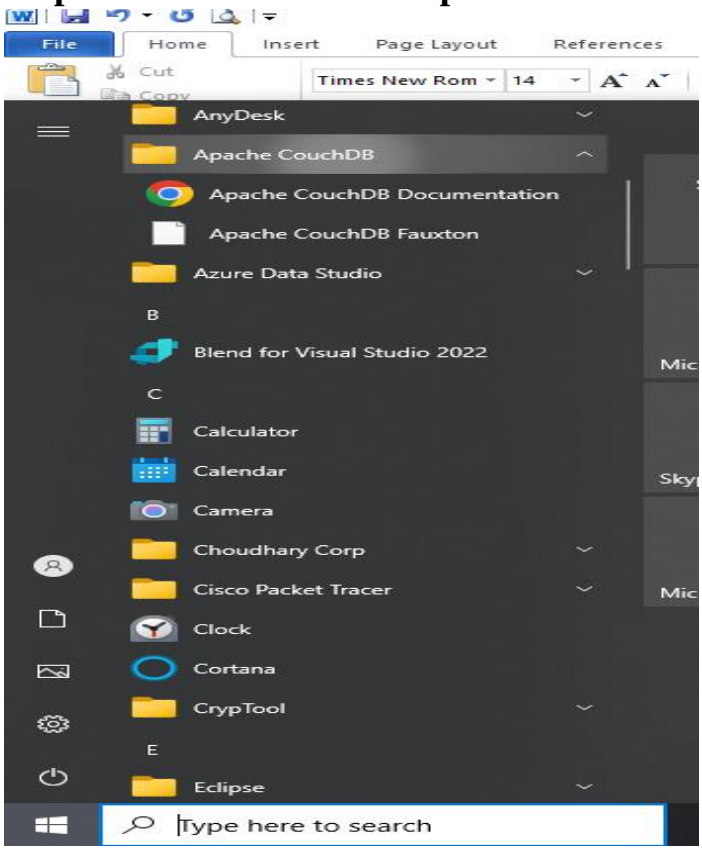
PRACTICAL-1

AIM- Data Curation and Management using NoSQL and R

Step1: Install Couch db from https://couchdb.apache.org/

Step 2: Check in folder of apache couch db like shown



Step 3: open Apache CouchDB Fauxton in couch db folder

Step 4: create a server admin account

Step 5:create some database

```
install.packages("sofa")
library(sofa)
install.packages("")
library(R4CouchDB)
library(couchDB)
install.packages("devtools")
devtools::install_github("ropensci/sofa")
library(sofa)
z<-Cushion$new(host="localhost",user="admin",pwd="admin")
#  host="stuff.cloudant.com",
#  transport="https",
# port=NULL,
#  user='foobar',
# pwd='things')
x<-Cushion$new()
```

```
z$ping()
db_list(z)
db_create(z,dbname="criminalsdb")
db_alldocs(z, dbname="criminalsdb")
doc1 <-'{"name":"criminals","crime":"theft"}'
doc_create(z,doc1,dbname = "criminalsdb",docid = "weapons")
doc2 <-'{"class":"regular","gang":"yes"}'
doc_create(z,doc2,dbname = "criminalsdb")
db_alldocs(z, dbname="criminalsdb")
doc_delete(z, dbname="criminalsdb", docid="weapons")
db_alldocs(z, dbname = "criminalsdb")
db_delete(z,dbname="student")
doc3<-'{"jailed":"yes"}'
doc_create(z,doc1,dbname = "criminalsdb",docid = "weapons")
doc_get(z, dbname = "criminalsdb", docid = "weapons")
revs <- db_revisions(z, dbname = "criminalsdb", docid = "weapons")
doc_update(z,dbname="criminalsdb",doc=doc3,docid="weapons",rev=revs[1])
db_revisions(z, dbname = "criminalsdb", docid = "weapons")
```

Output

```
> z$ping()
$couchdb
[1] "Welcome"

$version
[1] "3.3.0"

$git_sha
[1] "f6ddbe24c"

$uuid
[1] "ce8dcb65c759aa3797d54ebe03f6ffb7"

$features
$features[[1]]
[1] "access-ready"

$features[[2]]
[1] "partitioned"

$features[[3]]
[1] "pluggable-storage-engines"

$features[[4]]
[1] "reshard"

$features[[5]]
[1] "scheduler"


$vendor
$vendor$name
[1] "The Apache Software Foundation"
```

```
> db_list(z)
[1] "student"
> db_create(z,dbname="criminalsdb")
$ok
[1] TRUE

> db_alldocs(z, dbname="criminalsdb")
$total_rows
[1] 0

$offset
[1] 0

$rows
list()

> doc1 <-'{"name":"criminals","crime":"theft"}'
> doc_create(z,doc1,dbname = "criminalsdb",docid = "weapons")
$ok
[1] TRUE

$id
[1] "weapons"

$rev
[1] "1-dbcccf06a7265eadb0ad4b585252b659"

> doc2 <-'{"class":"regular","gang":"yes"}'
> doc_create(z,doc2,dbname = "criminalsdb")
$ok
[1] TRUE

$id
[1] "7a1a6b6b920eb3c47c52654b90000a1e"

$rev
[1] "1-1deff155f4245743daf518a51f354641"
```

PRACTICAL-2

AIM-Data Curation and Management using MongoDB and R.

Step 1: Install mongo db from https://www.mongodb.com/try/download/community
Step 2: Run it on local host

```
# installs development version of 'mongolite'
# devtools::install_github("jeroen/mongolite")
install.packages("mongolite")
# Init connection to local mongod
library(mongolite)
m <- mongo(collection = "diamonds")
# Insert test data
data(diamonds, package="ggplot2")
m$insert(diamonds)
# Check records
m$count()
nrow(diamonds)
# Perform a query and retrieve data
out <- m$find('{"cut" : "Premium", "price" : { "$lt" : 1000 } }')
# Compare
nrow(out)
nrow(subset(diamonds, cut == "Premium" & price < 1000))
# Cross-table
tbl <- m$mapreduce(
  map = "function(){emit({cut:this.cut, color:this.color}, 1)}",
  reduce = "function(id, counts){return Array.sum(counts)}")
# Same as:
data.frame(with(diamonds, table(cut, color)))
# Stream jsonlines into a connection
tmp <- tempfile()
m$export(file(tmp))

# Stream it back in R
library(jsonlite)
mydata <- stream_in(file(tmp))
# Or into mongo
m2 <- mongo("diamonds2")
m2$count()
m2$import(file(tmp))
m2$count()
# Remove the collection
m$drop()
m2$drop()
```

Output:

```
List of 5
 $ nInserted  : num 53940
 $ nMatched   : num 0
 $ nRemoved   : num 0
 $ nUpserted  : num 0
 $ writeErrors: list()
 [1] 107880
 [1] 53940
 [1] 6400
[1] 3200
 cut color Freq
1       Fair     D  163
2       Good     D  662
3  Very Good     D 1513
4    Premium     D 1603
5      Ideal     D 2834
6       Fair     E  224
7       Good     E  933
8  Very Good     E 2400
9    Premium     E 2337
10     Ideal     E 3903
11      Fair     F  312
12      Good     F  909
13 Very Good     F 2164
14   Premium     F 2331
15     Ideal     F 3826
16      Fair     G  314
17      Good     G  871
18 Very Good     G 2299
19   Premium     G 2924
20     Ideal     G 4884
21      Fair     H  303
22      Good     H  702
23 Very Good     H 1824
24   Premium     H 2360
25     Ideal     H 3115
26      Fair     I  175……
```

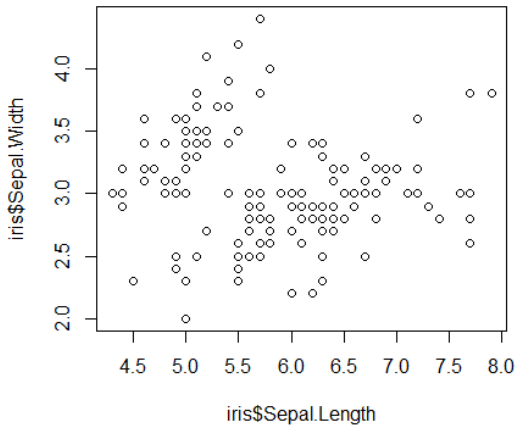
PRACTICAL-3

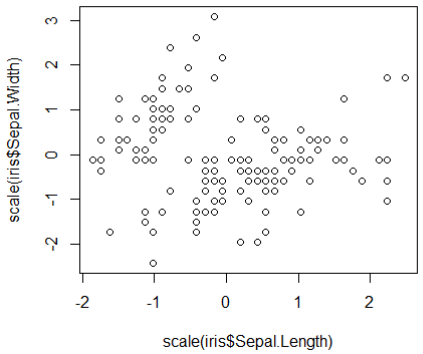
AIM- Practical of Principal Component Analysis.

```
data("iris")
head(iris)
summary(iris)
library()
"to find principal component"
mypr<-prcomp(iris[,-5],scale=T)
"to understand use of scale"
plot(iris$Sepal.Length,iris$Sepal.Width)
plot(scale(iris$Sepal.Length),scale(iris$Sepal.Width))
mypr
summary(mypr)
plot(mypr,type="l")
biplot(mypr,scale=0)
"extract pc scores"
str(mypr)
mypr$x
iris2<-cbind(iris,mypr$x[,1:2])
head(iris2)
cor(iris[,-5],iris2[,6:7])
install.packages("pls")
library(pls)
names(iris)
pcmodel<-
pcr(Sepal.Length~Species+Sepal.Width+Petal.Length+Petal.Width,ncomp=3,data=iris,scale=
T)
iris$pred<-predict(pcmodel,iris,ncomp = 2)
head(iris)
```

Output:

```
  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
  Sepal.Length    Sepal.Width     Petal.Length
 Min.   :4.300   Min.   :2.000   Min.   :1.000
 1st Qu.:5.100   1st Qu.:2.800   1st Qu.:1.600
 Median :5.800   Median :3.000   Median :4.350
 Mean   :5.843   Mean   :3.057   Mean   :3.758
 3rd Qu.:6.400   3rd Qu.:3.300   3rd Qu.:5.100
 Max.   :7.900   Max.   :4.400   Max.   :6.900
  Petal.Width          Species
 Min.   :0.100   setosa    :50
 1st Qu.:0.300   versicolor:50
 Median :1.300   virginica :50
 Mean   :1.199
 3rd Qu.:1.800
 Max.   :2.500
```

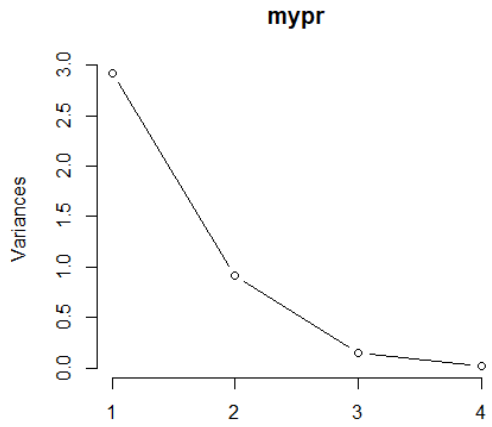


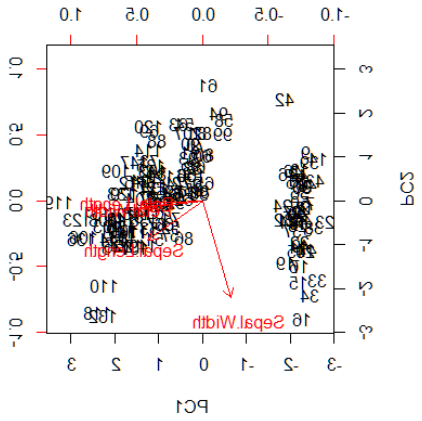



```
 Standard deviations (1, .., p=4):
[1] 1.7083611 0.9560494 0.3830886 0.1439265

Rotation (n x k) = (4 x 4):
                     PC1         PC2         PC3         PC4
```

```
Sepal.Length  0.5210659 -0.37741762  0.7195664  0.2612863
Sepal.Width  -0.2693474 -0.92329566 -0.2443818 -0.1235096
Petal.Length  0.5804131 -0.02449161 -0.1421264 -0.8014492
Petal.Width   0.5648565 -0.06694199 -0.6342727  0.5235971
> summary(mypr)
Importance of components:
                          PC1    PC2     PC3     PC4
Standard deviation     1.7084 0.9560 0.38309 0.14393
Proportion of Variance 0.7296 0.2285 0.03669 0.00518
Cumulative Proportion  0.7296 0.9581 0.99482 1.00000
> plot(mypr,type="l")
```






```
List of 5
 $ sdev    : num [1:4] 1.708 0.956 0.383 0.144
 $ rotation: num [1:4, 1:4] 0.521 -0.269 0.58 0.565 -0.377 ...
  ..- attr(*, "dimnames")=List of 2
  .. ..$ : chr [1:4] "Sepal.Length" "Sepal.Width" "Petal.Length"
"Petal.Width"
  .. ..$ : chr [1:4] "PC1" "PC2" "PC3" "PC4"
 $ center  : Named num [1:4] 5.84 3.06 3.76 1.2
  ..- attr(*, "names")= chr [1:4] "Sepal.Length" "Sepal.Width" "
Petal.Length" "Petal.Width"
 $ scale   : Named num [1:4] 0.828 0.436 1.765 0.762
  ..- attr(*, "names")= chr [1:4] "Sepal.Length" "Sepal.Width" "
Petal.Length" "Petal.Width"
 $ x       : num [1:150, 1:4] -2.26 -2.07 -2.36 -2.29 -2.38 ...
  ..- attr(*, "dimnames")=List of 2
  .. ..$ : NULL
```

```
  .. ..$ : chr [1:4] "PC1" "PC2" "PC3" "PC4"
 - attr(*, "class")= chr "pr
              PC1          PC2          PC3          PC4
  [1,] -2.25714118 -0.478423832  0.127279624  0.024087508
  [2,] -2.07401302  0.671882687  0.233825517  0.102662845
  [3,] -2.35633511  0.340766425 -0.044053900  0.028282305
  [4,] -2.29170679  0.595399863 -0.090985297 -0.065735340
  [5,] -2.38186270 -0.644675659 -0.015685647 -0.035802870
  [6,] -2.06870061 -1.484205297 -0.026878250  0.006586116
  [7,] -2.43586845 -0.047485118 -0.334350297 -0.036652767
  [8,] -2.22539189 -0.222403002  0.088399352 -0.024529919
  [9,] -2.32684533  1.111603700 -0.144592465 -0.026769540
 [10,] -2.17703491  0.467447569  0.252918268 -0.039766068
 [11,] -2.15907699 -1.040205867  0.267784001  0.016675503
 [12,] -2.31836413 -0.132633999 -0.093446191 -0.133037725
 [13,] -2.21104370  0.726243183  0.230140246  0.002416941
 [14,] -2.62430902  0.958296347 -0.180192423 -0.019151375
 [15,] -2.19139921 -1.853846555  0.471322025  0.194081578
 [16,] -2.25466121 -2.677315230 -0.030424684  0.050365010
 [17,] -2.20021676 -1.478655729  0.005326251  0.188186988
 [18,] -2.18303613 -0.487206131  0.044067686  0.092779618
 [19,] -1.89223284 -1.400327567  0.373093377  0.060891973
 [20,] -2.33554476 -1.124083597 -0.132187626 -0.037630354
 [21,] -1.90793125 -0.407490576  0.419885937  0.010884821
 [22,] -2.19964383 -0.921035871 -0.159331502  0.059398340
 [23,] -2.76508142 -0.456813301 -0.331069982  0.019582826
 [24,] -1.81259716 -0.085272854 -0.034373442  0.150636353
 [25,] -2.21972701 -0.136796175 -0.117599566 -0.269238379
 [26,] -1.94532930  0.623529705  0.304620475  0.043416203
 [27,] -2.04430277 -0.241354991 -0.086075649  0.067454082
 [28,] -2.16133650 -0.525389422  0.206125707  0.010241084
 [29,] -2.13241965 -0.312172005  0.270244895  0.083977887
 [30,] -2.25769799  0.336604248 -0.068207276 -0.107918349
 [31,] -2.13297647  0.502856075  0.074757996 -0.048027970
 [32,] -1.82547925 -0.422280389  0.269564311  0.239069476
 [33,] -2.60621687 -1.787587272 -0.047070727 -0.228470534
 [34,] -2.43800983 -2.143546796  0.082392024 -0.048053409
 [35,] -2.10292986  0.458665270  0.169706329  0.028926042
 [36,] -2.20043723  0.205419224  0.224688852  0.168343905
 [37,] -2.03831765 -0.659349230  0.482919584  0.195702902
 [38,] -2.51889339 -0.590315163 -0.019370918 -0.136048774
 [39,] -2.42152026  0.901161067 -0.192609402 -0.009705907
 [40,] -2.16246625 -0.267981199  0.175296561  0.007023875
 [41,] -2.27884081 -0.440240541 -0.034778398  0.106626042
 [42,] -1.85191836  2.329610745  0.203552303  0.288896090
 [43,] -2.54511203  0.477501017 -0.304745527 -0.066379077
 [44,] -1.95788857 -0.470749613 -0.308567588  0.176501717
 [45,] -2.12992356 -1.138415464 -0.247604064 -0.150539117
 [46,] -2.06283361  0.708678586  0.063716370  0.139801160
 [47,] -2.37677076 -1.116688691 -0.057026813 -0.151722682
 [48,] -2.38638171  0.384957230 -0.139002234 -0.048671707
 [49,] -2.22200263 -0.994627669  0.180886792 -0.014878291
 [50,] -2.19647504 -0.009185585  0.152518539  0.049206884
   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
        PC1        PC2
1 -2.257141 -0.4784238
2 -2.074013  0.6718827
3 -2.356335  0.3407664
4 -2.291707  0.5953999
5 -2.381863 -0.6446757
6 -2.068701 -1.4842053
                    PC1         PC2
Sepal.Length  0.8901688 -0.36082989
```

```
Sepal.Width  -0.4601427 -0.88271627
Petal.Length  0.9915552 -0.02341519
Petal.Width   0.9649790 -0.06399985
package ‘pls’ successfully unpacked and MD5 sums checked

The downloaded binary packages are in
        C:\Users\Administrator\AppData\Local\Temp\RtmpgZyY4l\down
loaded_packages
 [1] "Sepal.Length" "Sepal.Width"  "Petal.Length"
[4] "Petal.Width"  "Species"
  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
      pred
1 5.025168
2 5.125999
3 5.073053
4 5.118447
5 5.005002
6 5.041960
```

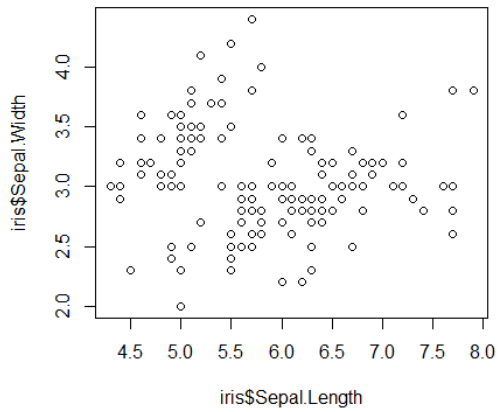
PRACTICAL-4

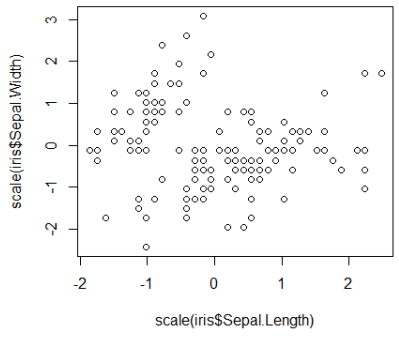
AIM- Practical of Clustering.

```
data("iris")
head(iris)
summary(iris)
library()
"to find principal component"
mypr<-prcomp(iris[,-5],scale=T)
"to understand use of scale"
plot(iris$Sepal.Length,iris$Sepal.Width)
plot(scale(iris$Sepal.Length),scale(iris$Sepal.Width))
mypr
summary(mypr)
plot(mypr,type="l")
biplot(mypr,scale=0)
"extract pc scores"
str(mypr)
mypr$x
iris2<-cbind(iris,mypr$x[,1:2])
head(iris2)
cor(iris[,-5],iris2[,6:7])
install.packages("pls")
library(pls)
names(iris)
pcmodel<-
pcr(Sepal.Length~Species+Sepal.Width+Petal.Length+Petal.Width,ncomp=3,data=iris,scale=
T)
iris$pred<-predict(pcmodel,iris,ncomp = 2)
head(iris)
```

Output:

```
  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
  Sepal.Length    Sepal.Width     Petal.Length
 Min.   :4.300   Min.   :2.000   Min.   :1.000
 1st Qu.:5.100   1st Qu.:2.800   1st Qu.:1.600
 Median :5.800   Median :3.000   Median :4.350
 Mean   :5.843   Mean   :3.057   Mean   :3.758
 3rd Qu.:6.400   3rd Qu.:3.300   3rd Qu.:5.100
 Max.   :7.900   Max.   :4.400   Max.   :6.900
  Petal.Width          Species
 Min.   :0.100   setosa    :50
 1st Qu.:0.300   versicolor:50
 Median :1.300   virginica :50
 Mean   :1.199
 3rd Qu.:1.800
 Max.   :2.500
```

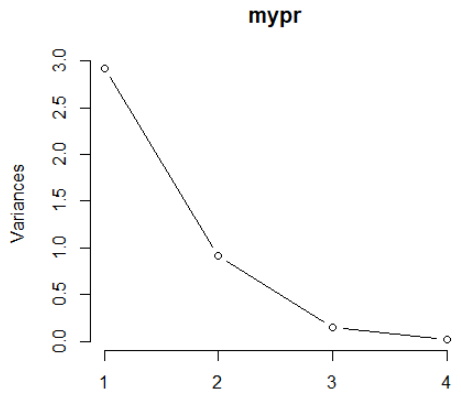


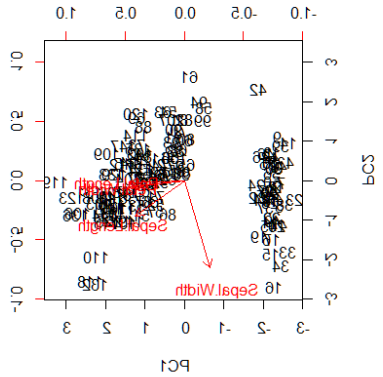



```
 Standard deviations (1, .., p=4):
[1] 1.7083611 0.9560494 0.3830886 0.1439265

Rotation (n x k) = (4 x 4):
                    PC1         PC2        PC3        PC4
Sepal.Length  0.5210659 -0.37741762  0.7195664  0.2612863
Sepal.Width  -0.2693474 -0.92329566 -0.2443818 -0.1235096
```

```
Petal.Length  0.5804131 -0.02449161 -0.1421264 -0.8014492
Petal.Width   0.5648565 -0.06694199 -0.6342727  0.5235971
> summary(mypr)
Importance of components:
                          PC1    PC2     PC3     PC4
Standard deviation     1.7084 0.9560 0.38309 0.14393
Proportion of Variance 0.7296 0.2285 0.03669 0.00518
Cumulative Proportion  0.7296 0.9581 0.99482 1.00000
> plot(mypr,type="l")
```






```
List of 5
 $ sdev     : num [1:4] 1.708 0.956 0.383 0.144
 $ rotation: num [1:4, 1:4] 0.521 -0.269 0.58 0.565 -0.377 ...
  ..- attr(*, "dimnames")=List of 2
  .. ..$ : chr [1:4] "Sepal.Length" "Sepal.Width" "Petal.Length"
"Petal.Width"
  .. ..$ : chr [1:4] "PC1" "PC2" "PC3" "PC4"
 $ center   : Named num [1:4] 5.84 3.06 3.76 1.2
  ..- attr(*, "names")= chr [1:4] "Sepal.Length" "Sepal.Width" "
Petal.Length" "Petal.Width"
 $ scale    : Named num [1:4] 0.828 0.436 1.765 0.762
  ..- attr(*, "names")= chr [1:4] "Sepal.Length" "Sepal.Width" "
Petal.Length" "Petal.Width"
 $ x        : num [1:150, 1:4] -2.26 -2.07 -2.36 -2.29 -2.38 ...
  ..- attr(*, "dimnames")=List of 2
  .. ..$ : NULL
  .. ..$ : chr [1:4] "PC1" "PC2" "PC3" "PC4"
 - attr(*, "class")= chr "pr
                PC1         PC2          PC3          PC4
  [1,] -2.25714118 -0.478423832  0.127279624  0.024087508
  [2,] -2.07401302  0.671882687  0.233825517  0.102662845
```

```
  [3,] -2.35633511  0.340766425 -0.044053900  0.028282305
  [4,] -2.29170679  0.595399863 -0.090985297 -0.065735340
  [5,] -2.38186270 -0.644675659 -0.015685647 -0.035802870
  [6,] -2.06870061 -1.484205297 -0.026878250  0.006586116
  [7,] -2.43586845 -0.047485118 -0.334350297 -0.036652767
  [8,] -2.22539189 -0.222403002  0.088399352 -0.024529919
  [9,] -2.32684533  1.111603700 -0.144592465 -0.026769540
 [10,] -2.17703491  0.467447569  0.252918268 -0.039766068
 [11,] -2.15907699 -1.040205867  0.267784001  0.016675503
 [12,] -2.31836413 -0.132633999 -0.093446191 -0.133037725
 [13,] -2.21104370  0.726243183  0.230140246  0.002416941
 [14,] -2.62430902  0.958296347 -0.180192423 -0.019151375
 [15,] -2.19139921 -1.853846555  0.471322025  0.194081578
 [16,] -2.25466121 -2.677315230 -0.030424684  0.050365010
 [17,] -2.20021676 -1.478655729  0.005326251  0.188186988
 [18,] -2.18303613 -0.487206131  0.044067686  0.092779618
 [19,] -1.89223284 -1.400327567  0.373093377  0.060891973
 [20,] -2.33554476 -1.124083597 -0.132187626 -0.037630354
 [21,] -1.90793125 -0.407490576  0.419885937  0.010884821
 [22,] -2.19964383 -0.921035871 -0.159331502  0.059398340
 [23,] -2.76508142 -0.456813301 -0.331069982  0.019582826
 [24,] -1.81259716 -0.085272854 -0.034373442  0.150636353
 [25,] -2.21972701 -0.136796175 -0.117599566 -0.269238379
 [26,] -1.94532930  0.623529705  0.304620475  0.043416203
 [27,] -2.04430277 -0.241354991 -0.086075649  0.067454082
 [28,] -2.16133650 -0.525389422  0.206125707  0.010241084
 [29,] -2.13241965 -0.312172005  0.270244895  0.083977887
 [30,] -2.25769799  0.336604248 -0.068207276 -0.107918349
 [31,] -2.13297647  0.502856075  0.074757996 -0.048027970
 [32,] -1.82547925 -0.422280389  0.269564311  0.239069476
 [33,] -2.60621687 -1.787587272 -0.047070727 -0.228470534
 [34,] -2.43800983 -2.143546796  0.082392024 -0.048053409
 [35,] -2.10292986  0.458665270  0.169706329  0.028926042
 [36,] -2.20043723  0.205419224  0.224688852  0.168343905
 [37,] -2.03831765 -0.659349230  0.482919584  0.195702902
 [38,] -2.51889339 -0.590315163 -0.019370918 -0.136048774
 [39,] -2.42152026  0.901161067 -0.192609402 -0.009705907
 [40,] -2.16246625 -0.267981199  0.175296561  0.007023875
 [41,] -2.27884081 -0.440240541 -0.034778398  0.106626042
 [42,] -1.85191836  2.329610745  0.203552303  0.288896090
 [43,] -2.54511203  0.477501017 -0.304745527 -0.066379077
 [44,] -1.95788857 -0.470749613 -0.308567588  0.176501717
 [45,] -2.12992356 -1.138415464 -0.247604064 -0.150539117
 [46,] -2.06283361  0.708678586  0.063716370  0.139801160
 [47,] -2.37677076 -1.116688691 -0.057026813 -0.151722682
 [48,] -2.38638171  0.384957230 -0.139002234 -0.048671707
 [49,] -2.22200263 -0.994627669  0.180886792 -0.014878291
 [50,] -2.19647504 -0.009185585  0.152518539  0.049206884
   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
        PC1        PC2
1 -2.257141 -0.4784238
2 -2.074013  0.6718827
3 -2.356335  0.3407664
4 -2.291707  0.5953999
5 -2.381863 -0.6446757
6 -2.068701 -1.4842053
                    PC1         PC2
Sepal.Length  0.8901688 -0.36082989
Sepal.Width  -0.4601427 -0.88271627
Petal.Length  0.9915552 -0.02341519
Petal.Width   0.9649790 -0.06399985
package ‘pls’ successfully unpacked and MD5 sums checked
```

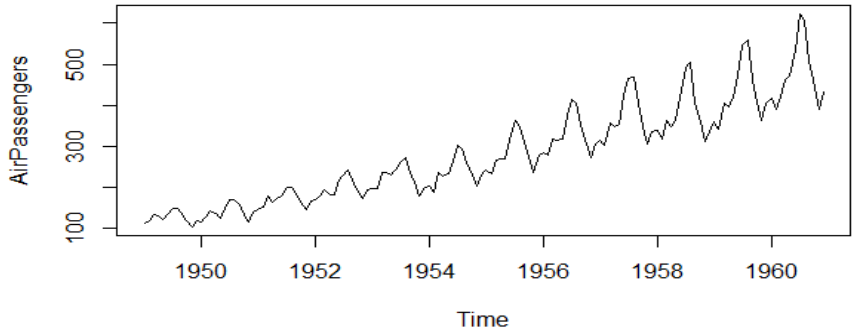
```
The downloaded binary packages are in
        C:\Users\Administrator\AppData\Local\Temp\RtmpgZyY4l\down
loaded_packages
 [1] "Sepal.Length" "Sepal.Width"  "Petal.Length"
[4] "Petal.Width"  "Species"
  Sepal.Length Sepal.Width Petal.Length Petal.Width Species
1          5.1         3.5          1.4         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
      pred
1 5.025168
2 5.125999
3 5.073053
4 5.118447
5 5.005002
6 5.041960
```

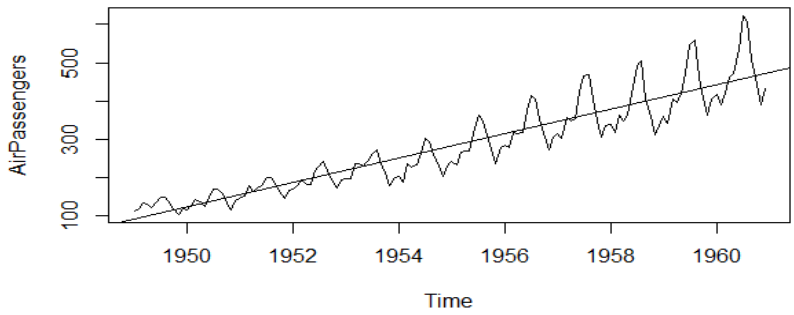
PRACTICAL-5

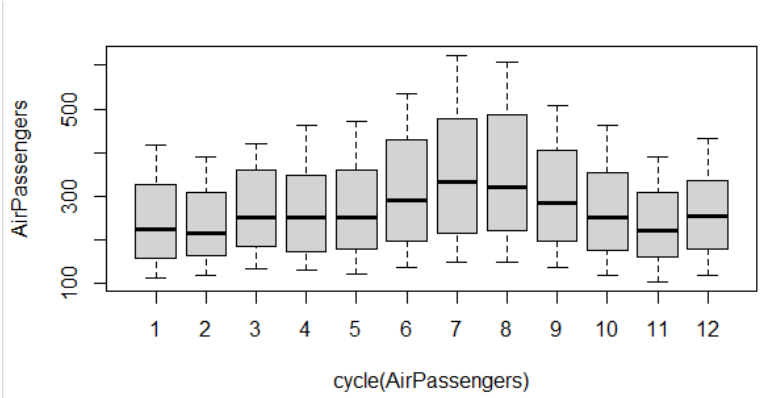
AIM- Practical of Time Series Forecasting

```
#consider the inbuilt data set Air Passengers
data("AirPassengers")
#to know the format of data set here ts will tell that the
#data set belongs to time series format
class(AirPassengers)
#to know the start of time series
start(AirPassengers)
#to know the end of time series
end(AirPassengers)
#to know the frequency of the data set here 12 means that
#the time series is on monthly basis
frequency(AirPassengers)
#to know the mean, median etc of the dataset
summary(AirPassengers)
#to plot the time series model
plot(AirPassengers)
#to plot the best fit line which can be used for regression
abline(reg=lm(AirPassengers~time(AirPassengers)))
#to plot the cycle across years
cycle(AirPassengers)
#to aggregate the cycle and display its trend per year
plot(aggregate(AirPassengers,FUN=mean))
#to get the box plot
boxplot(AirPassengers~cycle(AirPassengers))
```

Output:


AirPassengers
100
300
500
1950
1952
1954
1956
1958
1960
Time


AirPassengers
100
300
500
1950
1952
1954
1956
1958
1960
Time


AirPassengers
100
300
500
1
2
3
4
5
6
7
8
9
10
11
12
cycle(AirPassengers)

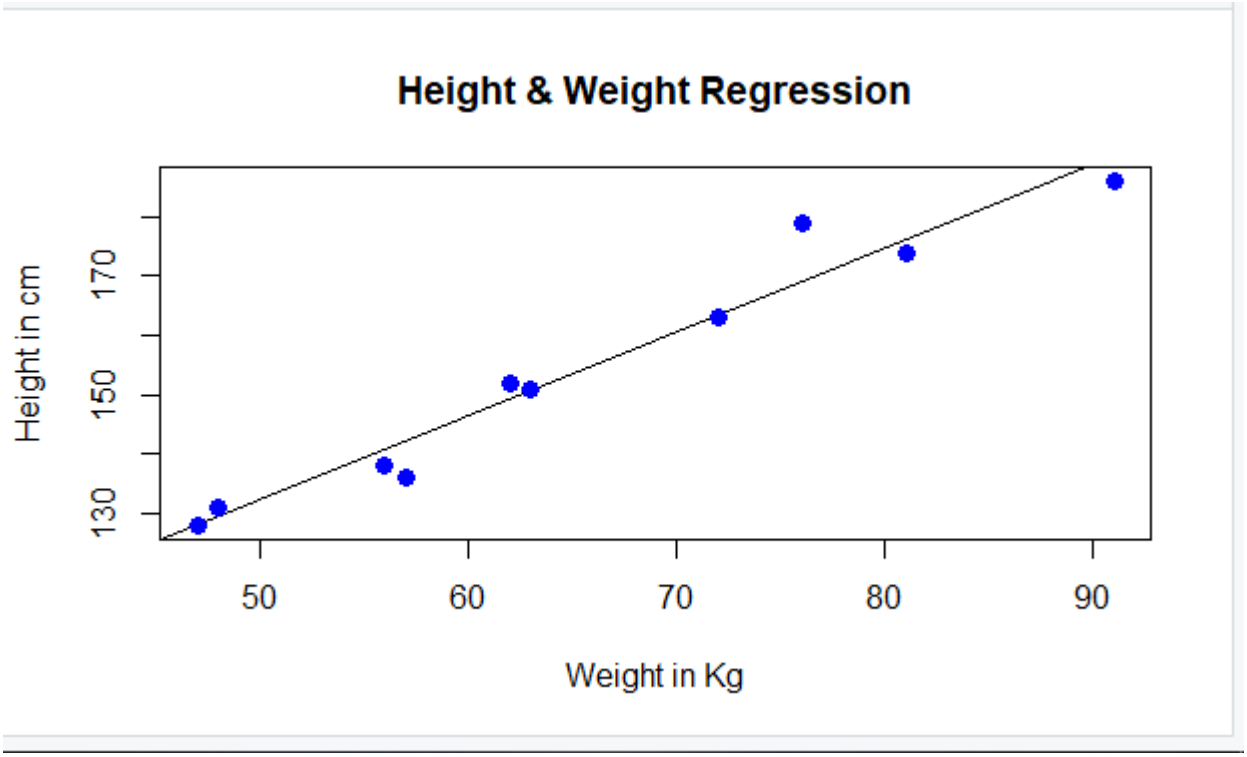
PRACTICAL-6(A)

AIM-Practical of Simple Regression with data values

```
x <- c(151, 174, 138, 186, 128, 136, 179, 163, 152, 131)
y <- c(63, 81, 56, 91, 47, 57, 76, 72, 62, 48)
relation <- lm(y~x)
a <- data.frame(x = 170)
result <-  predict(relation,a)
print(result)
# Give the chart file a name.
#png(file = "linearregression.png")
library()
# Plot the chart.
plot(y,x,col = "blue",main = "Height & Weight Regression",
    abline(lm(x~y)),cex = 1.3,pch = 16,xlab = "Weight in Kg",ylab = "Height in cm")
```

Output:

```
       1
76.22869
```



PRACTICAL-6(B)

AIM-Practical of Multiple Regression with data values

```
input <- mtcars[,c("mpg","disp","hp","wt")]
print(head(input))
model <- lm(mpg~disp+hp+wt, data = input)

# Show the model.
print(model)

# Get the Intercept and coefficients as vector elements.
cat("# # # # The Coefficient Values # # # ","\n")

a <- coef(model)[1]
print(a)

Xdisp <- coef(model)[2]
Xhp <- coef(model)[3]
Xwt <- coef(model)[4]

print(Xdisp)
print(Xhp)
print(Xwt)
```

Output:

```
Mazda RX4          21.0  160 110 2.620
Mazda RX4 Wag      21.0  160 110 2.875
Datsun 710         22.8  108  93 2.320
Hornet 4 Drive     21.4  258 110 3.215
Hornet Sportabout 18.7  360 175 3.440
Valiant            18.1  225 105 3.460

Call:
lm(formula = mpg ~ disp + hp + wt, data = input)

Coefficients:
(Intercept)         disp           hp           wt
  37.105505    -0.000937    -0.031157    -3.800891
(Intercept)
   37.10551

-0.0009370091
    hp
-0.03115655

  wt
-3.800891
```

PRACTICAL-6(C)

AIM-Practical of Simple Regression with data set

```
# install usingR and ggplot2 packages; packages already installed; loading them using
library()
library(UsingR)
# Require ggplot2 and UsingR
require(UsingR)
require(ggplot2)
# The first 10 observation of our dataset using the print(head(data, n = 10)) function
print(head(father.son, n = 10))
print(tail(father.son, n = 10))
str(father.son)
summary(father.son)
# Histogram of father's height distribution
ggplot(data = father.son, mapping = aes(x = fheight)) +
  geom_histogram(bins = 30, fill = "seagreen") +
  ggtitle("Histogram of Father's Height") +
  theme(plot.title = element_text(hjust = 0.5))
# Calculate Linear regression using lm() function
(height.lm <- lm(sheight ~ fheight, data = father.son))
# Complete regression results using summary() function
(summary(height.lm))
```

Output:

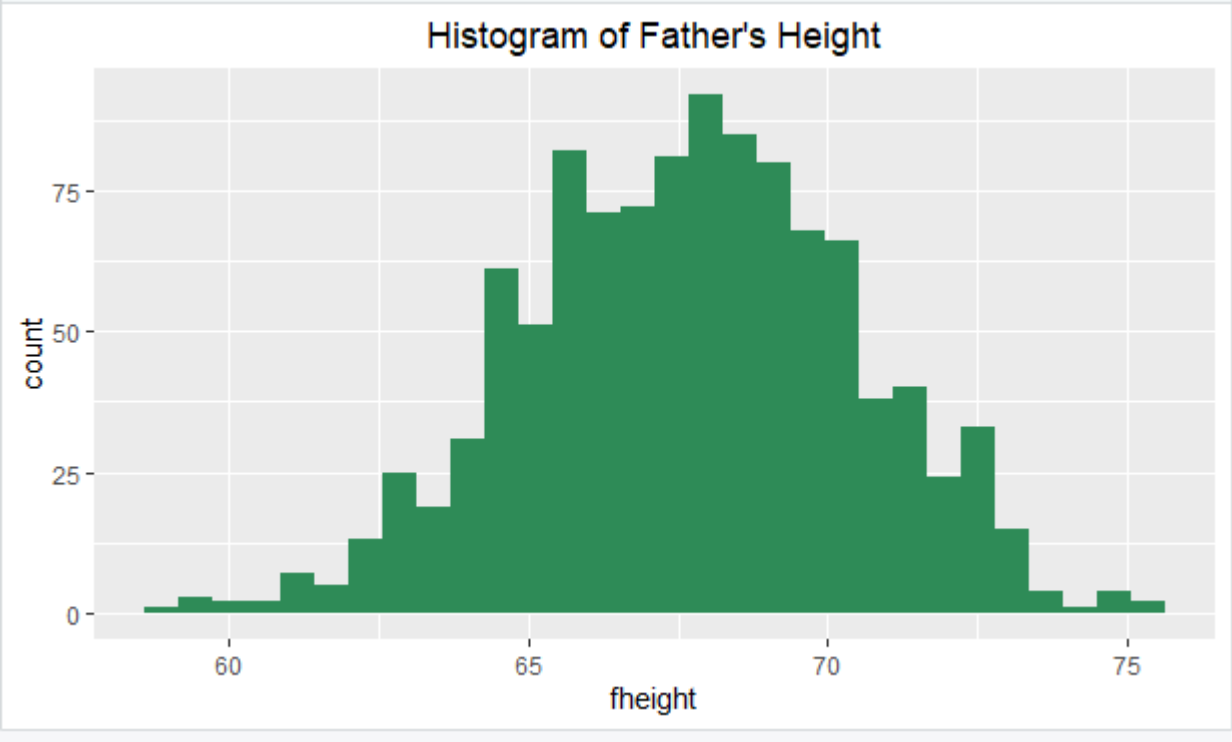
```
fheight  sheight
1  65.04851 59.77827
2  63.25094 63.21404
3  64.95532 63.34242
4  65.75250 62.79238
5  61.13723 64.28113
6  63.02254 64.24221
7  65.37053 64.08231
8  64.72398 63.99574
9  66.06509 64.61338
10 66.96738 63.97944

   fheight  sheight
1069 72.15051 66.72684
1070 63.22006 58.79456
1071 73.26450 67.89277
1072 65.81296 61.04946
1073 67.70657 59.81693
1074 66.99681 70.75232
1075 71.33181 68.26774
1076 71.78314 69.30589
1077 70.73837 69.30199
1078 70.30609 67.01500


summary(father.son)
    fheight         sheight
 Min.   :59.01   Min.   :58.51
 1st Qu.:65.79   1st Qu.:66.93
 Median :67.77   Median :68.62
 Mean   :67.69   Mean   :68.68
```

```
 3rd Qu.:69.60   3rd Qu.:70.47
 Max.   :75.43   Max.   :78.36

> # Histogram of father's height distribution
> ggplot(data = father.son, mapping = aes(x = fheight)) +
+   geom_histogram(bins = 30, fill = "seagreen") +
+   ggtitle("Histogram of Father's Height") +
+   theme(plot.title = element_text(hjust = 0.5))
```




```
> (height.lm <- lm(sheight ~ fheight, data = father.son))

Call:
lm(formula = sheight ~ fheight, data = father.son)

Coefficients:
(Intercept)      fheight
    33.8866       0.5141


> (summary(height.lm))

Call:
lm(formula = sheight ~ fheight, data = father.son)

Residuals:
    Min      1Q  Median      3Q     Max
-8.8772 -1.5144 -0.0079  1.6285  8.9685

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) 33.88660    1.83235   18.49   <2e-16 ***
fheight      0.51409    0.02705   19.01   <2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 2.437 on 1076 degrees of freedom
Multiple R-squared:  0.2513,   Adjusted R-squared:  0.2506
F-statistic: 361.2 on 1 and 1076 DF,  p-value: < 2.2e-16
```

PRACTICAL-7

AIM-Practical of Logistic Regression

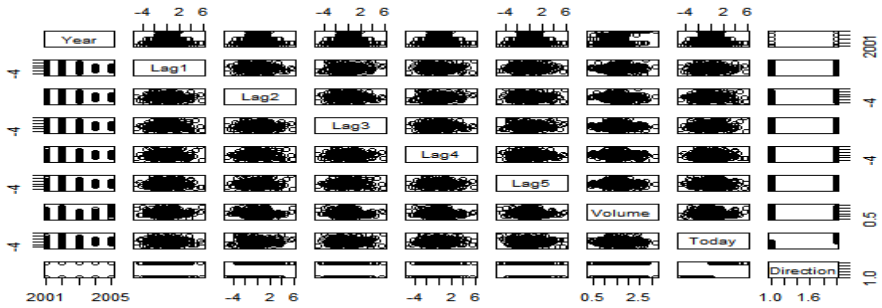
```
rm(list=ls())
library(ISLR)
names(Smarket)
dim(Smarket)
summary(Smarket)
pairs(Smarket)
?Smarket
cor(Smarket[,-9])
attach(Smarket)
par(mfrow=c(1,1))
plot(Volume)
glm.fits=glm(Direction~Lag1+Lag2+Lag3+Lag4+Lag5+Volume,data=Smarket,family=bino
mial)
summary(glm.fits)
coef(glm.fits)
summary(glm.fits)$coef
summary(glm.fits)$coef[,4]
glm.probs=predict(glm.fits,type="response")
glm.probs[1:10]
contrasts(Direction)
glm.pred=rep("Down",1250)
glm.pred[glm.probs>.5]="Up"
glm.probs[1:10]
glm.pred[1:10]
table(glm.pred,Direction)
(507+145)/1250
mean(glm.pred==Direction)
train=(Year<2005)
Smarket.2005=Smarket[!train,]
dim(Smarket.2005)
Direction.2005=Direction[!train]
glm.fits=glm(Direction~Lag1+Lag2+Lag3+Lag4+Lag5+Volume,data=Smarket,family=bino
mial,subset=train)
summary(glm.fits)
glm.probs=predict(glm.fits,Smarket.2005,type="response")
glm.pred=rep("Down",252)
glm.pred[glm.probs>.5]="Up"
table(glm.pred,Direction.2005)
mean(glm.pred==Direction.2005)
mean(glm.pred!=Direction.2005)
glm.fits=glm(Direction~Lag1+Lag2,data=Smarket,family=binomial,subset=train)
glm.probs=predict(glm.fits,Smarket.2005,type="response")
glm.pred=rep("Down",252)
```

```
glm.pred[glm.probs>.5]="Up"
table(glm.pred,Direction.2005)
mean(glm.pred==Direction.2005)
(106+35)/252
106/(106+35)
76/(36+76)
```

Output:

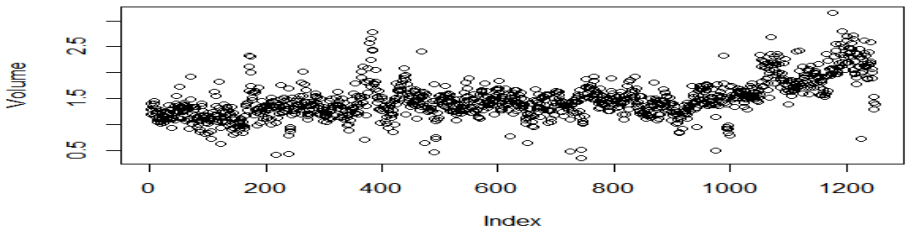
```
[1] "Year"      "Lag1"      "Lag2"      "Lag3"      "Lag4"      "Lag5"
[7] "Volume"    "Today"     "Direction"
 [1] 1250    9

      Year           Lag1                Lag2                Lag3
 Min.   :2001   Min.   :-4.922000   Min.   :-4.922000   Min.   :-4.922000
 1st Qu.:2002   1st Qu.:-0.639500   1st Qu.:-0.639500   1st Qu.:-0.640000
 Median :2003   Median : 0.039000   Median : 0.039000   Median : 0.038500
 Mean   :2003   Mean   : 0.003834   Mean   : 0.003919   Mean   : 0.001716
 3rd Qu.:2004   3rd Qu.: 0.596750   3rd Qu.: 0.596750   3rd Qu.: 0.596750
 Max.   :2005   Max.   : 5.733000   Max.   : 5.733000   Max.   : 5.733000
      Lag4                Lag5              Volume           Today
 Min.   :-4.922000   Min.   :-4.92200   Min.   :0.3561   Min.   :-4.922000
 1st Qu.:-0.640000   1st Qu.:-0.64000   1st Qu.:1.2574   1st Qu.:-0.639500
 Median : 0.038500   Median : 0.03850   Median :1.4229   Median : 0.038500
 Mean   : 0.001636   Mean   : 0.00561   Mean   :1.4783   Mean   : 0.003138
 3rd Qu.: 0.596750   3rd Qu.: 0.59700   3rd Qu.:1.6417   3rd Qu.: 0.596750
 Max.   : 5.733000   Max.   : 5.73300   Max.   :3.1525   Max.   : 5.733000
 Direction
 Down:602
 Up  :648
```




```
              Year         Lag1         Lag2         Lag3         Lag4
Lag5
Year   1.00000000  0.029699649  0.030596422  0.033194581  0.035688718  0.0
29787995
Lag1   0.02969965  1.000000000 -0.026294328 -0.010803402 -0.002985911 -0.0
05674606
Lag2   0.03059642 -0.026294328  1.000000000 -0.025896670 -0.010853533 -0.0
03557949
Lag3   0.03319458 -0.010803402 -0.025896670  1.000000000 -0.024051036 -0.0
18808338
Lag4   0.03568872 -0.002985911 -0.010853533 -0.024051036  1.000000000 -0.0
27083641
Lag5   0.02978799 -0.005674606 -0.003557949 -0.018808338 -0.027083641  1.0
00000000
Volume 0.53900647  0.040909908 -0.043383215 -0.041823686 -0.048414246 -0.0
22002315
Today  0.03009523 -0.026155045 -0.010250033 -0.002447647 -0.006899527 -0.0
34860083
             Volume        Today
Year    0.53900647  0.030095229
Lag1    0.04090991 -0.026155045
Lag2   -0.04338321 -0.010250033
Lag3   -0.04182369 -0.002447647
```

```
Lag4   -0.04841425 -0.006899527
Lag5   -0.02200231 -0.034860083
Volume  1.00000000  0.014591823
Today   0.01459182  1.000000000
```




```
Call:
glm(formula = Direction ~ Lag1 + Lag2 + Lag3 + Lag4 + Lag5 +
    Volume, family = binomial, data = Smarket)
Deviance Residuals:
   Min      1Q  Median      3Q     Max
-1.446  -1.203   1.065   1.145   1.326
Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept) -0.126000   0.240736  -0.523    0.601
Lag1        -0.073074   0.050167  -1.457    0.145
Lag2        -0.042301   0.050086  -0.845    0.398
Lag3         0.011085   0.049939   0.222    0.824
Lag4         0.009359   0.049974   0.187    0.851
Lag5         0.010313   0.049511   0.208    0.835
Volume       0.135441   0.158360   0.855    0.392
(Dispersion parameter for binomial family taken to be 1)
    Null deviance: 1731.2  on 1249  degrees of freedom
Residual deviance: 1727.6  on 1243  degrees of freedom
AIC: 1741.6
Number of Fisher Scoring iterations: 3
 (Intercept)         Lag1         Lag2         Lag3         Lag4         L
ag5
-0.126000257 -0.073073746 -0.042301344  0.011085108  0.009358938  0.010313
068
      Volume
 0.135440659
                Estimate Std. Error    z value  Pr(>|z|)
 (Intercept)        Lag1        Lag2        Lag3        Lag4        Lag5
Volume
  0.6006983   0.1452272   0.3983491   0.8243333   0.8514445   0.8349974
0.3924004
        1         2         3         4         5         6         7
8
0.5070841 0.4814679 0.4811388 0.5152224 0.5107812 0.5069565 0.4926509 0.50
92292
        9        10
0.5176135 0.4888378
     Up
Down  0
Up    1
        1         2         3         4         5         6         7
8
0.5070841 0.4814679 0.4811388 0.5152224 0.5107812 0.5069565 0.4926509 0.50
92292
        9        10
0.5176135 0.4888378
        1         2         3         4         5         6         7
8
0.5070841 0.4814679 0.4811388 0.5152224 0.5107812 0.5069565 0.4926509 0.50
92292
        9        10
[1] "Up"   "Down" "Down" "Up"   "Up"   "Up"   "Down" "Up"   "Up"   "Down"
        Direction    Down  145 141
    Up    457 507
```

```
 [1] 0.5216
 [1] 252
Call:
glm(formula = Direction ~ Lag1 + Lag2 + Lag3 + Lag4 + Lag5 +
    Volume, family = binomial, data = Smarket, subset = train)
Deviance Residuals:
   Min      1Q  Median      3Q     Max
-1.302  -1.190   1.079   1.160   1.350
Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept)  0.191213   0.333690   0.573    0.567
Lag1        -0.054178   0.051785  -1.046    0.295
Lag2        -0.045805   0.051797  -0.884    0.377
Lag3         0.007200   0.051644   0.139    0.889
Lag4         0.006441   0.051706   0.125    0.901
Lag5        -0.004223   0.051138  -0.083    0.934
Volume      -0.116257   0.239618  -0.485    0.628
AIC: 1395.1
Number of Fisher Scoring iterations: 3
        Direction.2005
glm.pred Down Up
    Down   77 97
    Up     34 44
 [1] 0.5198413
        Direction.2005
glm.pred Down  Up
    Down   35  35
    Up     76 106
 [1] 0.5595238
 [1] 0.5595238
 [1] 0.751773
 [1] 0.678571
```

PRACTICAL-8

AIM-Practical of Hypothesis Testing

```
dataf<-seq(1,20,by=1)
dataf
mean(dataf)
sd(dataf)
a<-t.test(dataf,alternative = "two.sided",mu=10,conf.int=0.95)
a
a$p.value
a$statistic
(10.5-10)/(sd(dataf)/sqrt(length(dataf)))
length(dataf)=1
length(dataf)
dataf
dataf<-seq(1,20,by=1)
length(dataf)-1
```

Output:

```
[1]  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20
 [1] 10.5
 [1] 5.91608
One Sample t-test
data:  dataf
t = 0.37796, df = 19, p-value = 0.7096
alternative hypothesis: true mean is not equal to 10
95 percent confidence interval:
  7.731189 13.268811
sample estimates:
mean of x
     10.5
 [1] 0.7096465
        t
0.3779645
 [1] 0.3779645
 [1] 1
 [1] 1
 [1] 19
```

PRACTICAL-9

AIM-Practical of Analysis of Variance

```
ftest<-read.csv(file.choose(),sep=",",header=T)
var.test(ftest$density,ftest$block,alternative = "two.sided")
"one way anova"
data1<-read.csv(file.choose(),sep = ",",header = T)
names(data1)
summary(data1)
head(data1)
one.way <- aov(yield ~ fertilizer, data = data1)
summary(one.way)
"two way anova"
data2<-read.csv(file.choose(),sep=",",header = T)
names(data2)
summary(data2)
two.way <- aov(yield ~ fertilizer + density, data = data2)
summary(two.way)
```

Output:

```
> ftest<-read.csv(file.choose(),sep=",",header=T)
> var.test(ftest$density,ftest$block,alternative = "two.sided")

        F test to compare two variances

data:  ftest$density and ftest$block
F = 0.2, num df = 95, denom df = 95, p-value = 9.06e-14
alternative hypothesis: true ratio of variances is not equal to 1
95 percent confidence interval:
 0.1334488 0.2997404
sample estimates:
ratio of variances
               0.2

> "one way anova"
[1] "one way anova"
> data1<-read.csv(file.choose(),sep = ",",header = T)
> names(data1)
[1] "density"    "block"      "fertilizer" "yield"
> summary(data1)
    density        block        fertilizer     yield
 Min.   :1.0   Min.   :1.00   Min.   :1    Min.   :175.4
 1st Qu.:1.0   1st Qu.:1.75   1st Qu.:1    1st Qu.:176.5
 Median :1.5   Median :2.50   Median :2    Median :177.1
 Mean   :1.5   Mean   :2.50   Mean   :2    Mean   :177.0
 3rd Qu.:2.0   3rd Qu.:3.25   3rd Qu.:3    3rd Qu.:177.4
 Max.   :2.0   Max.   :4.00   Max.   :3    Max.   :179.1
> head(data1)
  density block fertilizer    yield
1       1     1          1 177.2287
2       2     2          1 177.5500
3       1     3          1 176.4085
4       2     4          1 177.7036
5       1     1          1 177.1255
6       2     2          1 176.7783
> one.way <- aov(yield ~ fertilizer, data = data1)
> summary(one.way)
```

```
> summary(one.way)
            Df Sum Sq Mean Sq F value   Pr(>F)
fertilizer   1   5.74   5.743   14.91 0.000207 ***
Residuals   94  36.21   0.385
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
> "two way anova"
[1] "two way anova"
> data2<-read.csv(file.choose(),sep=",",header = T)
> names(data2)
[1] "density"    "block"      "fertilizer" "yield"
> summary(data2)
    density        block        fertilizer     yield
 Min.   :1.0   Min.   :1.00   Min.   :1    Min.   :175.4
 1st Qu.:1.0   1st Qu.:1.75   1st Qu.:1    1st Qu.:176.5
 Median :1.5   Median :2.50   Median :2    Median :177.1
 Mean   :1.5   Mean   :2.50   Mean   :2    Mean   :177.0
 3rd Qu.:2.0   3rd Qu.:3.25   3rd Qu.:3    3rd Qu.:177.4
 Max.   :2.0   Max.   :4.00   Max.   :3    Max.   :179.1
> two.way <- aov(yield ~ fertilizer + density, data = data2)
> summary(two.way)
            Df Sum Sq Mean Sq F value   Pr(>F)
fertilizer   1  5.743   5.743   17.18 7.49e-05 ***
density      1  5.122   5.122   15.32 0.000173 ***
Residuals   93 31.089   0.334
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
>
```

PRACTICAL-10

AIM- Practical of Decision Tree

```
rm(list=ls())
library(ISLR)
data(package="ISLR")
data <- Carseats
head(data)        #First few rows for each column of the data
library(tree)
require(tree)
names(data)
hist(data$Sales)
#creating Sales_bin based on the Sales variable
data$Sales_bin <- as.factor(ifelse(data$Sales >= 8, "yes", "no"))
#droping the original Sales variable
data$Sales = NULL
#Take a look at the data
head(data)
set.seed(200)
#Developing the model
train_m <- sample(1: nrow(data), nrow(data)*0.70)
#Making the split
Train_data <- data[train_m,]
Test_data <- data[-train_m,]
rm(data, train_m)
head(Train_data)
head(Test_data)
Des_tree_model <- tree(Sales_bin~., Train_data)
plot(Des_tree_model)
text(Des_tree_model, pretty = 0)
#Using the model on testing dataset to check how good it is going
Pred_tree <- predict(Des_tree_model, Test_data, type = "class")
mean(Pred_tree != Test_data$Sales_bin)
```

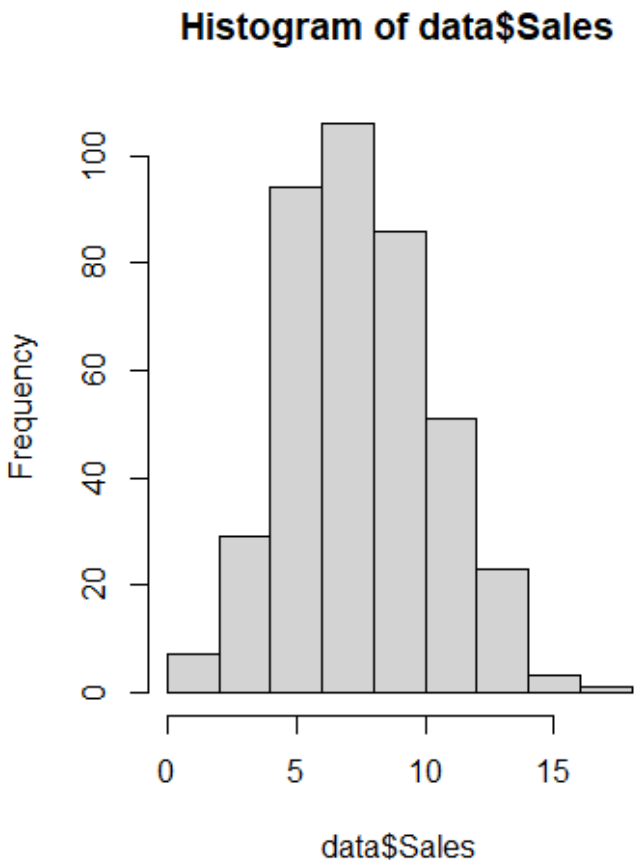
Output:

```
head(data)          #First few rows for each column of the data
  Sales CompPrice Income Advertising Population Price ShelveLoc Age Educat
ion Urban  US
1  9.50       138     73          11        276   120       Bad  42
17   Yes Yes
2 11.22       111     48          16        260    83      Good  65
10   Yes Yes
3 10.06       113     35          10        269    80    Medium  59
12   Yes Yes
4  7.40       117    100           4        466    97    Medium  55
14   Yes Yes
```

```
5  4.15       141     64          3        340   128       Bad  38
13   Yes  No
6 10.81       124    113         13        501    72       Bad  78
16    No Yes
```

```
names(data)
 [1] "Sales"       "CompPrice"   "Income"      "Advertising" "Population"
"Price"       "ShelveLoc"   "Age"
 [9] "Education"   "Urban"       "US"
```

```
Hist(data$Sales)
```




```
head(data)
  CompPrice Income Advertising Population Price ShelveLoc Age Education Ur
ban  US Sales_bin
1       138     73          11        276   120       Bad  42        17
Yes Yes       yes
2       111     48          16        260    83      Good  65        10
Yes Yes       yes
3       113     35          10        269    80    Medium  59        12
Yes Yes       yes
4       117    100           4        466    97    Medium  55        14
Yes Yes        no
5       141     64           3        340   128       Bad  38        13
Yes  No        no
6       124    113          13        501    72       Bad  78        16
No Yes       yes
```

```
head(Train_data)
    CompPrice Income Advertising Population Price ShelveLoc Age Education
Urban  US Sales_bin
166       147     58           7        100   191       Bad  27        15
Yes Yes        no
370       135    100          22        463   122    Medium  36        14
Yes Yes       yes
239       121     24           0        200   133      Good  73        13
Yes  No        no
232       132     69           0        123   122    Medium  27        11
No  No       yes
215       115    115           3         48   107    Medium  73        18
Yes Yes        no
220       116     79          19        359   116      Good  58        17
Yes Yes       yes

> head(Test_data)
   CompPrice Income Advertising Population Price ShelveLoc Age Education U
rban  US Sales_bin
6         124    113          13        501    72       Bad  78        16
No Yes       yes
9         132    110           0        108   124    Medium  76        10
No  No        no
17        118     32           0        284   110      Good  63        13
Yes  No        no
18        147     74          13        251   131      Good  52        10
Yes Yes       yes
19        110    110           0        408    68      Good  46        17
No Yes       yes
21        125     90           2        367   131    Medium  35        18
Yes Yes        no
```