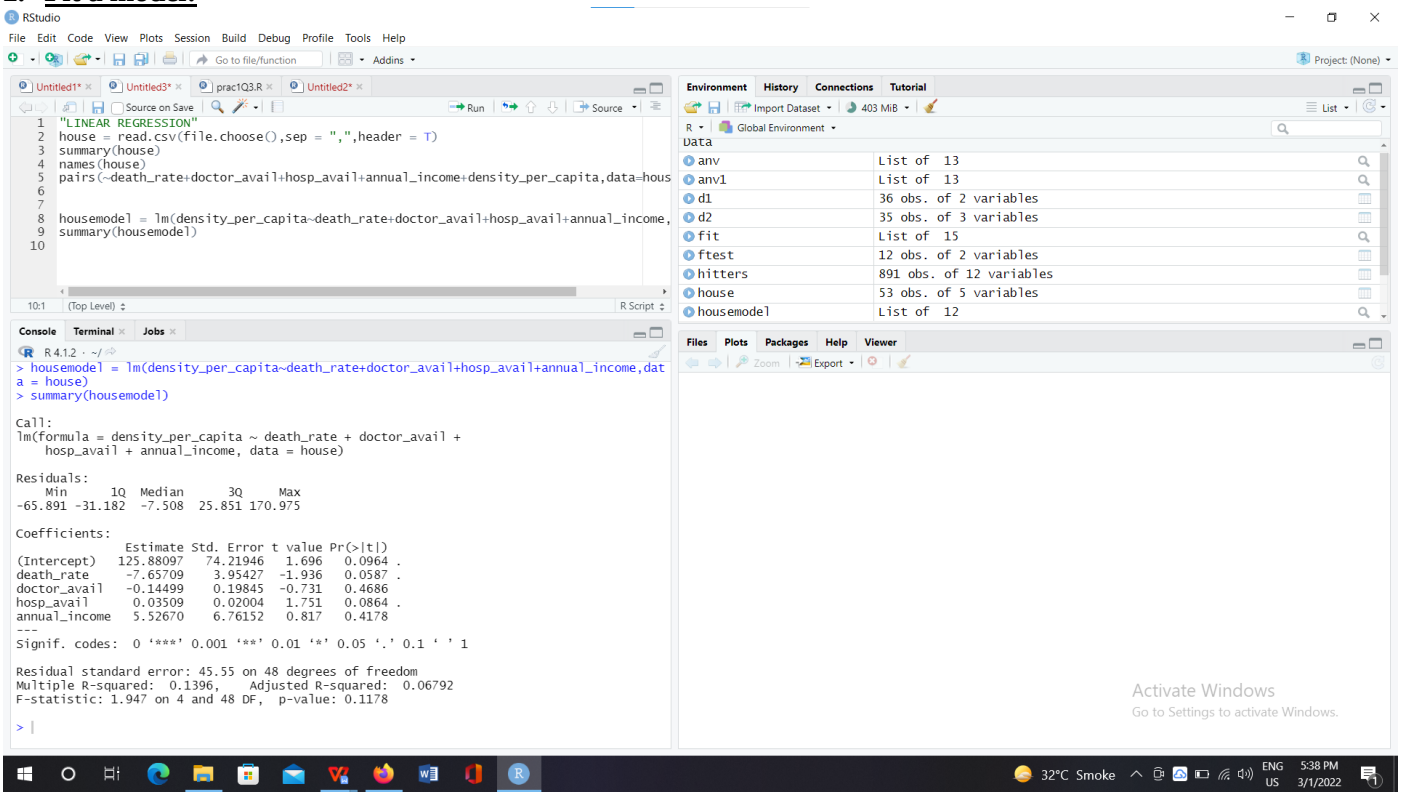
Practical No:05

Aim:Practical of Simple/Multiple Linear Regression

1.Scatter plot:



2. Fit a model:

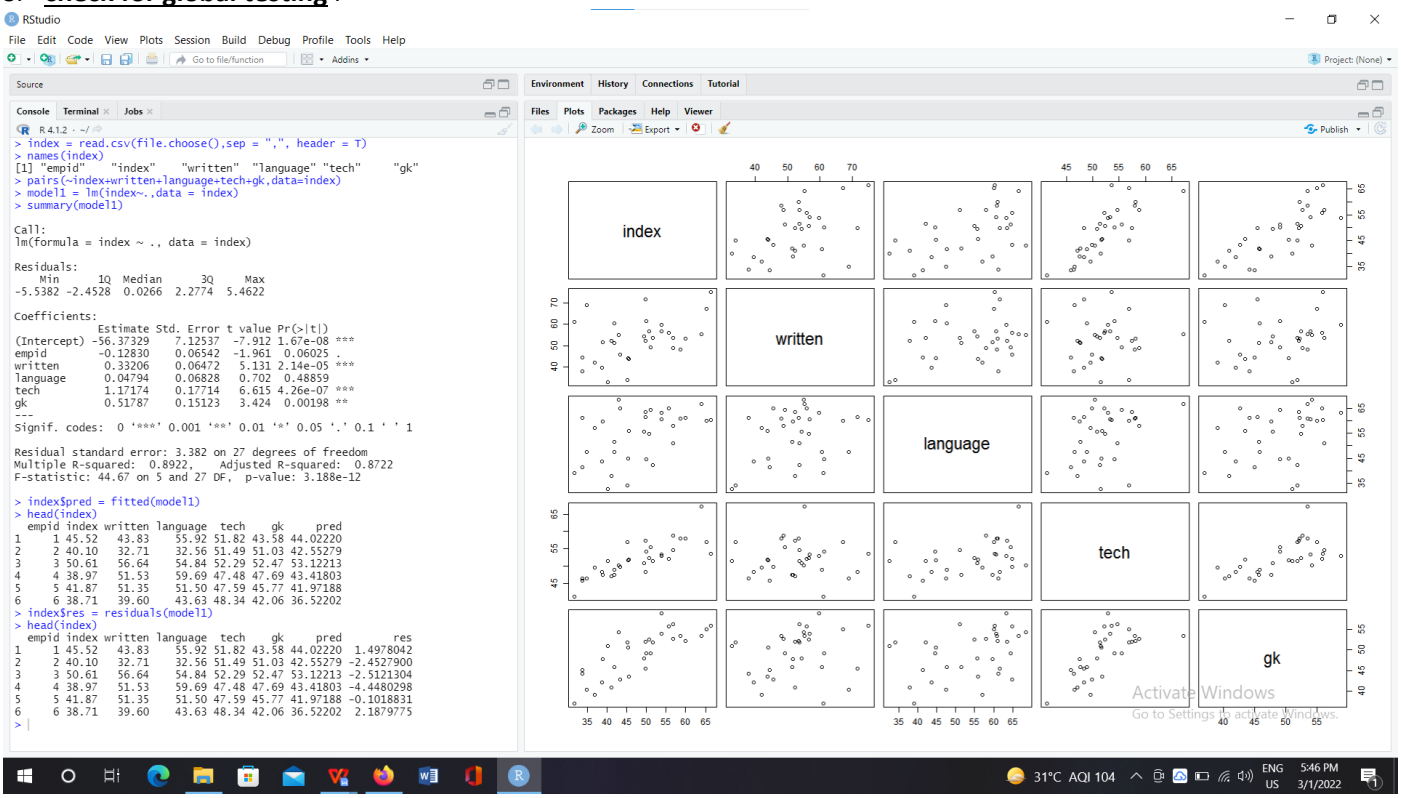


Conclusion:

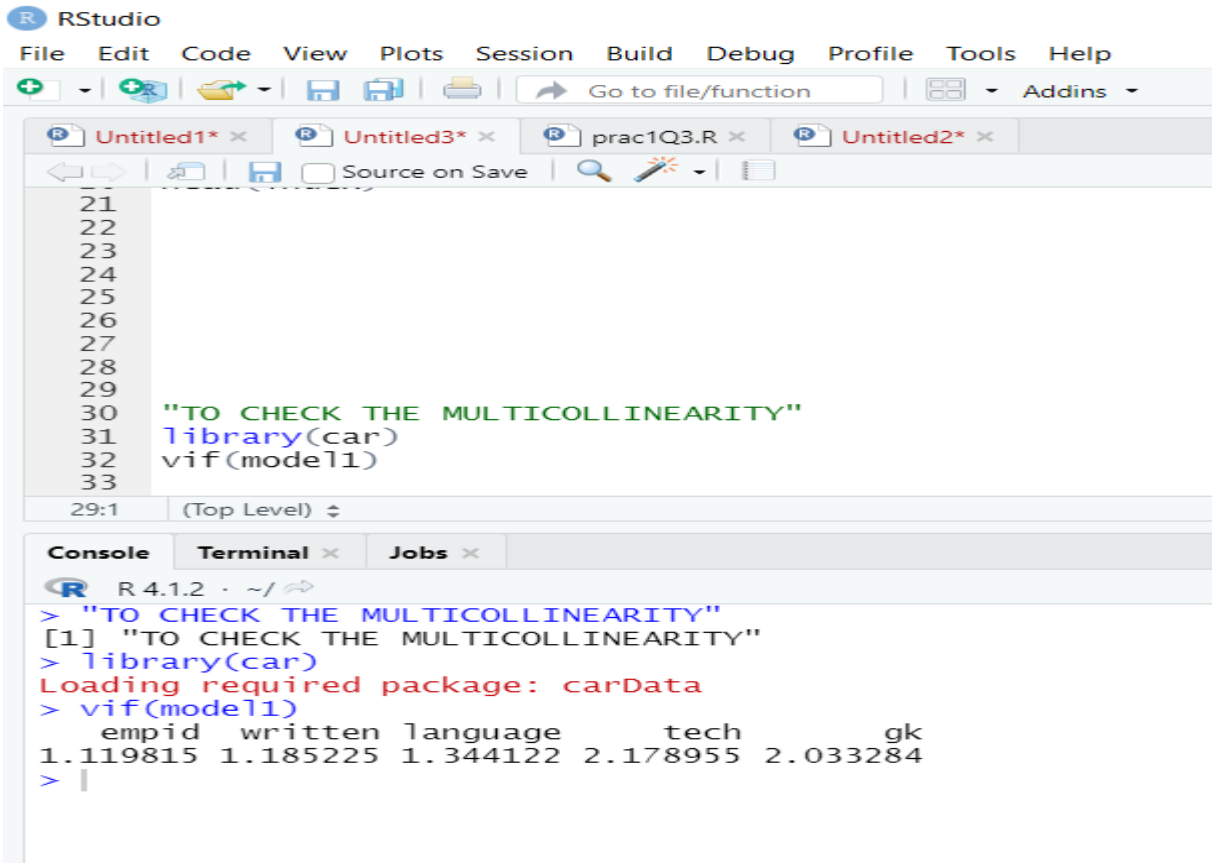
As value of multiple r(square) is 0.8922

So 89% of variation in index is explained by the model and 11% is not explained by the model

3. **check for global testing** :



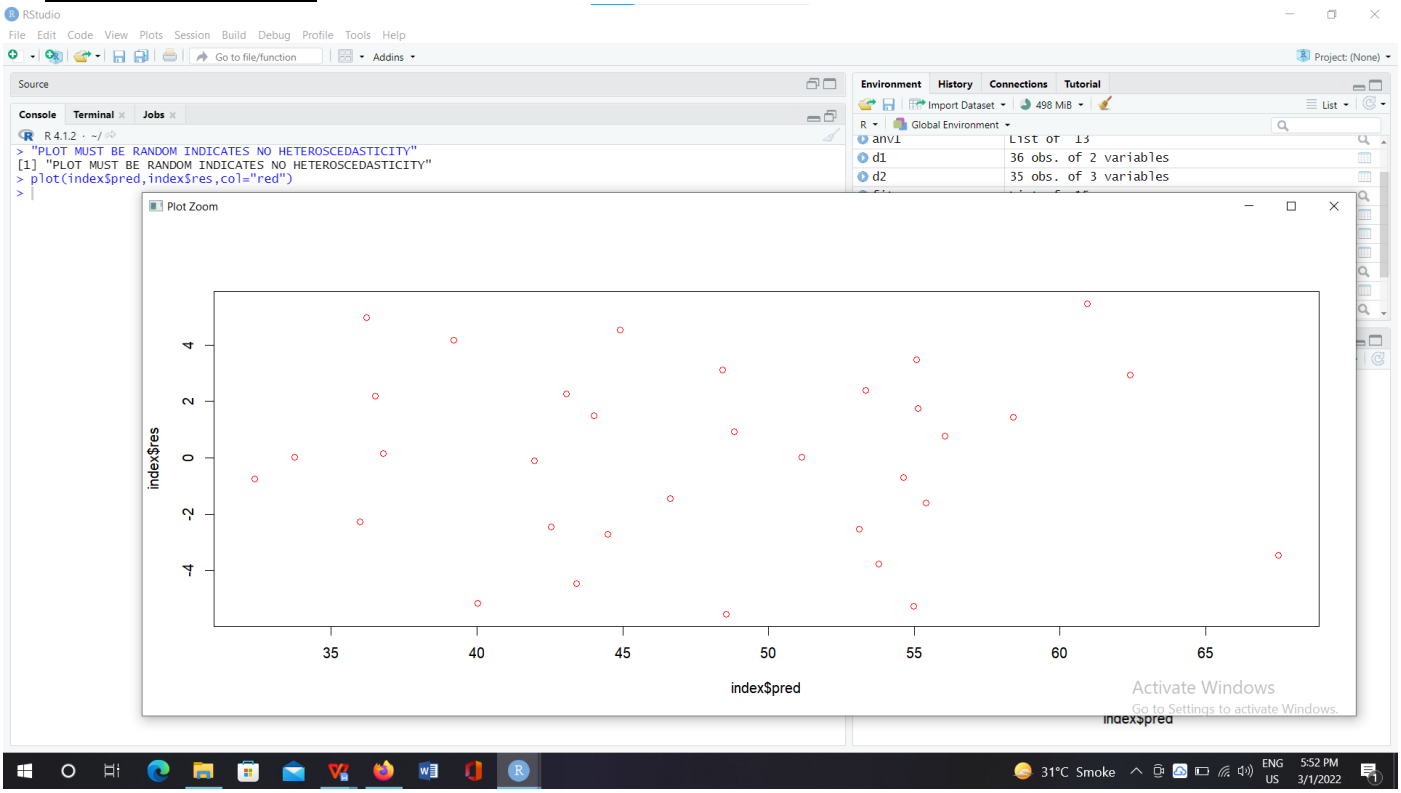
4. **check the multicollinearity:**



Conclusion:

As all VIF are less than 5
Multicollinearity is not present.

5. check heteroscedasticity



Conclusion:

since errors are generated randomly. There is no heteroscedasticity.

6. Shapiro test to check normality of errors.

```
> "ERROR ARE ASSUMED TO BE NORMALLY DESTRIBUTED"
[1] "ERROR ARE ASSUMED TO BE NORMALLY DESTRIBUTED"
> shapiro.test(index$res)

        Shapiro-Wilk normality test

data:  index$res
W = 0.97172, p-value = 0.5293
```

Conclusion:

Ho: There is homoscedasticity

H1: there is no constant variance

As P-value is greater than 0.05 we accept Ho.

7. Detecting autocorrelation using Durbin Watson Test d=2(1-r):

```
> "Detecting autocorrelation using Durbin Watson Test d=2(1-r)"
[1] "Detecting autocorrelation using Durbin Watson Test d=2(1-r)"
> library(car)
> durbinWatsonTest(model1)
 lag Autocorrelation D-W Statistic p-value
   1       0.2268414      1.500571   0.106
 Alternative hypothesis: rho != 0
```

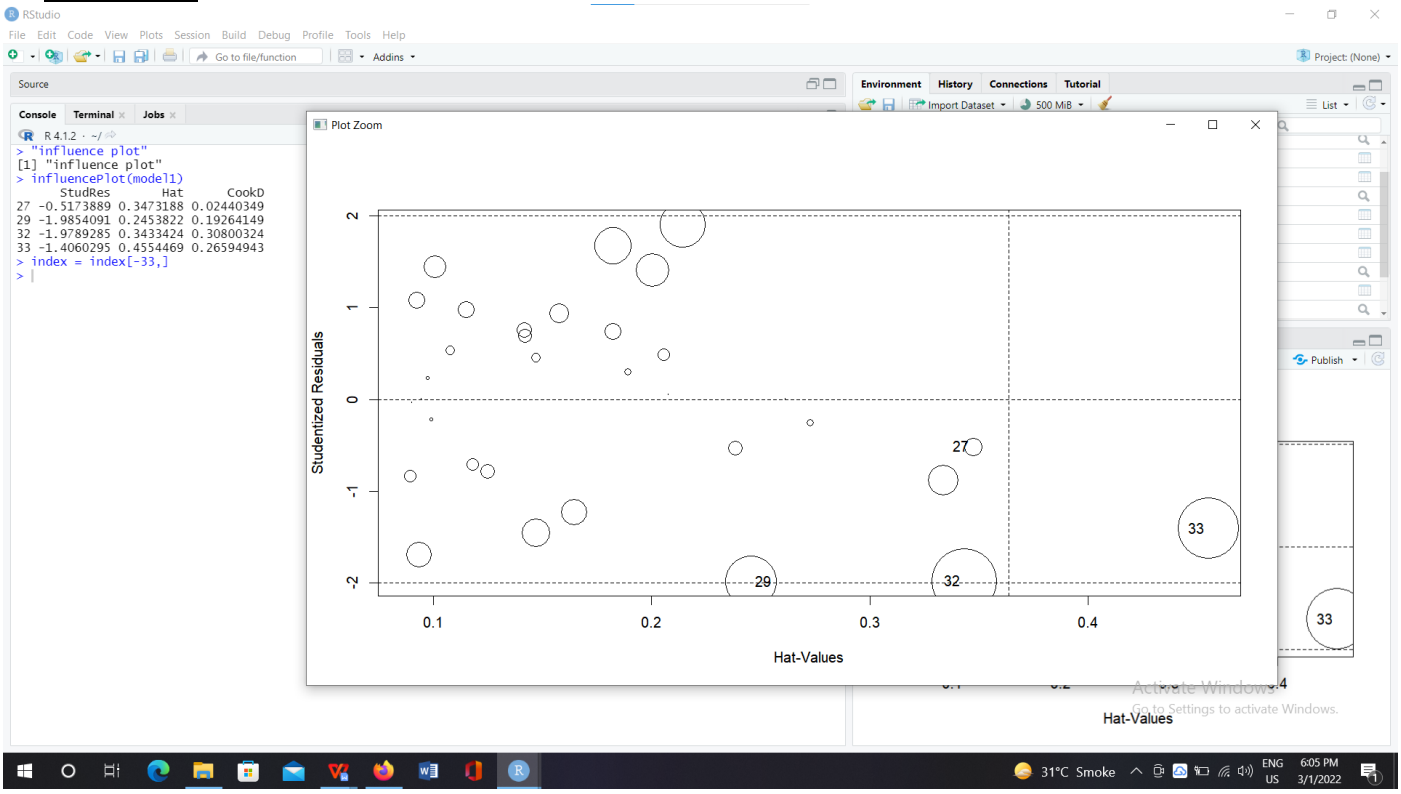
Conclusion:

Ho: Auto correlation is not present among errors

H1: not Ho

As P-value is greater than 0.05 we accept Ho.

8. influence plot



9. Validation using Hold-Out method:

```
> "Validation using Hold-Out method:"
[1] "Validation using Hold-Out method:"
> library("caret")
> library("lattice")
> library("ggplot2")
> index = read.csv(file.choose(),sep = ",",header = T)
> summary(index)
     empid          index          written         language         tech            gk
 Min.   : 1   Min.   :31.64   Min.   :32.71   Min.   :32.56   Min.   :41.25   Min.   :37.00
 1st Qu.: 9   1st Qu.:41.19   1st Qu.:45.59   1st Qu.:44.89   1st Qu.:48.34   1st Qu.:45.07
 Median :17   Median :49.45   Median :53.38   Median :57.04   Median :51.64   Median :50.53
 Mean   :17   Mean   :47.87   Mean   :52.66   Mean   :53.99   Mean   :52.02   Mean   :49.04
 3rd Qu.:25   3rd Qu.:53.92   3rd Qu.:56.75   3rd Qu.:61.28   3rd Qu.:54.68   3rd Qu.:53.50
 Max.   :33   Max.   :66.39   Max.   :75.03   Max.   :68.53   Max.   :67.27   Max.   :58.90
> d = createDataPartition(index$empid,p=0.8,list= F)
> head(d)
     Resample1
[1,]         1
[2,]         2
[3,]         3
[4,]         4
[5,]         5
[6,]         7
> dim(d)
[1] 29  1
> traindata = index[d,]
> testdata = index[-d,]
> dim(traindata)
[1] 29  6
> dim(testdata)
[1] 4 6
>
```

10. Validation using k fold method:

```
Console  Terminal  Jobs
R 4.1.2 · ~/
> "Validation using k fold method:"
[1] "Validation using k fold method:"
> kfolds = trainControl(method = "CV" , number = 4)
> modelkfold = train(index~written+language+tech+gk,data = index,method="lm",trControl=kfolds)
> modelkfold
Linear Regression

33 samples
 4 predictor

No pre-processing
Resampling: Cross-Validated (4 fold)
Summary of sample sizes: 25, 25, 25, 24
Resampling results:

  RMSE      Rsquared   MAE
  4.033235  0.8876647  3.202412

Tuning parameter 'intercept' was held constant at a value of TRUE
>
```

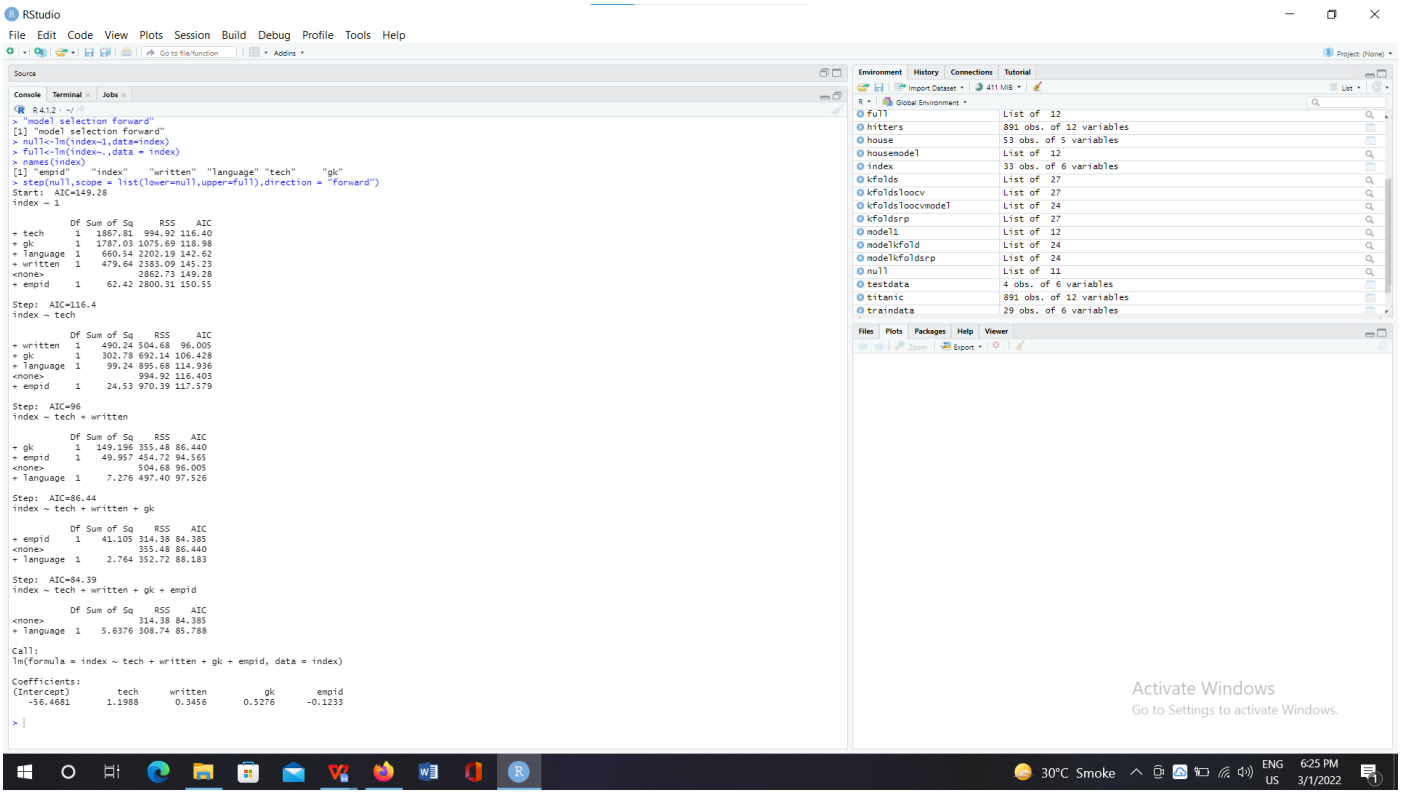
Conclusion:

As the value of RMSE is sufficiently large the model is stable.

```
Console  Terminal  Jobs
R 4.1.2 · ~/
Tuning parameter 'intercept' was held constant at a value of TRUE
> "validation using repetative k fold"
[1] "validation using repetative k fold"
> kfoldsrp<-trainControl(method = "repeatedcv",number = 4,repeats = 5)
> modelkfoldsrp<-train(index~written+language+tech+gk,data = index,method="lm",trControl=kfoldsrp)
> modelkfoldsrp
Linear Regression

33 samples
 4 predictor

No pre-processing
Resampling: Cross-Validated (4 fold, repeated 5 times)
Summary of sample sizes: 24, 25, 25, 25, 25, 24, ...
Resampling results:

  RMSE     Rsquared   MAE
  4.06407  0.8538584  3.38513

Tuning parameter 'intercept' was held constant at a value of TRUE
> "validation usning leave one out"
[1] "validation usning leave one out"
> kfoldsloocv<-trainControl(method = "LOOCV")
> kfoldsloocvmodel<-train(index~written+language+tech+gk,data = index,method="lm",trControl=kfoldsloocv)
> kfoldsloocvmodel
Linear Regression

33 samples
 4 predictor

No pre-processing
Resampling: Leave-One-Out Cross-Validation
Summary of sample sizes: 32, 32, 32, 32, 32, 32, ...
Resampling results:

  RMSE      Rsquared   MAE
  4.044207  0.8147009  3.254919

Tuning parameter 'intercept' was held constant at a value of TRUE
>
```

11. Model selection forward method.

"model selection forward"
null<-lm(index~1,data=index)
full<-lm(index~.,data = index)
names(index)
step(null,scope = list(lower=null,upper=full),direction = "forward")



12. Model selection backward method:

Source

Console Terminal Jobs

R 4.1.2 · ~/

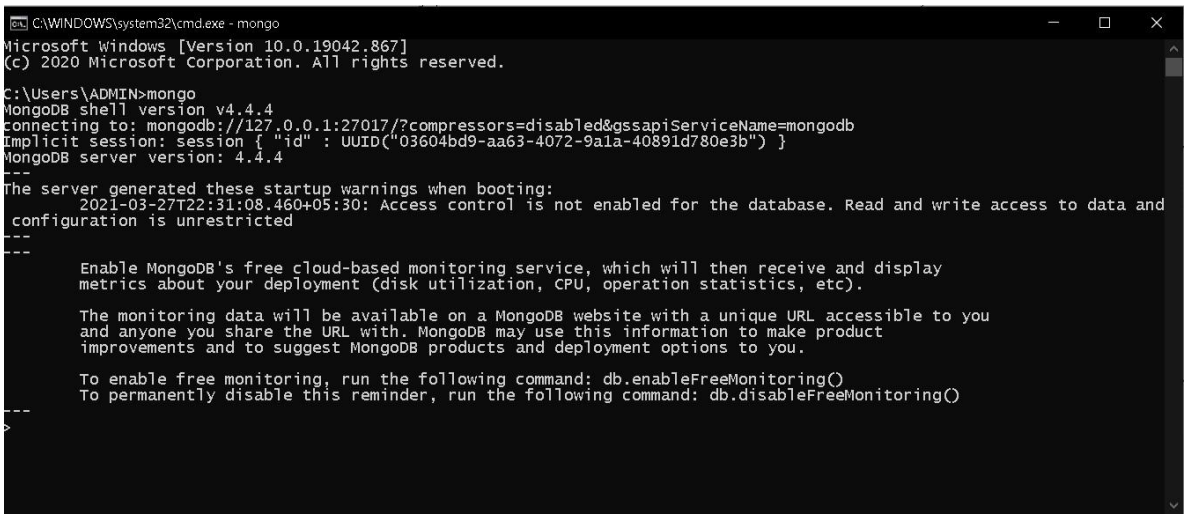
```
> "model selection backword"
[1] "model selection backword"
> step(full,scope=list(lower=null,upper=full),direction = "backward")
Start:  AIC=85.79
index ~ empid + written + language + tech + gk

           Df Sum of Sq    RSS     AIC
- language  1      5.64 314.38  84.385
<none>                 308.74  85.788
- empid     1     43.98 352.72  88.183
- gk        1    134.09 442.83  95.691
- written   1    300.99 609.74 106.245
- tech      1    500.35 809.10 115.581

Step:  AIC=84.39
index ~ empid + written + tech + gk

           Df Sum of Sq    RSS     AIC
<none>                 314.38  84.385
- empid     1     41.11 355.48  86.440
- gk        1    140.34 454.72  94.565
- written   1    357.94 672.32 107.469
- tech      1    549.77 864.15 115.753

Call:
lm(formula = index ~ empid + written + tech + gk, data = index)

Coefficients:
(Intercept)        empid      written         tech           gk
   -56.4681      -0.1233       0.3456       1.1988       0.5276

>
```

Practical No. 6

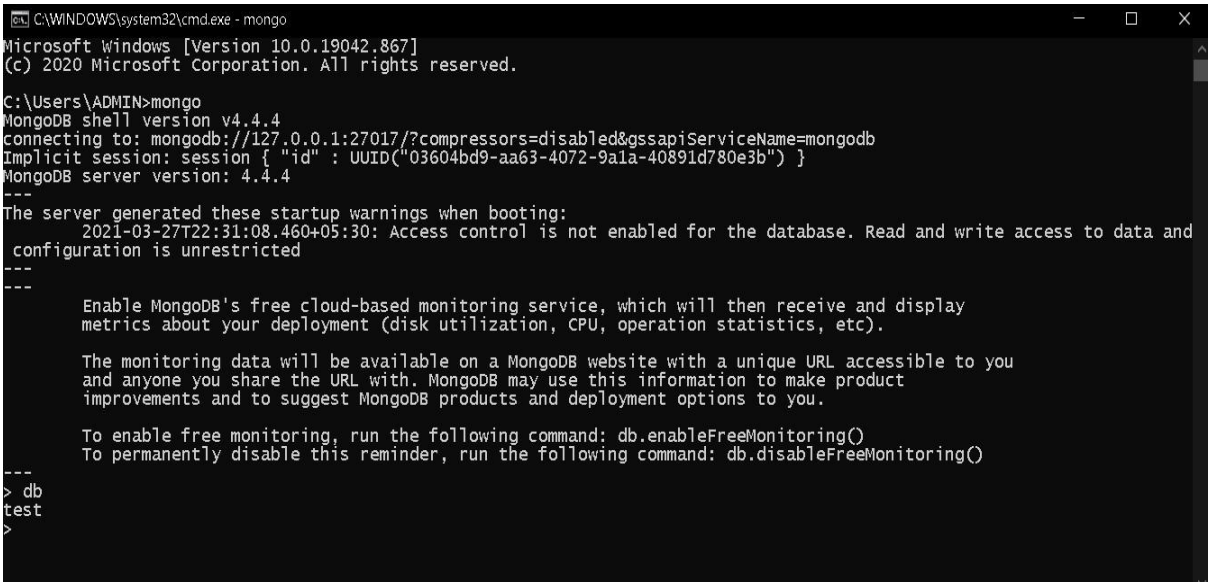
Aim:- Practical of Mongo DB.

1) To start mongodb we need to run the following command. This command will start the server.

Now run **mongo** command in cmd this will start the mongodb shell.

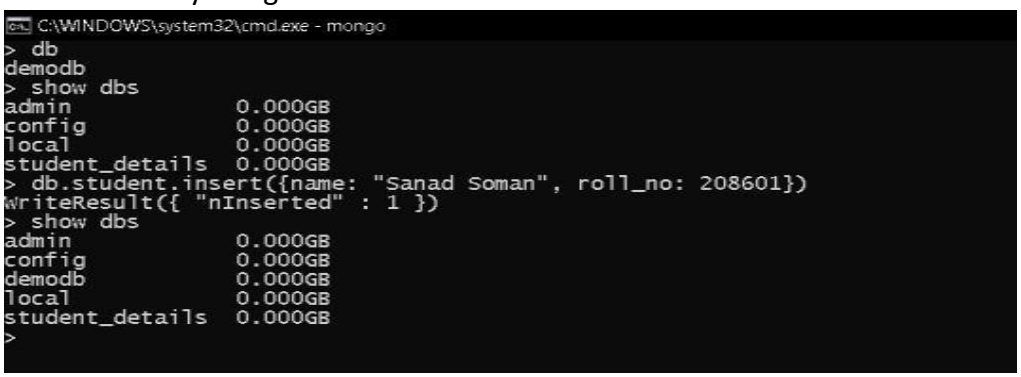


2) To display name of current database we use command "**db**".



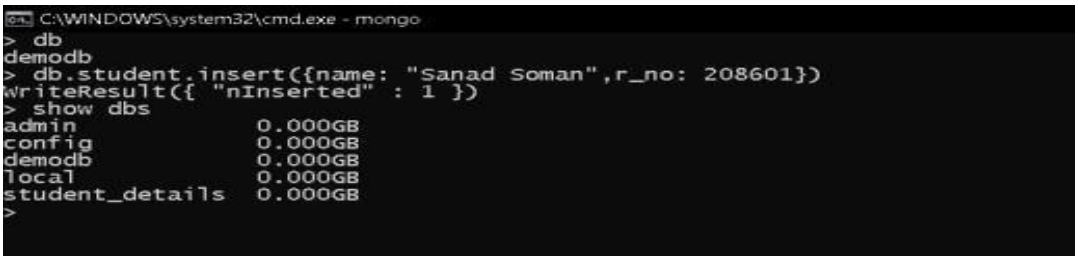
3) To create database or use a particular database we can use command "**use**".

4) Now to see the created database we use command "**show dbs**" but at first it will not show us the created database as soon as we create the collection within database we can see our database by using "**show dbs** " command.

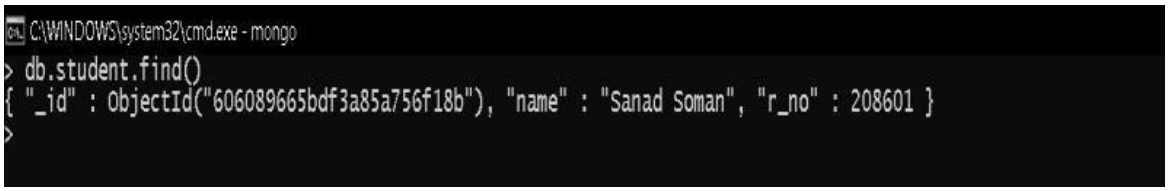


5) To drop database we can use command "**db.dropDatabase()**".

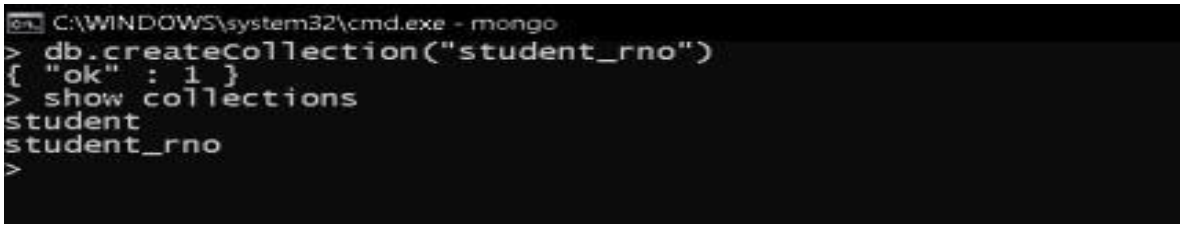
6) To create collection we can use the command "**db.collection_name.insert({key_value pairs})**".



7) To see the records within collection we can use following command:"**db.collection_name.find()**".



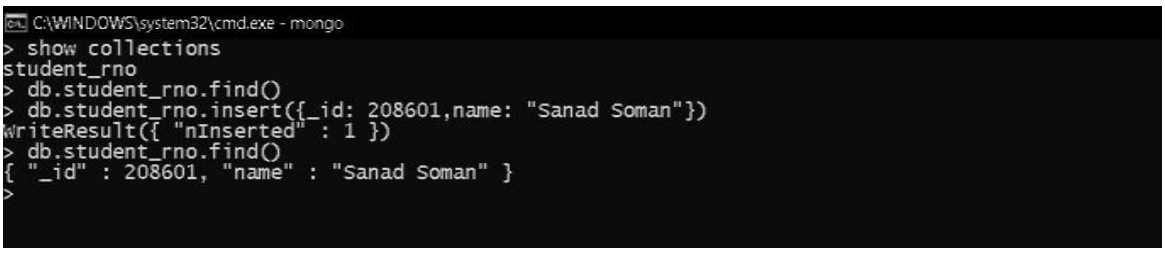
8) To create collections with options before inserting data use command "**db.createCollection("collection_name")**".



9) To drop a collection we use command : "**db.collection_name.drop()**".

10) To insert values into collection we use command:

"**db.collection_name.insert({key_value pairs})**".



11) To display data in json format.

```
> db.book.find().forEach(printjson)
{
        "_id" : ObjectId("605e22e6b0865f1560e0d470"),
        "name" : "Peter",
        "age" : 30,
        "profession" : "Web_Designer"
}
{
        "_id" : ObjectId("605f30b1eb3638cd57492e46"),
        "name" : "Harry",
        "age" : 28,
        "profession" : "Software_Developer"
}
>
```

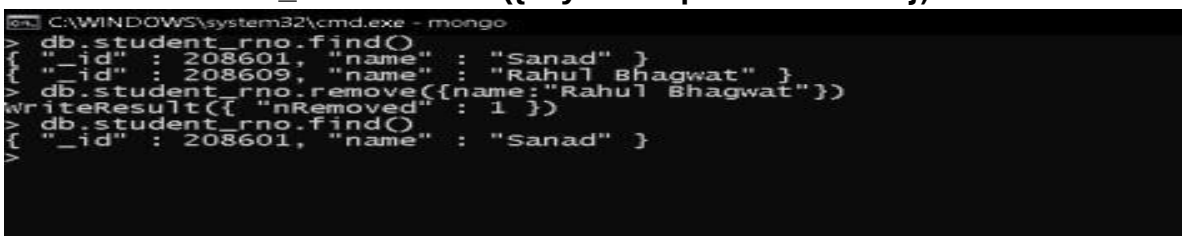
12) To fetch specific data based on criteria we use command: "**db.collection_name.find({keyvaluepair}).pretty()**".

```
C:\WINDOWS\system32\cmd.exe - mongo
> db.student_rno.find({name:"Sanad Soman"}).pretty()
{ "_id" : 208601, "name" : "Sanad Soman" }
> db.student_rno.find()
{ "_id" : 208601, "name" : "Sanad Soman" }
{ "_id" : 208609, "name" : "Rahul Bhagwat" }
>
```

13) To update we use command:"db.collection_name.update({key_value condition},{$set:keyvalue pair})".

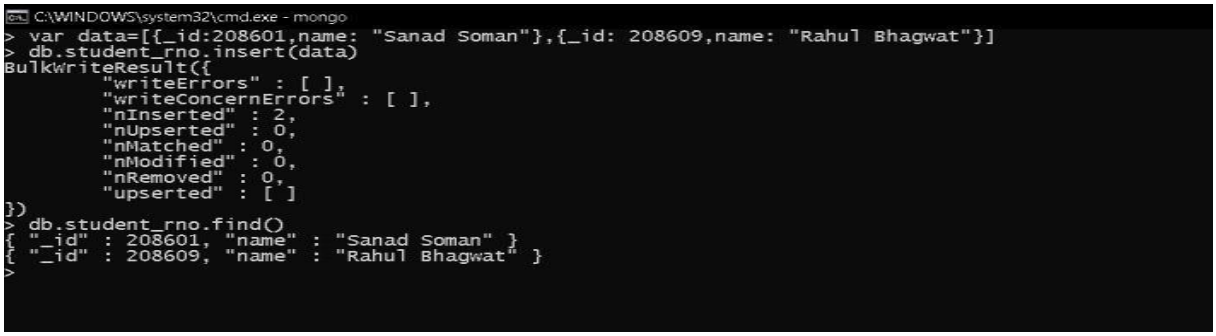
```
C:\WINDOWS\system32\cmd.exe - mongo
> db.student_rno.update({name: "Sanad Soman"},{$set:{name:"Sanad"}})
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
> db.student_rno.find()
{ "_id" : 208601, "name" : "Sanad" }
{ "_id" : 208609, "name" : "Rahul Bhagwat" }
>
```

14) We can remove a record from a collection using command :"**db.collection_name.remove({key-value pair condition"})**".



15) To remove all the records use command: "**db.collection_name.remove({})**".

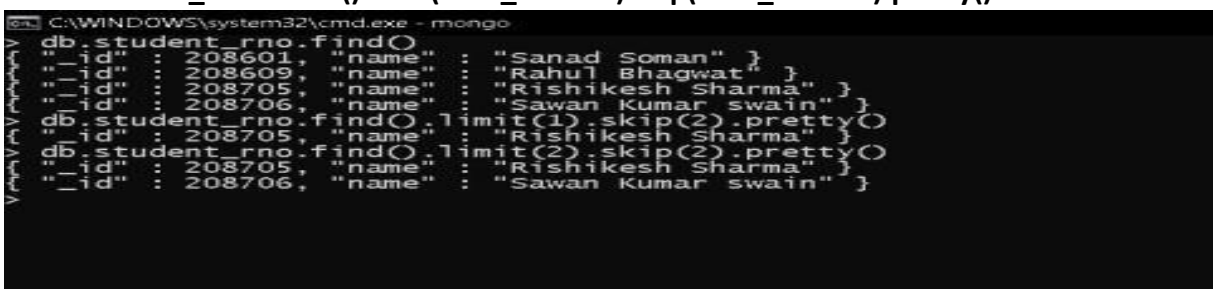
16) To insert multiple records in collection we must create an array of records and pass that array as parameter to command:"**db.collection_name.insert(array_name)**".



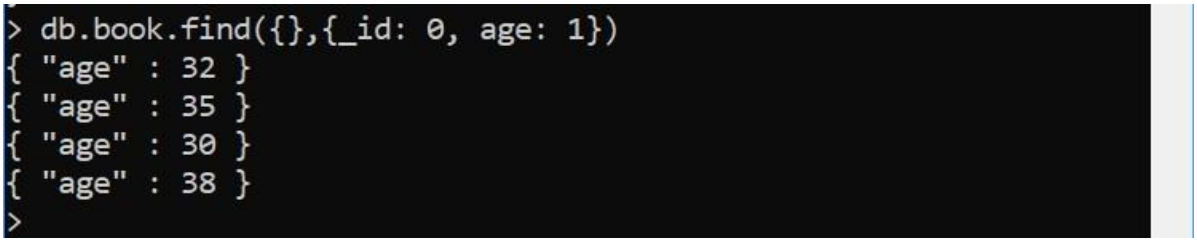
17) To limit records we need to use command **"db.collection_name.find().limit(num_records)".**

18) To skip records use command:

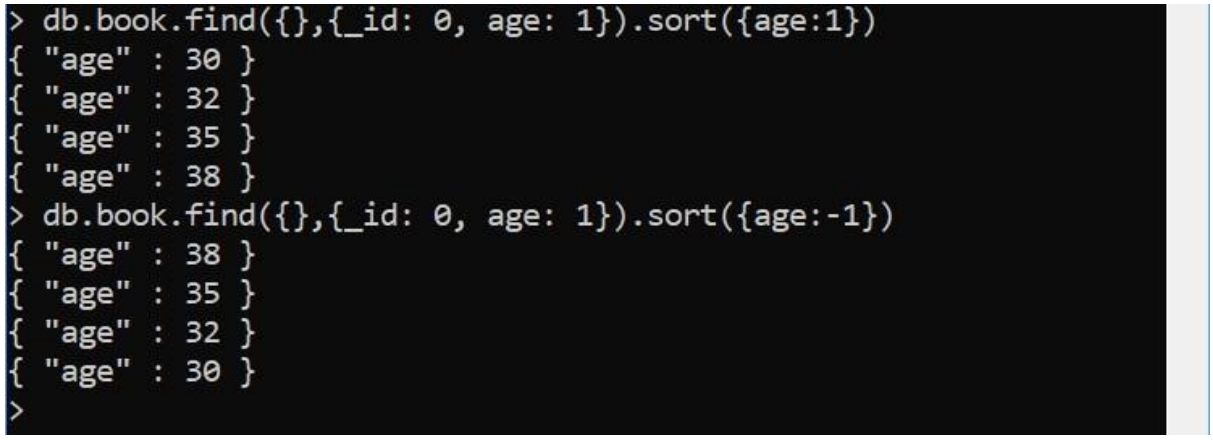
"**db.collection_name.find().limit(num_records).skip(num_records).pretty()**".



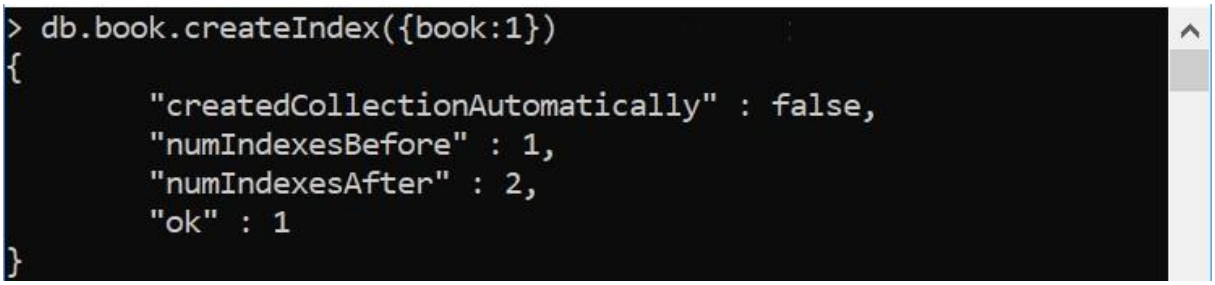
19) To fetch a specific field.



20) To sort the data: 1 will sort it in ascending order and -1 will sort it in descending order.



21) To create the index:



22) To find the index:

```
> db.book.getIndexes()
[
        {
                "v" : 2,
                "key" : {
                        "_id" : 1
                },
                "name" : "_id_"
        },
        {
                "v" : 2,
                "key" : {
                        "book" : 1
                },
                "name" : "book_1"
        }
]
>
```

23) To drop the index:

```
> db.book.dropIndex({book:1})
{ "nIndexesWas" : 2, "ok" : 1 }
>
```

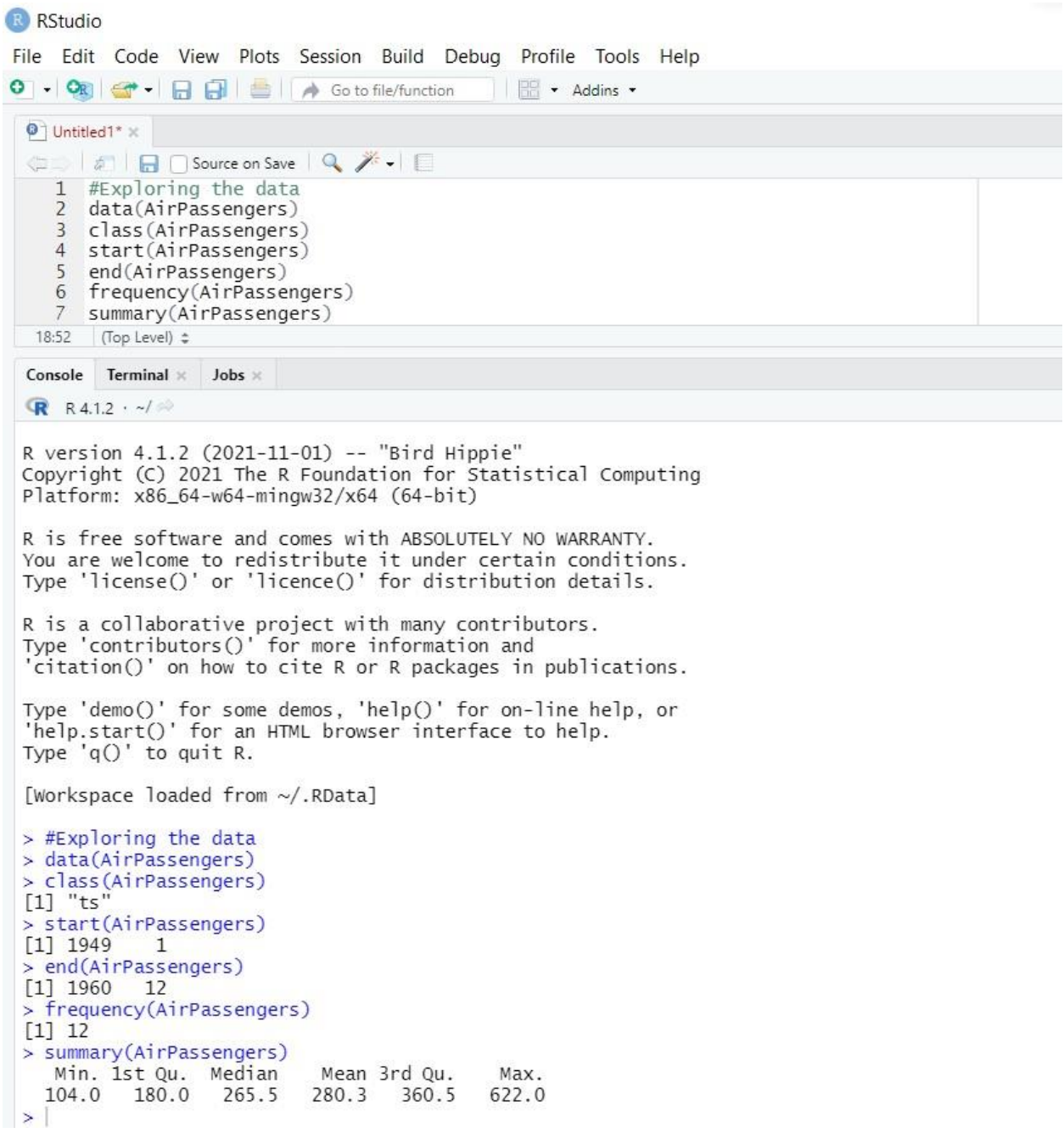
24) To drop all the indexes:

```
> db.book.dropIndexes()
{
        "nIndexesWas" : 1,
        "msg" : "non-_id indexes dropped for collection",
        "ok" : 1
}
>
```

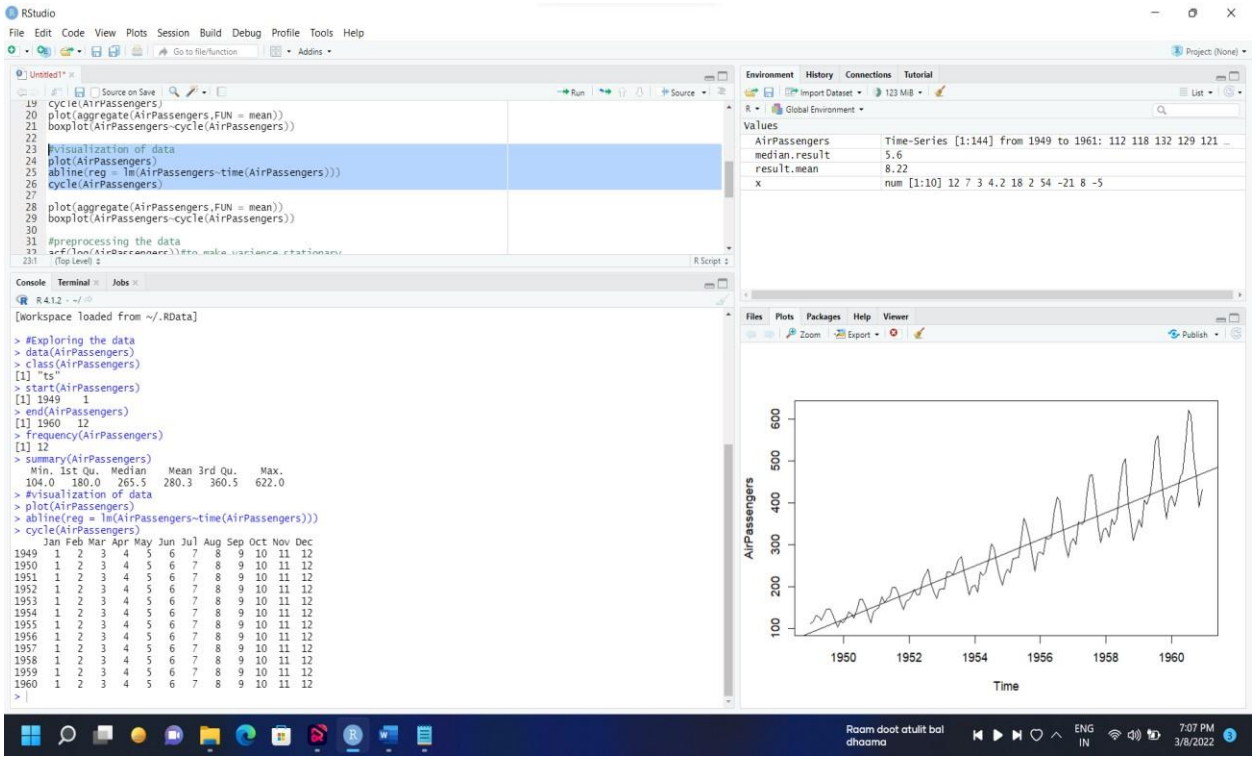
Practical No: 7

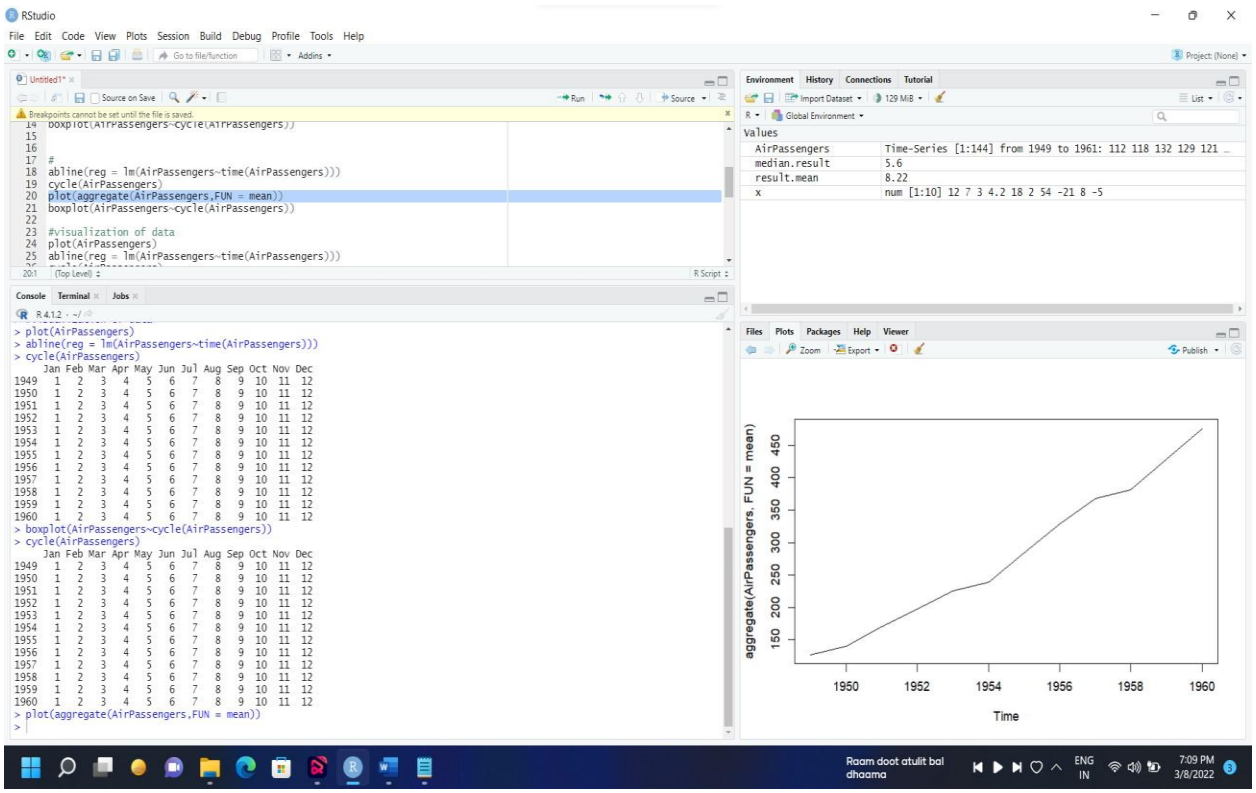
Aim: Practical of Time Series.

1) The AirPassenger dataset in R provides monthly totals of US airline passengers, from 1949 to 1960. This dataset is already of a time series class therefore no further class or date manipulation is required.

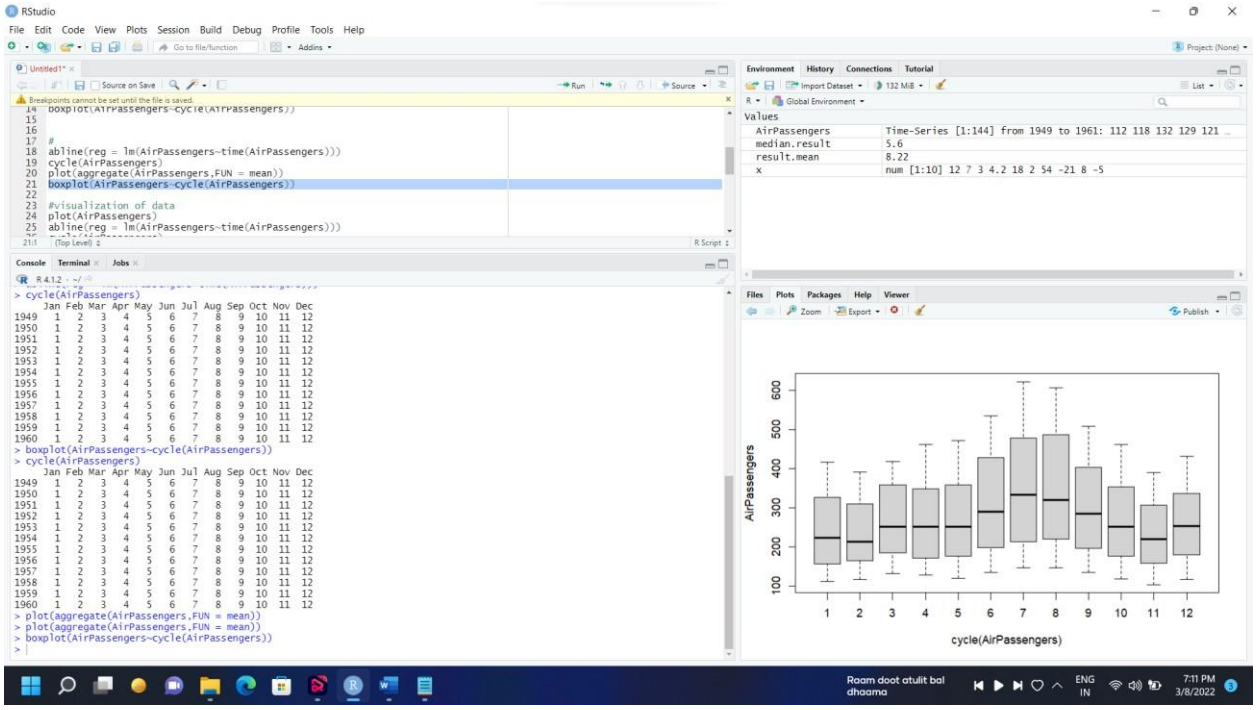


2) From the following plot we can say that the passenger numbers increase over time with each year in linear form.

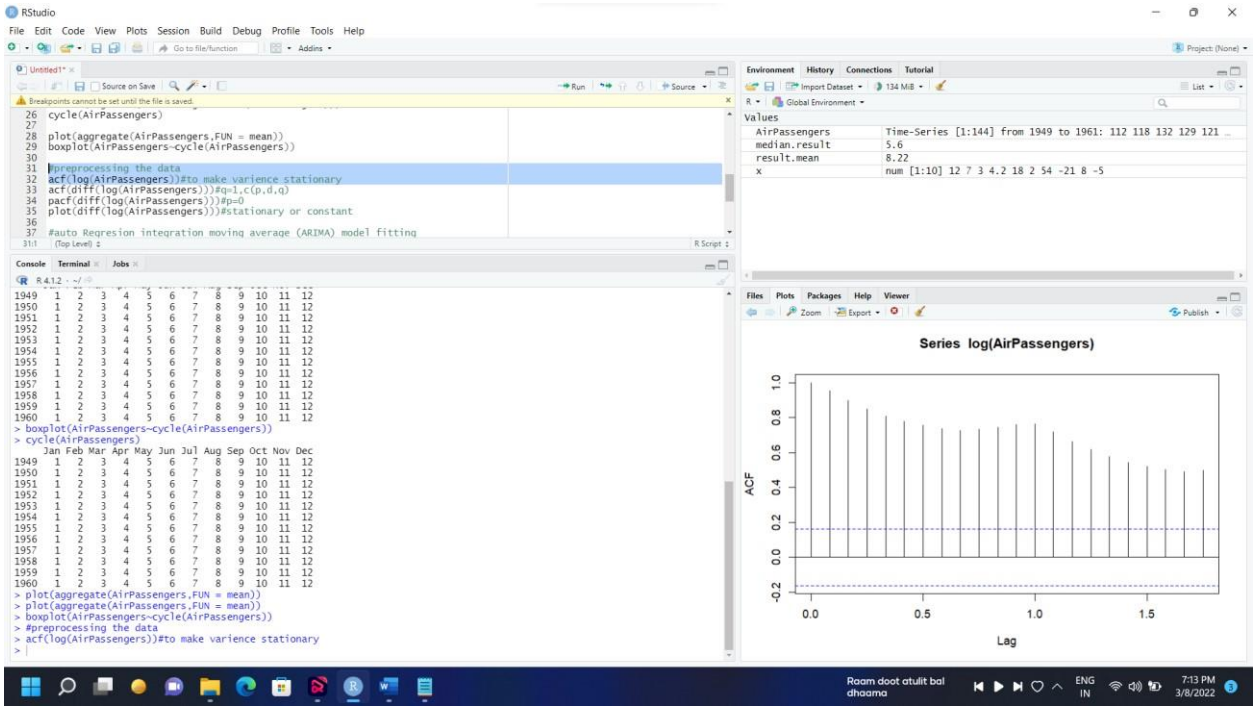




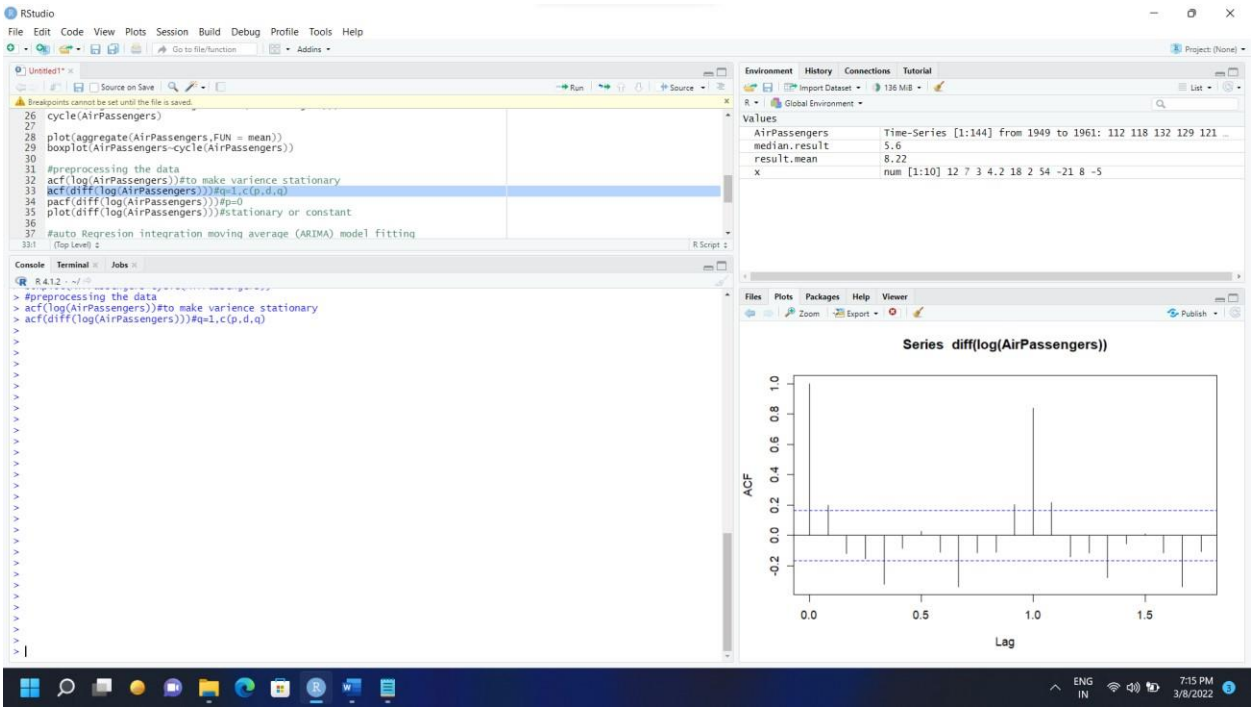
3) In the boxplot there are more passengers travelling in months 7 and 8 with higher means and higher variances than the other months.



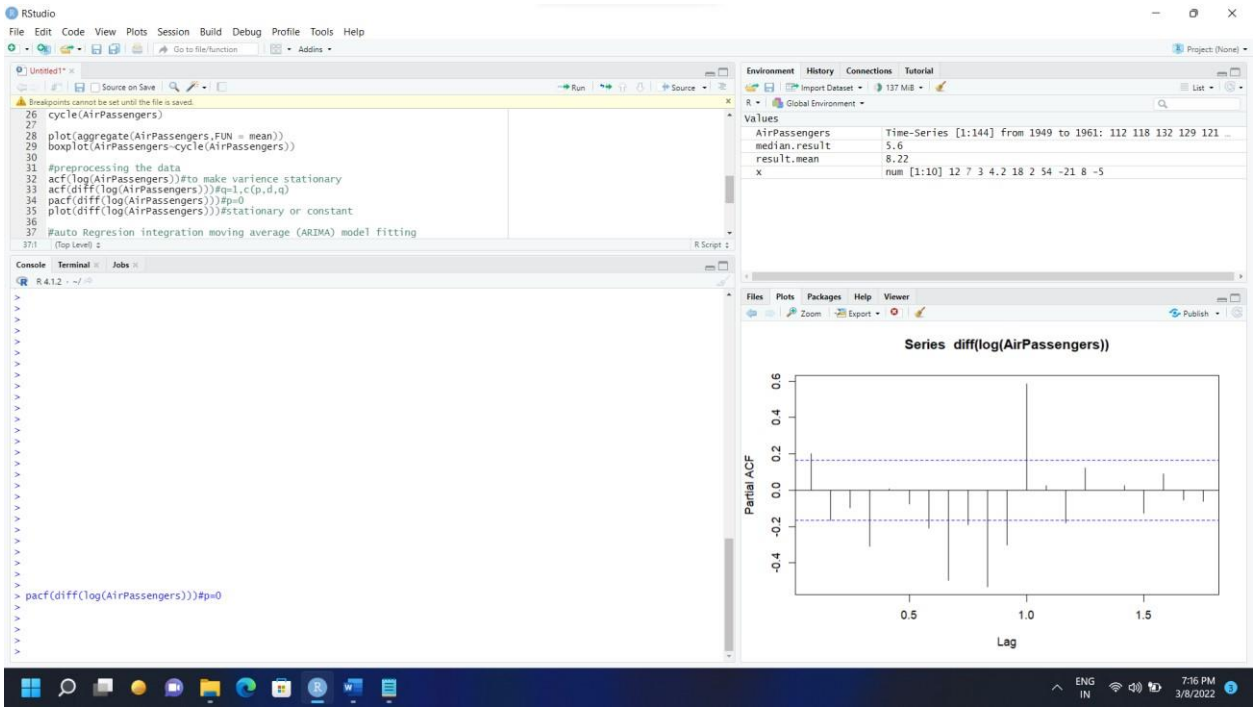
4) To make the variance stationary we will use the Autocorrelation Function (acf).



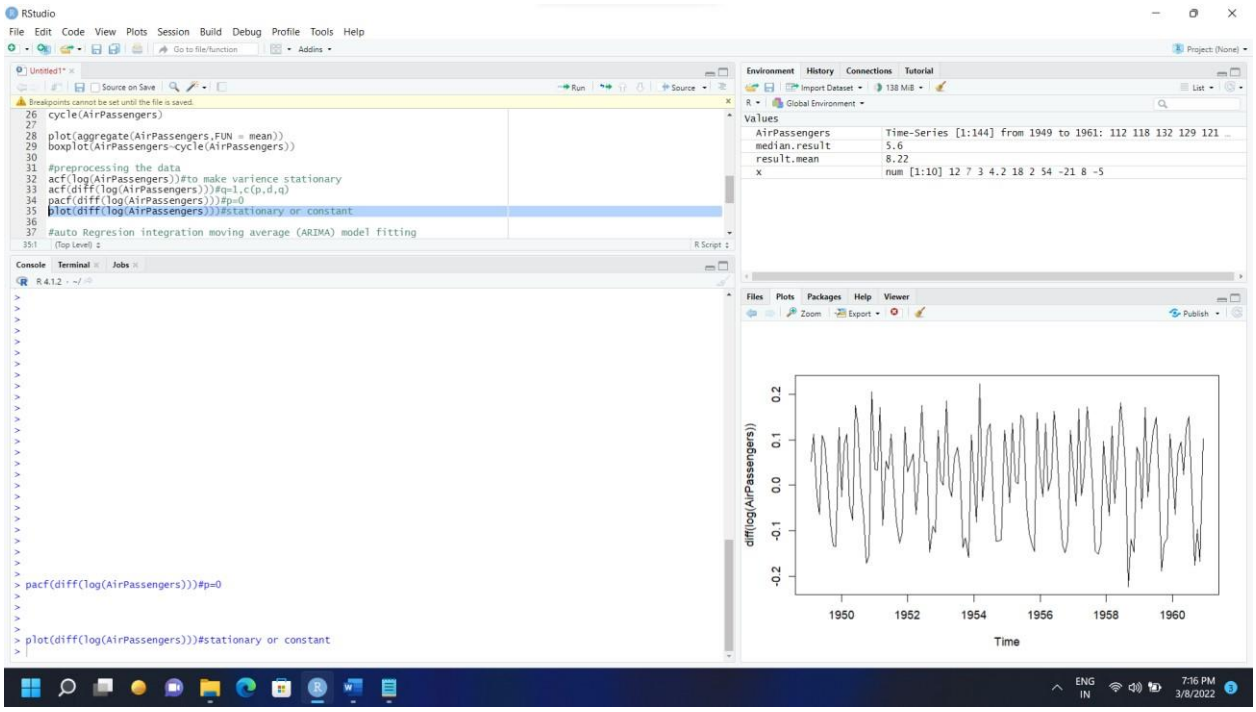
5) We can see that the acf of the residuals is centered around 0.



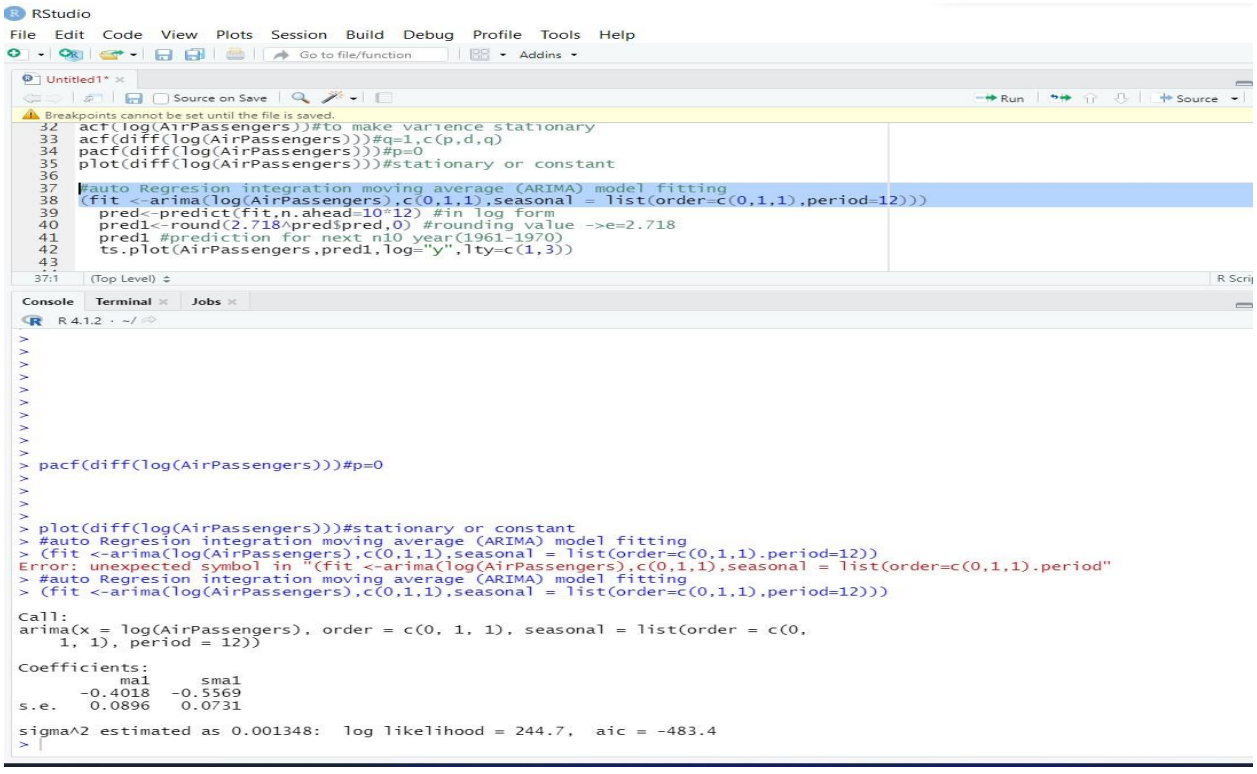
6) Partial acf(Autocorrelation Function).



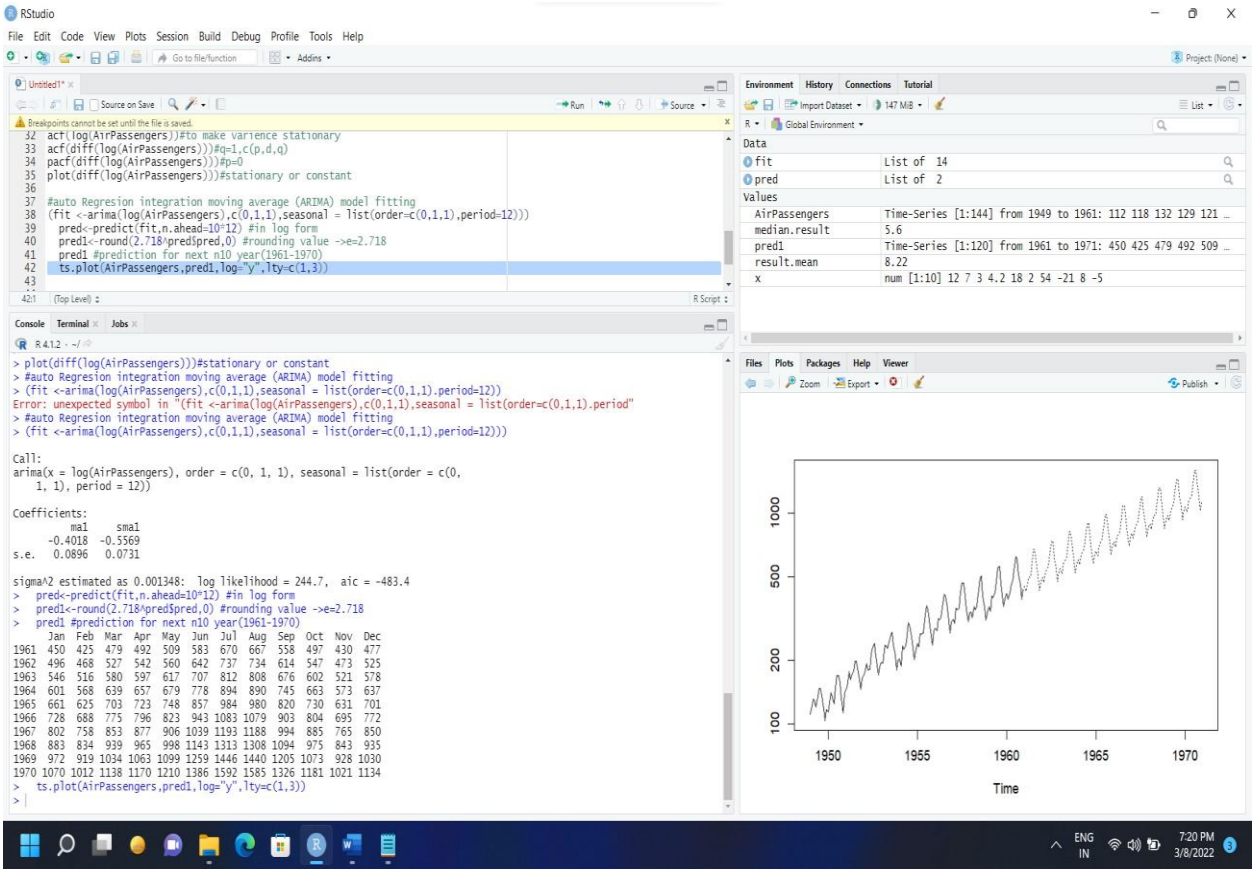
7) From the following plot we can say that the Means and variance are stationary.



8) ARIMA Model fitting.



9) In the following data we can see the prediction value for the next 10 year. Which is from Year 1961 to 1970. In the graph the dotted lines represent that value is increasing in linear form for the next 10 year as well.



Practical No:8

Aim: Practical of Principal Component Analysis

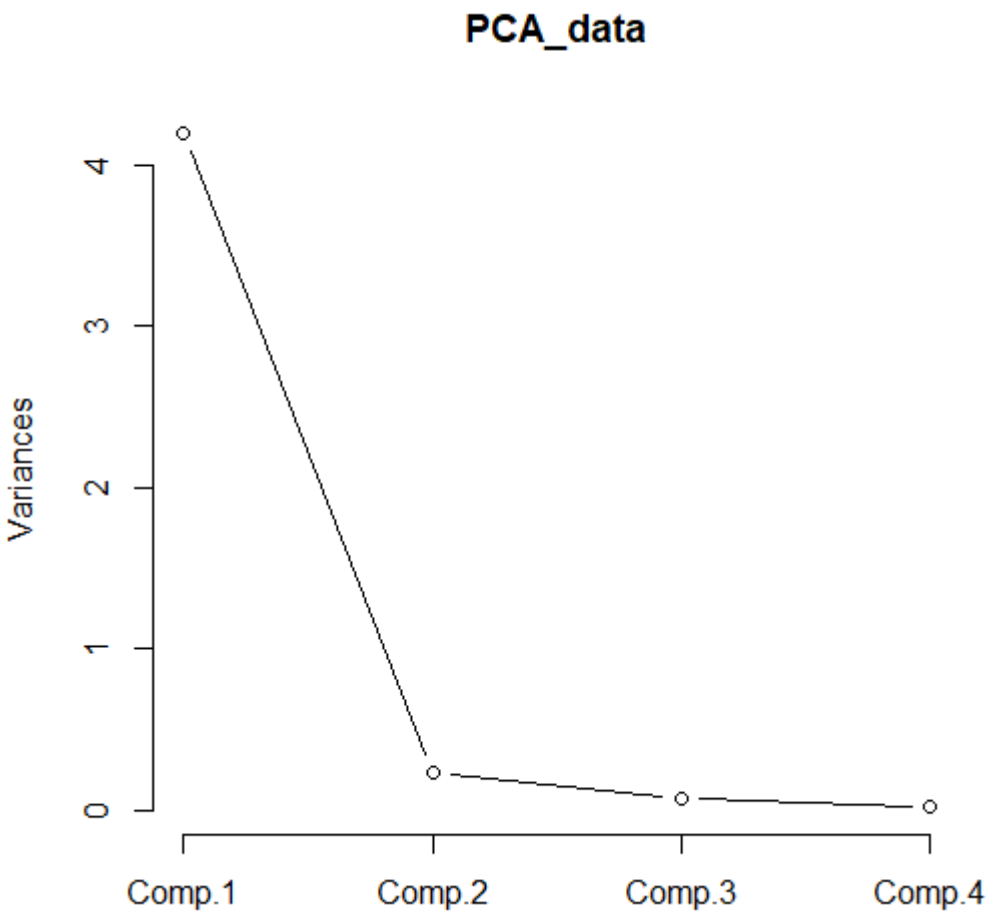
Code:

```
data_iris <- iris[1:4]
Cov_data <- cov(data_iris )
 # Find out the eigenvectors and eigenvalues using the covariance matrix
Eigen_data <- eigen(Cov_data)
# Using the inbuilt function
PCA_data <- princomp(data_iris ,cor="False")
# Let's now compare the output variances
Eigen_data$values
PCA_data$sdev^2
PCA_data$loadings[,1:4]
Eigen_data$vectors
summary(PCA_data)
biplot (PCA_data)
screeplot(PCA_data, type="lines")
#Select the first principal component for the second model
model2 = PCA_data$loadings[,1]
#For the second model, we need to calculate scores by multiplying our loadings with the data
model2_scores <- as.matrix(data_iris) %*% model2
#Loading libraries for naiveBayes model
library(class)
install.packages("e1071")
library(e1071)
#Fitting the first model over the entire data
mod1<-naiveBayes(iris[,1:4], iris[,5])
 #Fitting the second model using the first principal component
mod2<-naiveBayes(model2_scores, iris[,5])
# Accuracy for the first model
table(predict(mod1, iris[,1:4]), iris[,5])
# Accuracy for the second model
 table(predict(mod2, model2_scores), iris[,5])
```

Output:

```
> data_iris <- iris[1:4]
> Cov_data <- cov(data_iris )
>  # Find out the eigenvectors and eigenvalues using the covariance matrix
> Eigen_data <- eigen(Cov_data)
> # Using the inbuilt function
> PCA_data <- princomp(data_iris ,cor="False")
> # Let□s now compare the output variances
> Eigen_data$values
[1] 4.22824171 0.24267075 0.07820950 0.02383509
> PCA_data$sdev^2
    Comp.1     Comp.2     Comp.3     Comp.4
4.20005343 0.24105294 0.07768810 0.02367619
> PCA_data$loadings[,1:4]
                Comp.1      Comp.2      Comp.3     Comp.4
Sepal.Length  0.36138659  0.65658877  0.58202985  0.3154872
Sepal.Width  -0.08452251  0.73016143 -0.59791083 -0.3197231
Petal.Length  0.85667061 -0.17337266 -0.07623608 -0.4798390
Petal.Width   0.35828920 -0.07548102 -0.54583143  0.7536574
> Eigen_data$vectors
            [,1]        [,2]        [,3]       [,4]
[1,]  0.36138659 -0.65658877 -0.58202985  0.3154872
[2,] -0.08452251 -0.73016143  0.59791083 -0.3197231
[3,]  0.85667061  0.17337266  0.07623608 -0.4798390
[4,]  0.35828920  0.07548102  0.54583143  0.7536574
```

```
> summary(PCA_data)
Importance of components:
                          Comp.1     Comp.2     Comp.3      Comp.4
Standard deviation     2.0494032 0.49097143 0.27872586 0.153870700
Proportion of Variance 0.9246187 0.05306648 0.01710261 0.005212184
Cumulative Proportion  0.9246187 0.97768521 0.99478782 1.000000000
> biplot (PCA_data)
> screeplot(PCA_data, type="lines")
> #Select the first principal component for the second model
> model2 = PCA_data$loadings[,1]
> #For the second model, we need to calculate scores by multiplying our loadings with the data
> model2_scores <- as.matrix(data_iris) %*% model2
> #Loading libraries for naiveBayes model
> library(class)
> install.packages("e1071")
Error in install.packages : Updating loaded packages
> library(e1071)
> #Fitting the first model over the entire data
> mod1<-naiveBayes(iris[,1:4], iris[,5])
>  #Fitting the second model using the first principal component
> mod2<-naiveBayes(model2_scores, iris[,5])
> # Accuracy for the first model
> table(predict(mod1, iris[,1:4]), iris[,5])

             setosa versicolor virginica
  setosa         50          0         0
  versicolor      0         47         3
  virginica       0          3        47
> # Accuracy for the second model
>  table(predict(mod2, model2_scores), iris[,5])

             setosa versicolor virginica
  setosa         50          0         0
  versicolor      0         46         5
  virginica       0          4        45
```



Practical No:9

Aim: Practical of Clustering

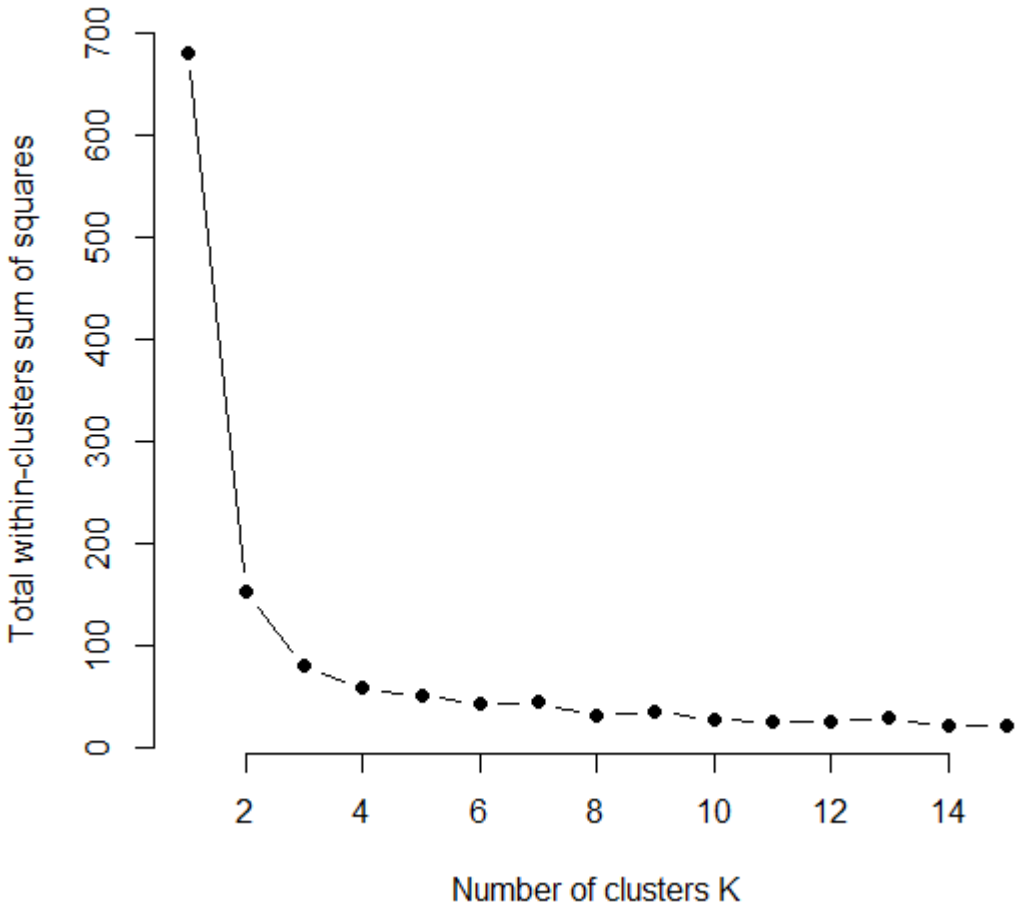
Code:

```
"k-means clustering "
data("iris")
names(iris)
new_data<-subset(iris,select = c(-Species))
new_data
cl<-kmeans(new_data,3)
cl

data <- new_data

wss <- sapply(1:15,
              function(k){kmeans(data, k )$tot.withinss})
wss

plot(1:15, wss,
     type="b", pch = 19, frame = FALSE,
     xlab="Number of clusters K",
     ylab="Total within-clusters sum of squares")

install.packages("cluster")
library(cluster)
clusplot(new_data, cl$cluster, color=TRUE, shade=TRUE,
         labels=2, lines=0)
cl$cluster

cl$centers


"agglomarative clustering "
clusters <- hclust(dist(iris[, 3:4]))
plot(clusters)
install.packages("ggplot")
clusterCut <- cutree(clusters, 3)
table(clusterCut, iris$Species)
ggplot(iris, aes(Petal.Length, Petal.Width, color = iris$Species)) +
  geom_point(alpha = 0.4, size = 3.5) + geom_point(col = clusterCut) +
  scale_color_manual(values = c('black', 'red', 'green'))

clusters <- hclust(dist(iris[, 3:4]), method = 'average')
clusterCut1 <- cutree(clusters, 3)
table(clusterCut1, iris$Species)

plot(clusters)
ggplot(iris, aes(Petal.Length, Petal.Width, color = iris$Species)) +
  geom_point(alpha = 0.4, size = 3.5) + geom_point(col = clusterCut1) +
  scale_color_manual(values = c('black', 'red', 'green'))
```

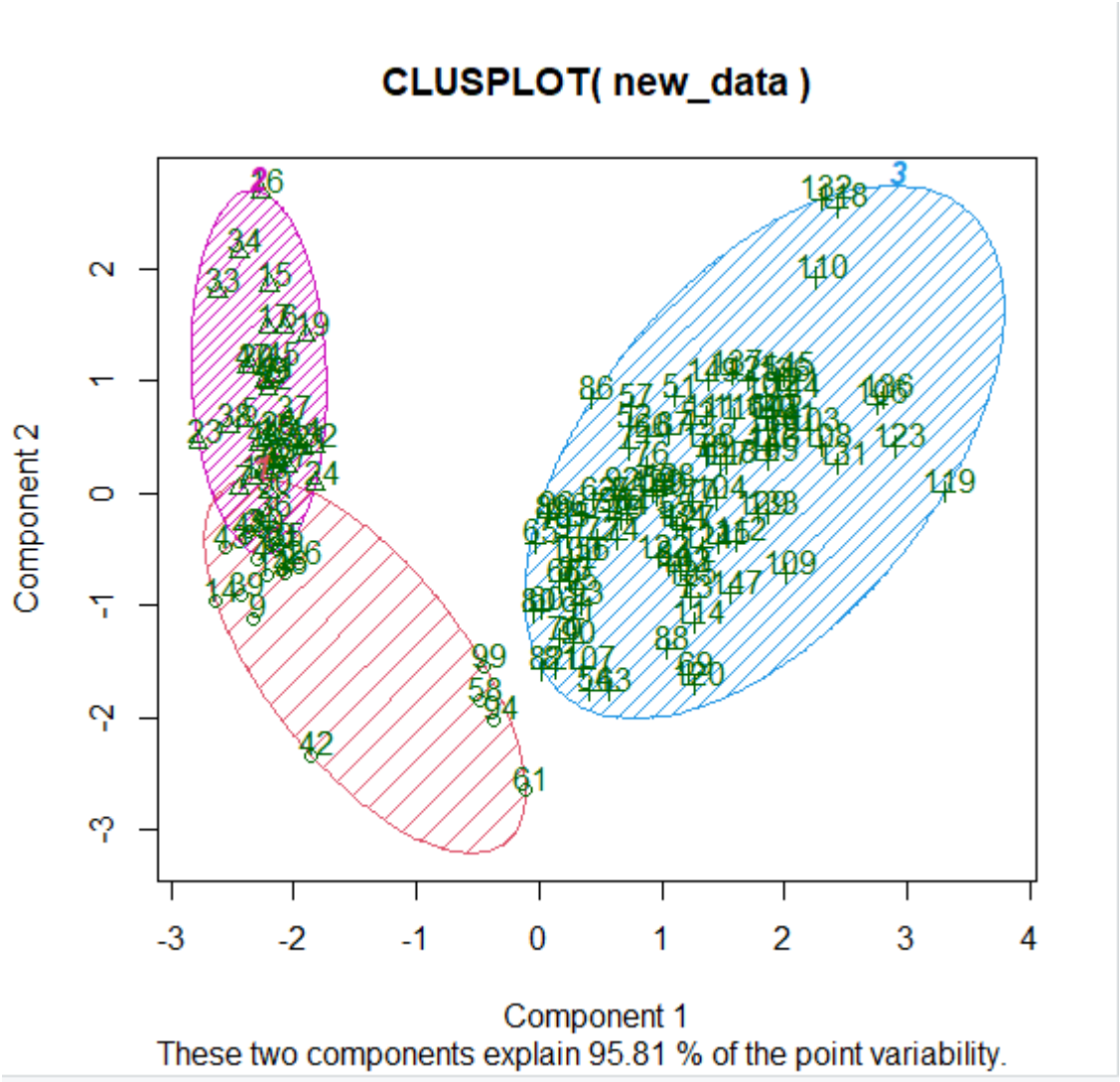
Output:

```
> "k-means clustering "
[1] "k-means clustering "
> data("iris")
> names(iris)
[1] "Sepal.Length" "Sepal.Width"  "Petal.Length" "Petal.Width"  "Species"
> new_data<-subset(iris,select = c(-Species))
> new_data
    Sepal.Length Sepal.Width Petal.Length Petal.Width
1            5.1         3.5          1.4         0.2
2            4.9         3.0          1.4         0.2
3            4.7         3.2          1.3         0.2
4            4.6         3.1          1.5         0.2
5            5.0         3.6          1.4         0.2
6            5.4         3.9          1.7         0.4
7            4.6         3.4          1.4         0.3
8            5.0         3.4          1.5         0.2
9            4.4         2.9          1.4         0.2
10           4.9         3.1          1.5         0.1
11           5.4         3.7          1.5         0.2
12           4.8         3.4          1.6         0.2
13           4.8         3.0          1.4         0.1
14           4.3         3.0          1.1         0.1
15           5.8         4.0          1.2         0.2
16           5.7         4.4          1.5         0.4
17           5.4         3.9          1.3         0.4
18           5.1         3.5          1.4         0.3
19           5.7         3.8          1.7         0.3
20           5.1         3.8          1.5         0.3
21           5.4         3.4          1.7         0.2
22           5.1         3.7          1.5         0.4
23           4.6         3.6          1.0         0.2
24           5.1         3.3          1.7         0.5
25           4.8         3.4          1.9         0.2
26           5.0         3.0          1.6         0.2
27           5.0         3.4          1.6         0.4
28           5.2         3.5          1.5         0.2
29           5.2         3.4          1.4         0.2
30           4.7         3.2          1.6         0.2
31           4.8         3.1          1.6         0.2
32           5.4         3.4          1.5         0.4
33           5.2         4.1          1.5         0.1
34           5.5         4.2          1.4         0.2
35           4.9         3.1          1.5         0.2
36           5.0         3.2          1.2         0.2
37           5.5         3.5          1.3         0.2
38           4.9         3.6          1.4         0.1
39           4.4         3.0          1.3         0.2
40           5.1         3.4          1.5         0.2
41           5.0         3.5          1.3         0.3
42           4.5         2.3          1.3         0.3
43           4.4         3.2          1.3         0.2
44           5.0         3.5          1.6         0.6
45           5.1         3.8          1.9         0.4
46           4.8         3.0          1.4         0.3
47           5.1         3.8          1.6         0.2
48           4.6         3.2          1.4         0.2
49           5.3         3.7          1.5         0.2
```

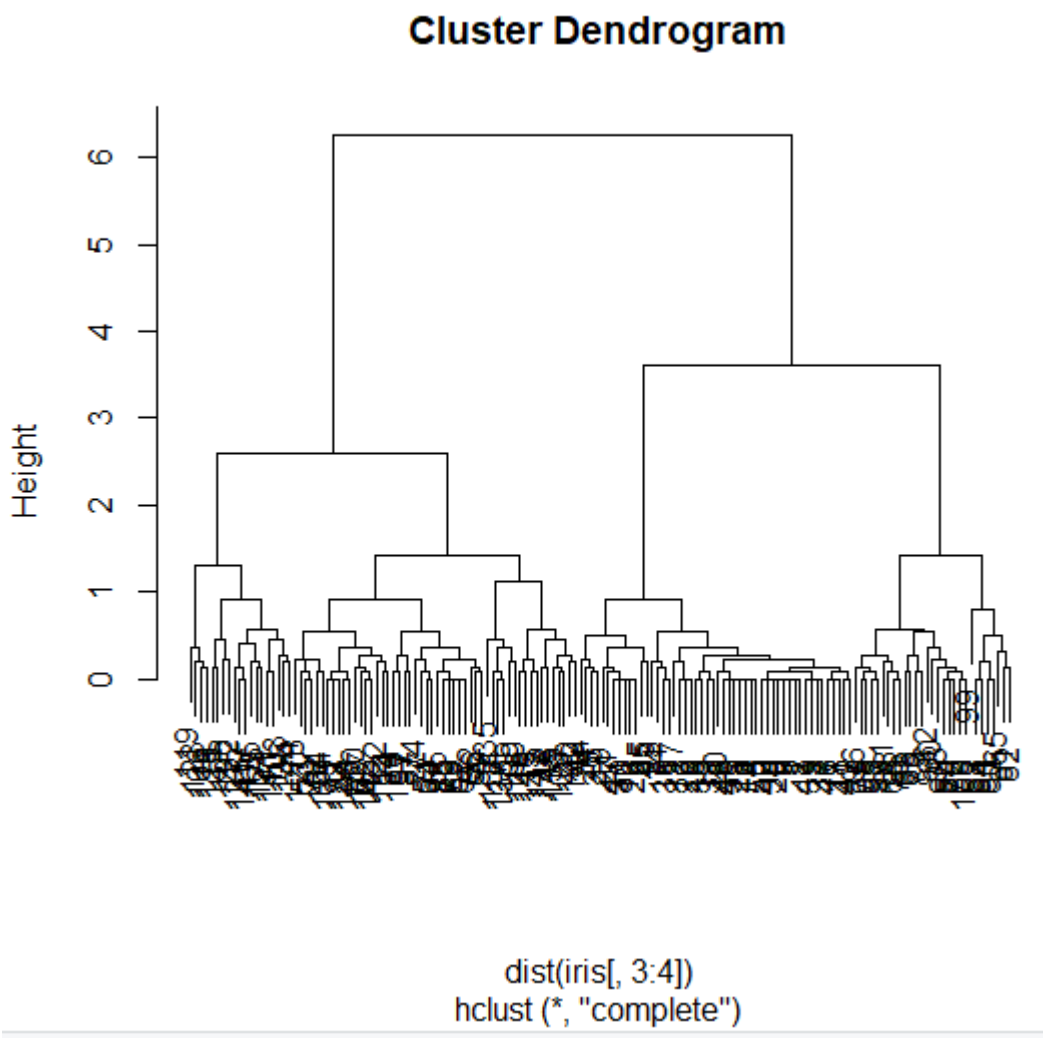
```
> cl<-kmeans(new_data,3)
> cl
K-means clustering with 3 clusters of sizes 21, 33, 96

Cluster means:
  Sepal.Length Sepal.Width Petal.Length Petal.Width
1     4.738095    2.904762     1.790476   0.3523810
2     5.175758    3.624242     1.472727   0.2727273
3     6.314583    2.895833     4.973958   1.7031250

Clustering vector:
  1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39
  2   1   1   1   2   2   2   2   1   1   2   2   1   1   2   2   2   2   2   2   2   2   2   2   1   1   2   2   2   1   1   2   2   2   1   2   2   2   1
 40  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55  56  57  58  59  60  61  62  63  64  65  66  67  68  69  70  71  72  73  74  75  76  77  78
  2   2   1   1   2   2   1   2   1   2   2   3   3   3   3   3   3   3   1   3   3   1   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3
 79  80  81  82  83  84  85  86  87  88  89  90  91  92  93  94  95  96  97  98  99 100 101 102 103 104 105 106 107 108 109 110 111 112 113 114 115 116 117
  3   3   3   3   3   3   3   3   3   3   3   3   3   3   1   3   3   3   3   1   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3
118 119 120 121 122 123 124 125 126 127 128 129 130 131 132 133 134 135 136 137 138 139 140 141 142 143 144 145 146 147 148 149 150
  3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3

Within cluster sum of squares by cluster:
[1]  17.669524   6.432121 118.651875
 (between_SS / total_SS =  79.0 %)

Available components:

[1] "cluster"      "centers"      "totss"        "withinss"     "tot.withinss" "betweenss"    "size"         "iter"         "ifault"
> data <- new_data
>
> wss <- sapply(1:15,
+               function(k){kmeans(data, k )$tot.withinss})
> wss
 [1] 681.37060 152.34795  78.85144  57.26562  50.13655  41.70442  43.55224  30.30321  34.55763  26.30156  25.27685  24.92178  28.18946  20.57355  20.96531
>
> plot(1:15, wss,
+      type="b", pch = 19, frame = FALSE,
+      xlab="Number of clusters K",
+      ylab="Total within-clusters sum of squares")
```



```
> library(cluster)
> clusplot(new_data, cl$cluster, color=TRUE, shade=TRUE,
+          labels=2, lines=0)
> cl$cluster
  1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39
  2   1   1   1   2   2   2   2   1   1   2   2   1   1   2   2   2   2   2   2   2   2   2   1   1   2   2   2   1   1   2   2   2   1   2   2   2   1
 40  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55  56  57  58  59  60  61  62  63  64  65  66  67  68  69  70  71  72  73  74  75  76  77  78
  2   2   1   1   2   2   1   2   1   2   2   3   3   3   3   3   3   3   1   3   3   1   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3
 79  80  81  82  83  84  85  86  87  88  89  90  91  92  93  94  95  96  97  98  99 100 101 102 103 104 105 106 107 108 109 110 111 112 113 114 115 116 117
  3   3   3   3   3   3   3   3   3   3   3   3   3   3   1   3   3   3   3   1   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3
118 119 120 121 122 123 124 125 126 127 128 129 130 131 132 133 134 135 136 137 138 139 140 141 142 143 144 145 146 147 148 149 150
  3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3
>
> cl$centers
  Sepal.Length Sepal.Width Petal.Length Petal.Width
1     4.738095    2.904762     1.790476   0.3523810
2     5.175758    3.624242     1.472727   0.2727273
3     6.314583    2.895833     4.973958   1.7031250
```



```
> "agglomarative clustering "
[1] "agglomarative clustering "
> clusters <- hclust(dist(iris[, 3:4]))
> plot(clusters)
> install.packages("ggplot")
Installing package into 'C:/Users/rohan/OneDrive/Documents/R/win-library/4.0'
(as 'lib' is unspecified)
Warning in install.packages :
  package 'ggplot' is not available for this version of R

A version of this package for your version of R might be available elsewhere,
see the ideas at
https://cran.r-project.org/doc/manuals/r-patched/R-admin.html#Installing-packages
> clusterCut <- cutree(clusters, 3)
> table(clusterCut, iris$Species)

clusterCut setosa versicolor virginica
         1     50          0         0
         2      0         21        50
         3      0         29         0
> ggplot(iris, aes(Petal.Length, Petal.Width, color = iris$Species)) +
+   geom_point(alpha = 0.4, size = 3.5) + geom_point(col = clusterCut) +
+   scale_color_manual(values = c('black', 'red', 'green'))
```



Practical No:10

Aim: Practical of Logistic Regression.

Code:

```
loan<-read.csv(file.choose(),header=T,sep=",")
head(loan)
summary(loan)
str(loan)
loan$AGE<-as.factor(loan$AGE)
str(loan)
names(loan)
"creating model"
model1<-glm(DEFAULTER~.,family = binomial,data = loan)
summary(model1)
"global testing for the acceptance of the model"
null<-glm(DEFAULTER~1,family = binomial,data=loan)
anova(null,model1,test = "Chisq")
"predicting the probilities"
loan$predprob<-round(fitted(model1),2)
head(loan)
"classification and misclassification analysis "
library(gmodels)
table(loan$DEFAULTER,fitted(model1)>0.5)
sens<-95/(88+95)*100
sens
spc<-478/(478+39)*100
spc

"check the trade off between sensivity and specificity using different cut off values"
table(loan$DEFAULTER,fitted(model1)>0.1)
table(loan$DEFAULTER,fitted(model1)>0.2)
table(loan$DEFAULTER,fitted(model1)>0.3)
table(loan$DEFAULTER,fitted(model1)>0.4)
table(loan$DEFAULTER,fitted(model1)>0.5)

"goodness of fit using receiver Operational Curve "
pred<-predict(model1,loan,type="response")
install.packages("ROCR")
library(ROCR)
rocrpred<-prediction(pred,loan$DEFAULTER)
rocrperf<-performance(rocrpred,"tpr","fpr")
"to check proper cut off point"
plot(rocrperf,colorize=TRUE,print.cutoffs.at=seq(0.1,by=0.1))

"to check coeficients"
coef(model1)
exp(coef(model1))

"as credit to debit ratio of person incerases by 1 unit ,odds of the event increses by 77%"
"model validation same as linear regression"
"variable selection same as linear regresiion"
```

Output:

```
> loan<-read.csv(file.choose(),header=T,sep=",")
> head(loan)
  SN AGE EMPLOY ADDRESS DEBTINC CREDDEBT OTHDEBT DEFAULTER
1  1   3     17      12     9.3    11.36    5.01         1
2  2   1     10       6    17.3     1.36    4.00         0
3  3   2     15      14     5.5     0.86    2.17         0
4  4   3     15      14     2.9     2.66    0.82         0
5  5   1      2       0    17.3     1.79    3.06         1
6  6   3      5       5    10.2     0.39    2.16         0
> summary(loan)
       SN              AGE            EMPLOY          ADDRESS          DEBTINC         CREDDEBT          OTHDEBT          DEFAULTER
 Min.   :  1.0   Min.   :1.000   Min.   : 0.000   Min.   : 0.000   Min.   : 0.40   Min.   : 0.010   Min.   : 0.050   Min.   :0.0000
 1st Qu.:175.8   1st Qu.:1.000   1st Qu.: 3.000   1st Qu.: 3.000   1st Qu.: 5.00   1st Qu.: 0.370   1st Qu.: 1.048   1st Qu.:0.0000
 Median :350.5   Median :2.000   Median : 7.000   Median : 7.000   Median : 8.60   Median : 0.855   Median : 1.985   Median :0.0000
 Mean   :350.5   Mean   :1.903   Mean   : 8.389   Mean   : 8.279   Mean   :10.26   Mean   : 1.553   Mean   : 3.058   Mean   :0.2614
 3rd Qu.:525.2   3rd Qu.:2.000   3rd Qu.:12.000   3rd Qu.:12.000   3rd Qu.:14.12   3rd Qu.: 1.905   3rd Qu.: 3.928   3rd Qu.:1.0000
 Max.   :700.0   Max.   :3.000   Max.   :31.000   Max.   :34.000   Max.   :41.30   Max.   :20.560   Max.   :27.030   Max.   :1.0000
> str(loan)
'data.frame':   700 obs. of  8 variables:
 $ SN       : int  1 2 3 4 5 6 7 8 9 10 ...
 $ AGE      : int  3 1 2 3 1 3 2 3 1 2 ...
 $ EMPLOY   : int  17 10 15 15 2 5 20 12 3 0 ...
 $ ADDRESS  : int  12 6 14 14 0 5 9 11 4 13 ...
 $ DEBTINC  : num  9.3 17.3 5.5 2.9 17.3 10.2 30.6 3.6 24.4 19.7 ...
 $ CREDDEBT : num  11.36 1.36 0.86 2.66 1.79 ...
 $ OTHDEBT  : num  5.01 4 2.17 0.82 3.06 ...
 $ DEFAULTER: int  1 0 0 0 1 0 0 0 1 0 ...
> loan$AGE<-as.factor(loan$AGE)
> str(loan)
'data.frame':   700 obs. of  8 variables:
 $ SN       : int  1 2 3 4 5 6 7 8 9 10 ...
 $ AGE      : Factor w/ 3 levels "1","2","3": 3 1 2 3 1 3 2 3 1 2 ...
 $ EMPLOY   : int  17 10 15 15 2 5 20 12 3 0 ...
 $ ADDRESS  : int  12 6 14 14 0 5 9 11 4 13 ...
 $ DEBTINC  : num  9.3 17.3 5.5 2.9 17.3 10.2 30.6 3.6 24.4 19.7 ...
 $ CREDDEBT : num  11.36 1.36 0.86 2.66 1.79 ...
 $ OTHDEBT  : num  5.01 4 2.17 0.82 3.06 ...
 $ DEFAULTER: int  1 0 0 0 1 0 0 0 1 0 ...
> names(loan)
[1] "SN"        "AGE"       "EMPLOY"    "ADDRESS"   "DEBTINC"   "CREDDEBT"  "OTHDEBT"   "DEFAULTER"
> "creating model"
[1] "creating model"
> model1<-glm(DEFAULTER~.,family = binomial,data = loan)
> summary(model1)

Call:
glm(formula = DEFAULTER ~ ., family = binomial, data = loan)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.2903  -0.6562  -0.3092   0.2481   2.8942
```

```
Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept) -0.9571221  0.3267254  -2.929 0.003396 **
SN           0.0004689  0.0005275   0.889 0.374064
AGE2         0.2523596  0.2667267   0.946 0.344080
AGE3         0.6089838  0.3612509   1.686 0.091841 .
EMPLOY      -0.2607294  0.0318825  -8.178 2.89e-16 ***
ADDRESS     -0.0995857  0.0223934  -4.447 8.70e-06 ***
DEBTINC      0.0857756  0.0221648   3.870 0.000109 ***
CREDDEBT     0.5618315  0.0885848   6.342 2.26e-10 ***
OTHDEBT      0.0212219  0.0570848   0.372 0.710071
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 804.36  on 699  degrees of freedom
Residual deviance: 552.62  on 691  degrees of freedom
AIC: 570.62

Number of Fisher Scoring iterations: 6

> "global testing for the acceptance of the model"
[1] "global testing for the acceptance of the model"
> null<-glm(DEFAULTER~1,family = binomial,data=loan)
> anova(null,model1,test = "Chisq")
Analysis of Deviance Table

Model 1: DEFAULTER ~ 1
Model 2: DEFAULTER ~ SN + AGE + EMPLOY + ADDRESS + DEBTINC + CREDDEBT +
    OTHDEBT
  Resid. Df Resid. Dev Df Deviance  Pr(>Chi)
1       699     804.36
2       691     552.62  8   251.75 < 2.2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
> "predicting the probilities"
[1] "predicting the probilities"
> loan$predprob<-round(fitted(model1),2)
> head(loan)
  SN AGE EMPLOY ADDRESS DEBTINC CREDDEBT OTHDEBT DEFAULTER predprob
1  1   3     17      12     9.3    11.36    5.01         1     0.79
2  2   1     10       6    17.3     1.36    4.00         0     0.14
3  3   2     15      14     5.5     0.86    2.17         0     0.01
4  4   3     15      14     2.9     2.66    0.82         0     0.02
5  5   1      2       0    17.3     1.79    3.06         1     0.75
6  6   3      5       5    10.2     0.39    2.16         0     0.27
> "classification and misclassification analysis "
[1] "classification and misclassification analysis "
> library(gmodels)
> table(loan$DEFAULTER,fitted(model1)>0.5)
```

```
> table(loan$DEFAULTER,fitted(model1)>0.5)

    FALSE TRUE
  0   478   39
  1    88   95
> sens<-95/(88+95)*100
> sens
[1] 51.91257
> spc<-478/(478+39)*100
> spc
[1] 92.45648
>
> "check the trade off between sensivity and specificity using different cut off values"
[1] "check the trade off between sensivity and specificity using different cut off values"
> table(loan$DEFAULTER,fitted(model1)>0.1)

    FALSE TRUE
  0   250  267
  1    13  170
> table(loan$DEFAULTER,fitted(model1)>0.2)

    FALSE TRUE
  0   346  171
  1    25  158
> table(loan$DEFAULTER,fitted(model1)>0.3)

    FALSE TRUE
  0   407  110
  1    43  140
> table(loan$DEFAULTER,fitted(model1)>0.4)

    FALSE TRUE
  0   448   69
  1    69  114
> table(loan$DEFAULTER,fitted(model1)>0.5)

    FALSE TRUE
  0   478   39
  1    88   95
>
> "goodness of fit using receiver Operational Curve "
[1] "goodness of fit using receiver Operational Curve "
> pred<-predict(model1,loan,type="response")
> install.packages("ROCR")
Error in install.packages : Updating loaded packages
> library(ROCR)
> rocrpred<-prediction(pred,loan$DEFAULTER)
> rocrperf<-performance(rocrpred,"tpr","fpr")
> "to check proper cut off point"
[1] "to check proper cut off point"
> plot(rocrperf,colorize=TRUE,print.cutoffs.at=seq(0.1,by=0.1))
>
> "to check coeficients"
[1] "to check coeficients"
> coef(model1)
```

```
>  to check coeficients
[1] "to check coeficients"
> coef(model1)
  (Intercept)            SN          AGE2          AGE3        EMPLOY       ADDRESS       DEBTINC      CREDDEBT       OTHDEBT
-0.9571221261  0.0004689067  0.2523595660  0.6089837712 -0.2607293672 -0.0995856709  0.0857755990  0.5618315081  0.0212219271
> exp(coef(model1))
(Intercept)          SN        AGE2        AGE3      EMPLOY     ADDRESS     DEBTINC    CREDDEBT     OTHDEBT
  0.3839964   1.0004690   1.2870587   1.8385620   0.7704894   0.9052124   1.0895618   1.7538818   1.0214487
>
> "as credit to debit ratio of person incerases by 1 unit ,odds of the event increses by 77%"
[1] "as credit to debit ratio of person incerases by 1 unit ,odds of the event increses by 77%"
> "model validation same as linear regression"
[1] "model validation same as linear regression"
> "variable selection same as linear regresiion"
[1] "variable selection same as linear regresiion"
> install.packages("ROCR")
Warning in install.packages :
  package 'ROCR' is in use and will not be installed
>
```