DR B.R. AMBEDKAR NATIONAL INSTITUTE OF TECHNOLOGY, JALANDHAR

Data Mining and Data Warehousing Lab

CSX-425

Session: July-Dec 2020

SUBMITTED TO-

Dr. Kunwar Pal
Assistant Professor
CSE Department

SUBMITTED BY-

Name – Ankit Goyal
Roll no - 17103011
Group - G-1
Branch - CSE

Assignment – 1

1. For given A and B:

a) **Find matrix – matrix multiplication (AB)**
b) **Find $(AB)^T$ and $(AB)^{-1}$**
c) **Find the mean, standard deviation for each column and row for the matrices A, B, AB, $(AB)^{-1}$.**

Program:

```
A <- rbind(c(3,-2,1),c(-1,4,-2))
B <- rbind(c(-7,4),c(9,5),c(2,-1))

print("Matrix A : ")
print(A)
print("Matrix B :")
print(B)

#AB
C <-A%*%B
print("Multiplication AB :")
print(C)

#T(AB)
T <-t(C)
print("Transpose of Matrix AB :")
print(T)

#I(AB)
I <- solve(C)
print("Inverse of Matrix AB :")
print(I)

#Mean
print("Mean of Matrix A :")
#Row
mean(A[1,])
mean(A[2,])
#column
mean(A[,1])
mean(A[,2])
mean(A[,3])

print("Mean of Matrix B :")
#Row
mean(B[1,])
mean(B[2,])
mean(B[3,])
#column
mean(B[,1])
mean(B[,2])

print("Mean of Matrix AB :")
#Row
mean(C[1,])
mean(C[2,])
#column
```

```
mean(C[,1])
mean(C[,2])

print("Mean of Matrix Inverse of AB :")
#Row
mean(I[1,])
mean(I[2,])
#column
mean(I[,1])
mean(I[,2])

#Standard Deviations
print("Standard deviation of matrix A :")
sd(A,na.rm=TRUE)
print("Standard deviation of matrix B :")
sd(B,na.rm=TRUE)
print("Standard deviation of matrix AB :")
sd(C,na.rm=TRUE)
print("Standard deviation of matrix inverse of AB :")
sd(I,na.rm=TRUE)
```

Output:

```
> A <- rbind(c(3,-2,1),c(-1,4,-2))
> B <- rbind(c(-7,4),c(9,5),c(2,-1))
>
> print("Matrix A : ")
[1] "Matrix A : "
> print(A)
     [,1] [,2] [,3]
[1,]    3   -2    1
[2,]   -1    4   -2
> print("Matrix B :")
[1] "Matrix B :"
> print(B)
     [,1] [,2]
[1,]   -7    4
[2,]    9    5
[3,]    2   -1
>
> #AB
> C <-A%*%B
> print("Multiplication AB :")
[1] "Multiplication AB :"
> print(C)
     [,1] [,2]
[1,]  -37    1
[2,]   39   18
```

```
>
> #T(AB)
> T <-t(C)
> print("Transpose of Matrix AB :")
[1] "Transpose of Matrix AB :"
> print(T)
     [,1] [,2]
[1,]  -37   39
[2,]    1   18
>
> #I(AB)
> I <- solve(C)
> print("Inverse of Matrix AB :")
[1] "Inverse of Matrix AB :"
> print(I)
            [,1]       [,2]
[1,] -0.02553191 0.00141844
[2,]  0.05531915 0.05248227
>
```

```
> #Mean
> print("Mean of Matrix A :")
[1] "Mean of Matrix A :"
> #Row
> mean(A[1,])
[1] 0.6666667
> mean(A[2,])
[1] 0.3333333
> #column
> mean(A[,1])
[1] 1
> mean(A[,2])
[1] 1
> mean(A[,3])
[1] -0.5
>
> print("Mean of Matrix B :")
[1] "Mean of Matrix B :"
> #Row
> mean(B[1,])
[1] -1.5
> mean(B[2,])
[1] 7
> mean(B[3,])
[1] 0.5
> #column
> mean(B[,1])
[1] 1.333333
> mean(B[,2])
[1] 2.666667
>
> print("Mean of Matrix AB :")
[1] "Mean of Matrix AB :"
> #Row
> mean(C[1,])
[1] -18
> mean(C[2,])
[1] 28.5
> #column
> mean(C[,1])
[1] 1
> mean(C[,2])
[1] 9.5
>
> print("Mean of Matrix Inverse of AB :")
[1] "Mean of Matrix Inverse of AB :"
> #Row
> mean(I[1,])
[1] -0.01205674
> mean(I[2,])
[1] 0.05390071
```

```
> #column
> mean(I[,1])
[1] 0.01489362
> mean(I[,2])
[1] 0.02695035
>
> #Standard Deviations
> print("Standard deviation of matrix A :")
[1] "Standard deviation of matrix A :"
> sd(A,na.rm=TRUE)
[1] 2.588436
> print("Standard deviation of matrix B :")
[1] "Standard deviation of matrix B :"
> sd(B,na.rm=TRUE)
[1] 5.51362
> print("Standard deviation of matrix AB :")
[1] "Standard deviation of matrix AB :"
> sd(C,na.rm=TRUE)
[1] 32.17012
> print("Standard deviation of matrix inverse of AB :")
[1] "Standard deviation of matrix inverse of AB :"
> sd(I,na.rm=TRUE)
[1] 0.03965505
>
```

2. Write a "Function" program in R to find n! . Hence Find 13! , 32! ,.Do not name the function by "Factorial". You can initialize that 0!=1 and 1!=1.

Program:

```
findfactorial <- function(n){

  factorial <- 1
  if((n==0||n==1))
    factorial <- 1
  else{
    for(i in 1:n)
      factorial <- factorial*i
  }
  return (factorial)
}

print(findfactorial(13))
print(findfactorial(32))
```

Output:

```
> findfactorial <- function(n){
+
+   factorial <- 1
+   if((n==0||n==1))
+     factorial <- 1
+   else{
+     for(i in 1:n)
+       factorial <- factorial*i
+   }
+   return (factorial)
+ }
>
> print(findfactorial(13))
[1] 6227020800
> print(findfactorial(32))
[1] 2.631308e+35
>
```

3. Write a "Function" program in R to find maximum and minimum from a set of numbers. Do not name the function by "max" or "min". As an input you take (4,44.7,2,40,54,1,3,4).

Program:

```
vector1 <- c(4,44.7,2,40,54,1,3,4)
l <- length(vector1)

min1 = 10000
max1 = -10000

for(i in 1:l){
  if(min1>vector1[i]){
    min1 = vector1[i];
  }
  if(max1<vector1[i]){
    max1 = vector1[i];
  }
}

print(paste("Minimum is", min1))
print(paste("Maximum is", max1))
```

Output :

```
> vector1 <- c(4,44.7,2,40,54,1,3,4)
> l <- length(vector1)
>
> min1 = 10000
> max1 = -10000
>
> for(i in 1:l){
+   if(min1>vector1[i]){
+     min1 = vector1[i];
+   }
+   if(max1<vector1[i]){
+     max1 = vector1[i];
+   }
+ }
>
> print(paste("Minimum is", min1))
[1] "Minimum is 1"
> print(paste("Maximum is", max1))
[1] "Maximum is 54"
>
```

Assignment - 2

1. How to read/write data from the dataset in R.

Program:

```
library(readxl)
dataset=read_excel(file.choose())
View(dataset)
```

Output:

Filter

	...1	vote	age	economic.cond.national	economic.cond.household	Blair	Hague	Kennedy	Europe	political.knowledge	gender
1	1	Liberal Democrat	43	3	3	4	1	4	2	2	female
2	2	Labour	36	4	4	4	4	4	5	2	male
3	3	Labour	35	4	4	5	2	3	3	2	male
4	4	Labour	24	4	2	2	1	3	4	0	female
5	5	Labour	41	2	2	1	1	4	6	2	male
6	6	Labour	47	3	4	4	4	2	4	2	male
7	7	Liberal Democrat	57	2	2	4	4	2	11	2	male
8	8	Labour	77	3	4	4	1	4	1	0	male
9	9	Labour	39	3	3	4	4	4	11	0	female
10	10	Labour	70	3	2	5	1	1	11	2	male
11	11	Labour	39	3	3	1	2	1	7	0	female
12	12	Labour	66	4	3	4	4	4	9	2	male
13	13	Labour	59	4	4	4	1	4	10	2	female
14	14	Labour	66	3	3	2	5	4	8	0	female
15	15	Labour	77	2	3	2	1	4	11	2	female
16	16	Labour	51	4	4	4	4	4	5	0	male
17	17	Labour	43	2	4	1	4	3	8	0	female
18	18	Labour	41	4	4	5	4	2	7	2	female

Showing 1 to 19 of 1,525 entries, 11 total columns

2. Use different function in R

a) Read
b) Head
c) Tail
d) Names

Program:

```
data <- read_excel("BEPSxls.xlsx")
#data-read
print("************************");
print(data)
head(data,5)
tail(data,5)

print("******Names Data ***********")
print(names(data))
```

Output:

```
> data <- read_excel("BEPSxls.xlsx")
New names:
* `` -> ...1
> #data-read
> print("************************");
[1] "************************"
> print(data)
# A tibble: 1,525 x 11
    ...1 vote              age economic.cond.national economic.cond.household Blair Hague Kennedy Europe political.knowledge gender
   <dbl> <chr>           <dbl>                 <dbl>                   <dbl> <dbl> <dbl>   <dbl>  <dbl>               <dbl> <chr>
 1     1 Liberal Democrat    43                     3                       3     4     1       4      2                   2 female
 2     2 Labour              36                     4                       4     4     4       4      5                   2 male
 3     3 Labour              35                     4                       4     5     2       3      3                   2 male
 4     4 Labour              24                     4                       2     2     1       3      4                   0 female
 5     5 Labour              41                     2                       2     1     1       4      6                   2 male
 6     6 Labour              47                     3                       4     4     4       2      4                   2 male
 7     7 Liberal Democrat    57                     2                       2     4     4       2     11                   2 male
 8     8 Labour              77                     3                       4     4     1       4      1                   0 male
 9     9 Labour              39                     3                       3     4     4       4     11                   0 female
10    10 Labour              70                     3                       2     5     1       1     11                   2 male
# ... with 1,515 more rows
> head(data,5)
# A tibble: 5 x 11
   ...1 vote              age economic.cond.national economic.cond.household Blair Hague Kennedy Europe political.knowledge gender
  <dbl> <chr>           <dbl>                 <dbl>                   <dbl> <dbl> <dbl>   <dbl>  <dbl>               <dbl> <chr>
1     1 Liberal Democrat    43                     3                       3     4     1       4      2                   2 female
2     2 Labour              36                     4                       4     4     4       4      5                   2 male
3     3 Labour              35                     4                       4     5     2       3      3                   2 male
4     4 Labour              24                     4                       2     2     1       3      4                   0 female
5     5 Labour              41                     2                       2     1     1       4      6                   2 male
> tail(data,5)
# A tibble: 5 x 11
   ...1 vote           age economic.cond.national economic.cond.household Blair Hague Kennedy Europe political.knowledge gender
  <dbl> <chr>        <dbl>                 <dbl>                   <dbl> <dbl> <dbl>   <dbl>  <dbl>               <dbl> <chr>
1  1521 Conservative    67                     5                       3     2     4       4     11                   3 male
2  1522 Conservative    73                     2                       2     4     4       4      8                   2 male
3  1523 Labour          37                     3                       3     5     4       4      2                   2 male
4  1524 Conservative    61                     3                       3     1     4       1     11                   2 male
5  1525 Conservative    74                     2                       3     2     4       4     11                   0 female
>
> print("******Names Data ***********")
[1] "******Names Data ***********"
> print(names(data))
 [1] "...1"                "vote"                "age"                 "economic.cond.national"  "economic.cond.household"
 [6] "Blair"               "Hague"               "Kennedy"             "Europe"                  "political.knowledge"
[11] "gender"
>
```

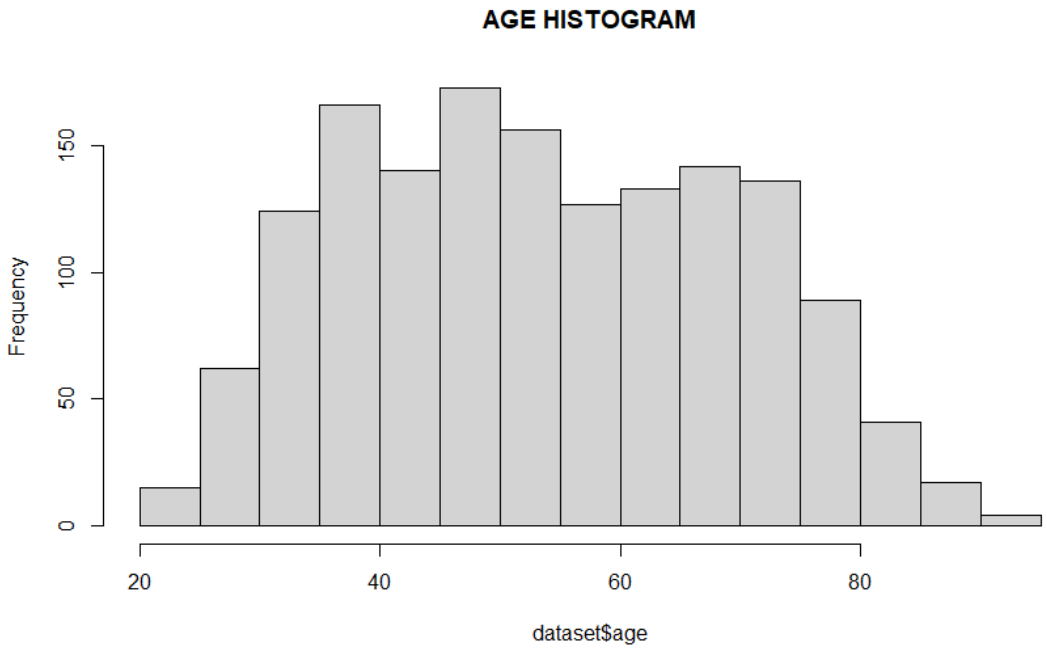
3. **Download the given dataset and perform the following.**
    a. **Mean**
    b. **Median**
    c. **Summary**
    d. **Histogram**
    e. **Plot**

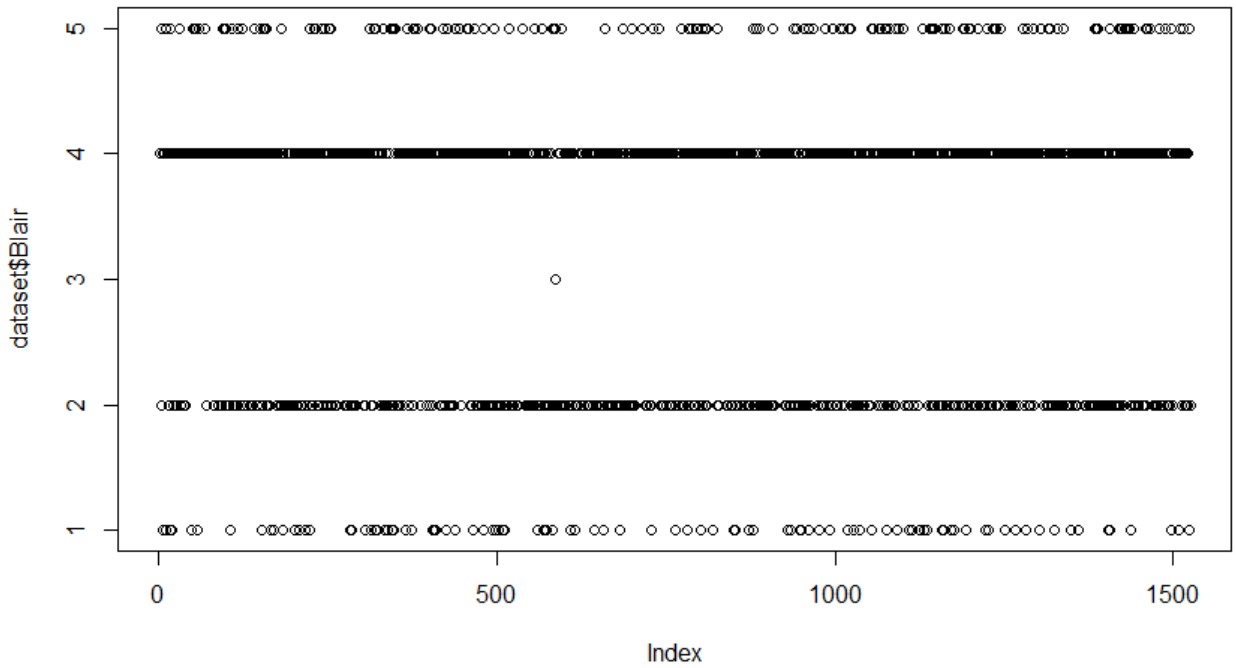
Program:

```
dataset<- read_excel("BEPSxls.xlsx")
mean(dataset$age)
median(dataset$age)
summary(dataset)
hist(dataset$age,main = 'AGE HISTOGRAM')
plot(dataset$Blair)
```

Output:

```
> mean(dataset$age)
[1] 54.1823
> median(dataset$age)
[1] 53
> summary(dataset)
      ...1          vote                age        economic.cond.national economic.cond.household     Blair           Hague          Kennedy     
 Min.   :   1   Length:1525        Min.   :24.00   Min.   :1.000          Min.   :1.00            Min.   :1.000   Min.   :1.000   Min.   :1.000  
 1st Qu.: 382   Class :character   1st Qu.:41.00   1st Qu.:3.000          1st Qu.:3.00            1st Qu.:2.000   1st Qu.:2.000   1st Qu.:2.000  
 Median : 763   Mode  :character   Median :53.00   Median :3.000          Median :3.00            Median :4.000   Median :2.000   Median :3.000  
 Mean   : 763                      Mean   :54.18   Mean   :3.246          Mean   :3.14            Mean   :3.334   Mean   :2.747   Mean   :3.135  
 3rd Qu.:1144                      3rd Qu.:67.00   3rd Qu.:4.000          3rd Qu.:4.00            3rd Qu.:4.000   3rd Qu.:4.000   3rd Qu.:4.000  
 Max.   :1525                      Max.   :93.00   Max.   :5.000          Max.   :5.00            Max.   :5.000   Max.   :5.000   Max.   :5.000  
     Europe       political.knowledge    gender         
 Min.   : 1.000   Min.   :0.000       Length:1525       
 1st Qu.: 4.000   1st Qu.:0.000       Class :character  
 Median : 6.000   Median :2.000       Mode  :character  
 Mean   : 6.729   Mean   :1.542                         
 3rd Qu.:10.000   3rd Qu.:2.000                         
 Max.   :11.000   Max.   :3.000                         
> hist(dataset$age,main = 'AGE HISTOGRAM')
> plot(dataset$Blair)
>
```


AGE HISTOGRAM
Frequency
0
50
100
150
20
40
60
80
dataset$age


dataset$Blair
1
2
3
4
5
0
500
1000
1500
Index

4. Attach and Detach the dataset in R.

Program:

```
data <- data.frame(x1 = c(9, 8, 3, 4, 8),
                   x2 = c(5, 4, 7, 1, 1),
                   x3 = c(1, 2, 3, 4, 5))
data
x1 #give error
attach(data)
x1 #run
detach(data)
x1 # give error
```

Output:

```
> data <- data.frame(x1 = c(9, 8, 3, 4, 8),
+                    x2 = c(5, 4, 7, 1, 1),
+                    x3 = c(1, 2, 3, 4, 5))
> data
  x1 x2 x3
1  9  5  1
2  8  4  2
3  3  7  3
4  4  1  4
5  8  1  5
> x1 #give error
Error: object 'x1' not found
> attach(data)
> x1 #run
[1] 9 8 3 4 8
> detach(data)
> x1 # give error
Error: object 'x1' not found
>
```

Assignment – 3

1. Demonstration of pre-processing on dataset mtcars. (R-studio)

Program:

```
mtcars
mtcars$mpg = ifelse(is.na(mtcars$mpg),ave(mtcars$mpg, FUN = function(x)
mean(x,na.rm='TRUE')),mtcars$mpg)
mtcars
```

Output:

```
> mtcars$mpg = ifelse(is.na(mtcars$mpg),ave(mtcars$mpg, FUN = function(x) mean(x,na.rm='TRUE')),mtcars$mpg)
> mtcars
                   mpg cyl  disp  hp drat    wt  qsec vs am gear carb
Mazda RX4         21.0   6 160.0 110 3.90 2.620 16.46  0  1    4    4
Mazda RX4 Wag     21.0   6 160.0 110 3.90 2.875 17.02  0  1    4    4
Datsun 710        22.8   4 108.0  93 3.85 2.320 18.61  1  1    4    1
Hornet 4 Drive    21.4   6 258.0 110 3.08 3.215 19.44  1  0    3    1
Hornet Sportabout 18.7   8 360.0 175 3.15 3.440 17.02  0  0    3    2
Valiant           18.1   6 225.0 105 2.76 3.460 20.22  1  0    3    1
Duster 360        14.3   8 360.0 245 3.21 3.570 15.84  0  0    3    4
Merc 240D         24.4   4 146.7  62 3.69 3.190 20.00  1  0    4    2
Merc 230          22.8   4 140.8  95 3.92 3.150 22.90  1  0    4    2
Merc 280          19.2   6 167.6 123 3.92 3.440 18.30  1  0    4    4
Merc 280C         17.8   6 167.6 123 3.92 3.440 18.90  1  0    4    4
Merc 450SE        16.4   8 275.8 180 3.07 4.070 17.40  0  0    3    3
Merc 450SL        17.3   8 275.8 180 3.07 3.730 17.60  0  0    3    3
Merc 450SLC       15.2   8 275.8 180 3.07 3.780 18.00  0  0    3    3
Cadillac Fleetwood  10.4   8 472.0 205 2.93 5.250 17.98  0  0    3    4
Lincoln Continental 10.4   8 460.0 215 3.00 5.424 17.82  0  0    3    4
Chrysler Imperial   14.7   8 440.0 230 3.23 5.345 17.42  0  0    3    4
Fiat 128          32.4   4  78.7  66 4.08 2.200 19.47  1  1    4    1
Honda Civic       30.4   4  75.7  52 4.93 1.615 18.52  1  1    4    2
Toyota Corolla    33.9   4  71.1  65 4.22 1.835 19.90  1  1    4    1
Toyota Corona     21.5   4 120.1  97 3.70 2.465 20.01  1  0    3    1
Dodge Challenger  15.5   8 318.0 150 2.76 3.520 16.87  0  0    3    2
AMC Javelin       15.2   8 304.0 150 3.15 3.435 17.30  0  0    3    2
Camaro Z28        13.3   8 350.0 245 3.73 3.840 15.41  0  0    3    4
Pontiac Firebird  19.2   8 400.0 175 3.08 3.845 17.05  0  0    3    2
Fiat X1-9         27.3   4  79.0  66 4.08 1.935 18.90  1  1    4    1
Porsche 914-2     26.0   4 120.3  91 4.43 2.140 16.70  0  1    5    2
Lotus Europa      30.4   4  95.1 113 3.77 1.513 16.90  1  1    5    2
Ford Pantera L    15.8   8 351.0 264 4.22 3.170 14.50  0  1    5    4
Ferrari Dino      19.7   6 145.0 175 3.62 2.770 15.50  0  1    5    6
Maserati Bora     15.0   8 301.0 335 3.54 3.570 14.60  0  1    5    8
Volvo 142E        21.4   4 121.0 109 4.11 2.780 18.60  1  1    4    2
>
```

2. **Demonstrate the filter function on dataset mtcars using (deplyr package)**
    a. **Show where gear attribute = 4,**
    b. **Show where disp = 160,**
    c. **Show different operations (and,or,not)**

Program:

```
library(dplyr)

#1 Show where gear attribute = 4,
gear_4 <- filter(mtcars, gear == 4)
head(gear_4)

#2 Show where disp = 160.
disp_160 <- filter(mtcars, disp == 160.0)
head(disp_160)

#3 Show different operations (and, or, not)
#AND
gear4_and_carb4 <- filter(mtcars, gear == 4 & carb == 4)
head(gear4_and_carb4)
#OR
gear4_or_hp110 <- filter(mtcars, gear == 4 | hp == 110)
head(gear4_or_hp110)
#Not
gearNot4 <- filter(mtcars, gear != 4)
head(gearNot4)
```

Output:

```
> library(dplyr)
>
> #1 Show where gear attribute = 4,
> gear_4 <- filter(mtcars, gear == 4)
> head(gear_4)
               mpg cyl  disp  hp drat    wt  qsec vs am gear carb
Mazda RX4     21.0   6 160.0 110 3.90 2.620 16.46  0  1    4    4
Mazda RX4 Wag 21.0   6 160.0 110 3.90 2.875 17.02  0  1    4    4
Datsun 710    22.8   4 108.0  93 3.85 2.320 18.61  1  1    4    1
Merc 240D     24.4   4 146.7  62 3.69 3.190 20.00  1  0    4    2
Merc 230      22.8   4 140.8  95 3.92 3.150 22.90  1  0    4    2
Merc 280      19.2   6 167.6 123 3.92 3.440 18.30  1  0    4    4
>
> #2 Show where disp = 160.
> disp_160 <- filter(mtcars, disp == 160.0)
> head(disp_160)
              mpg cyl disp  hp drat    wt  qsec vs am gear carb
Mazda RX4      21   6  160 110  3.9 2.620 16.46  0  1    4    4
Mazda RX4 Wag  21   6  160 110  3.9 2.875 17.02  0  1    4    4
>
> #3 Show different operations (and, or, not)
> #AND
> gear4_and_carb4 <- filter(mtcars, gear == 4 & carb == 4)
> head(gear4_and_carb4)
               mpg cyl  disp  hp drat    wt  qsec vs am gear carb
Mazda RX4     21.0   6 160.0 110 3.90 2.620 16.46  0  1    4    4
Mazda RX4 Wag 21.0   6 160.0 110 3.90 2.875 17.02  0  1    4    4
Merc 280      19.2   6 167.6 123 3.92 3.440 18.30  1  0    4    4
Merc 280C     17.8   6 167.6 123 3.92 3.440 18.90  1  0    4    4
> #OR
> gear4_or_hp110 <- filter(mtcars, gear == 4 | hp == 110)
> head(gear4_or_hp110)
               mpg cyl  disp  hp drat    wt  qsec vs am gear carb
Mazda RX4     21.0   6 160.0 110 3.90 2.620 16.46  0  1    4    4
Mazda RX4 Wag 21.0   6 160.0 110 3.90 2.875 17.02  0  1    4    4
Datsun 710    22.8   4 108.0  93 3.85 2.320 18.61  1  1    4    1
Hornet 4 Drive 21.4  6 258.0 110 3.08 3.215 19.44  1  0    3    1
Merc 240D     24.4   4 146.7  62 3.69 3.190 20.00  1  0    4    2
Merc 230      22.8   4 140.8  95 3.92 3.150 22.90  1  0    4    2
> #Not
> gearNot4 <- filter(mtcars, gear != 4)
> head(gearNot4)
                  mpg cyl  disp  hp drat    wt  qsec vs am gear carb
Hornet 4 Drive    21.4   6 258.0 110 3.08 3.215 19.44  1  0    3    1
Hornet Sportabout 18.7   8 360.0 175 3.15 3.440 17.02  0  0    3    2
Valiant           18.1   6 225.0 105 2.76 3.460 20.22  1  0    3    1
Duster 360        14.3   8 360.0 245 3.21 3.570 15.84  0  0    3    4
Merc 450SE        16.4   8 275.8 180 3.07 4.070 17.40  0  0    3    3
Merc 450SL        17.3   8 275.8 180 3.07 3.730 17.60  0  0    3    3
>
```

3. **Demonstrate the different function on dataset mtcars/Titanic**
    a. **arrange**
    b. **group_by**
    c. **summarise**
    d. **select**
    e. **intersect**
    f. **setdiff**

Program:

```
print("Arrange : ")
arrange(mtcars, desc(disp))

print("Group By : ")
group_by(mtcars,drat)

print("Summarise : ")
summarise(mtcars,mean(disp))

print("Select : ")
select(mtcars,qsec)

print("Intersect :")
A<- subset(mtcars,disp==160)
B<- subset(mtcars,cyl=100)
intersect(A,B)

print("SetDiff :")
setdiff(B,A)
```

Output:

```
> print("Arrange : ")
[1] "Arrange : "
> arrange(mtcars, desc(disp))
                     mpg cyl  disp  hp drat    wt  qsec vs am gear carb
Cadillac Fleetwood  10.4   8 472.0 205 2.93 5.250 17.98  0  0    3    4
Lincoln Continental 10.4   8 460.0 215 3.00 5.424 17.82  0  0    3    4
Chrysler Imperial   14.7   8 440.0 230 3.23 5.345 17.42  0  0    3    4
Pontiac Firebird    19.2   8 400.0 175 3.08 3.845 17.05  0  0    3    2
Hornet Sportabout   18.7   8 360.0 175 3.15 3.440 17.02  0  0    3    2
Duster 360          14.3   8 360.0 245 3.21 3.570 15.84  0  0    3    4
Ford Pantera L      15.8   8 351.0 264 4.22 3.170 14.50  0  1    5    4
Camaro Z28          13.3   8 350.0 245 3.73 3.840 15.41  0  0    3    4
Dodge Challenger    15.5   8 318.0 150 2.76 3.520 16.87  0  0    3    2
AMC Javelin         15.2   8 304.0 150 3.15 3.435 17.30  0  0    3    2
Maserati Bora       15.0   8 301.0 335 3.54 3.570 14.60  0  1    5    8
Merc 450SE          16.4   8 275.8 180 3.07 4.070 17.40  0  0    3    3
Merc 450SL          17.3   8 275.8 180 3.07 3.730 17.60  0  0    3    3
Merc 450SLC         15.2   8 275.8 180 3.07 3.780 18.00  0  0    3    3
Hornet 4 Drive      21.4   6 258.0 110 3.08 3.215 19.44  1  0    3    1
Valiant             18.1   6 225.0 105 2.76 3.460 20.22  1  0    3    1
Merc 280            19.2   6 167.6 123 3.92 3.440 18.30  1  0    4    4
Merc 280C           17.8   6 167.6 123 3.92 3.440 18.90  1  0    4    4
Mazda RX4           21.0   6 160.0 110 3.90 2.620 16.46  0  1    4    4
Mazda RX4 Wag       21.0   6 160.0 110 3.90 2.875 17.02  0  1    4    4
Merc 240D           24.4   4 146.7  62 3.69 3.190 20.00  1  0    4    2
Ferrari Dino        19.7   6 145.0 175 3.62 2.770 15.50  0  1    5    6
Merc 230            22.8   4 140.8  95 3.92 3.150 22.90  1  0    4    2
Volvo 142E          21.4   4 121.0 109 4.11 2.780 18.60  1  1    4    2
Porsche 914-2       26.0   4 120.3  91 4.43 2.140 16.70  0  1    5    2
Toyota Corona       21.5   4 120.1  97 3.70 2.465 20.01  1  0    3    1
Datsun 710          22.8   4 108.0  93 3.85 2.320 18.61  1  1    4    1
Lotus Europa        30.4   4  95.1 113 3.77 1.513 16.90  1  1    5    2
Fiat X1-9           27.3   4  79.0  66 4.08 1.935 18.90  1  1    4    1
Fiat 128            32.4   4  78.7  66 4.08 2.200 19.47  1  1    4    1
Honda Civic         30.4   4  75.7  52 4.93 1.615 18.52  1  1    4    2
Toyota Corolla      33.9   4  71.1  65 4.22 1.835 19.90  1  1    4    1
>
> print("Group By : ")
[1] "Group By : "
> group_by(mtcars,drat)
# A tibble: 32 x 11
# Groups:   drat [22]
     mpg   cyl  disp    hp  drat    wt  qsec    vs    am  gear  carb
   <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
 1  21       6  160    110  3.9   2.62  16.5     0     1     4     4
 2  21       6  160    110  3.9   2.88  17.0     0     1     4     4
 3  22.8     4  108     93  3.85  2.32  18.6     1     1     4     1
 4  21.4     6  258    110  3.08  3.22  19.4     1     0     3     1
 5  18.7     8  360    175  3.15  3.44  17.0     0     0     3     2
 6  18.1     6  225    105  2.76  3.46  20.2     1     0     3     1
 7  14.3     8  360    245  3.21  3.57  15.8     0     0     3     4
 8  24.4     4  147.    62  3.69  3.19  20       1     0     4     2
 9  22.8     4  141.    95  3.92  3.15  22.9     1     0     4     2
10  19.2     6  168.   123  3.92  3.44  18.3     1     0     4     4
# ... with 22 more rows
>
```

```
> print("Summarise : ")
[1] "Summarise : "
> summarise(mtcars,mean(disp))
  mean(disp)
1   230.7219
>
> print("Select : ")
[1] "Select : "
> select(mtcars,qsec)
                   qsec
Mazda RX4         16.46
Mazda RX4 Wag     17.02
Datsun 710        18.61
Hornet 4 Drive    19.44
Hornet Sportabout 17.02
Valiant           20.22
Duster 360        15.84
Merc 240D         20.00
Merc 230          22.90
Merc 280          18.30
Merc 280C         18.90
Merc 450SE        17.40
Merc 450SL        17.60
Merc 450SLC       18.00
Cadillac Fleetwood  17.98
Lincoln Continental 17.82
Chrysler Imperial   17.42
Fiat 128          19.47
Honda Civic       18.52
Toyota Corolla    19.90
Toyota Corona     20.01
Dodge Challenger  16.87
AMC Javelin       17.30
Camaro Z28        15.41
Pontiac Firebird  17.05
Fiat X1-9         18.90
Porsche 914-2     16.70
Lotus Europa      16.90
Ford Pantera L    14.50
Ferrari Dino      15.50
Maserati Bora     14.60
Volvo 142E        18.60
>
> print("Intersect :")
[1] "Intersect :"
> A<- subset(mtcars,disp==160)
> B<- subset(mtcars,cyl=100)
> intersect(A,B)
              mpg cyl disp  hp drat    wt  qsec vs am gear carb
Mazda RX4      21   6  160 110  3.9 2.620 16.46  0  1    4    4
Mazda RX4 Wag  21   6  160 110  3.9 2.875 17.02  0  1    4    4
>
```

```
> print("SetDiff :")
[1] "SetDiff :"
> setdiff(B,A)
                   mpg cyl  disp  hp drat    wt  qsec vs am gear carb
Datsun 710        22.8   4 108.0  93 3.85 2.320 18.61  1  1    4    1
Hornet 4 Drive    21.4   6 258.0 110 3.08 3.215 19.44  1  0    3    1
Hornet Sportabout 18.7   8 360.0 175 3.15 3.440 17.02  0  0    3    2
Valiant           18.1   6 225.0 105 2.76 3.460 20.22  1  0    3    1
Duster 360        14.3   8 360.0 245 3.21 3.570 15.84  0  0    3    4
Merc 240D         24.4   4 146.7  62 3.69 3.190 20.00  1  0    4    2
Merc 230          22.8   4 140.8  95 3.92 3.150 22.90  1  0    4    2
Merc 280          19.2   6 167.6 123 3.92 3.440 18.30  1  0    4    4
Merc 280C         17.8   6 167.6 123 3.92 3.440 18.90  1  0    4    4
Merc 450SE        16.4   8 275.8 180 3.07 4.070 17.40  0  0    3    3
Merc 450SL        17.3   8 275.8 180 3.07 3.730 17.60  0  0    3    3
Merc 450SLC       15.2   8 275.8 180 3.07 3.780 18.00  0  0    3    3
Cadillac Fleetwood 10.4  8 472.0 205 2.93 5.250 17.98  0  0    3    4
Lincoln Continental 10.4 8 460.0 215 3.00 5.424 17.82  0  0    3    4
Chrysler Imperial 14.7   8 440.0 230 3.23 5.345 17.42  0  0    3    4
Fiat 128          32.4   4  78.7  66 4.08 2.200 19.47  1  1    4    1
Honda Civic       30.4   4  75.7  52 4.93 1.615 18.52  1  1    4    2
Toyota Corolla    33.9   4  71.1  65 4.22 1.835 19.90  1  1    4    1
Toyota Corona     21.5   4 120.1  97 3.70 2.465 20.01  1  0    3    1
Dodge Challenger  15.5   8 318.0 150 2.76 3.520 16.87  0  0    3    2
AMC Javelin       15.2   8 304.0 150 3.15 3.435 17.30  0  0    3    2
Camaro Z28        13.3   8 350.0 245 3.73 3.840 15.41  0  0    3    4
Pontiac Firebird  19.2   8 400.0 175 3.08 3.845 17.05  0  0    3    2
Fiat X1-9         27.3   4  79.0  66 4.08 1.935 18.90  1  1    4    1
Porsche 914-2     26.0   4 120.3  91 4.43 2.140 16.70  0  1    5    2
Lotus Europa      30.4   4  95.1 113 3.77 1.513 16.90  1  1    5    2
Ford Pantera L    15.8   8 351.0 264 4.22 3.170 14.50  0  1    5    4
Ferrari Dino      19.7   6 145.0 175 3.62 2.770 15.50  0  1    5    6
Maserati Bora     15.0   8 301.0 335 3.54 3.570 14.60  0  1    5    8
Volvo 142E        21.4   4 121.0 109 4.11 2.780 18.60  1  1    4    2
>
```

4. Remove the not required columns from mtcars dataset.

Program:

```
DATA <- subset(mtcars,select=c(1:9))
print(DATA)
```

Output:

```
> DATA <- subset(mtcars,select=c(1:9))
> print(DATA)
                   mpg cyl  disp  hp drat    wt  qsec vs am
Mazda RX4         21.0   6 160.0 110 3.90 2.620 16.46  0  1
Mazda RX4 Wag     21.0   6 160.0 110 3.90 2.875 17.02  0  1
Datsun 710        22.8   4 108.0  93 3.85 2.320 18.61  1  1
Hornet 4 Drive    21.4   6 258.0 110 3.08 3.215 19.44  1  0
Hornet Sportabout 18.7   8 360.0 175 3.15 3.440 17.02  0  0
Valiant           18.1   6 225.0 105 2.76 3.460 20.22  1  0
Duster 360        14.3   8 360.0 245 3.21 3.570 15.84  0  0
Merc 240D         24.4   4 146.7  62 3.69 3.190 20.00  1  0
Merc 230          22.8   4 140.8  95 3.92 3.150 22.90  1  0
Merc 280          19.2   6 167.6 123 3.92 3.440 18.30  1  0
Merc 280C         17.8   6 167.6 123 3.92 3.440 18.90  1  0
Merc 450SE        16.4   8 275.8 180 3.07 4.070 17.40  0  0
Merc 450SL        17.3   8 275.8 180 3.07 3.730 17.60  0  0
Merc 450SLC       15.2   8 275.8 180 3.07 3.780 18.00  0  0
Cadillac Fleetwood  10.4   8 472.0 205 2.93 5.250 17.98  0  0
Lincoln Continental 10.4   8 460.0 215 3.00 5.424 17.82  0  0
Chrysler Imperial   14.7   8 440.0 230 3.23 5.345 17.42  0  0
Fiat 128          32.4   4  78.7  66 4.08 2.200 19.47  1  1
Honda Civic       30.4   4  75.7  52 4.93 1.615 18.52  1  1
Toyota Corolla    33.9   4  71.1  65 4.22 1.835 19.90  1  1
Toyota Corona     21.5   4 120.1  97 3.70 2.465 20.01  1  0
Dodge Challenger  15.5   8 318.0 150 2.76 3.520 16.87  0  0
AMC Javelin       15.2   8 304.0 150 3.15 3.435 17.30  0  0
Camaro Z28        13.3   8 350.0 245 3.73 3.840 15.41  0  0
Pontiac Firebird  19.2   8 400.0 175 3.08 3.845 17.05  0  0
Fiat X1-9         27.3   4  79.0  66 4.08 1.935 18.90  1  1
Porsche 914-2     26.0   4 120.3  91 4.43 2.140 16.70  0  1
Lotus Europa      30.4   4  95.1 113 3.77 1.513 16.90  1  1
Ford Pantera L    15.8   8 351.0 264 4.22 3.170 14.50  0  1
Ferrari Dino      19.7   6 145.0 175 3.62 2.770 15.50  0  1
Maserati Bora     15.0   8 301.0 335 3.54 3.570 14.60  0  1
Volvo 142E        21.4   4 121.0 109 4.11 2.780 18.60  1  1
>
```

5. Show the attribute containing NA values in a column in dataset.

Program:

```
myData <- data.frame(col1 = c(4,6,2, NA),
                     col2 = c("string", NA,"is", "ankit"),
                     col3 = c(TRUE, NA, TRUE, TRUE),
                     col4 = c(2.5, NA, 3.2, NA))

is.na(myData)
```

Output:

```
> myData <- data.frame(col1 = c(4,6,2, NA),
+                      col2 = c("string", NA,"is", "ankit"),
+                      col3 = c(TRUE, NA, TRUE, TRUE),
+                      col4 = c(2.5, NA, 3.2, NA))
>
> is.na(myData)
      col1  col2  col3  col4
[1,] FALSE FALSE FALSE FALSE
[2,] FALSE  TRUE  TRUE  TRUE
[3,] FALSE FALSE FALSE FALSE
[4,]  TRUE FALSE FALSE  TRUE
>
```

6. Repeat the above question on downloaded dataset.

Program:

```
Is.na(mtcars)
```

Output:

```
> is.na(mtcars)
                     mpg   cyl  disp    hp  drat    wt  qsec    vs    am  gear  carb
Mazda RX4          FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
Mazda RX4 Wag      FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
Datsun 710         FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
Hornet 4 Drive     FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
Hornet Sportabout  FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
Valiant            FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
Duster 360         FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
Merc 240D          FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
Merc 230           FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
Merc 280           FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
Merc 280C          FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
Merc 450SE         FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
Merc 450SL         FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
Merc 450SLC        FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
Cadillac Fleetwood FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
Lincoln Continental FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
Chrysler Imperial  FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
Fiat 128           FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
Honda Civic        FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
Toyota Corolla     FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
Toyota Corona      FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
Dodge Challenger   FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
AMC Javelin        FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
Camaro Z28         FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
Pontiac Firebird   FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
Fiat X1-9          FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
Porsche 914-2      FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
Lotus Europa       FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
Ford Pantera L     FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
Ferrari Dino       FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
Maserati Bora      FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
Volvo 142E         FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
```

Assignment- 4

1. **As a crop researcher, you want to test effect of three different fertilizers mixtures on crop yield. You can use a one-way ANOVA to find out if there is a difference in crop yields between the three groups. Using the data, perform a one-way analysis of variance using alpha = .05.**
    a. **Perform a one-way analysis of variance**
    b. **Calculate test statistics**
    c. **Interpreting the results**
    d. **State conclusion**
    e. **Plot the graph for the same**

Theory:

The one-way analysis of variance (ANOVA), also known as one-factor ANOVA, is an extension of independent two-samples t-test for comparing means in a situation where there are more than two groups. In one-way ANOVA, the data is organized into several groups base on one single grouping variable (also called factor variable). The one-way analysis of variance (ANOVA) is used to determine whether there are any statistically significant differences between the means of three or more independent (unrelated) groups.To clarify if the data comes from the same population, you can perform a one-way analysis of variance (one-way ANOVA hereafter). This test, like any other statistical tests, gives evidence whether the H0 hypothesis can be accepted or rejected.

Hypothesis in one-way ANOVA test:

- H0: The means between groups are identical
- H3: At least, the mean of one group is different

In other words, the H0 hypothesis implies that there is not enough evidence to prove the mean of the group (factor) are different from another.

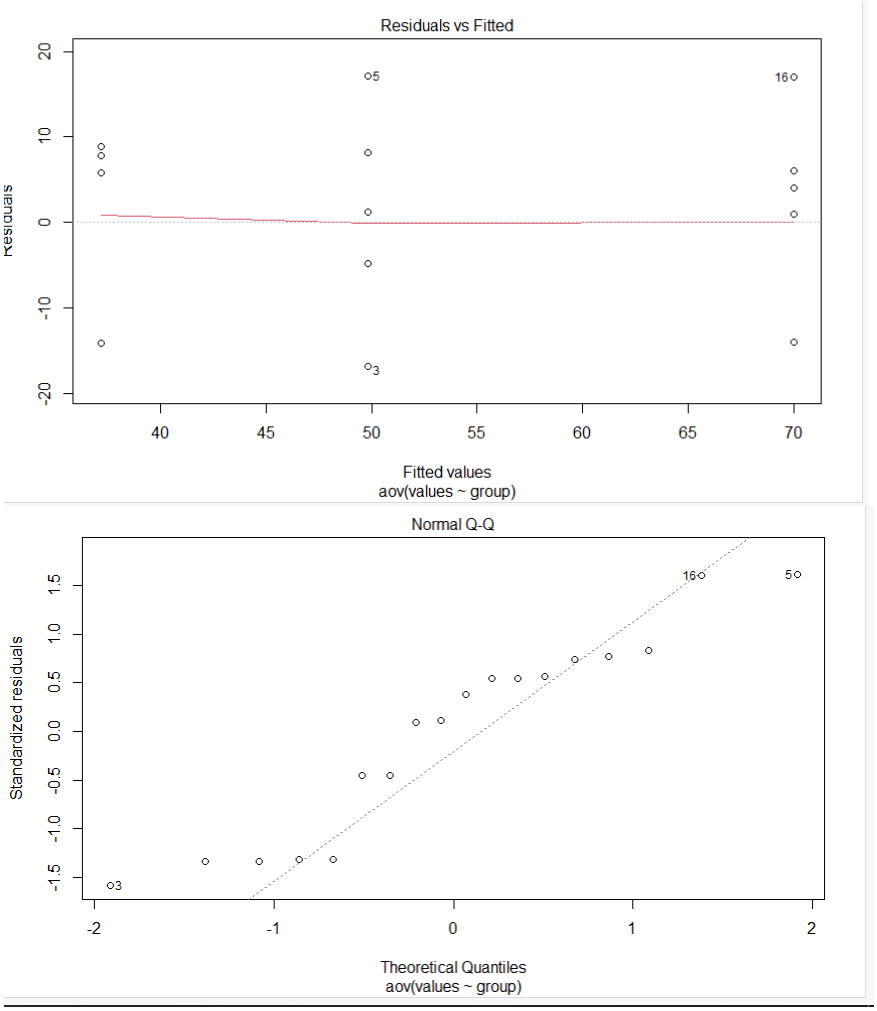
Program:

```
library(readxl)
my_data=read_excel(file.choose())

#my_data <- read_excel('DMDW_LAB4.xlsx')
View(my_data)

#check and display ordered levels
my_data$group <- ordered(my_data$group, levels = c("Group1", "Group2",
"Group3"))

#compute summary statistics by group
library(dplyr)
group_by(my_data, group) %>%
```

```
  summarise(
    count = n(),mean = mean(values, na.rm = TRUE),
    sd = sd(values, na.rm = TRUE)
  )

#compute one way ANOVA
#compute analysis of variance
res.aov <- aov(values ~ group, data = my_data)
#summary of analysis
summary(res.aov)

#interpret result of ANOVA
#multiple pairwise comparison
TukeyHSD(res.aov)
#homogeneity
plot(res.aov,1)
#normality
plot(res.aov,2)
```

Output:

```
`summarise()` ungrouping output (override with `.groups` argument)
# A tibble: 3 x 4
  group  count  mean    sd
  <ord>  <int> <dbl> <dbl>
1 Group1     6  49.8  11.8
2 Group2     6  37.2  11.0
3 Group3     6  70    12.1
>
> #compute one way ANOVA
> #compute analysis of variance
> res.aov <- aov(values ~ group, data = my_data)
> #summary of analysis
> summary(res.aov)
            Df Sum Sq Mean Sq F value   Pr(>F)
group        2   3290  1645.2   12.12 0.000737 ***
Residuals   15   2036   135.7
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
>
> #interpret result of ANOVA
> #multiple pairwise comparison
> TukeyHSD(res.aov)
  Tukey multiple comparisons of means
    95% family-wise confidence level

Fit: aov(formula = values ~ group, data = my_data)

$group
                 diff        lwr       upr     p adj
Group2-Group1 -12.66667 -30.136858  4.803524 0.1777035
Group3-Group1  20.16667   2.696476 37.636858 0.0230632
Group3-Group2  32.83333  15.363142 50.303524 0.0005503

> #homogeneity
```


Residuals vs Fitted
Residuals
Fitted values
aov(values ~ group)
Normal Q-Q
Standardized residuals
Theoretical Quantiles
aov(values ~ group)

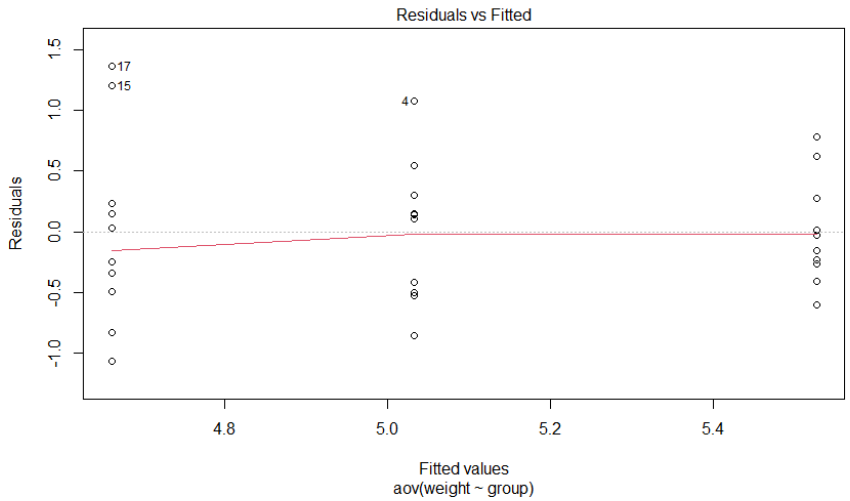
2. Repeat the quesion1 and perform one-way analysis of variance using inbuilt dataset in Rstudio.

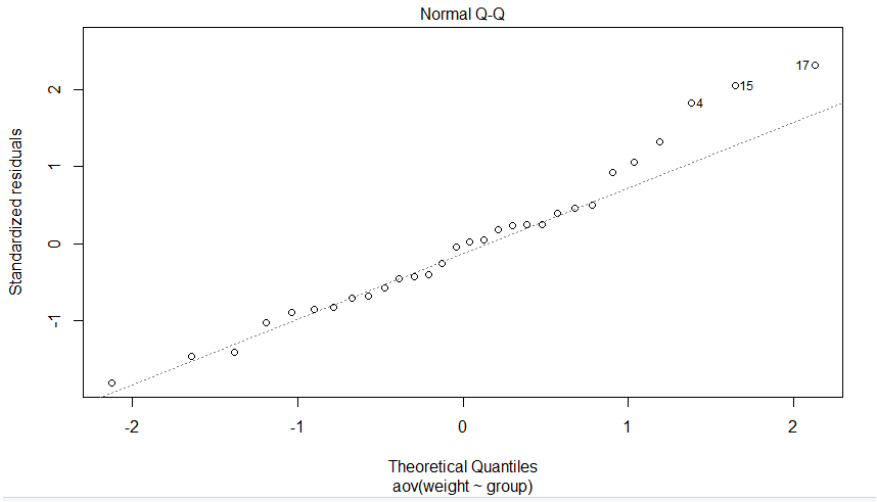
Program:

```
#build data
my_data<- PlantGrowth

#check data and display ordered levels
sample_n(my_data,10)

#show levels
levels(my_data$group)

#compute summary statistics
library(dplyr)
group_by(my_data, group) %>%
  summarise(
    count = n(),
    mean = mean(weight, na.rm = TRUE),
    sd = sd(weight, na.rm = TRUE)
  )

#compute anova test
# Compute the analysis of variance
res.aov <- aov(weight ~ group, data = my_data)
# Summary of the analysis
summary(res.aov)

#Interpret the result of one-way ANOVA tests
#multiple pairwise comparison
TukeyHSD(res.aov)
#Homogeneity of variances
plot(res.aov, 1)
#Normality
plot(res.aov, 2)
```

Output:

```
> sample_n(my_data,10)
   weight group
1    5.58  ctrl
2    5.87  trt1
3    5.54  trt2
4    5.80  trt2
5    4.81  trt1
6    4.53  ctrl
7    4.69  trt1
8    4.92  trt2
9    3.59  trt1
10   4.32  trt1
>
> #show levels
> levels(my_data$group)
[1] "ctrl" "trt1" "trt2"
>
> #compute summary statistics
> library(dplyr)
> group_by(my_data, group) %>%
+   summarise(
+     count = n(),
+     mean = mean(weight, na.rm = TRUE),
+     sd = sd(weight, na.rm = TRUE)
+   )
`summarise()` ungrouping output (override with `.groups` argument)
# A tibble: 3 x 4
  group count  mean    sd
  <fct> <int> <dbl> <dbl>
1 ctrl     10  5.03 0.583
2 trt1     10  4.66 0.794
3 trt2     10  5.53 0.443
>
> #compute anova test
> # Compute the analysis of variance
> res.aov <- aov(weight ~ group, data = my_data)
> # Summary of the analysis
> summary(res.aov)
            Df Sum Sq Mean Sq F value Pr(>F)
group        2  3.766  1.8832   4.846 0.0159 *
Residuals   27 10.492  0.3886
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
>
> #Interpret the result of one-way ANOVA tests
> #multiple pairwise comparison
> TukeyHSD(res.aov)
  Tukey multiple comparisons of means
    95% family-wise confidence level

Fit: aov(formula = weight ~ group, data = my_data)

$group
            diff        lwr       upr     p adj
trt1-ctrl -0.371 -1.0622161 0.3202161 0.3908711
trt2-ctrl  0.494 -0.1972161 1.1852161 0.1979960
trt2-trt1  0.865  0.1737839 1.5562161 0.0120064
```




Normal Q-Q
Standardized residuals
Theoretical Quantiles
aov(weight ~ group)
17
15
4
-2
-1
0
1
2

Assignment – 5

1. **Consider dataset "Groceries" and apply apriori algorithm on it. What are the first 5 rules generated when the min support is 0.001 and min confidence is 0.9?**

Program:

```
library(arules)
groceries <- read_excel("LAB5.csv")
rules=apriori(data= groceries, parameter =
list(support=0.001,confidence=0.9))
inspect(rules[1:5])
```

Output:

```
> inspect(rules[1:5])
    lhs                          rhs                          support confidence coverage lift count
[1] {Transaction ID=T1}       => {Items=Bread,Butter,Eggs}    0.25    1          0.25     4    1
[2] {Items=Bread,Butter,Eggs} => {Transaction ID=T1}          0.25    1          0.25     4    1
[3] {Transaction ID=T2}       => {Items=Butter,Eggs,Milk}     0.25    1          0.25     4    1
[4] {Items=Butter,Eggs,Milk}  => {Transaction ID=T2}          0.25    1          0.25     4    1
[5] {Transaction ID=T3}       => {Items=Butter}               0.25    1          0.25     4    1
>
```

2. The database has four transaction. What association rule can be found in this set, if the minimum support is 60% and minimum confidence is 80%.

Program:

```
library(arules)
library(readr)
groceries2 <- read_excel("LAB5-2.csv")

rules=apriori(data= groceries2,parameter= list(support=0.6,confidence=0.8))
rules
```

Output:

```
Parameter specification:
 confidence minval smax arem  aval originalSupport maxtime support minlen maxlen target  ext
        0.8    0.1    1 none FALSE            TRUE       5     0.6      1     10  rules TRUE

Algorithmic control:
 filter tree heap memopt load sort verbose
    0.1 TRUE TRUE  FALSE TRUE    2    TRUE

Absolute minimum support count: 2

set item appearances ...[0 item(s)] done [0.00s].
set transactions ...[8 item(s), 4 transaction(s)] done [0.00s].
sorting and recoding items ... [0 item(s)] done [0.00s].
creating transaction tree ... done [0.00s].
checking subsets of size 1 done [0.00s].
writing ... [0 rule(s)] done [0.00s].
creating S4 object  ... done [0.00s].
Warning message:
Column(s) 1, 2 not logical or factor. Applying default discretization (see '? discretizeDF').
> rules
set of 0 rules
```

3. Demonstration of association rule process on dataset titanic using apriori algorithm in rstudio.

Code:

```
library(arules)
library(readr)
titanic <- read_csv("titanic.csv")
data(titanic)
rules=apriori(data= titanic, parameter =
list(support=0.6,confidence=0.8))
rules
inspect(rules[1:5])
```

OUTPUT:

```
> inspect(rules[1:5])
    lhs           rhs                 support  confidence coverage lift count
[1] {}         => {Survived=[0,1]} 1.000000 1          1.000000 1    891
[2] {}         => {SibSp=[0,8]}    1.000000 1          1.000000 1    891
[3] {}         => {Parch=[0,6]}    1.000000 1          1.000000 1    891
[4] {Sex=male} => {Survived=[0,1]} 0.647587 1          0.647587 1    577
[5] {Sex=male} => {SibSp=[0,8]}    0.647587 1          0.647587 1    577
>
```

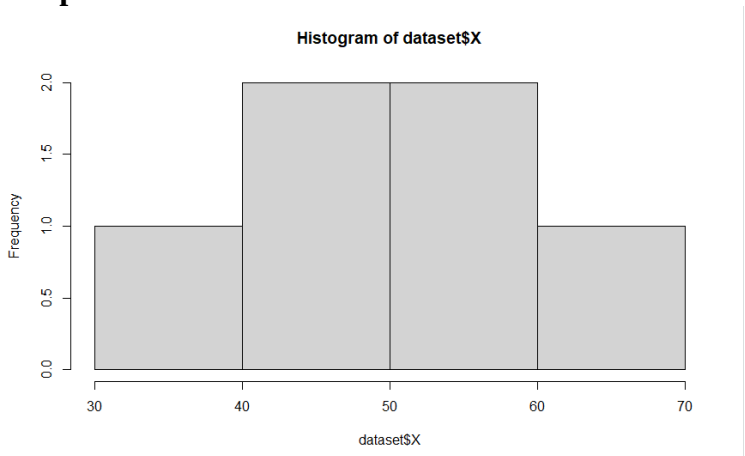
Assignment - 6

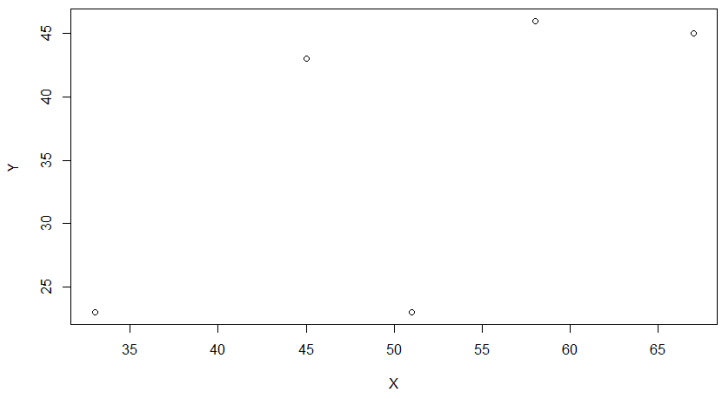
1. **Demonstrate performing linear regression on given data using R/Python.**
   a. **Plot the scattered graph**
   b. **Calculate test statistics**
   c. **Find coefficient and different performance matrix**

Program:

```
dataset <- read_excel("LAB6.xlsx")
summary(dataset)
hist(dataset$X)
plot(Y~X, data=dataset)
dataset.lm <- lm(Y~X, dataset)
summary(dataset.lm)
```

Output:





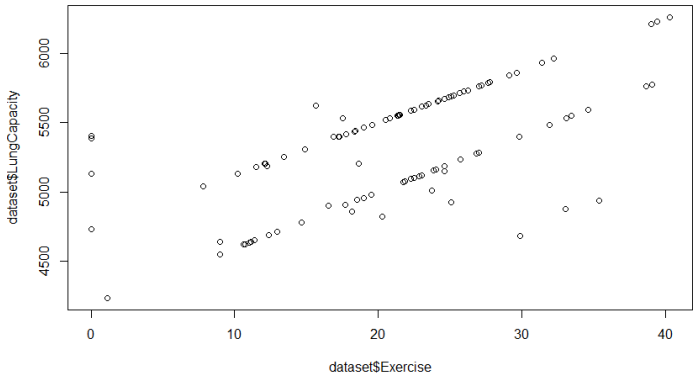

```
Call:
lm(formula = Y ~ X, data = dataset)

Residuals:
      1       2       3       4       5       6
-14.793   8.430  -5.124   8.430  -1.388   4.446

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  10.3979    19.5823   0.531    0.624
X             0.5372     0.3841   1.398    0.235

Residual standard error: 10.11 on 4 degrees of freedom
Multiple R-squared:  0.3284,    Adjusted R-squared:  0.1604
F-statistic: 1.956 on 1 and 4 DF,  p-value: 0.2345
```

2. Demonstrate performing linear regression on Lung capacity dataset using R/Python.

Program:

```
dataset <- read_excel("Lung Capacity.xls")
summary(dataset)
cor(dataset$Height, dataset$LungCapacity)
cor(dataset$Age, dataset$LungCapacity)
plot(dataset$Exercise, dataset$LungCapacity,data = dataset)


dataset.lm <- lm(dataset$LungCapacity ~ dataset$Gender + dataset$Height +
dataset$Smoker  +dataset$Exercise, data= dataset)
summary(dataset.lm)
```

Output:




```
Call:
lm(formula = dataset$LungCapacity ~ dataset$Gender + dataset$Height +
    dataset$Smoker + dataset$Exercise, data = dataset)

Residuals:
   Min     1Q Median     3Q    Max
-546.9 -126.9   -3.9  148.1  521.7

Coefficients:
                Estimate Std. Error t value Pr(>|t|)
(Intercept)     1656.937    467.790   3.542 0.000617 ***
dataset$Gender   202.105     41.557   4.863 4.57e-06 ***
dataset$Height    50.360      7.043   7.150 1.78e-10 ***
dataset$Smoker  -279.025     52.433  -5.322 6.85e-07 ***
dataset$Exercise  11.260      2.943   3.825 0.000234 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 199.2 on 95 degrees of freedom
Multiple R-squared:  0.7741,    Adjusted R-squared:  0.7646
F-statistic: 81.38 on 4 and 95 DF,  p-value: < 2.2e-16
```

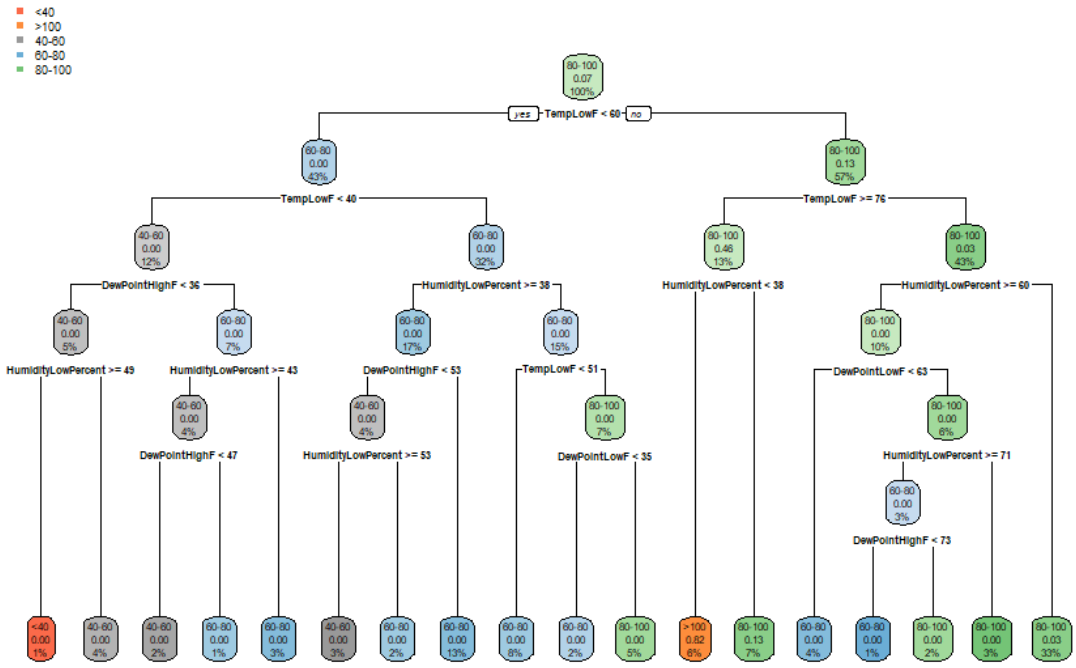
Assignment - 7

1. To construct Decision tree for weather data and classify it.

Program:

```
library(rpart.plot)
library(rpart)
dataset <- read_csv("austin_weather.csv")
head(dataset)
shuffle_index<-sample(1:nrow(dataset))
dataset <- dataset[shuffle_index,]
ls(dataset)
sum(is.na(dataset$Events))
dim(dataset)

sum(is.na(dataset$DewPointAvgF))
summary(dataset$TempHighF)

dataset = subset(dataset, select = -c(Date,Events,TempAvgF, DewPointAvgF,
HumidityAvgPercent,SeaLevelPressureAvgInches, VisibilityAvgMiles, WindAvgMPH ))

str(dataset)
dataset[] <- lapply(dataset, as.numeric)

dataset <- dataset %>%
  mutate(TempHighF = case_when(
    TempHighF < 40           ~ "<40",
    TempHighF >= 40 & TempHighF < 60 ~ "40-60",
    TempHighF >= 60 & TempHighF < 80 ~ "60-80",
    TempHighF >= 80 & TempHighF < 100 ~ "80-100",
    TempHighF >= 100                  ~ ">100",
    TRUE                              ~ "NA"
  ))

fit <- rpart(TempHighF~., data = dataset, method = 'class')
rpart.plot(fit, extra = 106)
```

Output:

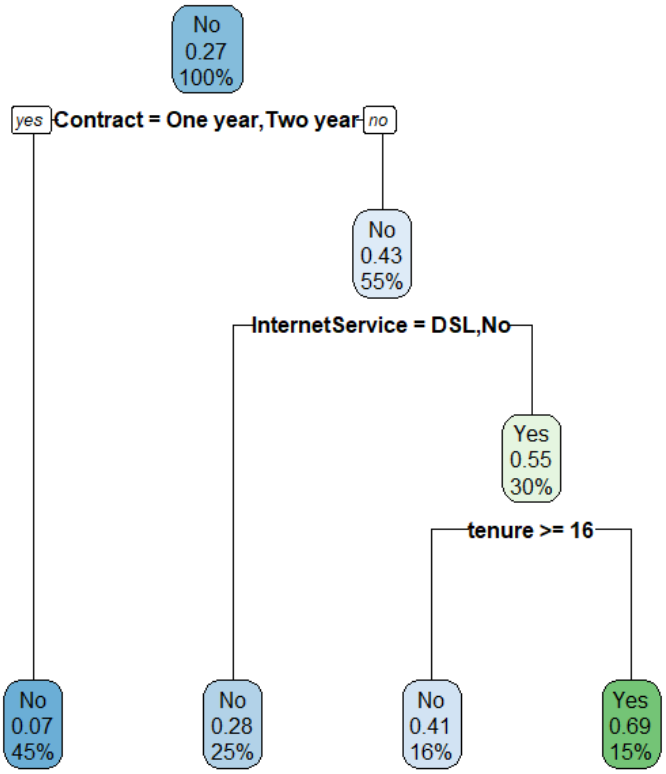

<40
>100
40-60
60-80
80-100
80-100 0.07 100%
yes
TempLowF < 60
no
60-80 0.00 43%
80-100 0.13 57%
TempLowF < 40
TempLowF >= 76
40-60 0.00 12%
60-80 0.00 32%
80-100 0.46 13%
80-100 0.03 43%
DewPointHighF < 36
HumidityLowPercent >= 38
HumidityLowPercent < 38
HumidityLowPercent >= 60
40-60 0.00 5%
60-80 0.00 7%
60-80 0.00 17%
60-80 0.00 15%
80-100 0.00 10%
HumidityLowPercent >= 49
HumidityLowPercent >= 43
DewPointHighF < 53
TempLowF < 51
DewPointLowF < 63
40-60 0.00 4%
80-100 0.00 7%
80-100 0.00 6%
DewPointHighF < 47
HumidityLowPercent >= 53
DewPointLowF < 35
HumidityLowPercent >= 71
60-80 0.00 3%
DewPointHighF < 73
<40 0.00 1%
40-60 0.00 4%
40-60 0.00 2%
60-80 0.00 1%
60-80 0.00 3%
40-60 0.00 3%
60-80 0.00 2%
60-80 0.00 13%
60-80 0.00 8%
60-80 0.00 2%
80-100 0.00 5%
>100 0.82 6%
80-100 0.13 7%
60-80 0.00 4%
60-80 0.00 1%
80-100 0.00 2%
80-100 0.00 3%
80-100 0.03 33%

2. To construct Decision tree for customer data and classify it.

Program:

```
dataset <- read_csv("WA_Fn-UseC_-Telco-Customer-Churn.csv")
dim(dataset)
ls(dataset)
dataset = subset(dataset, select = -c(customerID ))
fit <- rpart(Churn~., data = dataset, method = 'class')
rpart.plot(fit, extra = 106)
```

Output:



Assignment - 8

1. Write a procedure for clustering customer data using Simple KMeans Algorithm

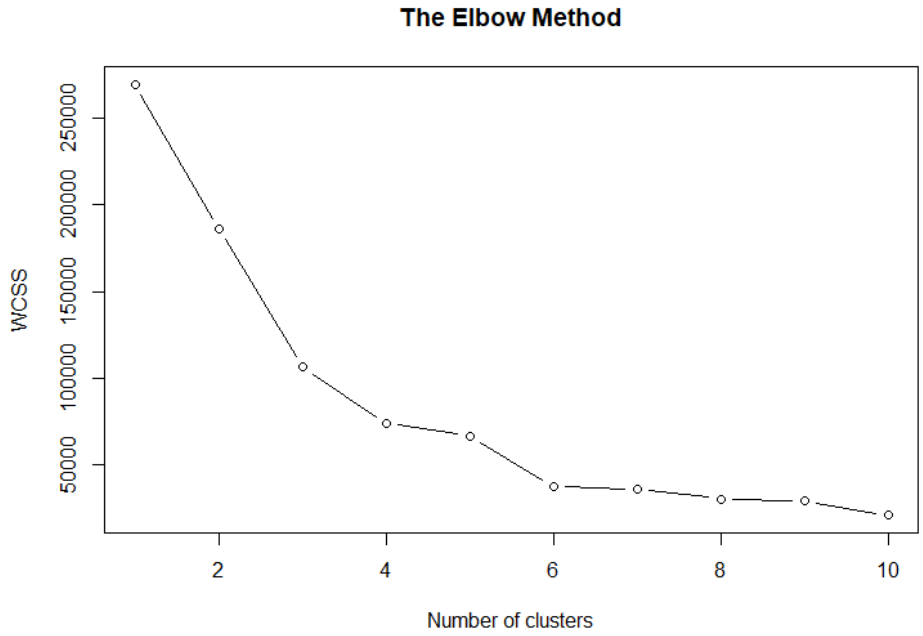
- Step 1: Choose groups in the feature plan randomly.
- Step 2: Minimize the distance between the cluster center and the different observations (**centroid**). It results in groups with observations.
- Step 3: Shift the initial centroid to the mean of the coordinates within a group.
- Step 4: Minimize the distance according to the new centroids. New boundaries are created. Thus, observations will move from one group to another.
- Repeat until no observation changes groups.

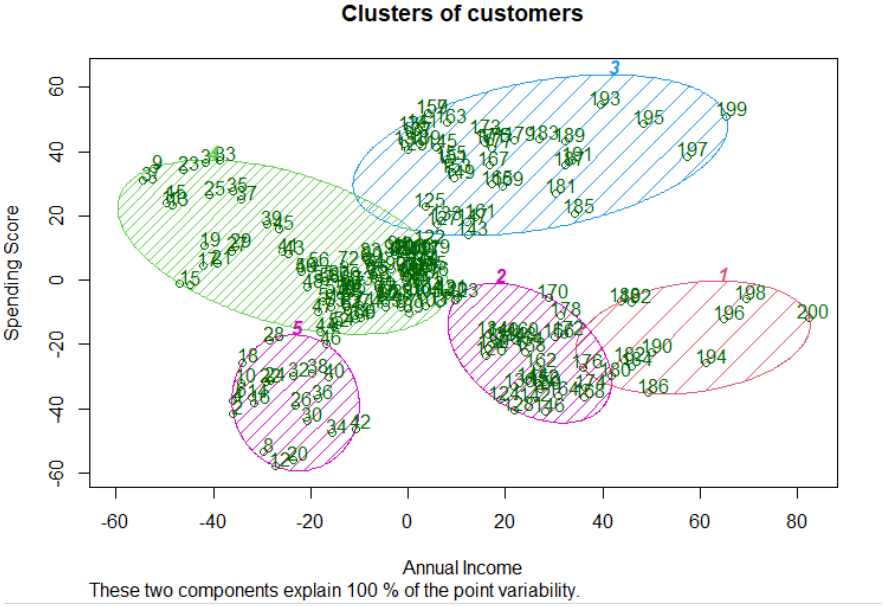
2. Demonstration of clustering rule process on dataset using simple k-means.

Program:

```
library(readr)
dataset = read_csv("Mall_Customers.csv")
dataset = dataset[4:5]
set.seed(6)
wcss = vector()
for (i in 1:10) wcss[i] = sum(kmeans(dataset, i)$withinss)
plot(1:10,
     wcss,
     type = 'b',
     main = paste('The Elbow Method'),
     xlab = 'Number of clusters',
     ylab = 'WCSS')
kmeans = kmeans(x = dataset, centers = 5)
y_kmeans = kmeans$cluster

# Visualising the clusters
library(cluster)
clusplot(dataset,
         y_kmeans,
         lines = 0,
         shade = TRUE,
         color = TRUE,
         labels = 2,
         plotchar = FALSE,
         span = TRUE,
         main = paste('Clusters of customers'),
         xlab = 'Annual Income',
         ylab = 'Spending Score')
```

Output:




Clusters of customers
Spending Score
Annual Income
These two components explain 100 % of the point variability.

Assignment - 9

1. **Demonstration of classification rule process on dataset using naïve bayes algorithm.**

Program:

```
# Installing Packages
install.packages("e1071")
install.packages("caTools")
install.packages("caret")

# Loading package
library(e1071)
library(caTools)
library(caret)
library(dplyr)

dataset = read_csv("Mall_Customers.csv")

dataset$Gender <- factor(dataset$Gender, levels = c("Male", "Female"),
labels = c(0,1))

dataset <- dataset %>%
  mutate(Age = case_when(
    Age < 30                  ~ "<30",
    Age >= 30 & Age < 45      ~ "30-45",
    Age >= 45 & Age < 60      ~ "45-60",
    Age >= 60                 ~ ">60",
    TRUE                      ~ "NA"
  ))

dataset <- dataset %>%
  mutate(Income = case_when(
    Income <40                    ~ "<40",
    Income >= 40 & Income < 60      ~ "40-60",
    Income >= 60                  ~ ">60",
    TRUE                      ~ "NA"
  ))

dataset <- dataset %>%
  mutate(Score = case_when(
    Score < 30                  ~ "<20",
    Score >= 20 & Score < 40      ~ "20-40",
    Score >= 40 & Score < 60      ~ "40-60",
    Score >= 60 & Score < 80      ~ "60-80",
    Score >= 80                 ~ ">80",
    TRUE                      ~ "NA"
  ))

trainIndex <- createDataPartition(dataset$Score, p = .7,
                                  list = FALSE,
                                  times = 1)

Train <- dataset[ trainIndex,]
Valid <- dataset[-trainIndex,]

# Fitting Naive Bayes Model
```

```
# to training dataset

classifier_cl <- naiveBayes(Score ~ ., data = Train)
classifier_cl
```

Output:

```
Naive Bayes Classifier for Discrete Predictors

Call:
naiveBayes.default(x = X, y = Y, laplace = laplace)

A-priori probabilities:
Y
       <20        >80      20-40      40-60      60-80
0.23076923 0.14685315 0.06993007 0.37762238 0.17482517

Conditional probabilities:
       CustomerID
Y            [,1]     [,2]
  <20   114.15152 72.06599
  >80   106.19048 71.32154
  20-40  75.40000 67.28909
  40-60  85.87037 23.96070
  60-80  97.96000 68.95196

       Gender
Y               0         1
  <20   0.5151515 0.4848485
  >80   0.4285714 0.5714286
  20-40 0.4000000 0.6000000
  40-60 0.4444444 0.5555556
  60-80 0.4400000 0.5600000

       Age
Y              <30        >60      30-45      45-60
  <20   0.18181818 0.03030303 0.36363636 0.42424242
  >80   0.47619048 0.00000000 0.52380952 0.00000000
  20-40 0.20000000 0.10000000 0.50000000 0.20000000
  40-60 0.31481481 0.22222222 0.16666667 0.29629630
  60-80 0.32000000 0.00000000 0.64000000 0.04000000

       Income
Y              <40        >60      40-60
  <20   0.36363636 0.63636364 0.00000000
  >80   0.38095238 0.61904762 0.00000000
  20-40 0.60000000 0.40000000 0.00000000
  40-60 0.01851852 0.42592593 0.55555556
  60-80 0.36000000 0.52000000 0.12000000
```

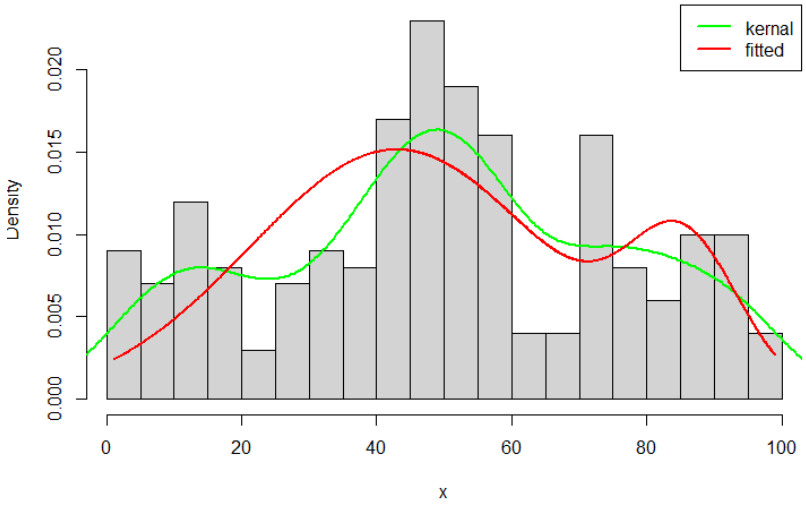
2. Demonstration of clustering rule process on dataset using EM algorithm.

Program:

```
install.packages("mixtools")
dataset = read_csv("Mall_Customers.csv")
summary(dataset$Score)
x <- dataset$Score
plot(density(x))

mem <- kmeans(x,2)$cluster
mu1 <- mean(x[mem==1])
mu2 <- mean(x[mem==2])
sigma1 <- sd(x[mem==1])
sigma2 <- sd(x[mem==2])
pi1 <- sum(mem==1)/length(mem)
pi2 <- sum(mem==2)/length(mem)
# modified sum only considers finite values
sum.finite <- function(x) {
  sum(x[is.finite(x)])
}

Q <- 0
# starting value of expected value of the log likelihood
Q[2] <- sum.finite(log(pi1)+log(dnorm(x, mu1, sigma1))) +
sum.finite(log(pi2)+log(dnorm(x, mu2, sigma2)))

k <- 2

while (abs(Q[k]-Q[k-1])>=1e-6) {
  # E step
  comp1 <- pi1 * dnorm(x, mu1, sigma1)
  comp2 <- pi2 * dnorm(x, mu2, sigma2)
  comp.sum <- comp1 + comp2

  p1 <- comp1/comp.sum
  p2 <- comp2/comp.sum

  # M step
  pi1 <- sum.finite(p1) / length(x)
  pi2 <- sum.finite(p2) / length(x)

  mu1 <- sum.finite(p1 * x) / sum.finite(p1)
  mu2 <- sum.finite(p2 * x) / sum.finite(p2)

  sigma1 <- sqrt(sum.finite(p1 * (x-mu1)^2) / sum.finite(p1))
  sigma2 <- sqrt(sum.finite(p2 * (x-mu2)^2) / sum.finite(p2))

  p1 <- pi1
  p2 <- pi2

  k <- k + 1
  Q[k] <- sum(log(comp.sum))
}

library(mixtools)
gm<-normalmixEM(x,k=2,lambda=c(0.9,0.1),mu=c(0.4,0.3),sigma=c(0.05,0.02))
gm$mu
gm$sigma
```

```
gm$lambda
hist(x, prob=T, breaks=32, xlim=c(range(x)[1], range(x)[2]), main='')
lines(density(x), col="green", lwd=2)
x1 <- seq(from=range(x)[1], to=range(x)[2], length.out=1000)
y <- pi1 * dnorm(x1, mean=mu1, sd=sigma1) + pi2 * dnorm(x1, mean=mu2,
sd=sigma2)
lines(x1, y, col="red", lwd=2)
legend('topright', col=c("green", 'red'), lwd=2, legend=c("kernal", "fitted"))
```

Output:



Assignment - 10

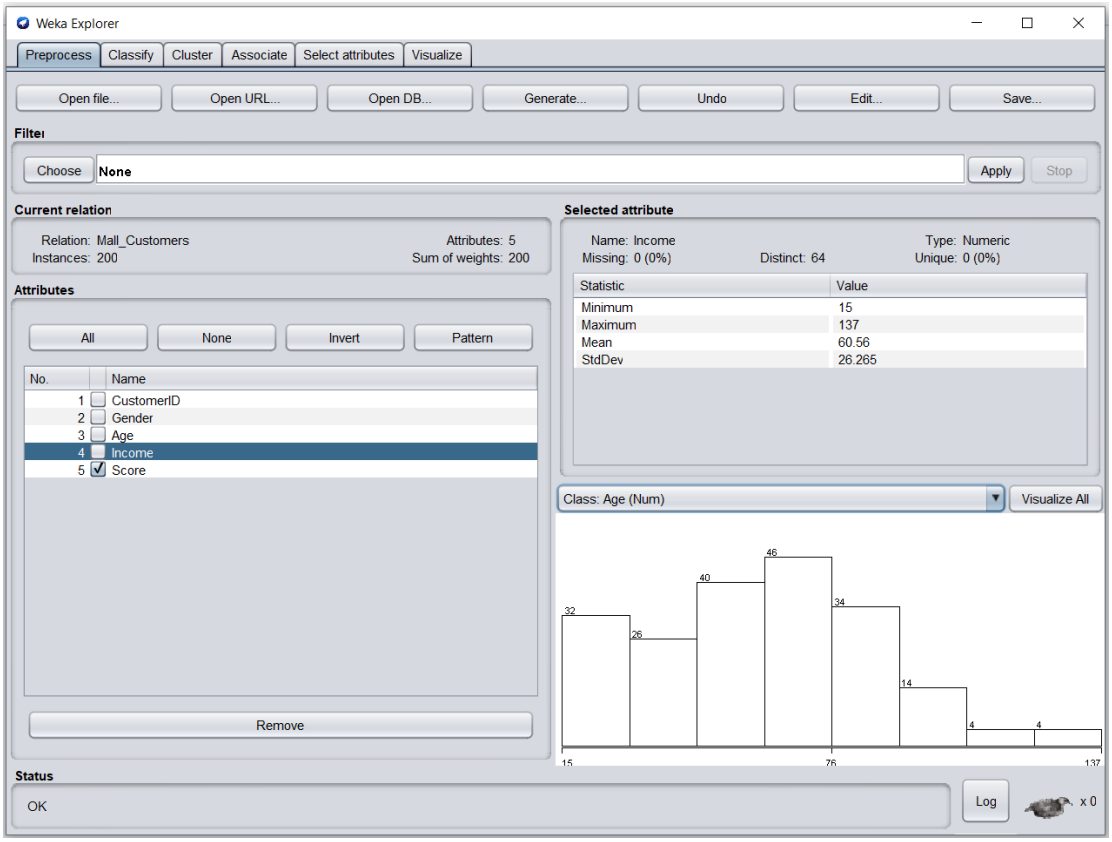
1. Build Data Warehouse, install and Explore WEKA.

Weka is a data mining visualization tool which contains collection of machine learning algorithms for data mining tasks.

Features of Weka:

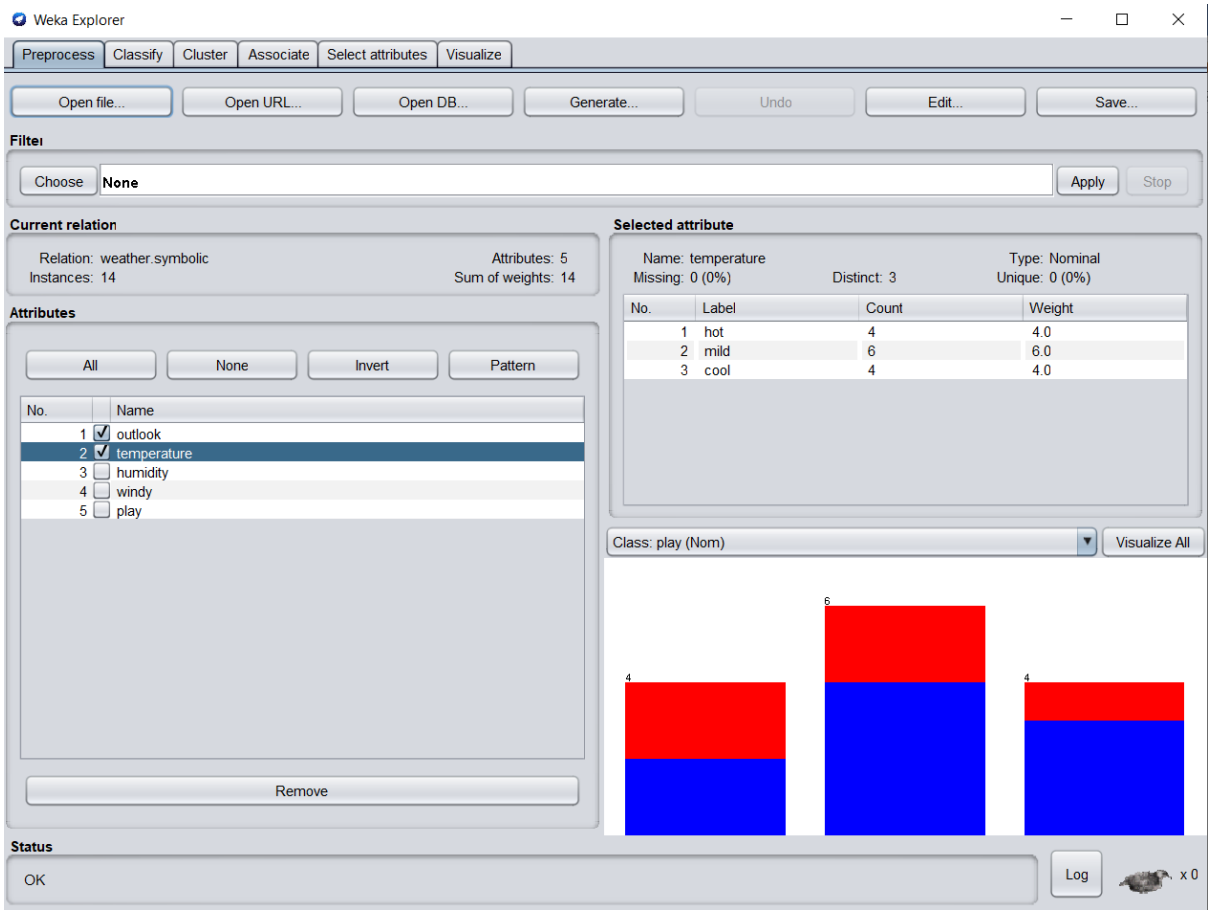
- **Data Preprocessing:** It is cleaning of data while data gathering and selection phase. It removes/adds default value to missing fields and resolve conflicts.
- **Data Classification and Prediction:** It classifies data based on relationship between things and predicts data label. For e.g., A Bank, based on available data of loan, classifies and predicts customer label 'risky' or 'safe'.
- **Clustering:** Group of related data into cluster, used to discover distinct group. For e.g., We have data of weather and based on that we want to decide whether to play outside or not, in such case, using Weka tool we can visualize overall data and can make decision according to the charts.

Below is a simple example of preprocess using Weka.



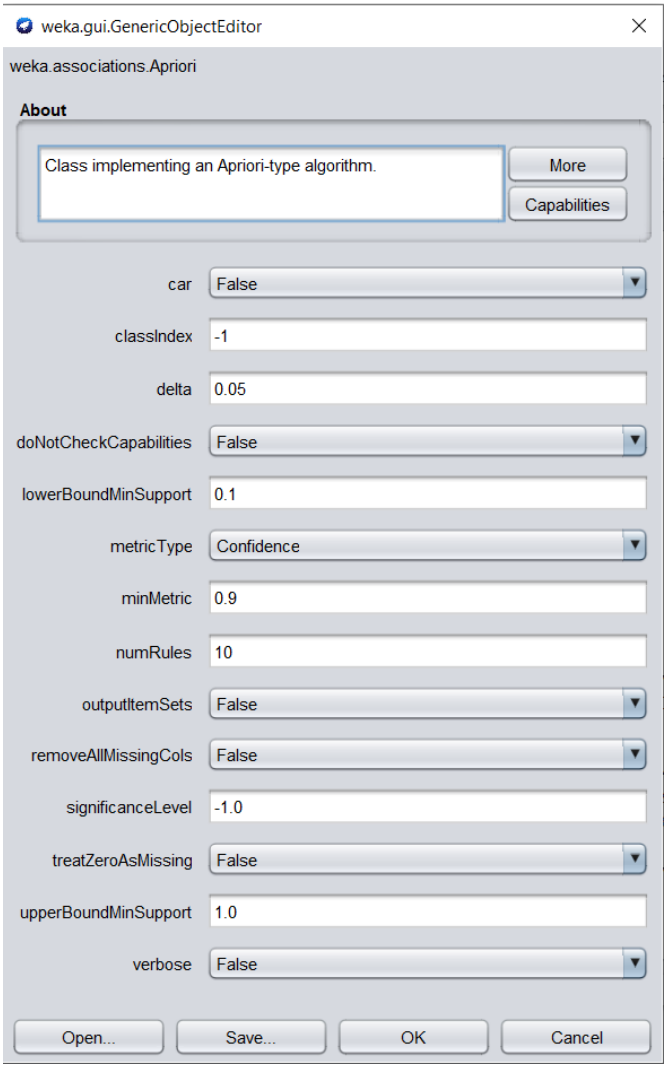
2. Perform data pre-processing tasks and Demonstrate performing association rule mining on datasets using WEKA.

Preprocessing:



Association:

We can set the parameters using the below interface.



Generating association rules-