Name: Sumit Singh

Roll No: 380

Class: TYBSC CS A

Subject: Data Science

Practical No : 1

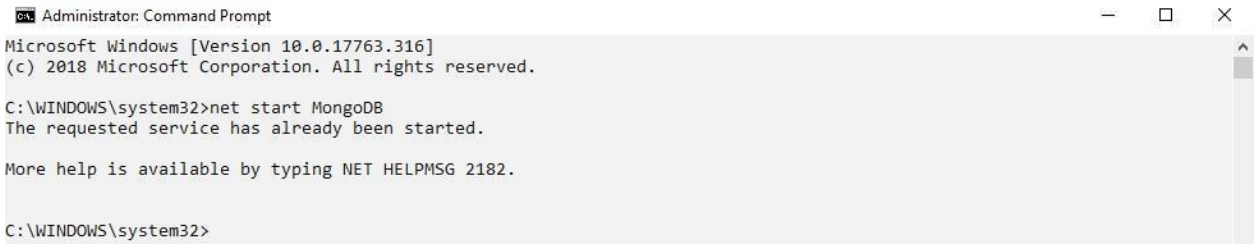
Aim: Practical of Data Collection, Data Curation and Management for large scale data system (such as MongoDB).

Procedure:

Install MongoDB

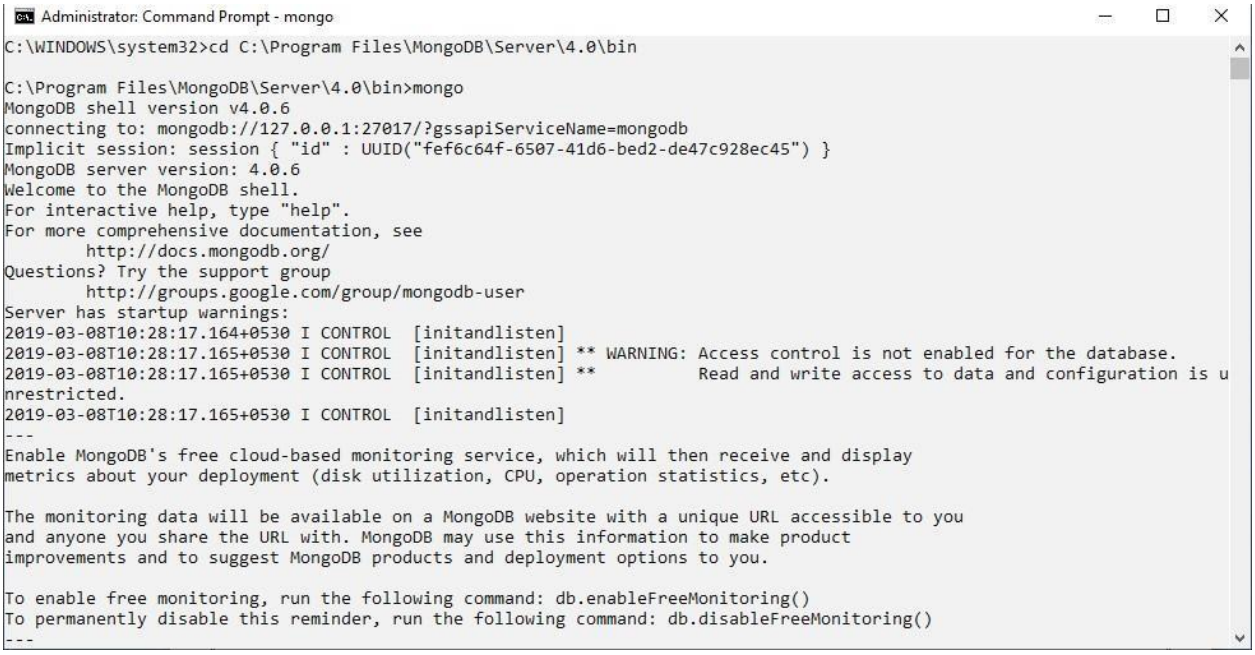
Run CMD as administrator

Start MongoDB via cmd



Enter the MongoDB bin folder via the cmd.

Type Mongo



Check Current Database

```
> db
test
>
```

Use database library

```
> use library
switched to db library
>
```

Create Collection(table)

```
> db.createCollection("book_info")
{ "ok" : 1 }
>
```

Insert document (records) in collection

```
> db.book_info.insert({title:'Programming with Java',status_info:{accession_no:BS0001,status:ISSUES},)
...
...
> db.book_info.insert({title:'PROGRAMMING WITH JAVA',
... status_info:{accession_no:'BS0001',status:'ISSUED'},
... author:'EBALAGURUSWAMY',
... cost:'350',
... publisher:{name:'TMH',location:'banglore'}})
WriteResult({ "nInserted" : 1 })
```

To insert multiple records in one column

```
> db.book_info.insert({title:'DISTRIBUTED SYSTEMS', status_info:[{accession_no:'BS0005',status:'ISSUED'},{accession_no:
BS0006',status:'ISSUED'},{accesion_no:'BS0007',status:'ISSUED'},{accession_no:'BS0008',status:'AVAIL'}], author:'ANDREW
TANENBAUM', cost:'350', publisher:{name:'PEARSON',location:'ANDHERI'}})
WriteResult({ "nInserted" : 1 })
```

To view records of a specific document:

```
> db.book_info.find({title:'LET US C'})
{ "_id" : ObjectId("5c8201d69cff5f2a54951b57"), "title" : "LET US C", "status_info" : { "accession_no" : "BS0009", "stat
us" : "AVAIL" }, "author" : "KANETKAR YASHWANT P", "cost" : "600", "publisher" : { "name" : "B.P.B", "location" : "RAJAS
THAN" } }
>
```

To view all records:

```
> db.book_info.find({})
{ "_id" : ObjectId("5c81fd569cff5f2a54951b52"), "title" : "PROGRAMMING WITH JAVA", "status_info" : { "accession_no" : "B
S0001", "status" : "ISSUED" }, "author" : "EBALAGURUSWAMY", "cost" : "350", "publisher" : { "name" : "TMH", "location" :
 "banglore" } }
{ "_id" : ObjectId("5c81fdd89cff5f2a54951b53"), "title" : "ASP.NET3.5 VB 2008", "status_info" : { "accession_no" : "BS00
02", "status" : "AVAIL" }, "author" : "ANNE BOEHM", "cost" : "650", "publisher" : { "name" : "MURACH", "location" : "JAI
PUR" } }
{ "_id" : ObjectId("5c81fe5e9cff5f2a54951b54"), "title" : "PRPGRAMMING IN VB", "status_info" : { "accession_no" : "BS000
3", "status" : "ISSUED" }, "author" : "JULIA CASE BRADLEY", "cost" : "600", "publisher" : { "name" : "TMH", "location" :
 "BANGLORE" } }
{ "_id" : ObjectId("5c81feaf9cff5f2a54951b55"), "title" : "DATABASE SYSTEM CONCEPTS", "status_info" : { "accession_no" :
 "BS0004", "status" : "ISSUED" }, "author" : "KORTH SUDARSHAN", "cost" : "500", "publisher" : { "name" : "TMH", "locatio
n" : "BANGLORE" } }
{ "_id" : ObjectId("5c82005f9cff5f2a54951b56"), "title" : "DISTRIBUTED SYSTEMS", "status_info" : [ { "accession_no" : "B
S0005", "status" : "ISSUED" }, { "accession_no" : "BS0006", "status" : "ISSUED" }, { "accesion_no" : "BS0007", "status"
: "ISSUED" }, { "accession_no" : "BS0008", "status" : "AVAIL" } ], "author" : "ANDREW TANENBAUM", "cost" : "350", "publi
sher" : { "name" : "PEARSON", "location" : "ANDHERI" } }
{ "_id" : ObjectId("5c8201d69cff5f2a54951b57"), "title" : "LET US C", "status_info" : { "accession_no" : "BS0009", "stat
us" : "AVAIL" }, "author" : "KANETKAR YASHWANT P", "cost" : "600", "publisher" : { "name" : "B.P.B", "location" : "RAJAS
THAN" } }
{ "_id" : ObjectId("5c8202f69cff5f2a54951b58"), "title" : "MODERN DIGITAL ELECTRONICS", "status_info" : [ { "accession_n
o" : "BS010", "status" : "ISSUED" }, { "accession_no" : "BS011", "status" : "AVAIL" } ], "author" : "JAIN R.P", "cost" :
 "650", "publisher" : { "name" : "TMH", "location" : "BANGLORE" } }
```

To view accession numbers:

```
> db.book_info.find({},{'status_info.accession_no':1})
{ "_id" : ObjectId("5c81fd569cff5f2a54951b52"), "status_info" : { "accession_no" : "BS0001" } }
{ "_id" : ObjectId("5c81fdd89cff5f2a54951b53"), "status_info" : { "accession_no" : "BS0002" } }
{ "_id" : ObjectId("5c81fe5e9cff5f2a54951b54"), "status_info" : { "accession_no" : "BS0003" } }
{ "_id" : ObjectId("5c81feaf9cff5f2a54951b55"), "status_info" : { "accession_no" : "BS0004" } }
{ "_id" : ObjectId("5c82005f9cff5f2a54951b56"), "status_info" : [ { "accession_no" : "BS0005" }, { "accession_no" : "BS0
006" }, {  }, { "accession_no" : "BS0008" } ] }
{ "_id" : ObjectId("5c8201d69cff5f2a54951b57"), "status_info" : { "accession_no" : "BS0009" } }
{ "_id" : ObjectId("5c8202f69cff5f2a54951b58"), "status_info" : [ { "accession_no" : "BS010" }, { "accession_no" : "BS01
1" } ] }
```

To print records with accession no “BS0001”

```
> db.book_info.find({'status_info.accession_no':'BS0001'})
{ "_id" : ObjectId("5c81fd569cff5f2a54951b52"), "title" : "PROGRAMMING WITH JAVA", "status_info" : { "accession_no" : "BS0001", "status" : "ISSUED" }, "author" : "EBALAGURUSWAMY", "cost" : "350", "publisher" : { "name" : "TMH", "location" : "banglore" } }
```

To print data in JSON format

Administrator: Command Prompt - mongo

```
> db.book_info.find().forEach(printjson)
{
        "_id" : ObjectId("5c81fd569cff5f2a54951b52"),
        "title" : "PROGRAMMING WITH JAVA",
        "status_info" : {
                "accession_no" : "BS0001",
                "status" : "ISSUED"
        },
        "author" : "EBALAGURUSWAMY",
        "cost" : "350",
        "publisher" : {
                "name" : "TMH",
                "location" : "banglore"
        }
}
{
        "_id" : ObjectId("5c81fdd89cff5f2a54951b53"),
        "title" : "ASP.NET3.5 VB 2008",
        "status_info" : {
                "accession_no" : "BS0002",
                "status" : "AVAIL"
        },
        "author" : "ANNE BOEHM",
        "cost" : "650",
        "publisher" : {
                "name" : "MURACH",
                "location" : "JAIPUR"
        }
}
{
```

To print the details of book where cost is more than 500

```
> db.book_info.find({"cost":{$gt:"500"}}).pretty()
{
        "_id" : ObjectId("5c81fdd89cff5f2a54951b53"),
        "title" : "ASP.NET3.5 VB 2008",
        "status_info" : {
                "accession_no" : "BS0002",
                "status" : "AVAIL"
        },
        "author" : "ANNE BOEHM",
        "cost" : "650",
        "publisher" : {
                "name" : "MURACH",
                "location" : "JAIPUR"
        }
}
{
        "_id" : ObjectId("5c81fe5e9cff5f2a54951b54"),
        "title" : "PRPGRAMMING IN VB",
        "status_info" : {
                "accession_no" : "BS0003",
                "status" : "ISSUED"
        },
        "author" : "JULIA CASE BRADLEY",
        "cost" : "600",
        "publisher" : {
                "name" : "TMH",
                "location" : "BANGLORE"
        }
```

Conclusion: We have successfully executed Data Collection, Data Curation and Management for large scale data system (such as MongoDB).

Name: Sumit Singh

Roll No: 380

Class: TYBSC CS A

Subject: Data Science

PRACTICAL 2

Aim: Simple /Multiple Linear Regressions.

#IMPOER DATASET:

Command:

>data=read.csv ("D://tycs/score.csv")

>data

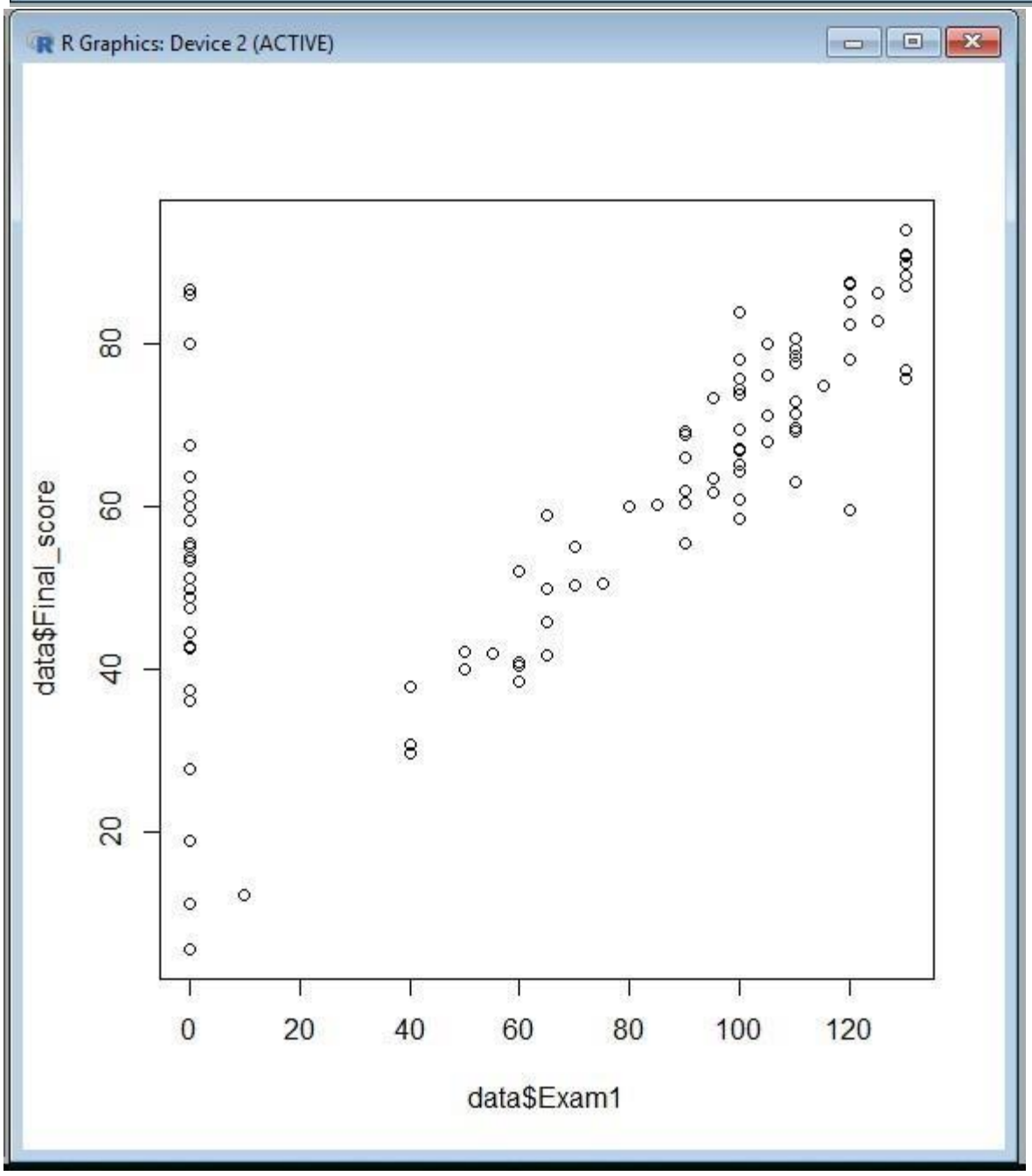
```
> data= read.csv("D://TYCS21/score.csv")
> data
   Exam1 Exam2 Exam3 Exam4 Final_score Grade
1     60    10    16   7.0       40.79     C
2     90     0     0   0.0       69.23     B
3    130    20    24   1.0       76.75     B
4    130    10    24   8.5       75.66     B
5     90     5    22   9.5       55.48     C
6    100    30    20   3.0       67.11     B
7    105    20    22   8.0       67.98     B
8    120    40    18  16.0       85.09     A
9    120    20    30  18.0       82.46     A
10   130    45    22  10.5       91.01     A
11    90    40    20   7.0       68.86     B
12   130    30    28  10.5       87.06     A
13   100    30    22   6.5       69.52     B
14     0    30    18   0.0       60.00     B
15     0    30    18   0.0       60.00     B
16    80     0    24   3.0       60.11     B
17   105    40    22   6.5       76.10     B
18    10     0     0   8.0       12.16     D
19   130    35    24   0.0       90.00     A
20     0    15    20   7.0       42.86     C
21    40    10    14   6.0       30.70     D
22    90    15    28   8.5       62.06     B
23   110     0    24   9.5       80.62     A
24    65     5    24   1.0       41.67     C
25    55    15    18   0.0       41.90     C
26   100    50    30  11.5       83.99     A
27    95    40    24   8.0       73.25     B
28     0    10    24   0.0       42.50     C
29     0     0    18   0.0       60.00     B
30    65    20    20   0.0       50.00     C
31   110    25    18   6.0       69.74     B
32   130    45    24   8.0       90.79     A
33   120    40    30   9.0       87.28     A
```

#PLOT THE DATASET:

COMMAND:

>plot(x=data$Exam1,y=data$Final_score)

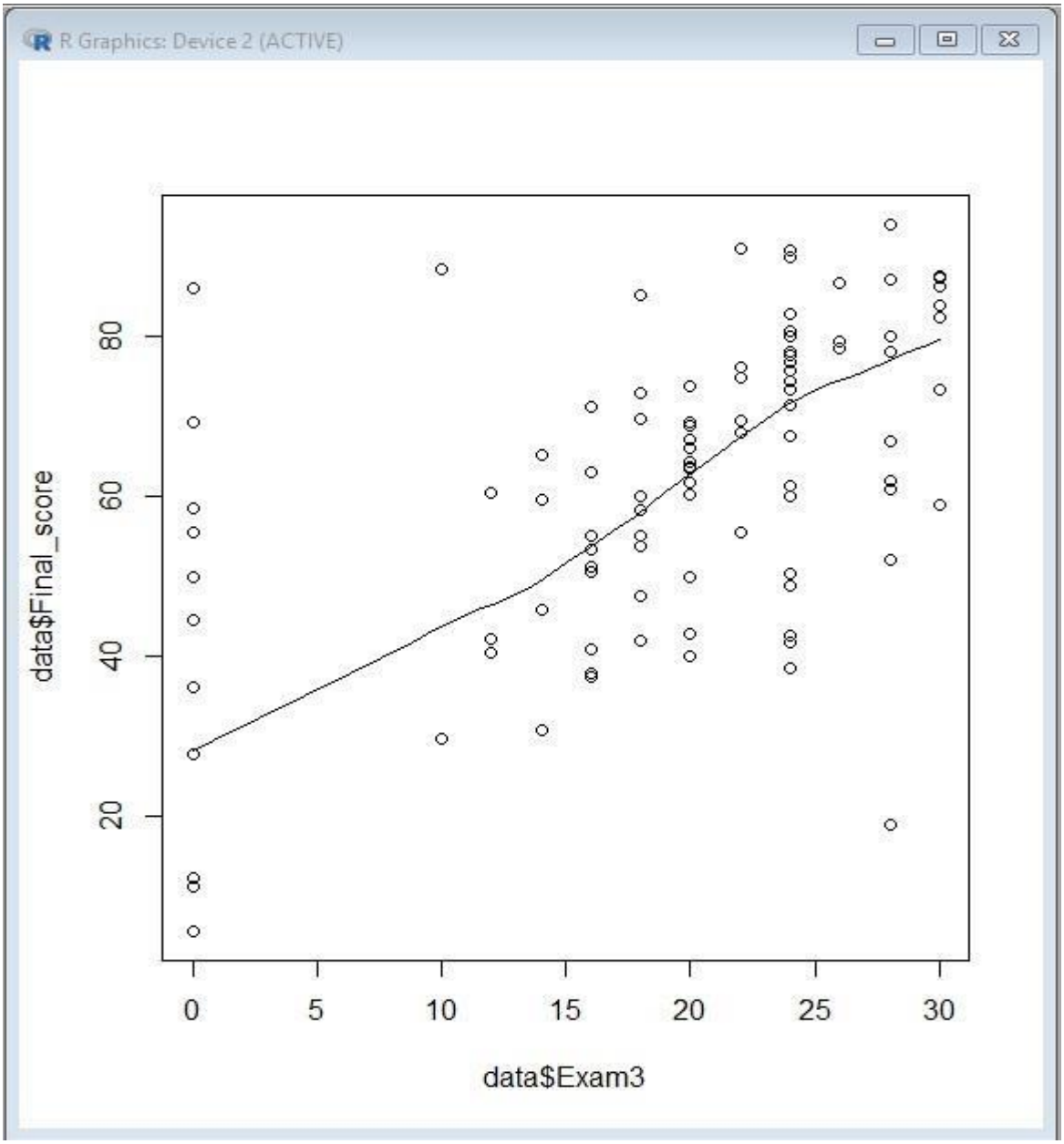
```
> plot(x=data$Exam1,y=data$Final_score)
>
```



#PLOT THE SCATTER DIAGRAM:

```
>scatter.smooth(x=data$Exam3,y=data$Final_score)
```

```
> scatter.smooth(x=data$Exam3,y=data$Final_score)
>
```




```
> cor(data$Exam3,data$Final_score)
[1] 0.6046352
>
```

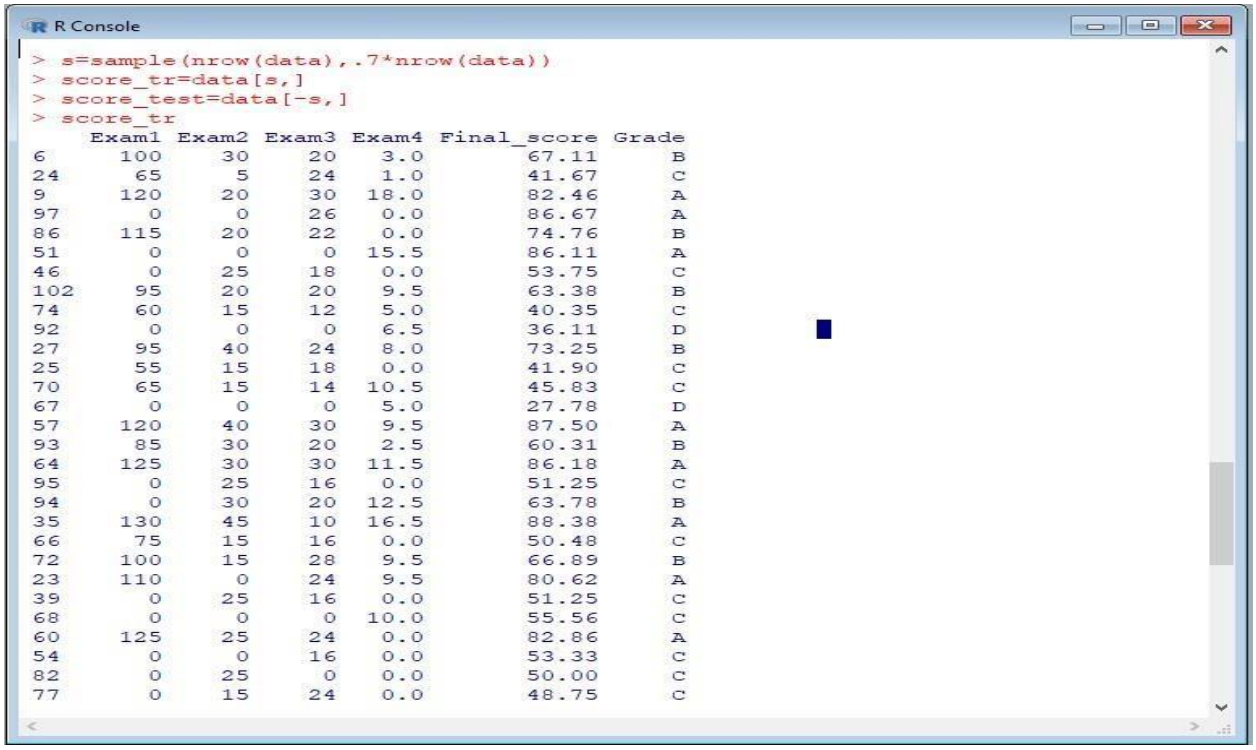
#PARTITIONING THE DATABASE INTO TRAINING AND TESTING SET

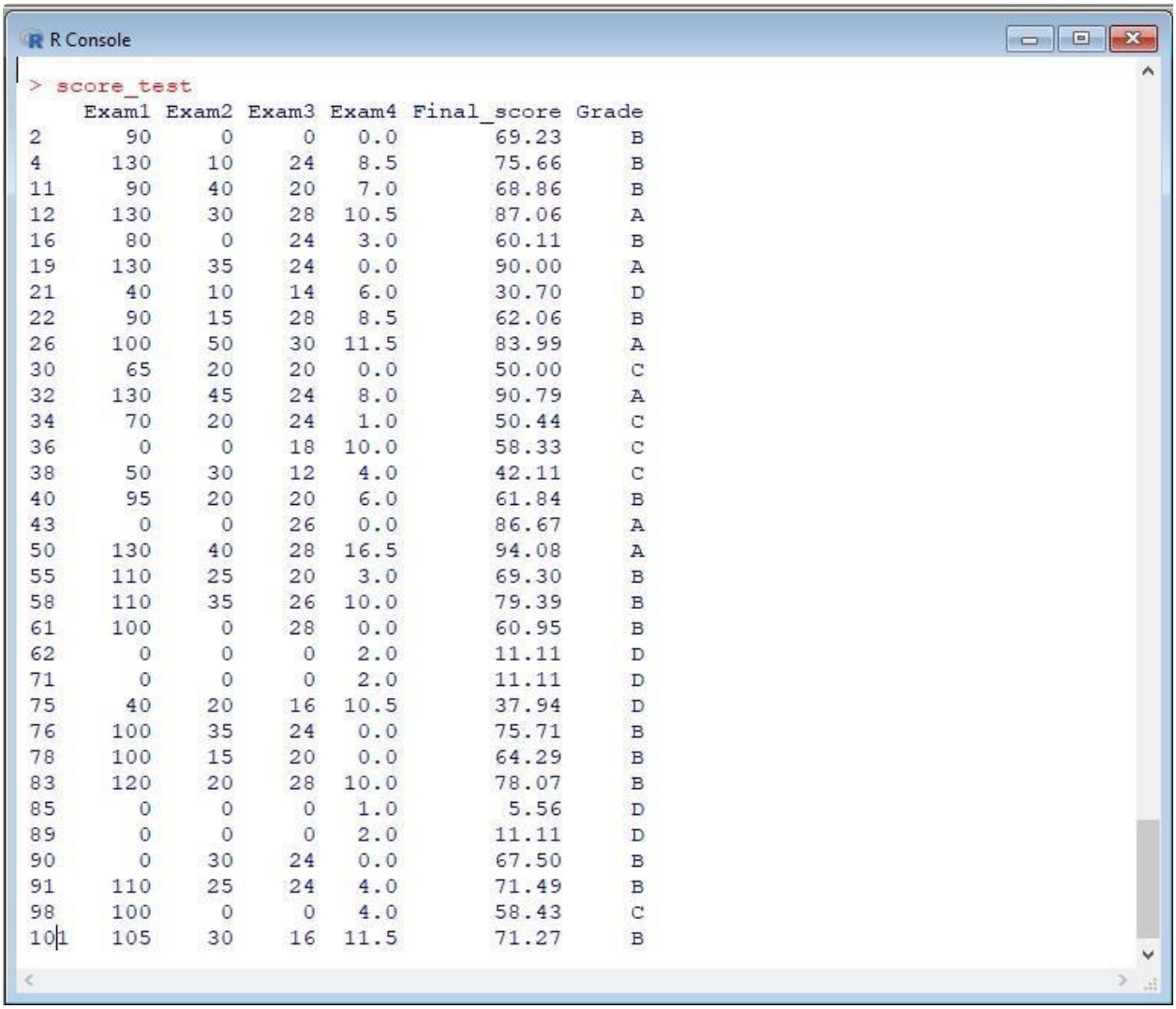
>s=sample(nrow(data),.7*nrow(data))

>score_tr=data[s,]

>score_test=[-s,]

Score_tr






```
> linmon=lm(Final_score~Exam3,data=score_tr)
> print(linmod)
Error in print(linmod) : object 'linmod' not found
> print(linmon)

Call:
lm(formula = Final_score ~ Exam3, data = score_tr)

Coefficients:
(Intercept)        Exam3
     39.537        1.119

> 
```

```
> pdata=predict(linmon,score_test)
> summary(linmon)

Call:
lm(formula = Final_score ~ Exam3, data = score_tr)

Residuals:
    Min      1Q  Median      3Q     Max
-52.005  -9.967   1.666  10.500  46.573

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  39.5367     4.8090   8.221 7.15e-12 ***
Exam3         1.1189     0.2362   4.737 1.10e-05 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 16.42 on 70 degrees of freedom
Multiple R-squared:  0.2427,    Adjusted R-squared:  0.2319
F-statistic: 22.44 on 1 and 70 DF,  p-value: 1.101e-05

>
```

#CREATING A MODEL

```
> actual_predict=data.frame(cbind(actuals=score_test$Final_score,predicteds=pdata))
> actual_predict
   actuals predicteds
2    69.23   39.53669
4    75.66   66.38965
11   68.86   61.91416
12   87.06   70.86515
16   60.11   66.38965
19   90.00   66.38965
21   30.70   55.20092
22   62.06   70.86515
26   83.99   73.10290
30   50.00   61.91416
32   90.79   66.38965
34   50.44   66.38965
36   58.33   59.67641
38   42.11   52.96317
40   61.84   61.91416
43   86.67   68.62740
50   94.08   70.86515
55   69.30   61.91416
58   79.39   68.62740
61   60.95   70.86515
62   11.11   39.53669
71   11.11   39.53669
75   37.94   57.43867
76   75.71   66.38965
78   64.29   61.91416
83   78.07   70.86515
85    5.56   39.53669
89   11.11   39.53669
90   67.50   66.38965
91   71.49   66.38965
98   58.43   39.53669
```

#PREDICTING THE OUTPUT ON TEST DATASET

```
> cor(actual_predict$actual,actual_predict$predict)
[1] 0.7674963
>
```

```
> mape= mean(abs((actual_predict$predicteds - actual_predict$actual))/ actual_predict$actual)*100
> mape
[1] 60.6191
> mape= mean(abs((actual_predict$predicteds - actual_predict$actual))/ actual_predict$actual)
> mape
[1] 0.606191
>
```

✞ Plot Scatter plot

```
>x=read.csv("D:/TYCS46/score.csv")
>x
   Exam1 Exam2 Exam3 Exam4 Final_score Grade
1     60    10    16   7.0       40.79     C
2     90     0     0   0.0       69.23     B
3    130    20    24   1.0       76.75     B
4    130    10    24   8.5       75.66     B
5     90     5    22   9.5       55.48     C
.
.
.
.
.
.
> s=sample(nrow(x),.7*nrow(x))
>score_tr=x[s,]
>score_test=x[-s,]
>scatter.smooth(x=score_tr$Exam3,y=score_tr$Final_score)
```

✞ Get Linear regression

```
>linmod=lm(Final_score~Exam3,data=score_tr)
>print(linmod)

Call:
lm(formula = Final_score ~ Exam3, data = score_tr)

Coefficients:
(Intercept)       Exam3
      35.90        1.32
```

✞ Prediction

```
> p=predict(linmod,score_test)
>actuals_preds = data.frame(cbind(actuals=score_test$Final_score,predicts=p))
>actuals_preds
actuals predicts
```

2 69.23 35.89681
11 68.86 62.30306
14 60.00 59.66244
15 60.00 59.66244
16 60.11 67.58432
26 83.99 75.50619
27 73.25 67.58432
32 90.79 67.58432
35 88.38 49.09994
39 51.25 57.02181
41 29.61 49.09994
44 65.13 54.38119
47 78.10 67.58432
50 94.08 72.86557
54 53.33 57.02181
56 51.97 72.86557
57 87.50 75.50619
62 11.11 35.89681
64 86.18 75.50619
70 45.83 54.38119
71 11.11 35.89681
78 64.29 62.30306
79 59.65 54.38119
84 74.34 67.58432
85 5.56 35.89681
88 47.50 59.66244
90 67.50 67.58432
91 71.49 67.58432
93 60.31 62.30306
94 63.78 62.30306
96 73.81 62.30306
99 18.86 72.86557

✞ Print Accuracy

```
> min_max_accuracy=mean(apply(actuals_preds,1,min)/apply(actuals_preds,1,max))
>min_max_accuracy
[1] 0.7806829
>mape=mean(abs((actuals_preds$predicteds-actuals_preds$actuals))/actuals_preds$actuals) >mape
[1] 0.68783456
```

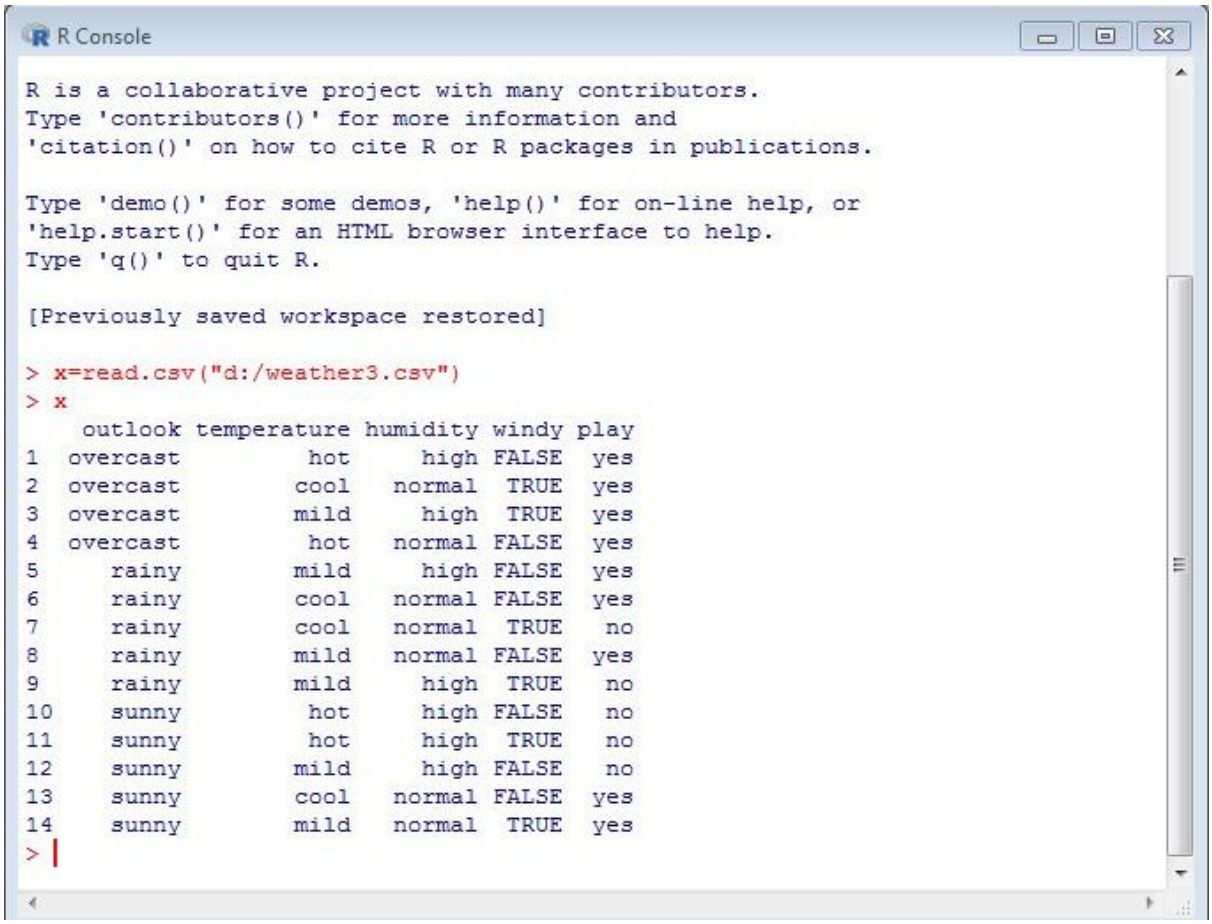
Conclusion: Hence we successfully performed Simple/ Multiple linear regression.

Name: SumitSingh
Roll No: 380
Class: TYBSC CS A
Subject: Data Science
Practical No : 3

Aim: Demonstration of Logistics Regression.

Code:

X<-read.csv("C:/Users/Admin/Documents/SampleStudentData.csv") >
X



PRINTING THE DATASET

>x$humidity=ifelse(test=x$humidity=="high",yes=1,no=0) >x

```
> x$humidity=ifelse(test=x$humidity=="high",yes=1,no=0)
> x
    outlook temperature humidity windy play
1  overcast         hot        1 FALSE  yes
2  overcast        cool        0  TRUE  yes
3  overcast        mild        1  TRUE  yes
4  overcast         hot        0 FALSE  yes
5     rainy        mild        1 FALSE  yes
6     rainy        cool        0 FALSE  yes
7     rainy        cool        0  TRUE   no
8     rainy        mild        0 FALSE  yes
9     rainy        mild        1  TRUE   no
10    sunny         hot        1 FALSE   no
11    sunny         hot        1  TRUE   no
12    sunny        mild        1 FALSE   no
13    sunny        cool        0 FALSE  yes
14    sunny        mild        0  TRUE  yes
```

>x$play=ifelse(test=x$play=="yes",yes=1,no=0) >x

```
> x$play=ifelse(test=x$play=="yes",yes=1,no=0)
> x
    outlook temperature humidity windy play
1  overcast         hot        1 FALSE    1
2  overcast        cool        0  TRUE    1
3  overcast        mild        1  TRUE    1
4  overcast         hot        0 FALSE    1
5     rainy        mild        1 FALSE    1
6     rainy        cool        0 FALSE    1
7     rainy        cool        0  TRUE    0
8     rainy        mild        0 FALSE    1
9     rainy        mild        1  TRUE    0
10    sunny         hot        1 FALSE    0
11    sunny         hot        1  TRUE    0
12    sunny        mild        1 FALSE    0
13    sunny        cool        0 FALSE    1
14    sunny        mild        0  TRUE    1
```

>x$windy=ifelse(test=x$windy=="FALSE",yes=0,no=1) >x

```
> x$windy=ifelse(test=x$windy=="FALSE",yes=0,no=1)
> x
    outlook temperature humidity windy play
1  overcast         hot        1     0    1
2  overcast        cool        0     1    1
3  overcast        mild        1     1    1
4  overcast         hot        0     0    1
5     rainy        mild        1     0    1
6     rainy        cool        0     0    1
7     rainy        cool        0     1    0
8     rainy        mild        0     0    1
9     rainy        mild        1     1    0
10    sunny         hot        1     0    0
11    sunny         hot        1     1    0
12    sunny        mild        1     0    0
13    sunny        cool        0     0    1
14    sunny        mild        0     1    1
> |
```

PARTIONING DATASET

> s=sample(nrow(x),.7*nrow(x))
>x_tr=x[s,]
>x_test=x[-s,]
>nrow(x)
>nrow(x_tr)
>nrow(x_test)

```
> s=sample(nrow(x),.7*nrow(x))
> x_tr=x[s,]
> x_test=x[-s,]
> nrow(x)
[1] 14
> nrow(x_tr)
[1] 9
> nrow(x_test)
[1] 5
> |
```

DATA MODELING

>lmod=glm(play~windy,data=x_tr,family=binomial,control=list(maxit=100)) >lmod

```
> lmod=glm(play~windy,data=x_tr,family=binomial,control=list(maxit=100))
> lmod

Call:  glm(formula = play ~ windy, family = binomial, data = x_tr, control = list(maxit = 100))

Coefficients:
(Intercept)        windy
      20.57       -19.87

Degrees of Freedom: 8 Total (i.e. Null);  7 Residual
Null Deviance:      6.279
Residual Deviance: 3.819        AIC: 7.819
> 
```

```
> summary(lmod)

Call:
glm(formula = play ~ windy, family = binomial, data = x_tr, control = list(maxit = 100))

Deviance Residuals:
     Min        1Q    Median        3Q       Max
-1.48230   0.00005   0.00005   0.00005   0.90052

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept)    20.57    7238.39   0.003    0.998
windy         -19.87    7238.39  -0.003    0.998

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 6.2790  on 8  degrees of freedom
Residual deviance: 3.8191  on 7  degrees of freedom
AIC: 7.8191

Number of Fisher Scoring iterations: 19

> 
```

>lmod=glm(play~humidity,data=x_tr,family=binomial,control=list(maxit=100))
>summary(lmod)

```
> lmod=glm(play~humidity,data=x_tr,family=binomial,control=list(maxit=100))
> summary(lmod)

Call:
glm(formula = play ~ humidity, family = binomial, data = x_tr,
    control = list(maxit = 100))

Deviance Residuals:
     Min        1Q    Median        3Q       Max
-1.97277   0.00008   0.55525   0.55525   0.55525

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept)    1.792      1.080   1.659   0.0971 .
humidity      17.774   7604.236   0.002   0.9981
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 6.2790  on 8  degrees of freedom
Residual deviance: 5.7416  on 7  degrees of freedom
AIC: 9.7416

Number of Fisher Scoring iterations: 18

>
```

>lmod=glm(play~temperature,data=x_tr,family=binomial,control=list(maxit=100))
>summary(lmod)

```
> lmod=glm(play~temperature,data=x_tr,family=binomial,control=list(maxit=100))
> summary(lmod)

Call:
glm(formula = play ~ temperature, family = binomial, data = x_tr,
    control = list(maxit = 100))

Deviance Residuals:
     Min        1Q    Median        3Q       Max
-1.66511   0.00005   0.00005   0.75853   0.75853

Coefficients:
                 Estimate Std. Error z value Pr(>|z|)
(Intercept)         1.099      1.155   0.951    0.341
temperaturehot     19.467  12537.265   0.002    0.999
temperaturemild    19.467  10236.634   0.002    0.998

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 6.2790  on 8  degrees of freedom
Residual deviance: 4.4987  on 6  degrees of freedom
AIC: 10.499

Number of Fisher Scoring iterations: 19

>
```

#PREDICTION:

> p=predict(lmod,x_test,type="response")

>p

```
> p=predict(lmod,x_test,type="response")
> p
           3            9           10           11           12
1.000000e+00 5.800756e-11 1.000000e+00 1.000000e+00 1.000000e+00
> |
```

(2) SECOND DATA SET:

#IMPORT THE DATA

>x2=read.csv("D:/grade_logit.csv")

>x2

```
> x2=read.csv("D:/grade_logit.csv")
> x2
   Exam1 Exam2 Exam3 Exam4 Final_score Grade
1     60    10    16   7.0       40.79     1
2     90     0     0   0.0       69.23     1
3    130    20    24   1.0       76.75     1
4    130    10    24   8.5       75.66     1
5     90     5    22   9.5       55.48     1
6    100    30    20   3.0       67.11     1
7    105    20    22   8.0       67.98     1
8    120    40    18  16.0       85.09     1
9    120    20    30  18.0       82.46     1
10   130    45    22  10.5       91.01     1
11    90    40    20   7.0       68.86     1
12   130    30    28  10.5       87.06     1
13   100    30    22   6.5       69.52     1
14     0    30    18   0.0       60.00     1
15     0    30    18   0.0       60.00     1
16    80     0    24   3.0       60.11     1
17   105    40    22   6.5       76.10     1
18    10     0     0   8.0       12.16     0
19   130    35    24   0.0       90.00     1
20     0    15    20   7.0       42.86     1
21    40    10    14   6.0       30.70     0
22    90    15    28   8.5       62.06     1
23   110     0    24   9.5       80.62     1
24    65     5    24   1.0       41.67     1
25    55    15    18   0.0       41.90     1
26   100    50    30  11.5       83.99     1
27    95    40    24   8.0       73.25     1
28     0    10    24   0.0       42.50     1
29     0     0    18   0.0       60.00     1
30    65    20    20   0.0       50.00     1
31   110    25    18   6.0       69.74     1
32   130    45    24   8.0       90.79     1
33   120    40    30   9.0       87.28     1
34    70    20    24   1.0       50.44     1
35   130    45    10  16.5       88.38     1
```

> lmod2=glm(Grade~Exam1,data=x2_train,family=binomial,control=list(maxit=100)) >summary(lmod2)

```
> lmod2=glm(Grade~Exam1,data=x2_train,family=binomial,control=list(maxit=100))
> summary(lmod2)

Call:
glm(formula = Grade ~ Exam1, family = binomial, data = x2_train,
    control = list(maxit = 100))

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.2051   0.1834   0.2442   0.4444   0.9351

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept) 0.600860   0.396710   1.515  0.12987
Exam1       0.028971   0.009424   3.074  0.00211 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 68.589  on 82  degrees of freedom
Residual deviance: 54.049  on 81  degrees of freedom
AIC: 58.049

Number of Fisher Scoring iterations: 6
```

Prediction data 1's and 0's form

>prediction=ifelse(p>.5,1,0)

>prediction

```
> prediction=ifelse(p>.5,1,0)
> prediction
 4 10 13 14 23 37 45 50 51 55 64 66 67 76 81 84 89 91 93 96 97
 1  1  1  1  1  1  1  1  0  1  1  1  0  1  1  1  0  1  1  1  1
>
```

PREDICTION MATRIX

>table(x2_test$Grade,prediction)

```
> table(x2_test$Grade,prediction)
   prediction
     0  1
  0  2  1
  1  1 17
```

> x2_test

```
> x2_test
   Exam1 Exam2 Exam3 Exam4 Final_score Grade
4    130    10    24   8.5       75.66     1
10   130    45    22  10.5       91.01     1
13   100    30    22   6.5       69.52     1
14     0    30    18   0.0       60.00     1
23   110     0    24   9.5       80.62     1
37     0    25    24   0.0       61.25     1
45    95    30    30  12.0       73.25     1
50   130    40    28  16.5       94.08     1
51     0     0     0  15.5       86.11     1
55   110    25    20   3.0       69.30     1
64   125    30    30  11.5       86.18     1
66    75    15    16   0.0       50.48     1
67     0     0     0   5.0       27.78     0
76   100    35    24   0.0       75.71     1
81    50    20    20   1.0       39.91     0
84   100    35    24  10.5       74.34     1
89     0     0     0   2.0       11.11     0
91   110    25    24   4.0       71.49     1
93    85    30    20   2.5       60.31     1
96   100    35    20   0.0       73.81     1
97     0     0    26   0.0       86.67     1
> |
```

#actuals predicted

>ac_pr<- data.frame(cbind(actuals=x2_test$Grade, predicteds=prediction)) >ac_pr

```
> ac_pr <- data.frame(cbind(actuals=x2_test$Grade, predicteds=prediction))
> ac_pr
   actuals predicteds
4        1          1
10       1          1
13       1          1
14       1          1
23       1          1
37       1          1
45       1          1
50       1          1
51       1          0
55       1          1
64       1          1
66       1          1
67       0          0
76       1          1
81       0          1
84       1          1
89       0          0
91       1          1
93       1          1
96       1          1
97       1          1
> |
```

>vif(lmod2) // variable influence factor

```
> vif(lmod2)
   Exam1    Exam2    Exam3
1.023350 1.117704 1.122152
> |
```

Conclusion: Hence we successfully performed Logistic regression.

Name: Sumit Singh

Roll No: 380

Class: TYBSC CS A

Subject: Data Science

Practical No: 4

Aim: Demonstration of Hypothesis testing.

Steps:

Step 1: First we have to create Excel file and Enter the 28 values so that we can fine deviation, Square of deviation, population, differentiate of mean, T-value, and system calculate standard deviation and save as .CSV file.

Output:

	A	B	C	D	E
1		C1	Deviation	Deviation sqr	
2	1	85.3	-12.22142857	149.3633163	
3		86.9	-10.62142857	112.8147449	
4		96.8	-0.721428571	0.520459184	
5		108.5	10.97857143	120.5290306	
6		113.8	16.27857143	264.9918878	
7		87.7	-9.821428571	96.46045918	
8		94.5	-3.021428571	9.129030612	
9		99.9	2.378571429	5.657602041	
10		92.9	-4.621428571	21.35760204	
11		67.3	-30.22142857	913.3347449	
12		90.6	-6.921428571	47.90617347	
13		129.8	32.27857143	1041.906173	
14		48.9	-48.62142857	2364.043316	
15		117.5	19.97857143	399.1433163	
16		100.8	3.278571429	10.74903061	
17		94.5	-3.021428571	9.129030612	
18		94.4	-3.121428571	9.743316327	
19		98.9	1.378571429	1.900459184	
20		96	-1.521428571	2.314744898	
21		99.4	1.878571429	3.529030612	
22		79.1	-18.42142857	339.3490306	
23		108.5	10.97857143	120.5290306	
24		84.6	-12.92142857	166.9633163	
25		117.5	19.97857143	399.1433163	
26		70	-27.52142857	757.4290306	
27		104.4	6.878571429	47.3147449	
28		127.1	29.57857143	874.8918878	
29		135	37.47857143	1404.643316	
30		97.52143	calculate varianc	346.242398	
31				t value	-0.69214
32	populatio	100		system calculate stdev	18.94904
33	Diff in me	-2.47857			
34					

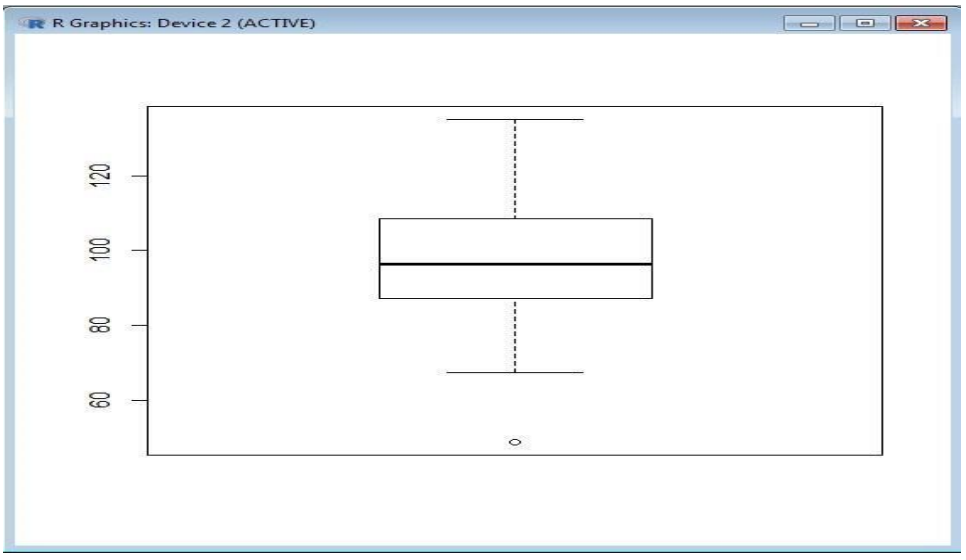
Step 2:Now we have to import Excel file (onetest.csv) type bellow command.

Output:

```
R Console
> datanew=read.csv("D:/onettest.csv")
> datanew
      C1
1   85.3
2   86.9
3   96.8
4  108.5
5  113.8
6   87.7
7   94.5
8   99.9
9   92.9
10  67.3
11  90.6
12 129.8
13  48.9
14 117.5
15 100.8
16  94.5
17  94.4
18  98.9
19  96.0
20  99.4
21  79.1
22 108.5
23  84.6
24 117.5
25  70.0
26 104.4
27 127.1
28 135.0
```

Step 3: After importing onetest.csv file we will plot Boxplot diagram type bellow command.

#boxplot(datanew) Output:



Step 4:After that find mean of respective data.

Output:

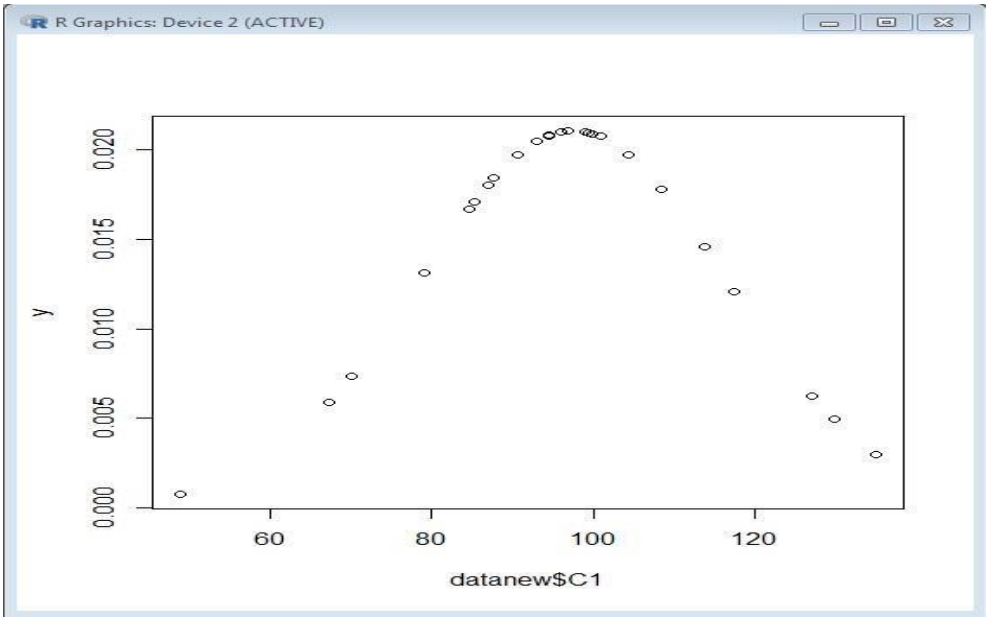
```
> m1=mean(datanew$C1)
> m1
[1] 97.52143
```

Step 5:Now calculate the standard deviation.
Output:

```
> sd1=sd(datanew$C1)
> sd1
[1] 18.94904
> mean1=mean(datanew$C1)
> mean1
[1] 97.52143
```

Step 6:Plot bell curve.

plot(datanew$C1) Output:



Step 7: At the end find T-Test value type following command.

Output:

```
> t.test(datanew$C1,alternative="greater",mu=100)

        One Sample t-test

data:  datanew$C1
t = -0.69214, df = 27, p-value = 0.7526
alternative hypothesis: true mean is greater than 100
95 percent confidence interval:
 91.4219     Inf
sample estimates:
mean of x
 97.52143
```

CONCLUSION:Thus we have implemented Hypothesis testing of a Single Population means successfully

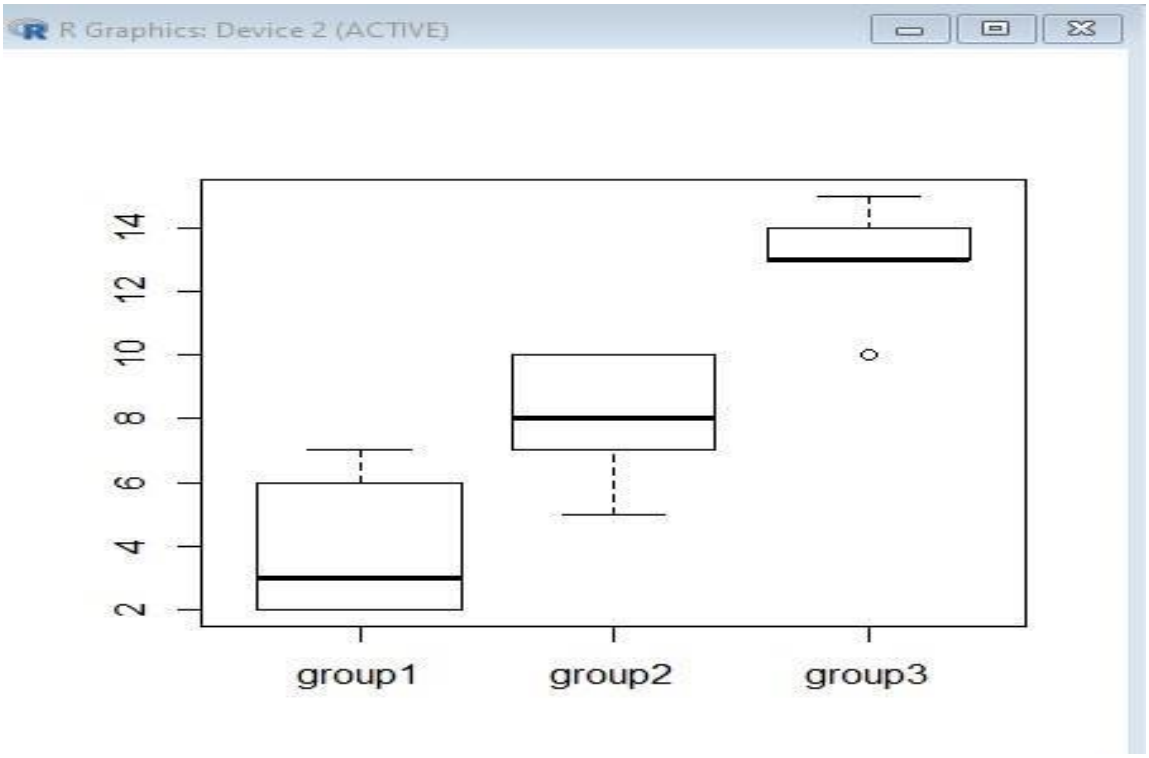
Name: Sumit Singh
Roll No: 380
Class: TYBSC CS A
Subject: Data Science
Practical No : 5

Aim: Demonstration of Analysis of Variance. Code:

#CREATE THE DATA IN TO THREE GROUPS #TO PRINT THE BOXPLOT

```
> group1=c(2,3,7,2,6)
> group2=c(10,8,7,5,10)
> group3=c(10,13,14,13,15)
> cg=data.frame(cbind(group1,group2,group3))
> cg
  group1 group2 group3
1      2     10     10
2      3      8     13
3      7      7     14
4      2      5     13
5      6     10     15
> boxplot(cg)
>
```

BOXPLOT



#TO PRINT THE DATA INTO STACK FORMATE

```
> stacked_g=stack(cg)
> stacked_g
   values     ind
1       2  group1
2       3  group1
3       7  group1
4       2  group1
5       6  group1
6      10  group2
7       8  group2
8       7  group2
9       5  group2
10     10  group2
11     10  group3
12     13  group3
13     14  group3
14     13  group3
15     15  group3
> |
```

TAKE ANOTHER DATASET AND WORK ON THAT.

 # CREATE THE DATA IN TO THREE GROUPS

```
> av=aov(values~ind,data=stacked_g)
> summary(av)
            Df Sum Sq Mean Sq F value   Pr(>F)
ind          2  203.3   101.7   22.59 8.54e-05 ***
Residuals   12   54.0     4.5
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
> g1=c(29,30,31,31,29)
> g2=c(28,29,27,30,29)
> g3=c(25,28,29,27,29)
> cg1=data.frame(cbind(g1,g2,g3))
> cg1
  g1 g2 g3
1 29 28 25
2 30 29 28
3 31 27 29
4 31 30 27
5 29 29 29
```

```
> stacked_g= stack(cgl)
> stacked_g
   values ind
1      29  g1
2      30  g1
3      31  g1
4      31  g1
5      29  g1
6      28  g2
7      29  g2
8      27  g2
9      30  g2
10     29  g2
11     25  g3
12     28  g3
13     29  g3
14     27  g3
15     29  g3
```

```
> av=aov(values~ind,data=stacked_g)
> avl=aov(values~ind,data=stacked_g)
> summary(avl)
            Df Sum Sq Mean Sq F value Pr(>F)
ind          2  14.53   7.267   4.275 0.0397 *
Residuals   12  20.40   1.700
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
>
```

Conclusion: Hence, we successfully performed Analysis of variance.

Name: Sumit Singh
Roll No: 380
Class: TYBSC CS A
Subject: Data Science
Practical No : 6

Aim: Demonstration of Decision Tree.
Code:
Steps:
Step1: click on packages and set cran mirror.
Step2: click on packages and select install packages and install 3 packages (rpart,tree,rattle)
Step3:(OPTIONAL Application for version 4.2)
install.packages("rpart") install.packages("tree")
install.packages("rattle")

```
[Previously saved workspace restored]

> chooseCRANmirror()
> utils:::menuInstallPkgs()

  There is a binary version available but the source version is later:
      binary source needs_compilation
rpart 4.1.16 4.1.19              TRUE

  Binaries will be installed
trying URL 'http://ftp.ussg.iu.edu/CRAN/bin/windows/contrib/4.0/rpart_4.1.16.zi$
Content type 'application/zip' length 982973 bytes (959 KB)
downloaded 959 KB

package 'rpart' successfully unpacked and MD5 sums checked

The downloaded binary packages are in
        C:\Users\admin\AppData\Local\Temp\Rtmp2BLQPi\downloaded_packages
> x=read.csv("C:/Users/admin/Desktop/weather1.csv")
> x
    outlook temp humidity windy play.golf
1     rainy  hot     high FALSE        no
2     rainy  hot     high  TRUE        no
3  overcast  hot     high FALSE       yes
```

Step4: **Create an excel data save it with .csv extension.**

Code:

Read excel data in rstudio

> x=read.csv("C:/weather1.csv")
> x

```
> x=read.csv("C:/Users/admin/Desktop/weather1.csv")
> x
    outlook temp humidity windy play.golf
1     rainy  hot     high FALSE        no
2     rainy  hot     high  TRUE        no
3  overcast  hot     high FALSE       yes
4     sunny mild     high FALSE       yes
5     sunny cool   normal FALSE       yes
6     sunny cool   normal  TRUE        no
7  overcast cool   normal  TRUE       yes
8     rainy mild     high FALSE       yes
9     rainy cool   normal FALSE       yes
10    sunny mild   normal FALSE       yes
11    rainy mild   normal  TRUE       yes
12 overcast mild     high  TRUE       yes
13 overcast  hot   normal FALSE       yes
14    sunny mild     high  TRUE        no
> sample_weather=sample(nrow(x),.7*nrow(x))
> weather_tr=x[sample_weather,]
> weather_test=x[-sample_weather,]
> weather_test
    outlook temp humidity windy play.golf
2     rainy  hot     high  TRUE        no
3  overcast  hot     high FALSE       yes
```

Create sample partition of the excel data

> sample_weather=sample(nrow(x),.7*nrow(x))

Create a weather partition for training

> weather_tr=x[sample_weather,]

Create a weather partition for testing

> weather_test=x[-sample_weather,]

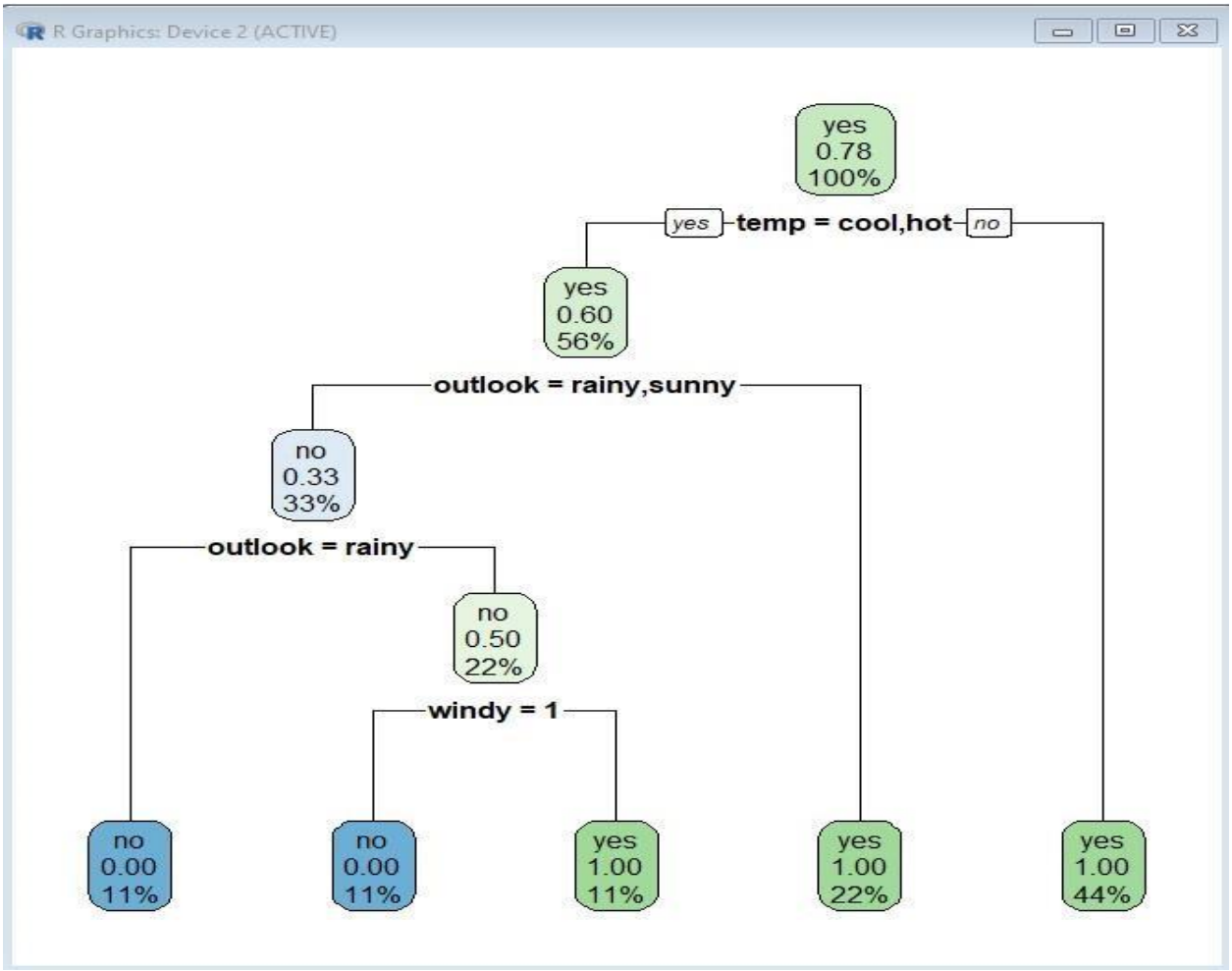
> weather_test

Call rpart packages

> library(rpart)

> library(rpart.plot) **Plot tree**

dtreemod=rpart(play.golf~.,data=weather_tr,method="class",control=rpart.control(minsplit=1,min bucket=1)) rpart.plot(dtreemod)



Predict Tree:

> p=predict(dtreemod,weather_test,type="class")
> weather_test
> table(weather_test$play.golf,p)

```
> p=predict(dtreemod,weather_test,type="class")
> weather_test
    outlook temp humidity windy play.golf
2     rainy  hot     high  TRUE        no
3  overcast  hot     high FALSE       yes
6     sunny cool   normal  TRUE        no
13 overcast  hot   normal FALSE       yes
14    sunny mild     high  TRUE        no
> table(weather_test$play.golf,p)
     p
      no yes
  no   1   2
  yes  2   0
>
```

Printing rules with rpart.rules

rpart.rules(dtreemod) play.golf

0.00 when temp is hot
1.00 when temp is cool or mild
>

Regression Tree:

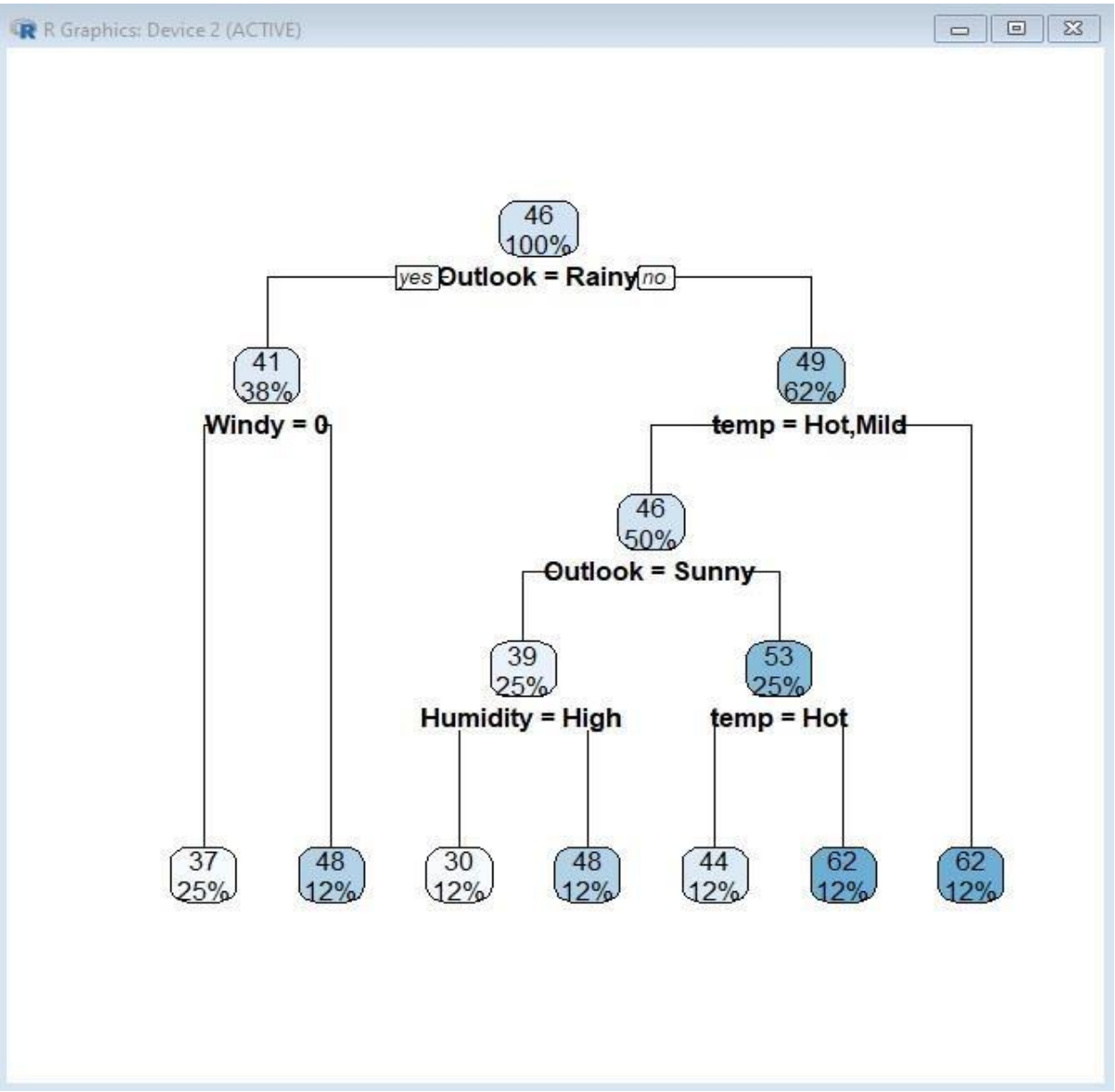
> x2=read.csv("C:/Users/admin/Desktop/weather2.csv") >
x2

```
> x2=read.csv("C:/Users/admin/Desktop/weather2.csv")
> x2
    Outlook  temp Humidity Windy Hours.Played
1     Rainy  Hot      High FALSE           26
2     Rainy  Hot      High  TRUE           30
3  Overcast  Hot      High FALSE           48
4     Sunny  Mild     High FALSE           46
5     Sunny  Cool   Normal FALSE           62
6  Overcast  Cool   Normal  TRUE           43
7     Rainy  Mild     High FALSE           36
8     Rainy  Cool   Normal FALSE           38
9     Sunny  Mild   Normal FALSE           48
10    Rainy  Mild   Normal  TRUE           48
11 Overcast  Mild     High  TRUE           62
12 Overcast   Hot   Normal FALSE           44
13    Sunny  Mild     High  TRUE           30
```

weather_tr2=x2[S2,] >
s2=sample(nrow(x),.7*nrow(x)) > weather_tr2=x2[s2,]
> weather_test2=x2[-s2,]
> weather_test2

```
> weather_tr2=x2[S2,]
> s2=sample(nrow(x),.7*nrow(x))
> weather_tr2=x2[s2,]
> weather_test2=x2[-s2,]
> weather_test2
   Outlook  temp Humidity Windy Hours.Played
1    Rainy  Hot      High FALSE           26
2    Rainy  Hot      High  TRUE           30
3 Overcast  Hot      High FALSE           48
4    Sunny  Mild     High FALSE           46
6 Overcast  Cool   Normal  TRUE           43
```

dtreemod2=rpart(Hours.Played~.,data=weather_tr2,method="anova",control=rpart.control(minsplit=1,minbucket=1)) > rpart.rules(dtreemod2)



Prediction:

> actuals_preds<- data.frame(cbind(actuals=weather_test2$Hours.played,predicts=p)) >

actuals_preds

```
> actuals_preds<- data.frame(cbind(actuals=weather_test2$Hours.played,predicts=p))
> actuals_preds
   predicts
2         1
7         2
9         1
12        2
14        2
> 
```

Conclusion: Hence we successfully implemented decision tree.

Name: Sumit Singh
Roll No: 380
Class: TYBSC CS A
Subject: Data Science Practical
No :7

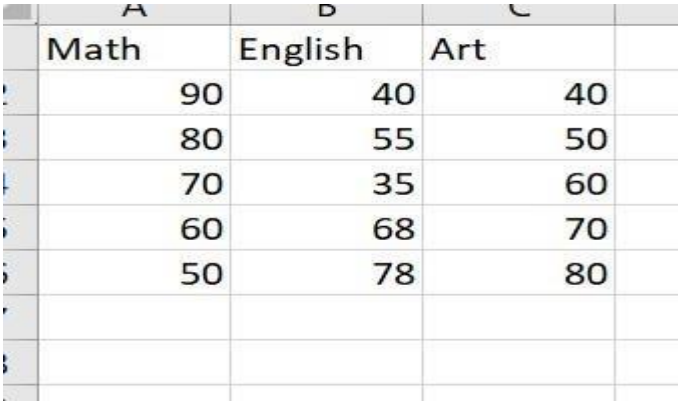
Aim: Demonstration of Principal Component Analysis.

Steps:

Step1: click on packages and set cran mirror(click on other and select USA IN)
Step2: click on packages and select install packages and install package FactoMineR. **install.packages("FactoMine**

R") library(FactoMineR)

Step3: Create Excel Sheet.



	A	B	C	
	Math	English	Art	
2	90	40	40	
3	80	55	50	
4	70	35	60	
5	60	68	70	
6	50	78	80	
7				
8				

Code:

x=read.csv("C:/Users/Akki/OneDrive/Desktop/JEENAL/students.csv") x

```
> x=read.csv("C:/Users/Akki/OneDrive/Desktop/JEENAL/students.csv")
> x
  Math English Art
1   90      40  40
2   80      55  50
3   70      35  60
4   60      68  70
5   50      78  80
```

cov_mat=cov(x)
cov_mat

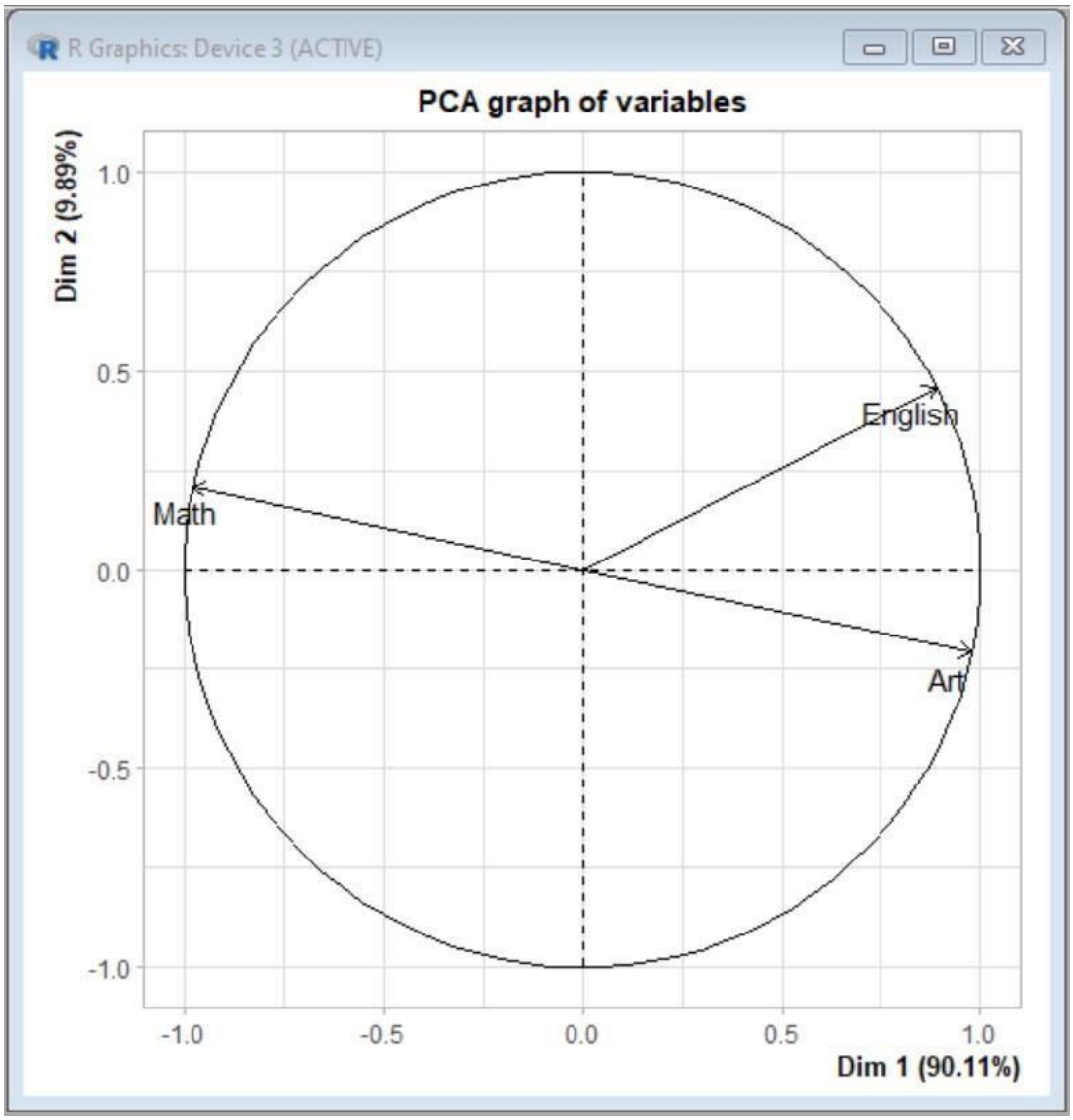
```
> cov_mat=cov(x)
> cov_mat
          Math English    Art
Math     250.0  -222.5 -250.0
English -222.5   330.7  222.5
Art     -250.0   222.5  250.0
```

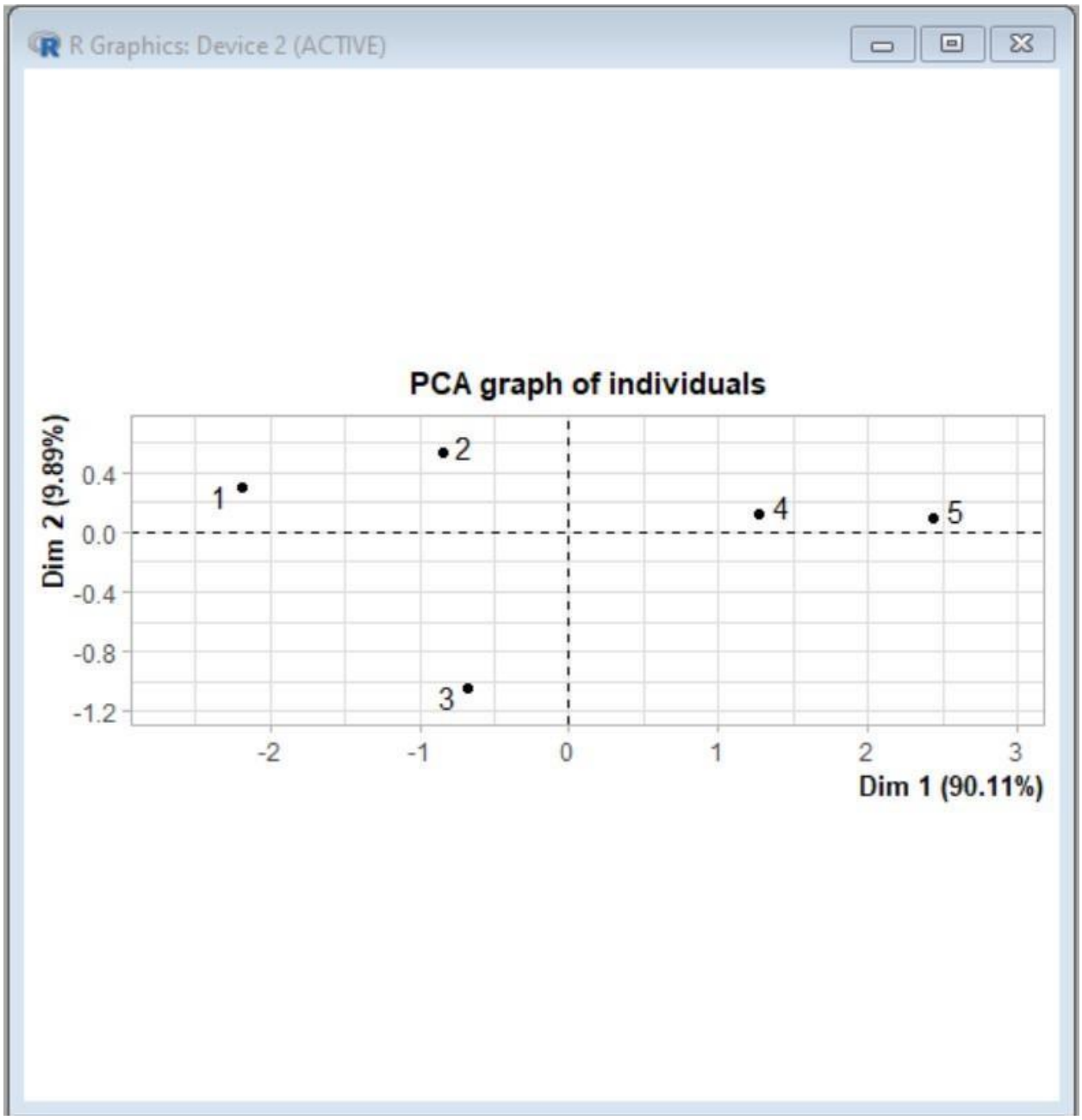
ex=eigen(cov_mat)

ex

```
> ex=eigen(cov_mat)
> ex
eigen() decomposition
$values
[1] 7.411998e+02 8.950015e+01 2.991369e-14

$vectors
           [,1]       [,2]         [,3]
[1,]  0.5612001 -0.4301795 7.071068e-01
[2,] -0.6083657 -0.7936568 2.220446e-16
[3,] -0.5612001  0.4301795 7.071068e-01

> datapca=PCA(x,ncp=3,graph=TRUE)
```

datapca=PCA(x,ncp=3,graph=TRUE)


R Graphics: Device 3 (ACTIVE)
PCA graph of variables
Dim 2 (9.89%)
1.0
0.5
0.0
-0.5
-1.0
English
Math
Art
-1.0
-0.5
0.0
0.5
1.0
Dim 1 (90.11%)



datapca$eig

```
> datapca$eig
       eigenvalue percentage of variance cumulative percentage of variance
comp 1  2.7031671              90.105571                           90.10557
comp 2  0.2968329               9.894429                          100.00000
comp 3  0.0000000               0.000000                          100.00000
```

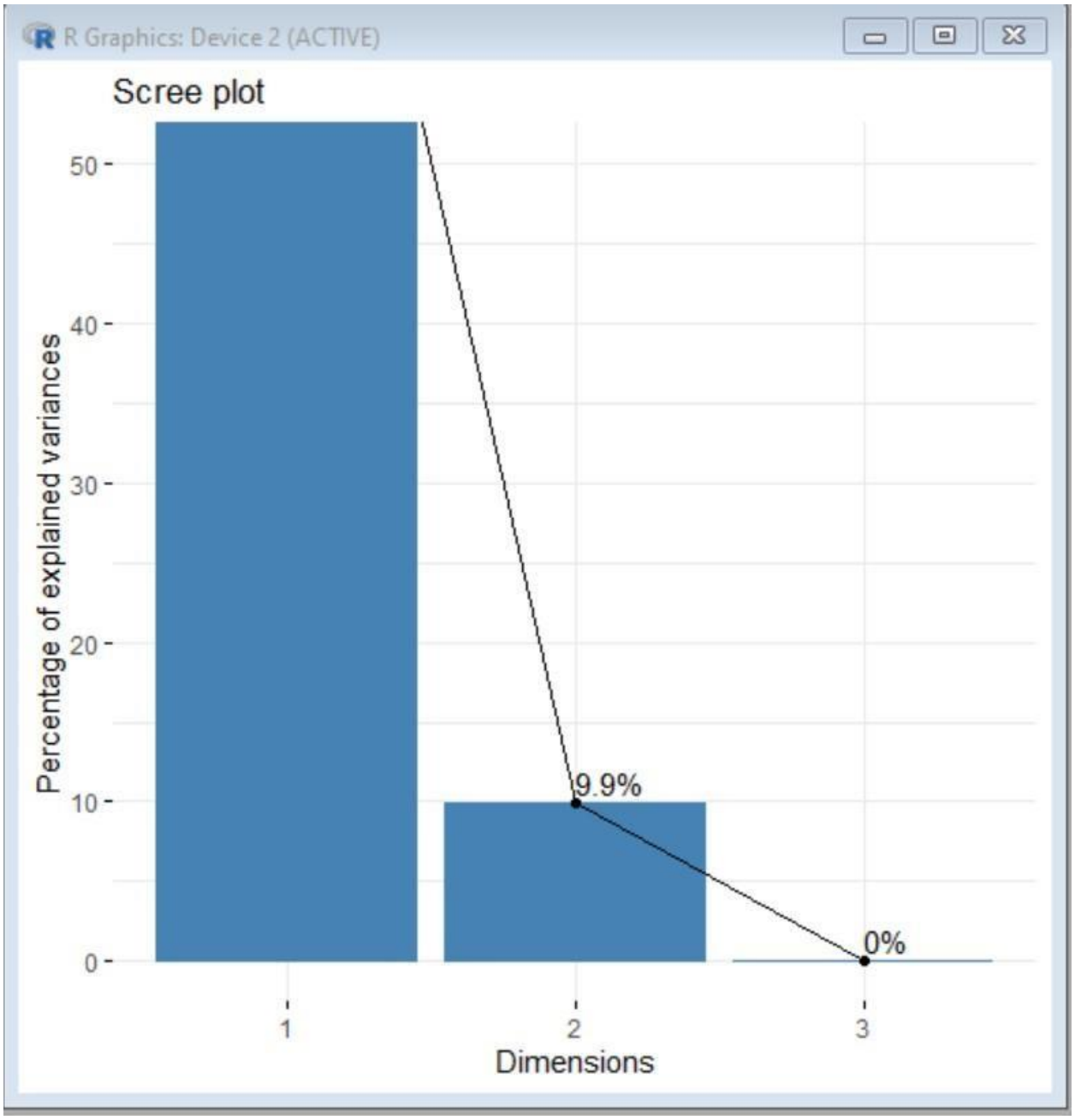
datapca$var

```
> datapca$var
$coord
             Dim.1       Dim.2 Dim.3
Math    -0.9780749  0.2082535     0
English  0.8887667  0.4583599     0
Art      0.9780749 -0.2082535     0

$cor
             Dim.1       Dim.2 Dim.3
Math    -0.9780749  0.2082535     0
English  0.8887667  0.4583599     0
Art      0.9780749 -0.2082535     0

$cos2
            Dim.1       Dim.2 Dim.3
Math    0.9566305 0.04336952     0
English 0.7899062 0.21009384     0
Art     0.9566305 0.04336952     0

$contrib
           Dim.1    Dim.2 Dim.3
Math    35.38925 14.61075   NaN
English 29.22151 70.77849   NaN
Art     35.38925 14.61075   NaN
```

datapcavarcoord

```
> datapca$var$coord
             Dim.1       Dim.2 Dim.3
Math    -0.9780749  0.2082535     0
English  0.8887667  0.4583599     0
Art      0.9780749 -0.2082535     0
> |
```

fviz_screeplot(datapca,addlabels=TRUE,ylim=c(0,50))


R Graphics: Device 2 (ACTIVE)
Scree plot
Percentage of explained variances
50
40
30
20
10
0
9.9%
0%
1
2
3
Dimensions

```
> install.packages('factoextra',repos="http://cran.us.r-project.org")
Installing package into 'C:/Users/Akki/AppData/Local/R/win-library/4.2'
(as 'lib' is unspecified)
also installing the dependencies 'corrplot', 'viridis', 'ggsci', 'cowplot', 'gg$

trying URL 'http://cran.us.r-project.org/bin/windows/contrib/4.2/corrplot_0.92.$
Content type 'application/zip' length 3844780 bytes (3.7 MB)
downloaded 3.7 MB

trying URL 'http://cran.us.r-project.org/bin/windows/contrib/4.2/viridis_0.6.2.$
Content type 'application/zip' length 2999945 bytes (2.9 MB)
downloaded 2.9 MB

trying URL 'http://cran.us.r-project.org/bin/windows/contrib/4.2/ggsci_2.9.zip'
Content type 'application/zip' length 2978453 bytes (2.8 MB)
downloaded 2.8 MB

trying URL 'http://cran.us.r-project.org/bin/windows/contrib/4.2/cowplot_1.1.1.$
```

install.packages('factoextra',repos="http://cran.us.r-project.org") library("factoextra")
fviz_screeplot(datapca,addlabels=TRUE,ylim=c(0,50)) head(iris)
x=iris[,-5]

```
> library("factoextra")
Loading required package: ggplot2
Learn more about the underlying theory at https://ggplot2-book.org/
Welcome! Want to learn more? See two factoextra-related books at https://go
> fviz_screeplot(datapca,addlabels=TRUE,ylim=c(0,50))
> 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
> xiri[,-5]
Error: object 'xiri' not found
> x=iris[,-5]
```

x=iris[,-5]

x

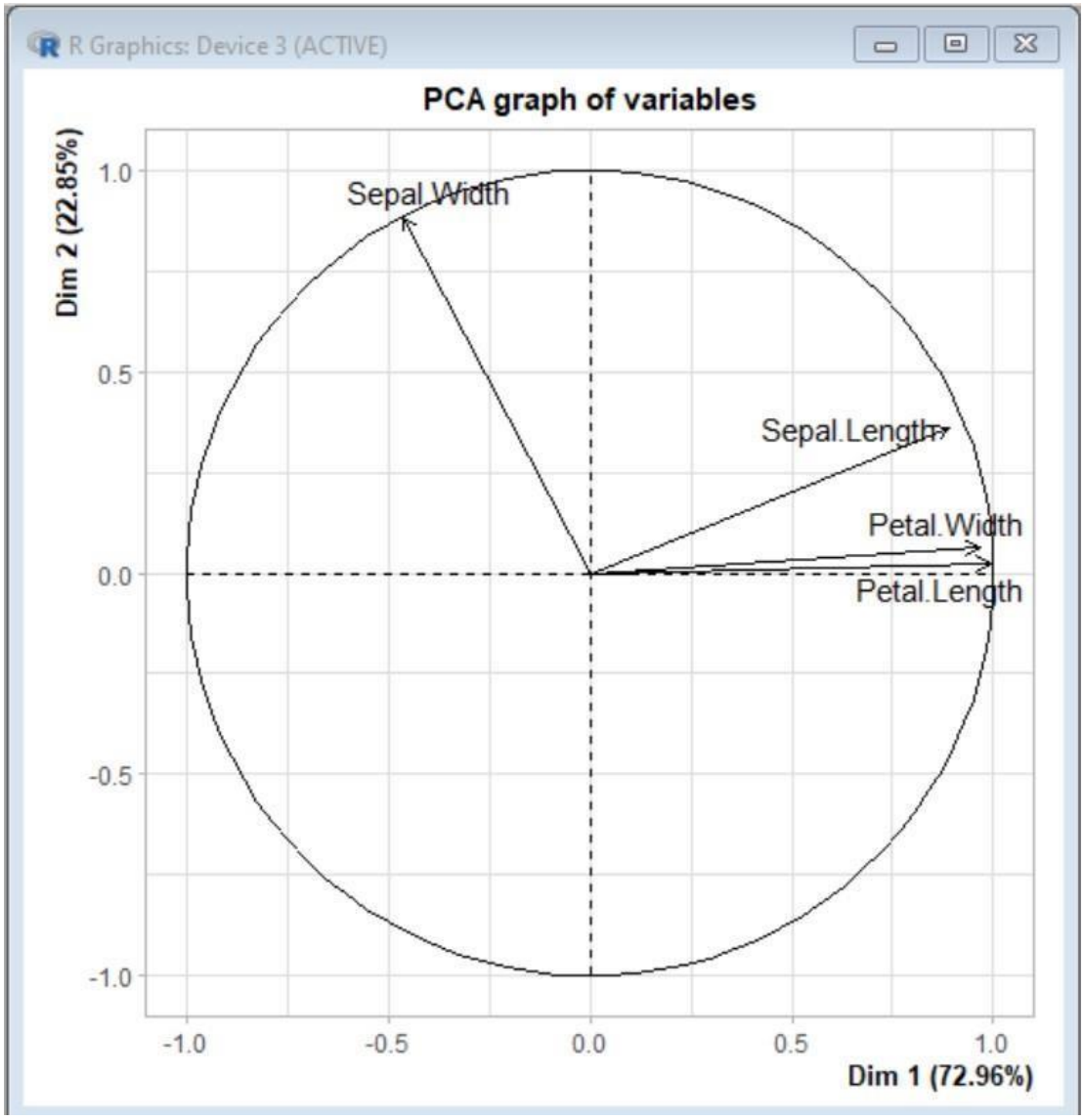
```
> x=iris[,-5]
> x
    Sepal.Length Sepal.Width Petal.Length Petal.Width
1            5.1         3.5          1.4         0.2
2            4.9         3.0          1.4         0.2
3            4.7         3.2          1.3         0.2
4            4.6         3.1          1.5         0.2
5            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
```

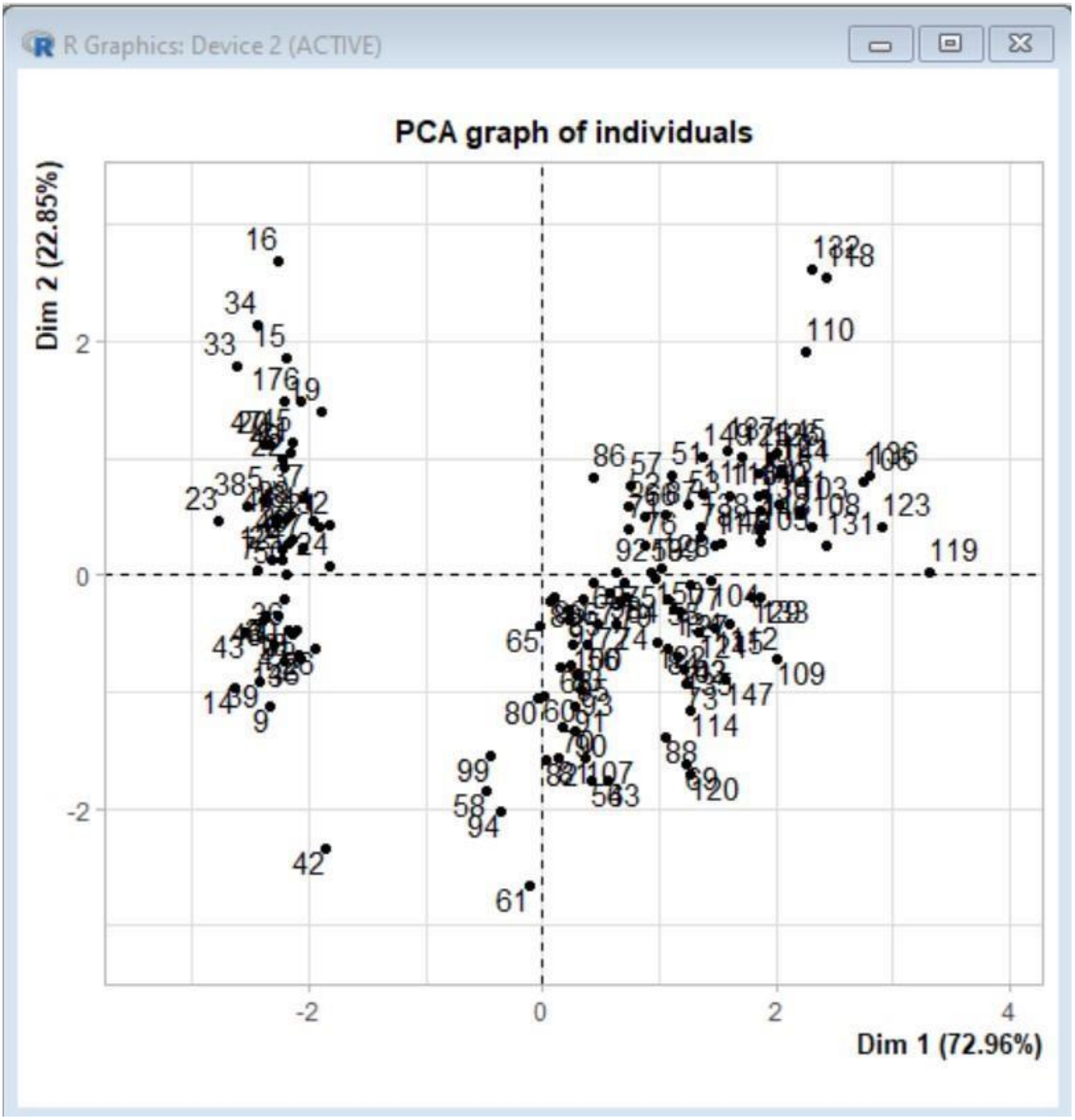
cov_iris=cov(x)

cov_iris

```
> cov_iris=cov(x)
> cov_iris
             Sepal.Length Sepal.Width Petal.Length Petal.Width
Sepal.Length    0.6856935  -0.0424340    1.2743154   0.5162707
Sepal.Width    -0.0424340   0.1899794   -0.3296564  -0.1216394
Petal.Length    1.2743154  -0.3296564    3.1162779   1.2956094
Petal.Width     0.5162707  -0.1216394    1.2956094   0.5810063
> irispca=PCA(x,ncp=3,graph=TRUE)
```

> irispca=PCA(x,ncp=3,graph=TRUE)


R Graphics: Device 3 (ACTIVE)
PCA graph of variables
Dim 2 (22.85%)
1.0
0.5
0.0
-0.5
-1.0
Sepal.Width
Sepal.Length
Petal.Width
Petal.Length
-1.0
-0.5
0.0
0.5
1.0
Dim 1 (72.96%)


R Graphics: Device 2 (ACTIVE)
PCA graph of individuals
Dim 2 (22.85%)
Dim 1 (72.96%)
2
0
-2
-2
0
2
4

irispca

```
> irispca
**Results for the Principal Component Analysis (PCA)**
The analysis was performed on 150 individuals, described by 4 variables
*The results are available in the following objects:

   name               description
1  "$eig"             "eigenvalues"
2  "$var"             "results for the variables"
3  "$var$coord"       "coord. for the variables"
4  "$var$cor"         "correlations variables - dimensions"
5  "$var$cos2"        "cos2 for the variables"
6  "$var$contrib"     "contributions of the variables"
7  "$ind"             "results for the individuals"
8  "$ind$coord"       "coord. for the individuals"
9  "$ind$cos2"        "cos2 for the individuals"
10 "$ind$contrib"     "contributions of the individuals"
11 "$call"            "summary statistics"
12 "$call$centre"     "mean of the variables"
13 "$call$ecart.type" "standard error of the variables"
14 "$call$row.w"      "weights for the individuals"
15 "$call$col.w"      "weights for the variables"
>
```

summary(irispca)

```
> summary(irispca)

Call:
PCA(X = x, ncp = 3, graph = TRUE)


Eigenvalues
                       Dim.1   Dim.2   Dim.3   Dim.4
Variance               2.918   0.914   0.147   0.021
% of var.             72.962  22.851   3.669   0.518
Cumulative % of var.  72.962  95.813  99.482 100.000

Individuals (the 10 first)
                  Dist    Dim.1    ctr   cos2    Dim.2    ctr   cos2    Dim.3
1              |  2.319 | -2.265  1.172  0.954 |  0.480  0.168  0.043 | -0.128
2              |  2.202 | -2.081  0.989  0.893 | -0.674  0.331  0.094 | -0.235
3              |  2.389 | -2.364  1.277  0.979 | -0.342  0.085  0.020 |  0.044
4              |  2.378 | -2.299  1.208  0.935 | -0.597  0.260  0.063 |  0.091
5              |  2.476 | -2.390  1.305  0.932 |  0.647  0.305  0.068 |  0.016
6              |  2.555 | -2.076  0.984  0.660 |  1.489  1.617  0.340 |  0.027
7              |  2.468 | -2.444  1.364  0.981 |  0.048  0.002  0.000 |  0.335
8              |  2.246 | -2.233  1.139  0.988 |  0.223  0.036  0.010 | -0.089
9              |  2.592 | -2.335  1.245  0.812 | -1.115  0.907  0.185 |  0.145
```

Conclusion: Hence, we successfully implemented Principal component analysis.

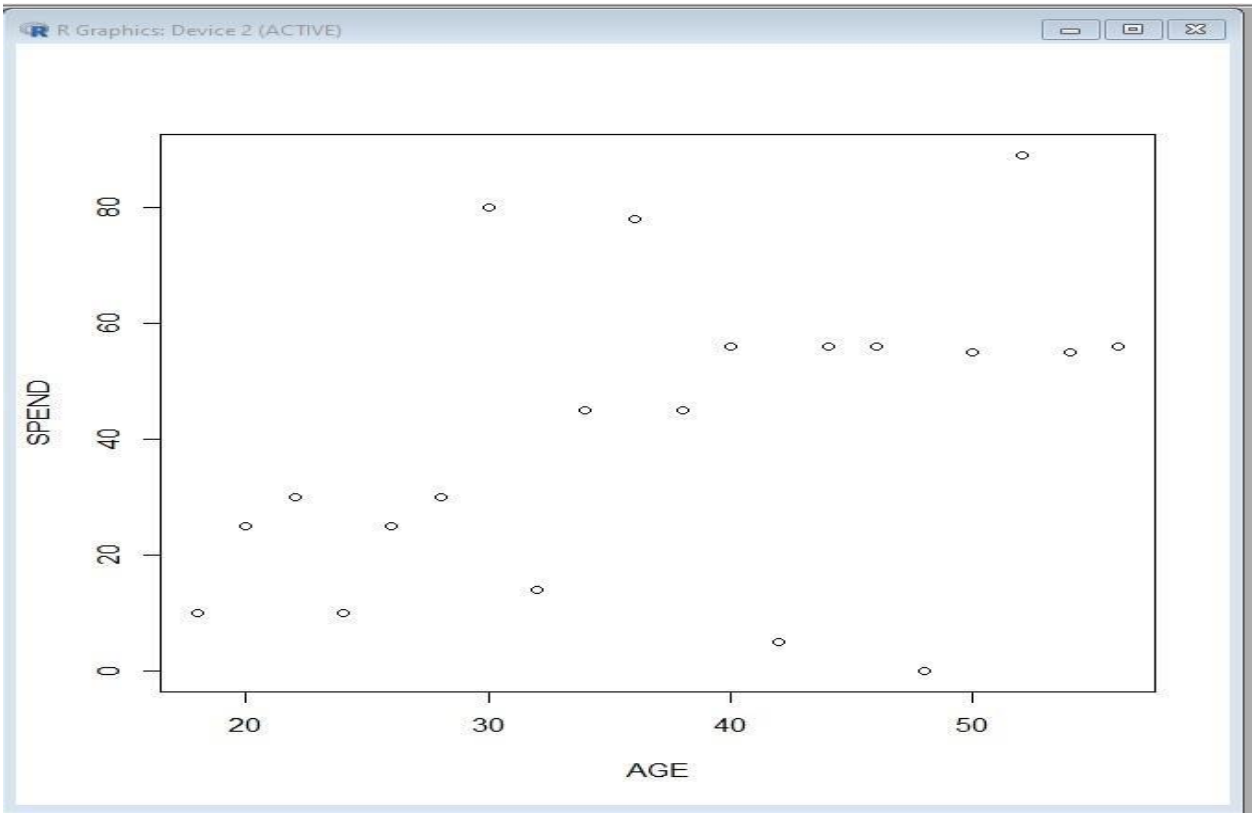
Name: Sumit Singh
Roll No: 380
Class: TYBSC CS A
Subject: Data Science
Practical No : 8

Practical No 8

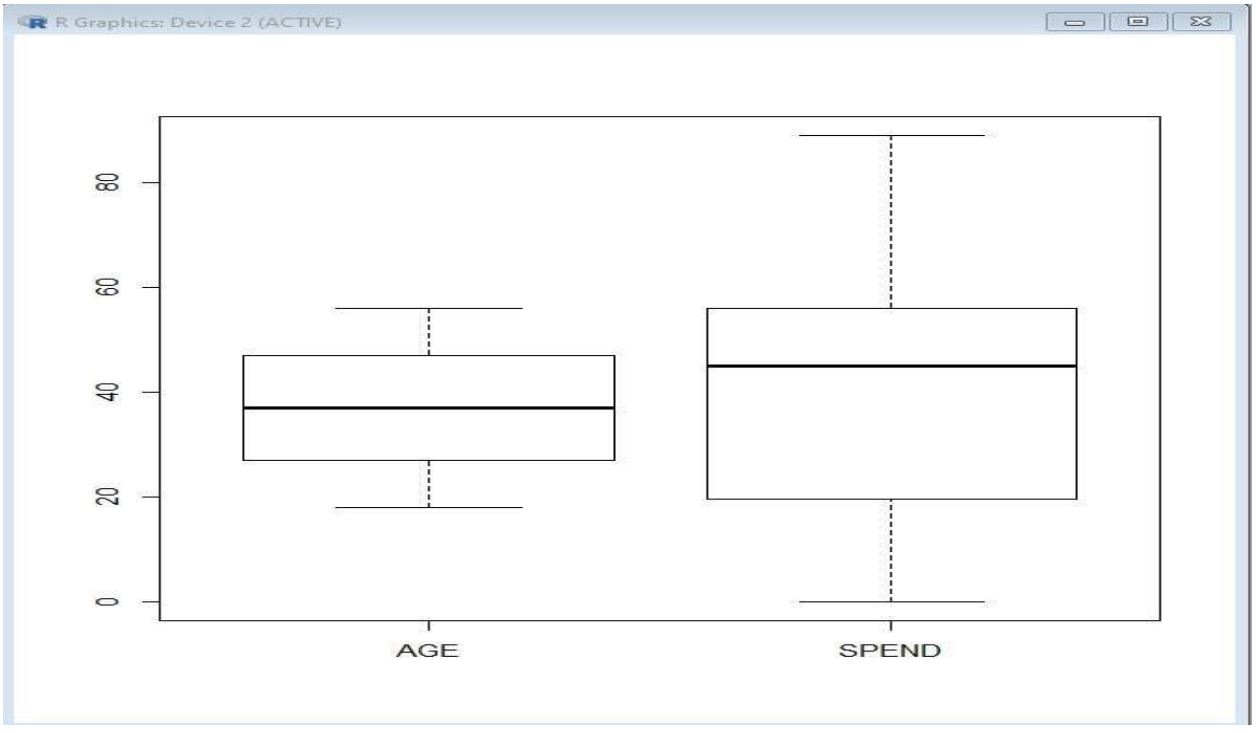
Aim: Demonstration of Clustering Code: df=read.csv("C:/Users/admin/Documents/AGE.csv") df

```
> df=read.csv("C:/Users/admin/Documents/AGE.csv")
> df
   AGE SPEND
1   18    10
2   20    25
3   22    30
4   24    10
5   26    25
6   28    30
7   30    80
8   32    14
9   34    45
10  36    78
11  38    45
12  40    56
13  42     5
14  44    56
15  46    56
16  48     0
17  50    55
18  52    89
19  54    55
20  56    56
```

plot(df)



boxplot(df)



Make the cluster

>set.seed(20)

> c1=kmeans(df[,1:2],3)

> c1

```
> set.seed(20)
> c1=kmeans(df[,1:2],3)
> c1
K-means clustering with 3 clusters of sizes 3, 8, 9

Cluster means:
       AGE    SPEND
1 39.33333 82.33333
2 45.25000 53.00000
3 28.88889 16.55556

Clustering vector:
 [1] 3 3 3 3 3 3 1 3 2 1 2 2 3 2 2 3 2 1 2 2

Within cluster sum of squares by cluster:
[1]  327.3333  595.5000 1829.1111
 (between_SS / total_SS =  82.3 %)

Available components:

[1] "cluster"      "centers"      "totss"        "withinss"     "tot.withinss"
[6] "betweenss"    "size"         "iter"         "ifault"
```

#SHOW THE IRIS DATA SET

>iris

```
> 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
7            4.6         3.4          1.4         0.3     setosa
8            5.0         3.4          1.5         0.2     setosa
9            4.4         2.9          1.4         0.2     setosa
10           4.9         3.1          1.5         0.1     setosa
11           5.4         3.7          1.5         0.2     setosa
12           4.8         3.4          1.6         0.2     setosa
13           4.8         3.0          1.4         0.1     setosa
```

#View(iris)

Data: iris

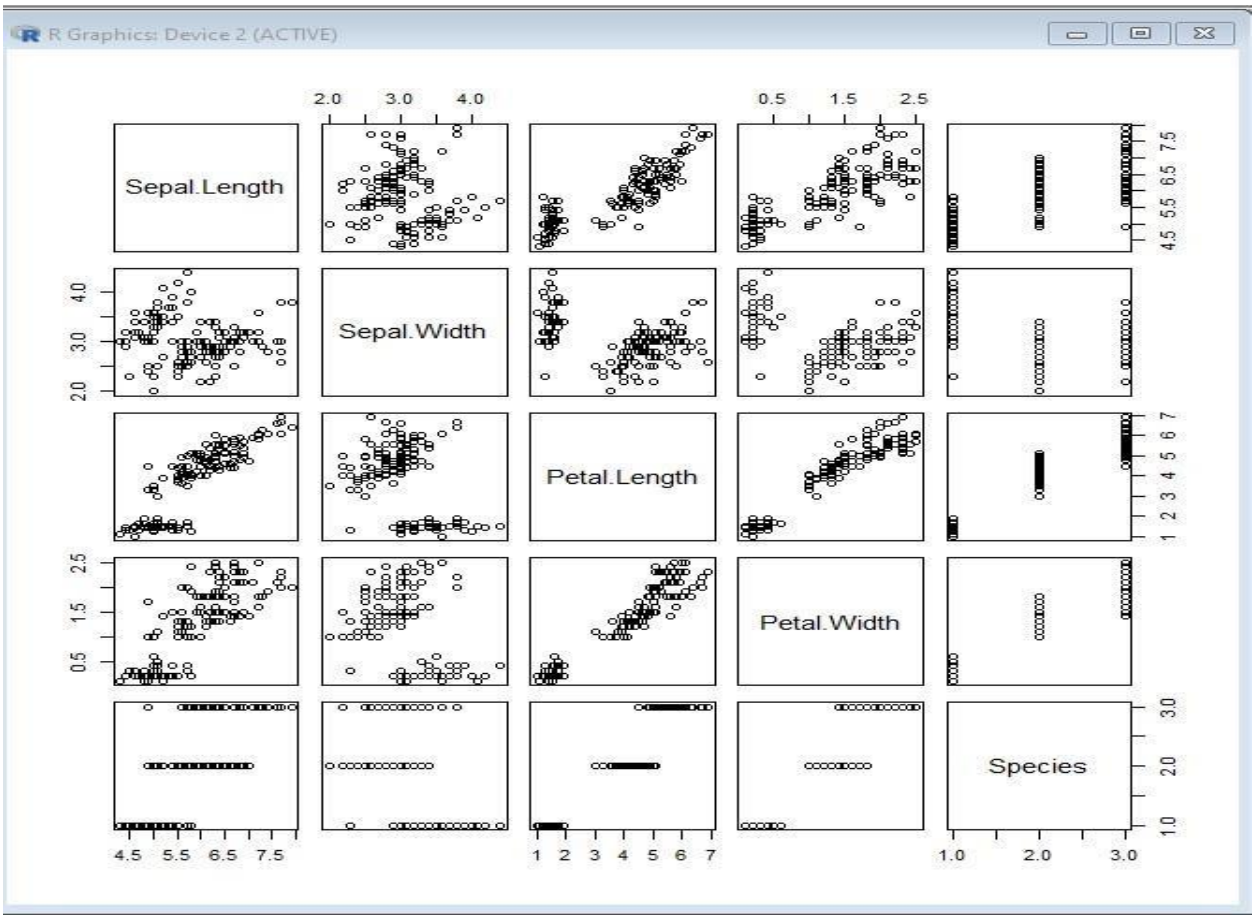
	Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Species
1	5.1	3.5	1.4	0.2	setosa
2	4.9	3.0	1.4	0.2	setosa
3	4.7	3.2	1.3	0.2	setosa
4	4.6	3.1	1.5	0.2	setosa
5	5.0	3.6	1.4	0.2	setosa
6	5.4	3.9	1.7	0.4	setosa
7	4.6	3.4	1.4	0.3	setosa
8	5.0	3.4	1.5	0.2	setosa
9	4.4	2.9	1.4	0.2	setosa
10	4.9	3.1	1.5	0.1	setosa
11	5.4	3.7	1.5	0.2	setosa
12	4.8	3.4	1.6	0.2	setosa
13	4.8	3.0	1.4	0.1	setosa
14	4.3	3.0	1.1	0.1	setosa
15	5.8	4.0	1.2	0.2	setosa
16	5.7	4.4	1.5	0.4	setosa
17	5.4	3.9	1.3	0.4	setosa
18	5.1	3.5	1.4	0.3	setosa
19	5.7	3.8	1.7	0.3	setosa

head(iris)

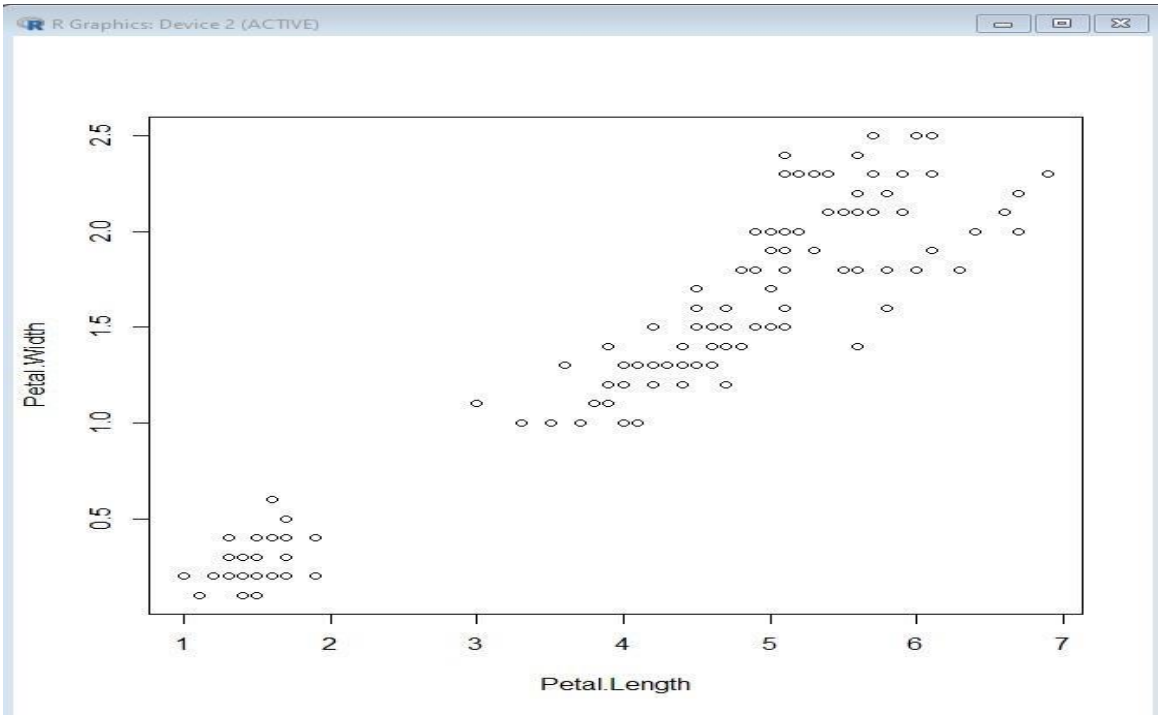
summary(iris)

```
> 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
```

plot(iris)



plot(iris[,3:4])



kmeansc1=kmeans(iris[,3:4],3) kmeansc1

```
> kmeanscl=kmeans(iris[,3:4],3)
> kmeanscl
K-means clustering with 3 clusters of sizes 50, 46, 54

Cluster means:
  Petal.Length Petal.Width
1     1.462000    0.246000
2     5.626087    2.047826
3     4.292593    1.359259

Clustering vector:
  [1] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 [38] 1 1 1 1 1 1 1 1 1 1 1 1 1 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
 [75] 3 3 3 2 3 3 3 3 3 2 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 2 2 2 2 2 2 3 2 2 2 2
[112] 2 2 2 2 2 2 2 2 3 2 2 2 3 2 2 3 3 2 2 2 2 2 2 2 2 2 2 3 2 2 2 2 2 2 2 2 2
[149] 2 2

Within cluster sum of squares by cluster:
[1]  2.02200 15.16348 14.22741
 (between_SS / total_SS =  94.3 %)

Available components:

[1] "cluster"      "centers"      "totss"        "withinss"     "tot.withinss"
[6] "betweenss"    "size"         "iter"         "ifault"
> |
```

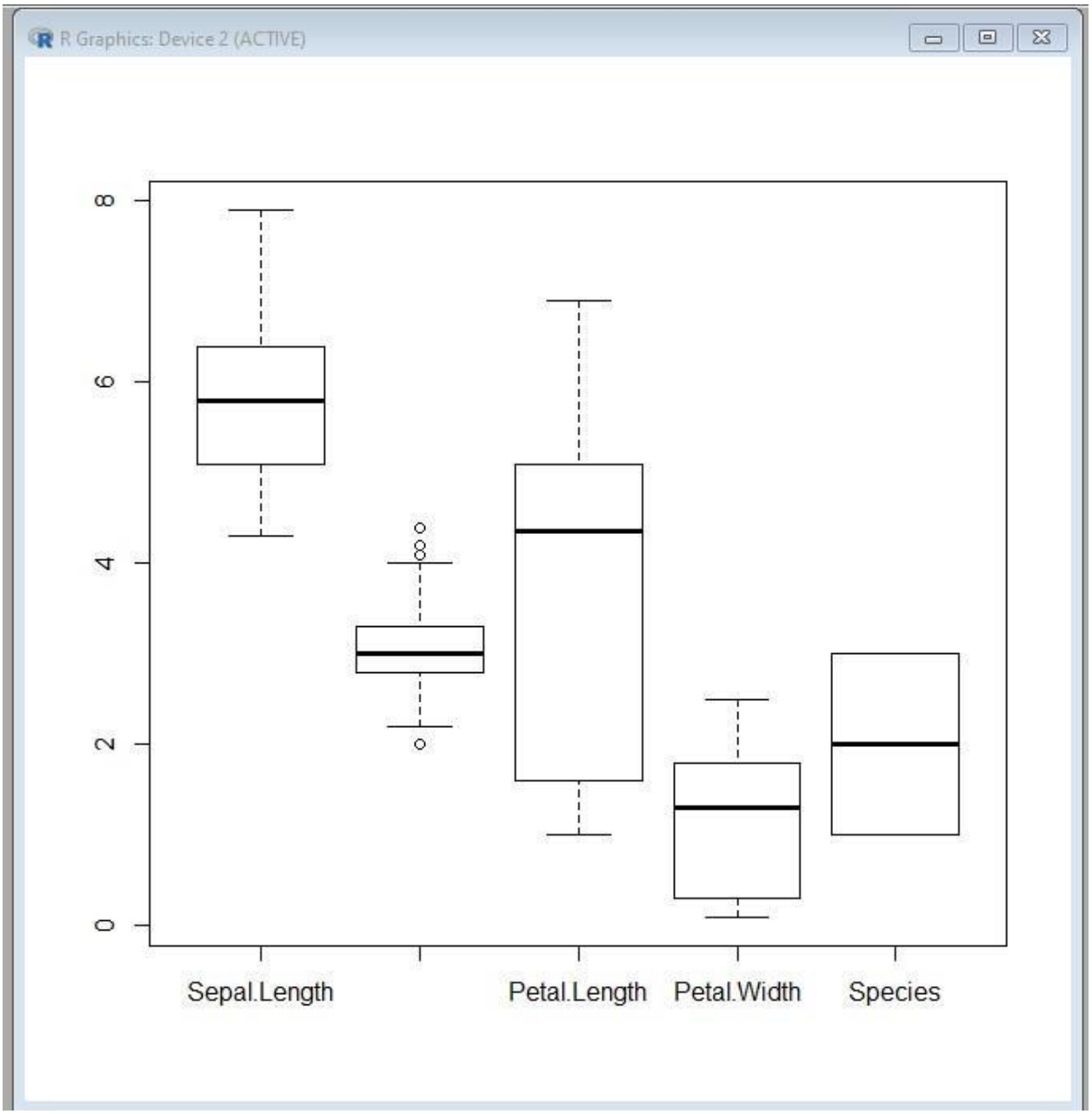
PRINT CONFUSION MATRIX

>table(kmeansc1$cluster,iris$Species)

```
> table(kmeanscl$cluster,iris$Species)

   setosa versicolor virginica
1      50          0         0
2       0          2        44
3       0         48         6
> |
```

CALCULATION OF ACCURACY 94.6%

boxplot(iris)



Conclusion: Hence, we successfully Implemented clustering.

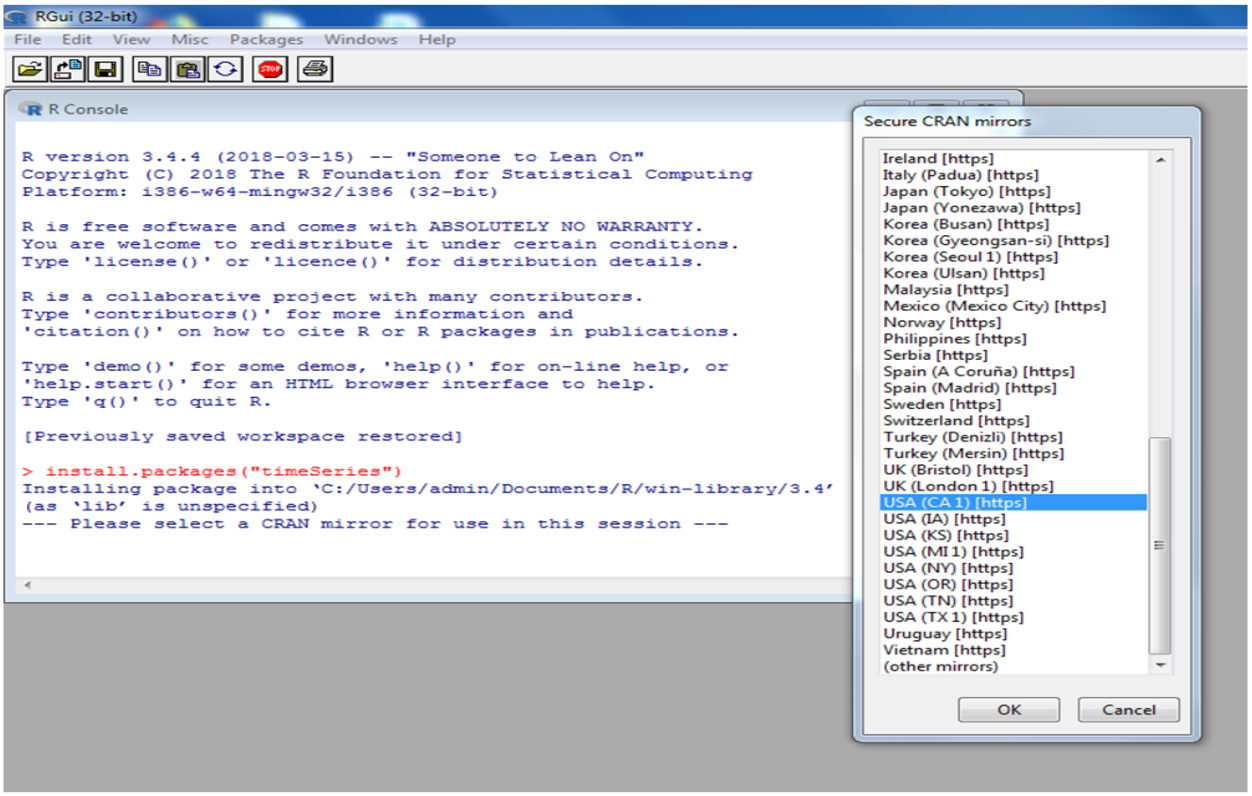
Name: Sumit Singh
Roll No:380
Class: TYBSC CS A
Subject: Data Science
Practical No : 9

Practical No 9

Aim: Demonstration of Time-series forecasting.

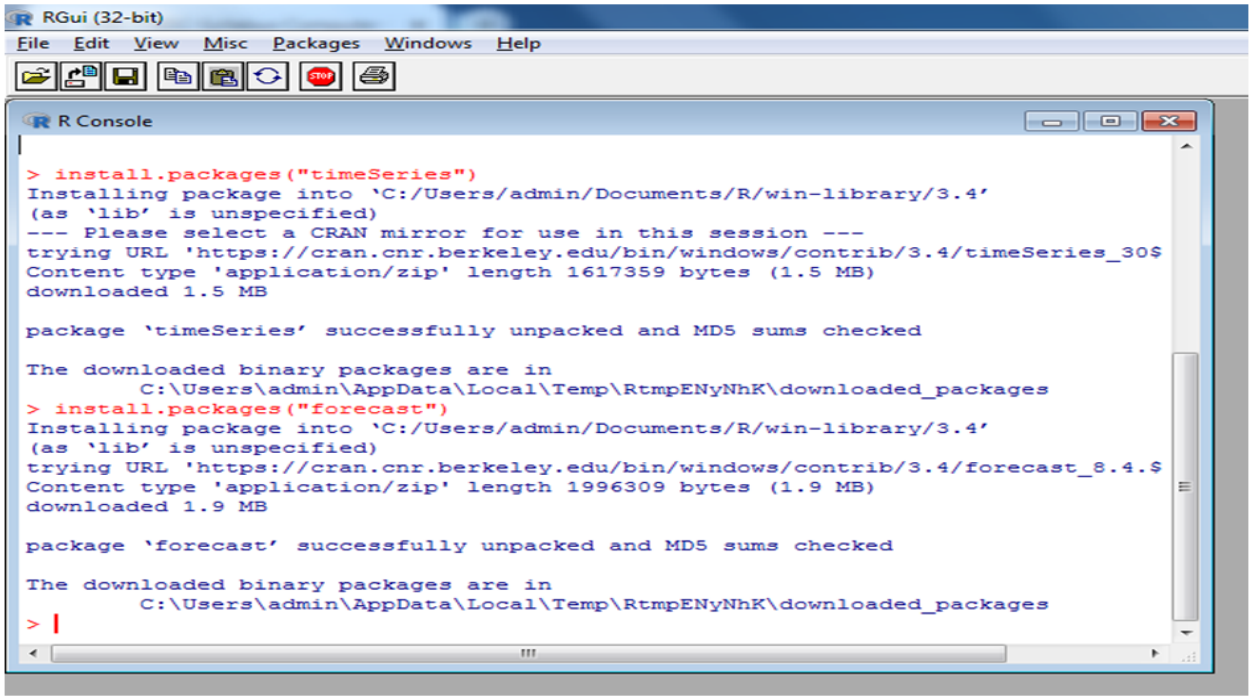
STEP1: Install time series

install.packages("timeSeries")



Step 2: Install package forecast

install.packages("forecast")



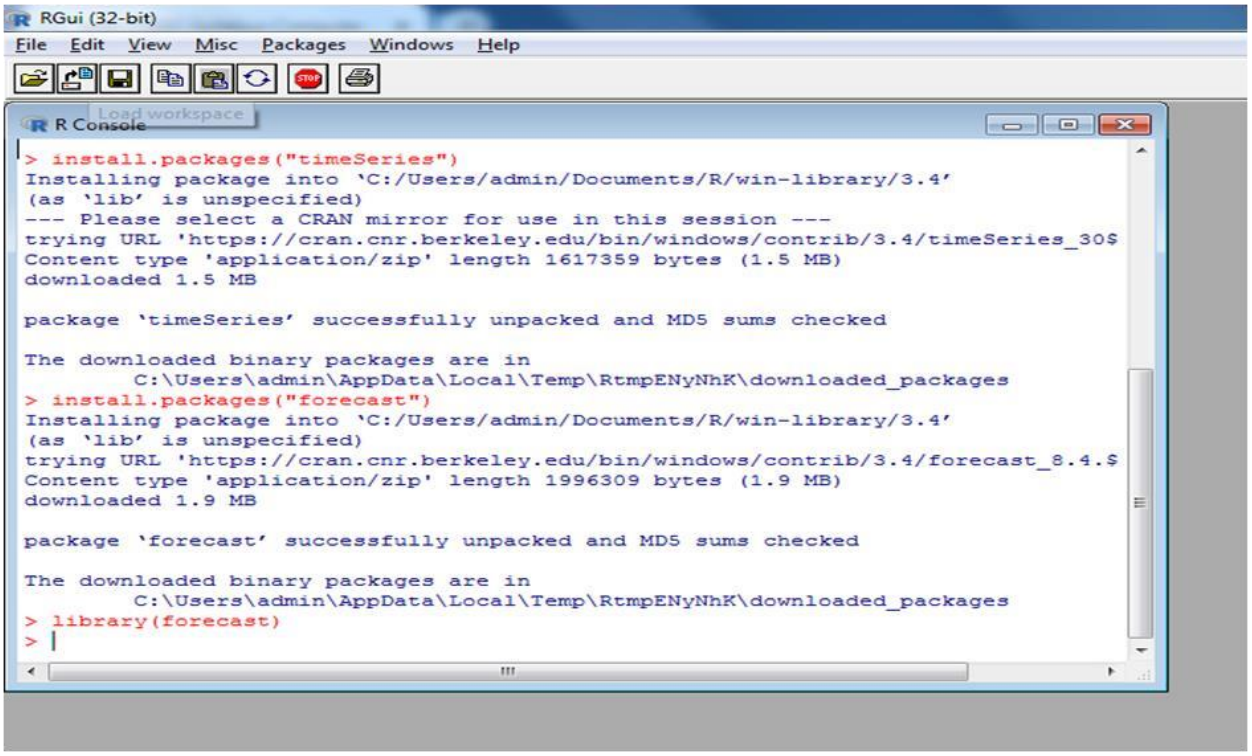
data1=table(AirPassengers)

Data1 view(data1)

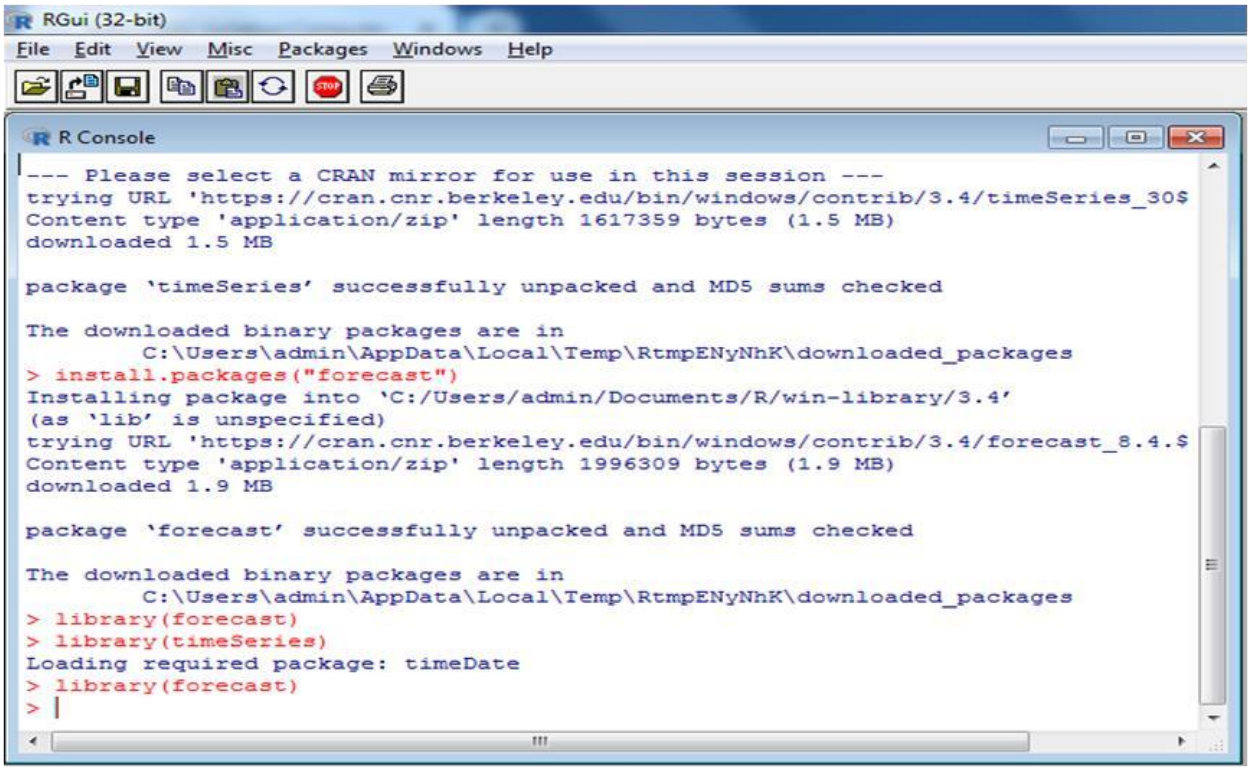
```
> data1
AirPassengers
104 112 114 115 118 119 121 125 126 129 132 133 135 136 140 141 145 146 148 149
  1   1   1   1   2   1   1   1   1   1   1   1   2   1   1   1   1   1   2   1
150 158 162 163 166 170 171 172 178 180 181 183 184 188 191 193 194 196 199 201
  1   1   1   1   1   2   1   2   2   2   1   1   1   1   1   1   1   2   2   1
203 204 209 211 218 227 229 230 233 234 235 236 237 242 243 259 264 267 269 270
  1   1   1   1   1   1   3   1   1   1   2   1   2   2   1   1   2   1   1   1
271 272 274 277 278 284 293 301 302 305 306 310 312 313 315 317 318 336 337 340
  1   1   1   1   1   1   1   1   1   1   2   1   1   1   2   1   2   1   1   1
342 347 348 355 356 359 360 362 363 364 374 390 391 396 404 405 406 407 413 417
  1   2   2   2   1   1   1   2   1   1   1   1   1   1   2   2   1   1   1   1
419 420 422 432 435 461 463 465 467 472 491 505 508 535 548 559 606 622
  1   1   1   1   1   2   1   1   1   2   1   1   1   1   1   1   1   1
> View (data1)
>
```

Step 3: library (timeSeries)

#library(forecast)



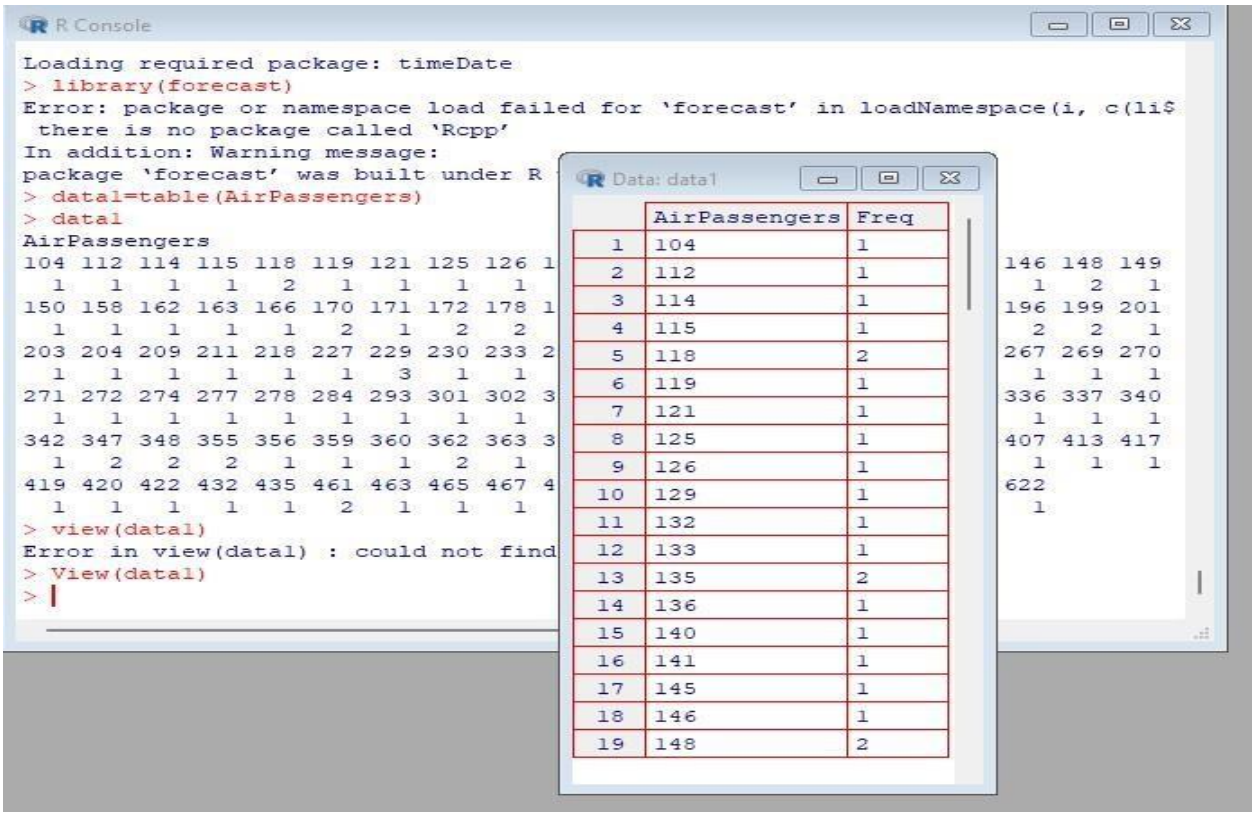
library(forecast)



Step 5: Air Passengers data

data1=table(AirPassengers)

data1



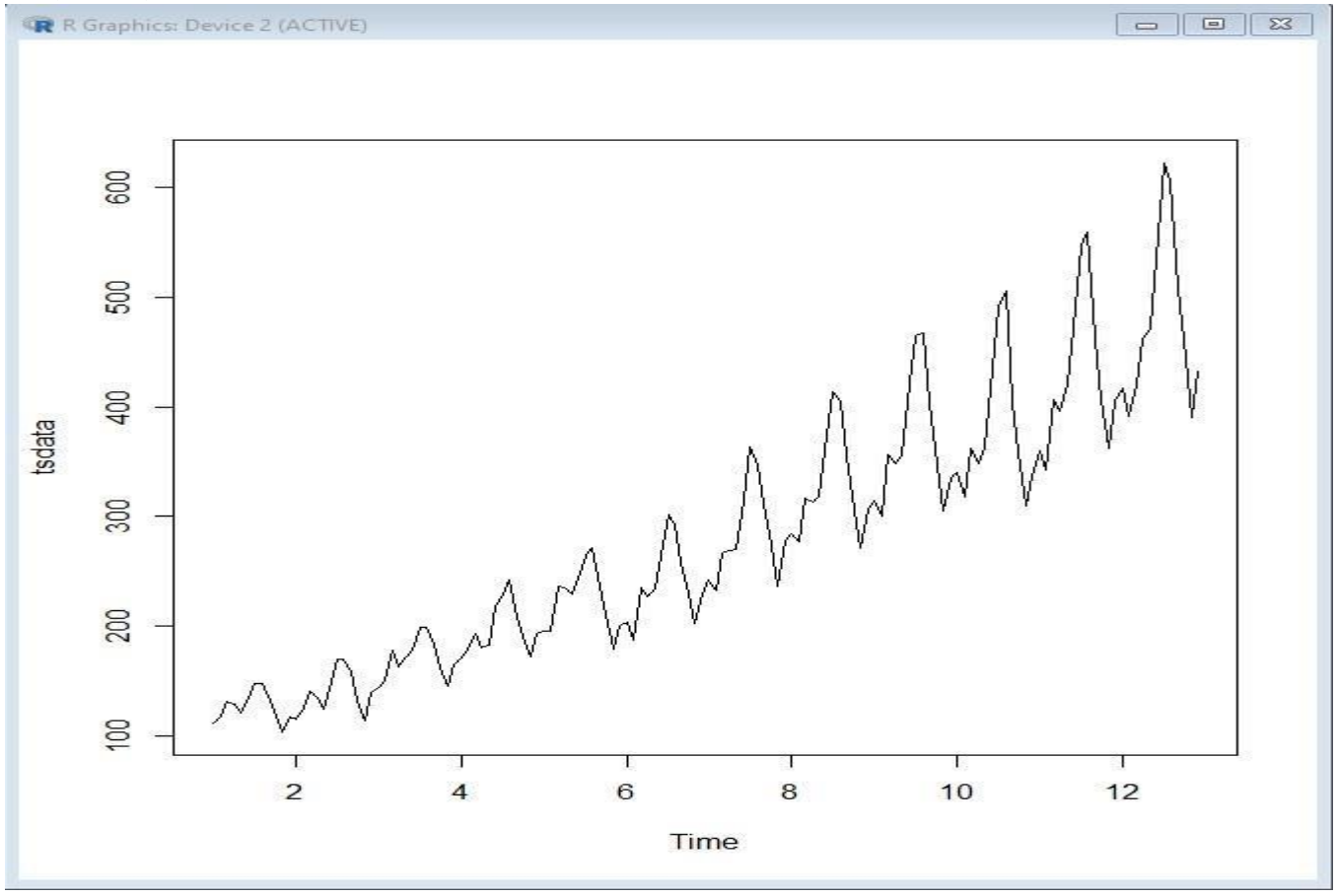
frequency (AirPassengers)

```
> frequency (AirPassengers)
[1] 12
>
```

tsdata=ts(AirPassengers,frequency=12) >
tsdata

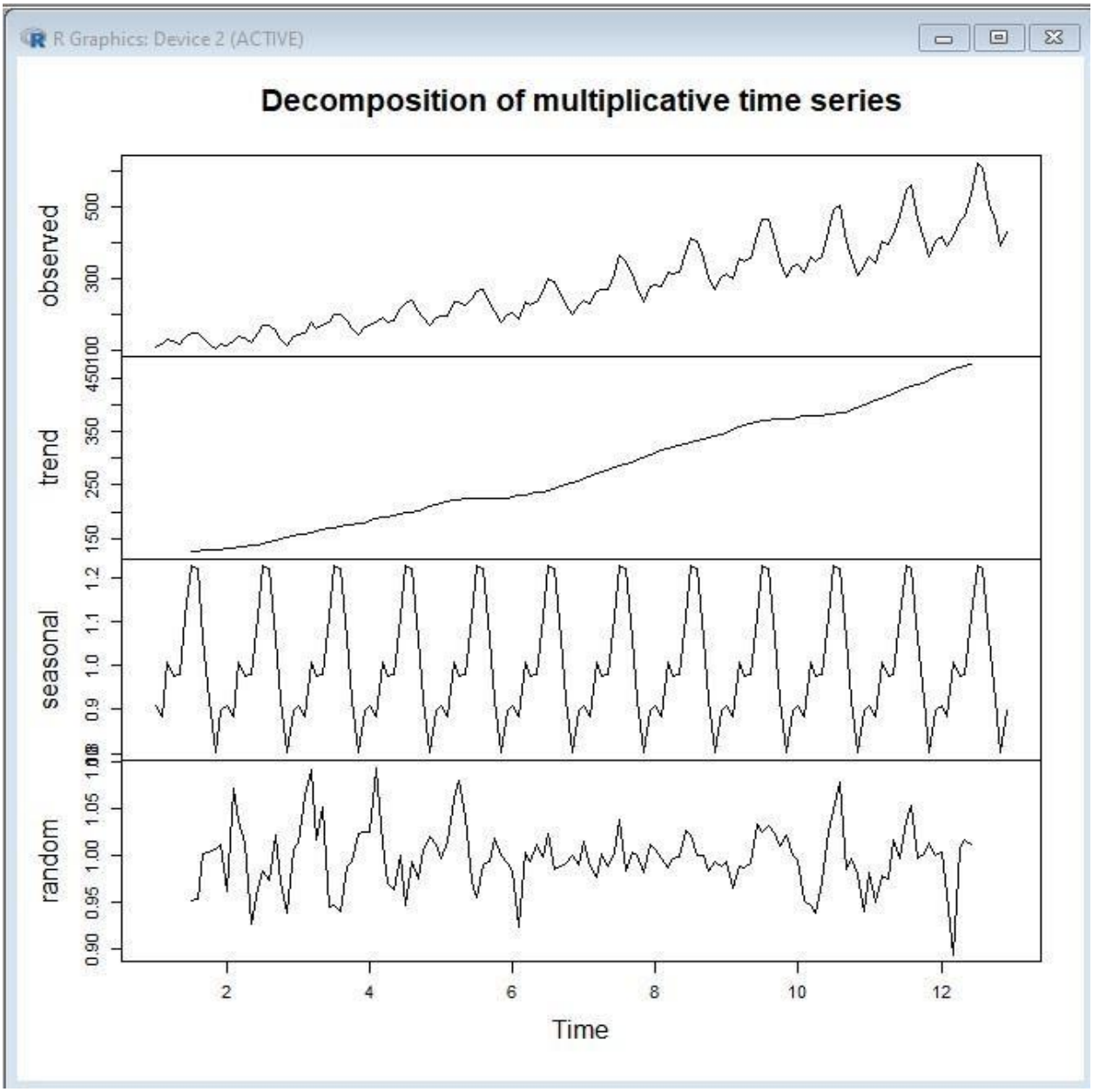
```
> tsdata=ts(AirPassengers,frequency=12)
> tsdata
    Jan Feb Mar Apr May Jun Jul Aug Sep Oct Nov Dec
1   112 118 132 129 121 135 148 148 136 119 104 118
2   115 126 141 135 125 149 170 170 158 133 114 140
3   145 150 178 163 172 178 199 199 184 162 146 166
4   171 180 193 181 183 218 230 242 209 191 172 194
5   196 196 236 235 229 243 264 272 237 211 180 201
6   204 188 235 227 234 264 302 293 259 229 203 229
7   242 233 267 269 270 315 364 347 312 274 237 278
8   284 277 317 313 318 374 413 405 355 306 271 306
9   315 301 356 348 355 422 465 467 404 347 305 336
10  340 318 362 348 363 435 491 505 404 359 310 337
11  360 342 406 396 420 472 548 559 463 407 362 405
12  417 391 419 461 472 535 622 606 508 461 390 432
> |
```

plot(tsdata)

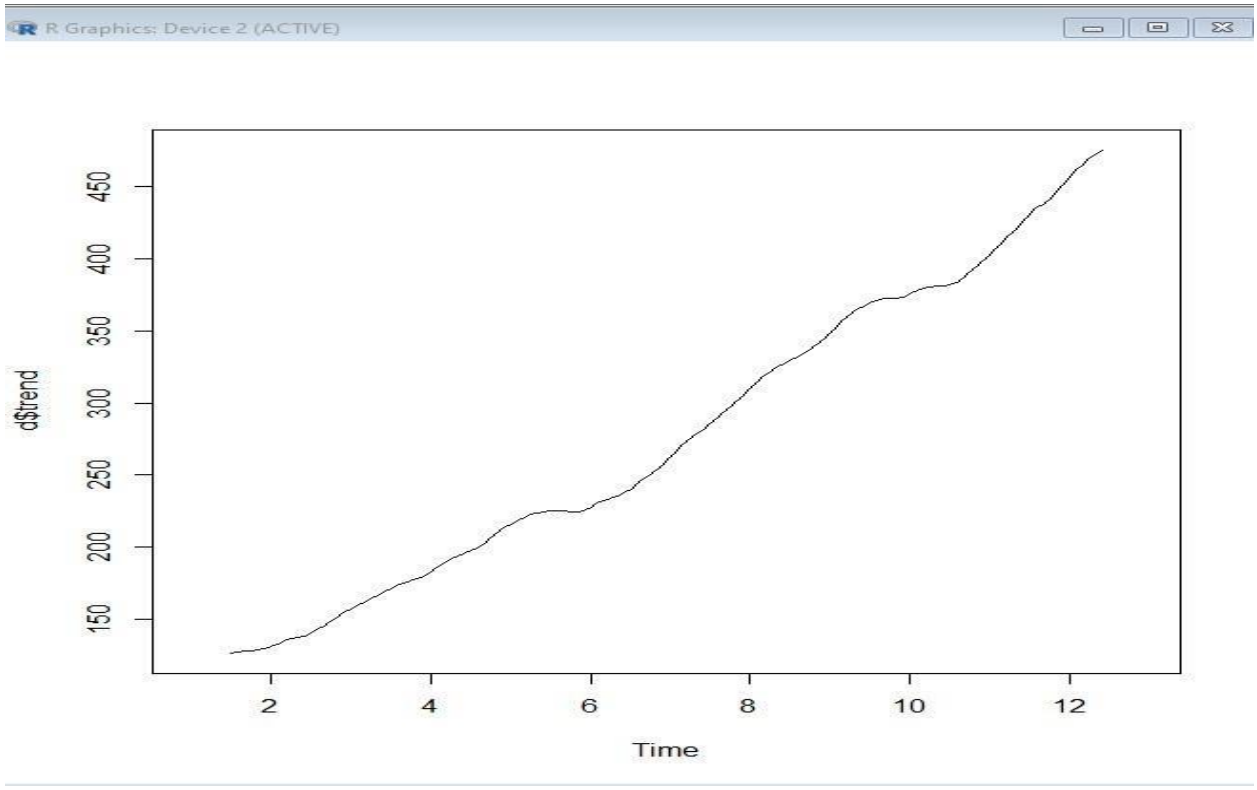


> d=decompose(tsdata,"multiplicative")

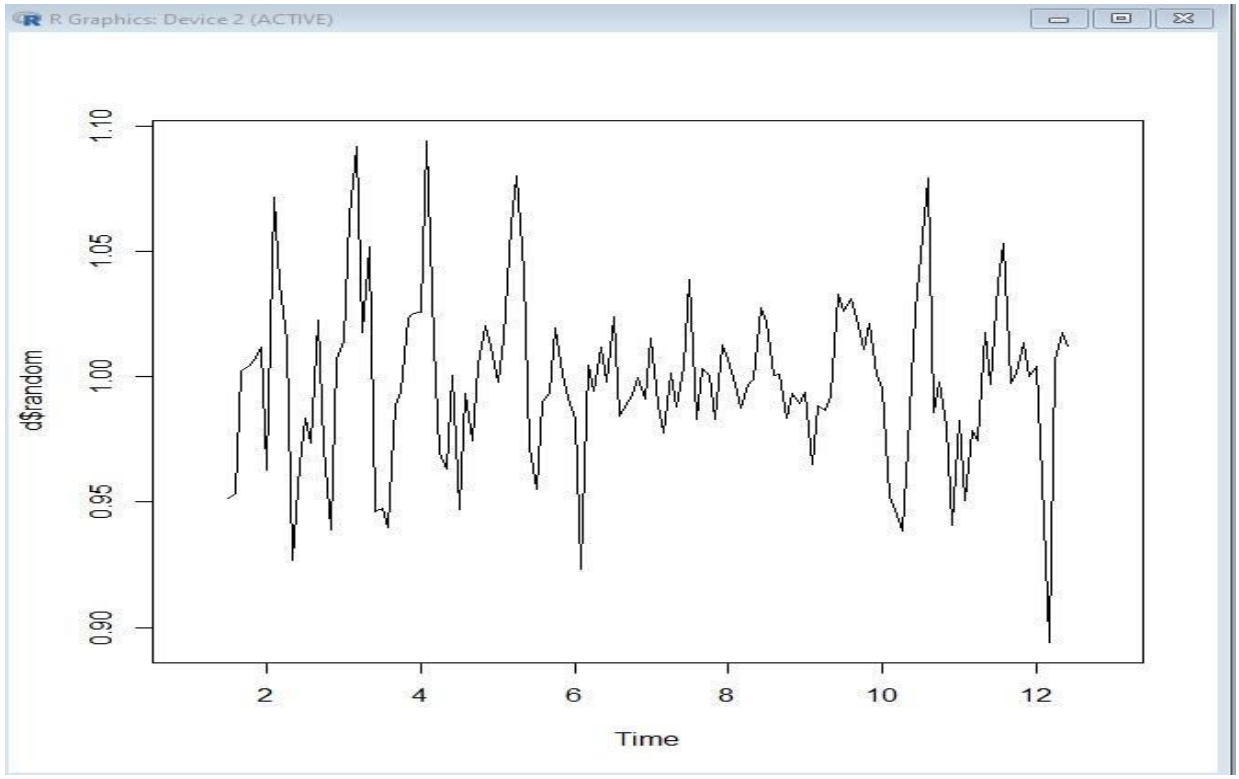
> plot(d)


R Graphics: Device 2 (ACTIVE)
Decomposition of multiplicative time series
observed
trend
seasonal
random
Time
2
4
6
8
10
12

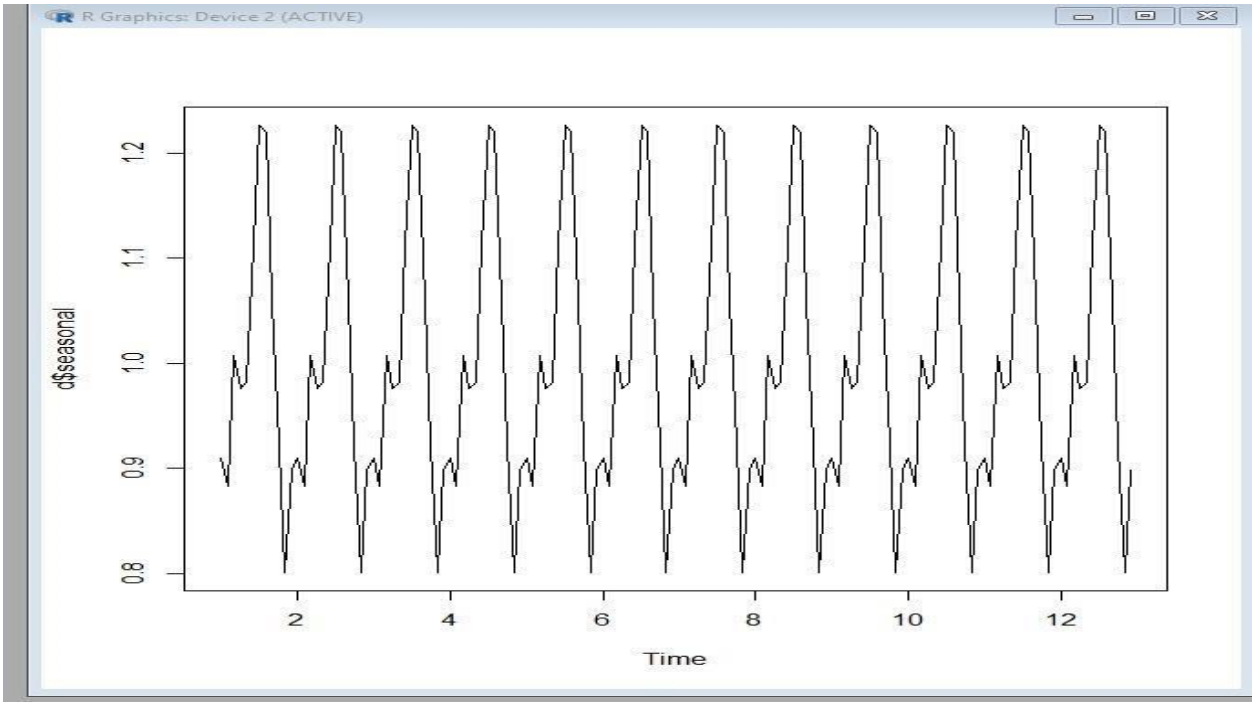
plot(d$trend)



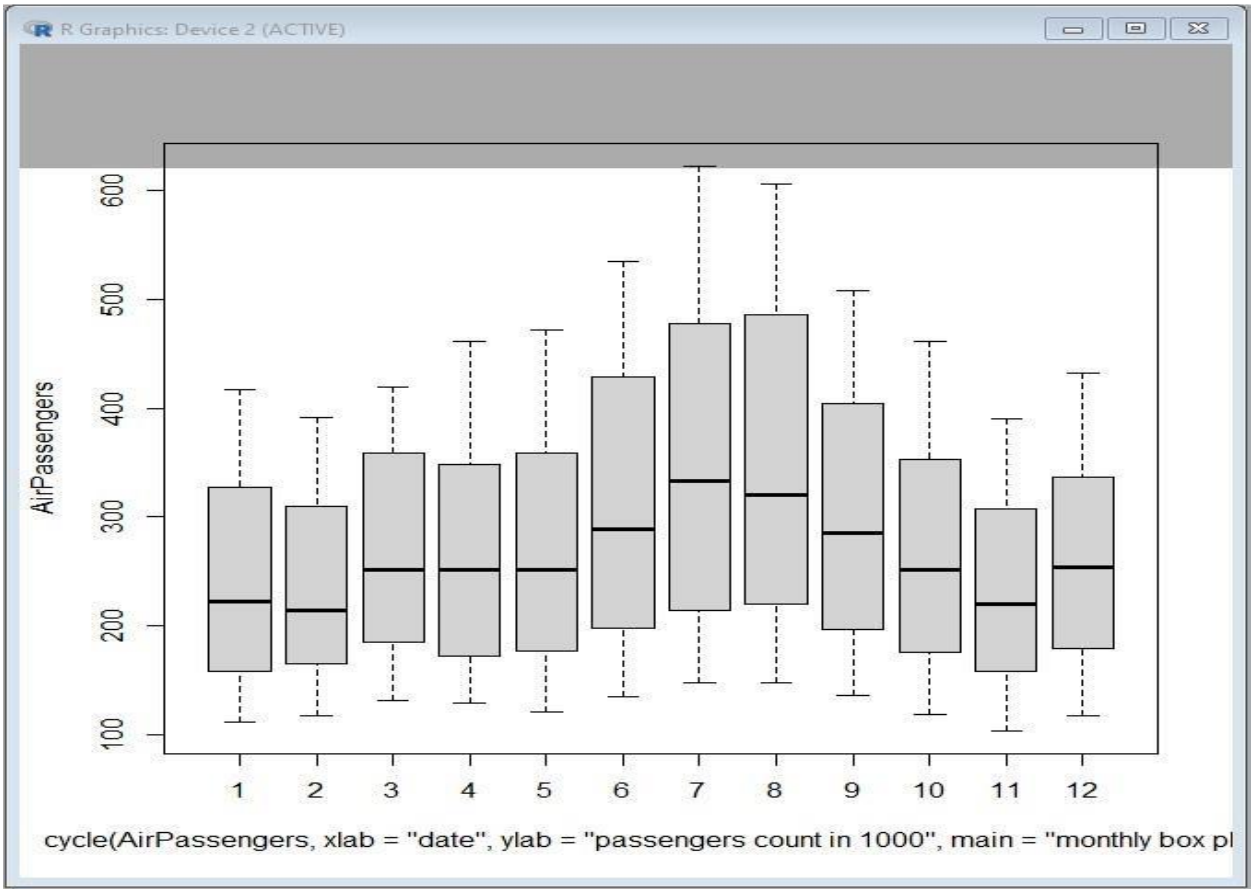
plot(d$random)



plot(d$seasonal)



boxplot(AirPassengers~cycle(AirPassengers,xlab="date",ylab="passengers count in 1000",main="monthly box plot"))



mymodel<- arima(AirPassengers) mymodel

```
> mymodel<- arima(AirPassengers)
> mymodel

Call:
arima(x = AirPassengers)

Coefficients:
      intercept
       280.2986
s.e.     9.9624

sigma^2 estimated as 14292:  log likelihood = -893.18,  aic = 1790.37
>
```

Conclusion: Hence we successfully implemented Time series forecasting.

Name: Sumit Singh

Roll No: 380

Class: TYBSC CS A

Subject: Data Science

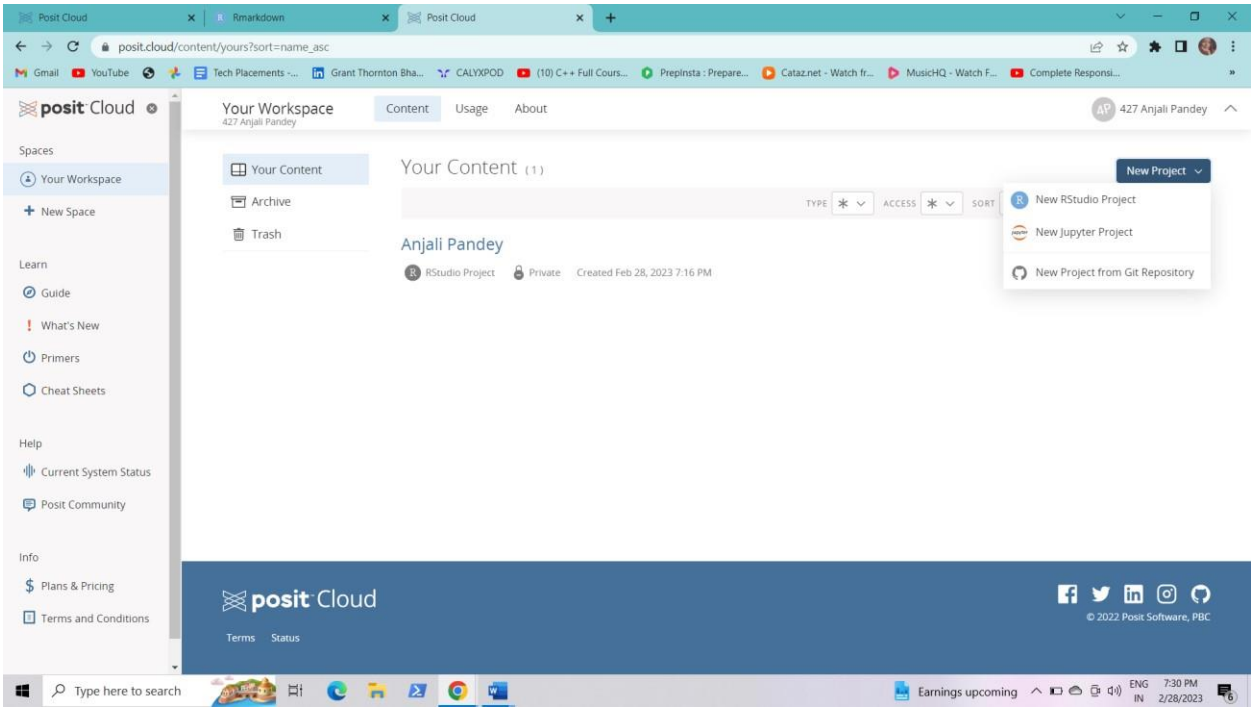
Practical No : 10

Aim: Use of R Markdown and RStudio Cloud (Store mini project in RStudio Cloud) **Steps:**

Step1: Create a rstudio cloud account and login to that account.

An user interface like this will appear on the screen.

Click on New project > New Rstudio Project.



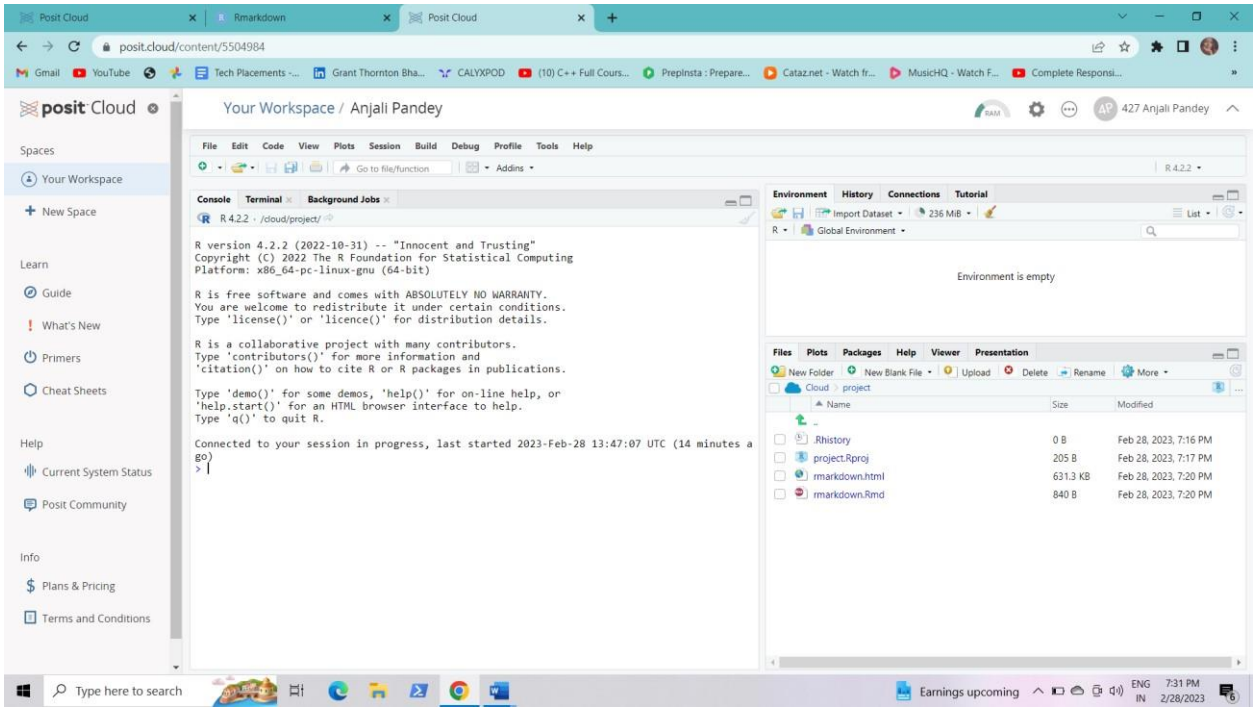
Step2: A window like this will appear on the screen.

Click on file> Rmarkdown

A list of packages will be installed on your Rstudio.

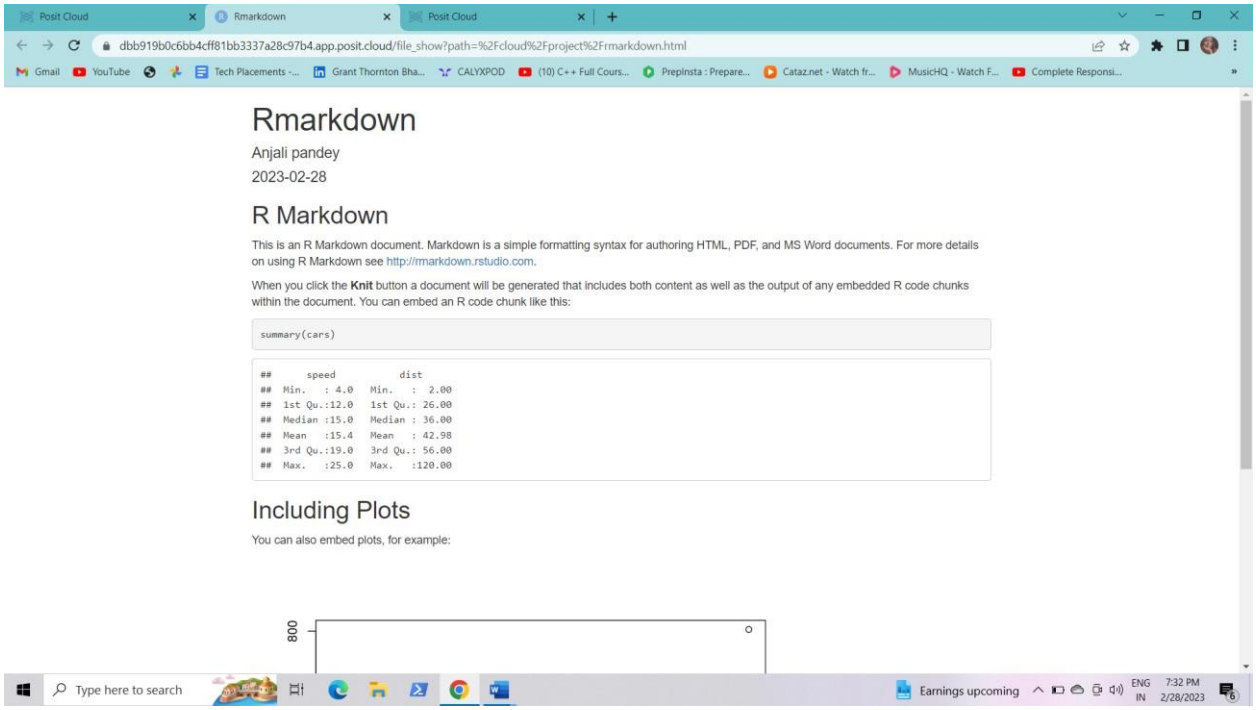
A window will appear on the screen where you have to put title and the author name.

Click on save.

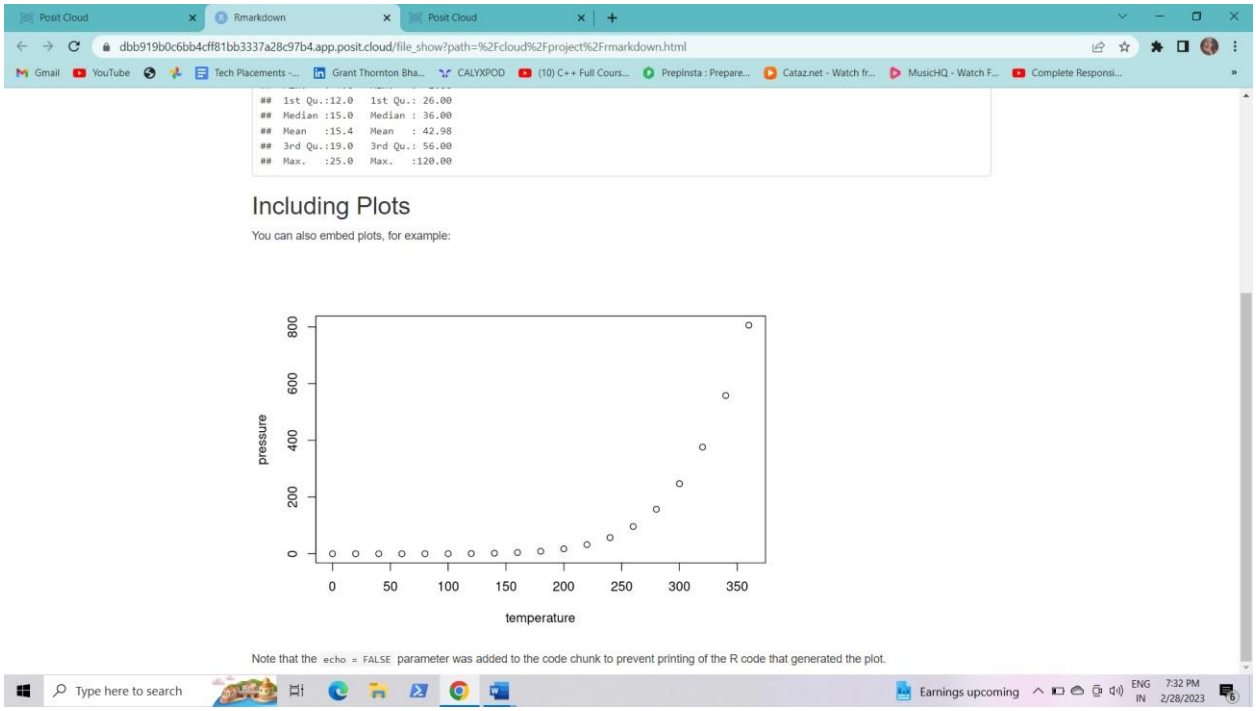


Step3: Save your project and give it HTML doctype and click on knit.

A new tab will be opened with the following content on the screen.



Step4: Apart from code, your graph will also be displayed here.



Conclusion: Hence we successfully Used of R Markdown and RStudio Cloud (Stored mini project in RStudio Cloud)