

## DWDM R PROGRAMMING-PRACTICALS

### 1.List of Programs:

**1The intervals and corresponding frequencies are as follows. age frequency**

**1-5. 200**

**5-15 450**

**15-20 300**

**20-50 1500**

**50-80 700**

**80-110 44**

**Compute an approximate median value for the data**

#### CODING-

```
intervals <- c("1-5", "5-15", "15-20", "20-50", "50-80", "80-110")
frequencies <- c(200, 450, 300, 1500, 700, 44)
cumulative_freq <- cumsum(frequencies)
median_interval_index <- which(cumulative_freq >= sum(frequencies)/2)[1]
lower_bound <- as.numeric(strsplit(intervals[median_interval_index], "-")[[1]][1])
upper_bound <- as.numeric(strsplit(intervals[median_interval_index], "-")[[1]][2])
cumulative_freq_before <- cumulative_freq[median_interval_index] -
frequencies[median_interval_index]
frequency_median <- frequencies[median_interval_index]
width <- upper_bound - lower_bound
median_value <- lower_bound + ((sum(frequencies)/2 - cumulative_freq_before) /
frequency_median) * width
print(paste("Approximate Median Value:", median_value))
```

#### OUTPUT-

```
[1] "Approximate Median Value: 32.94"
```

**2. Suppose that the data for analysis includes the attribute age. The age values for the data tuples are (in increasing order) 13, 15, 16, 16, 19, 20, 20, 21, 22, 22, 25, 25, 25, 25, 30, 33, 33, 35, 35, 35, 35, 36, 40, 45, 46, 52, 70.**

**(a) What is the mean of the data? What is the median?**

**(b) What is the mode of the data? Comment on the data's modality (i.e., bimodal, trimodal, etc.).**

**(c) What is the midrange of the data?**

**(d) Can you find (roughly) the first quartile (Q1) and the third quartile (Q3) of the data?**

**Coding:**

**#2a**

```
x<-c(13,15,16,16,19,20,20,21,22,22,25,25,25,25,30,33,33,35,35,35,35,36,40,45,46,52,70)
```

```
#mean
```

```
mean(x)
```

```
#median
```

```
median(x)
```

**output:**

```
mean(x)
```

```
[1] 29.96296
```

```
> #median
```

```
> median(x)
```

```
[1] 25
```

**CODING FOR 2b-**

**#2b**

```
#mode
```

```
MultipleModes <- function(x) {
```

```
  uniqx <- unique(x)
```

```
  freq_table <- tabulate(match(x, uniqx))
```

```
  modes <- uniqx[freq_table == max(freq_table)]
```

```
  modes
```

```
}
```

```
age_values <- c(13, 15, 16, 16, 19, 20, 20, 21, 22, 22, 25, 25, 25, 25, 30, 33, 33, 35, 35, 35, 35, 36, 40, 45, 46, 52, 70)
```

```
multiple_modes <- MultipleModes(age_values)
```

```
print(multiple_modes)
```

**output:**

25 35

### **CODING FOR 2c-**

```
#midrange
```

```
c) age_values <- c(13, 15, 16, 16, 19, 20, 20, 21, 22, 22, 25, 25, 25, 25, 30, 33, 33, 35, 35, 35, 35, 36, 40, 45, 46, 52, 70)
```

```
X<-(min(age_values)+max(age_values))/2
```

```
print(X)
```

OUTPUT-

41.5

### **CODING FOR 2d-**

```
d) #quartile
```

```
age_values <- c(13, 15, 16, 16, 19, 20, 20, 21, 22, 22, 25, 25, 25, 25, 30, 33, 33, 35, 35, 35, 35, 36, 40, 45, 46, 52, 70)
```

```
quantile(age_values)
```

**output:** 0% 25% 50% 75% 100%

13.0 20.5 25.0 35.0 70.0

## **3.Data Preprocessing :Reduction and Transformation**

**Use the two methods below to normalize the following group of data:**  
**200, 300, 400, 600, 1000 (a) min-max normalization by setting min = 0 and max = 1 (b) z-score normalization**

**Coding:**

```
#3a
```

```
data <- c(200, 300, 400, 600, 1000)
```

```
min<-min(data)
```

```
max<-max(data)
for (i in data)
{
  result1=i-min
  result2=max-min
  result3=result1/result2
  print(result3)
}
```

**OUTPUT:**

```
[1] 0
[1] 0.125
[1] 0.25
[1] 0.5
[1] 1
```

**#3b**

```
data <- c(200, 300, 400, 600, 1000)
mean1<-mean(data)
deviation<-sd(data)
for (i in data)
{
  result1=i-mean1
  result2=result1/deviation
  print(result2)
}
```

**OUTPUT:**

```
[1] -0.9486833
[1] -0.6324555
[1] -0.3162278
[1] 0.3162278
[1] 1.581139
```

**4.Data:11,13,13,15,15,16,19,20,20,20,21,21,22,23,24,30,40,45,45,45,71,72,73,75**

**a) Smoothing by bin mean**

**b) Smoothing by bin median**

**c) Smoothing by bin boundaries**

**CODING-**

```
#binning
data <- c(11, 13, 13, 15, 15, 16, 19, 20, 20, 20, 21, 21, 22, 23, 24, 30, 40, 45, 45, 45, 71, 72, 73, 75)
range=6
bin1=c()
bin2=c()
bin3=c()
bin4=c()
for(i in data[1:range]){
  bin1=append(bin1,i)
}
range1=range+1
range2=range*2
for(j in data[range1:range2])
{
  bin2=append(bin2,j)
}
range3=range2+1
range4=range*3
for(k in data[range3:range4])
{
  bin3=append(bin3,k)
}
range5=range4+1
```

```
range6=range*4
for(l in data[range5:range6]){
  bin4=append(bin4,l)
}
```

#### **#4a**

```
mean(bin1)
mean(bin2)
mean(bin3)
mean(bin4)
```

#### **#4b**

```
median(bin1)
median(bin2)
median(bin3)
median(bin4)
```

#### **OUTPUT:**

#### **#4a**

```
> mean(bin1)
[1] 13.83333
> mean(bin2)
[1] 20.16667
> mean(bin3)
[1] 30.66667
> mean(bin4)
[1] 63.5
```

>

#### **#4b**

```
> median(bin1)
[1] 14
> median(bin2)
[1] 20
> median(bin3)
```

```
[1] 27
```

```
> median(bin4)
```

```
[1] 71.5
```

**5) 5. Suppose that a hospital tested the age and body fat data for 18 randomly selected adults with the following results:**

<i>age</i>	23	23	27	27	39	41	47	49	50
<i>%fat</i>	9.5	26.5	7.8	17.8	31.4	25.9	27.4	27.2	31.2
<i>age</i>	52	54	54	56	57	58	58	60	61
<i>%fat</i>	34.6	42.5	28.8	33.4	30.2	34.1	32.9	41.2	35.7

#### **CODING-**

```
age <- c(23,23,27,27,39,41,47,49,50,52,54,54,56,57,58,58,60,61)
```

```
body_fat_percent <-
```

```
c(9.5,26.5,7.8,17.8,31.4,25.9,27.4,27.2,31.2,34.6,42.5,28.8,33.4,30.2,34.1,32.9,41.2,35.7)
```

#### **#5.a**

```
mean(age)
```

```
mean(body_fat_percent)
```

```
median(age)
```

```
median(body_fat_percent)
```

```
sd(age)
```

```
sd(body_fat_percent)
```

#### **#5.b**

```
#create dataframe
```

```
df<-data.frame(age,body_fat_percent)
```

```
#box plot
```

```
boxplot(df)
```

```
#scatter plot
```

```
plot(df)
#qq plot
qqnorm(age)
qqline(age)
qqnorm(body_fat_percent)
qqline(body_fat_percent)
```

#### **OUTPUT-**

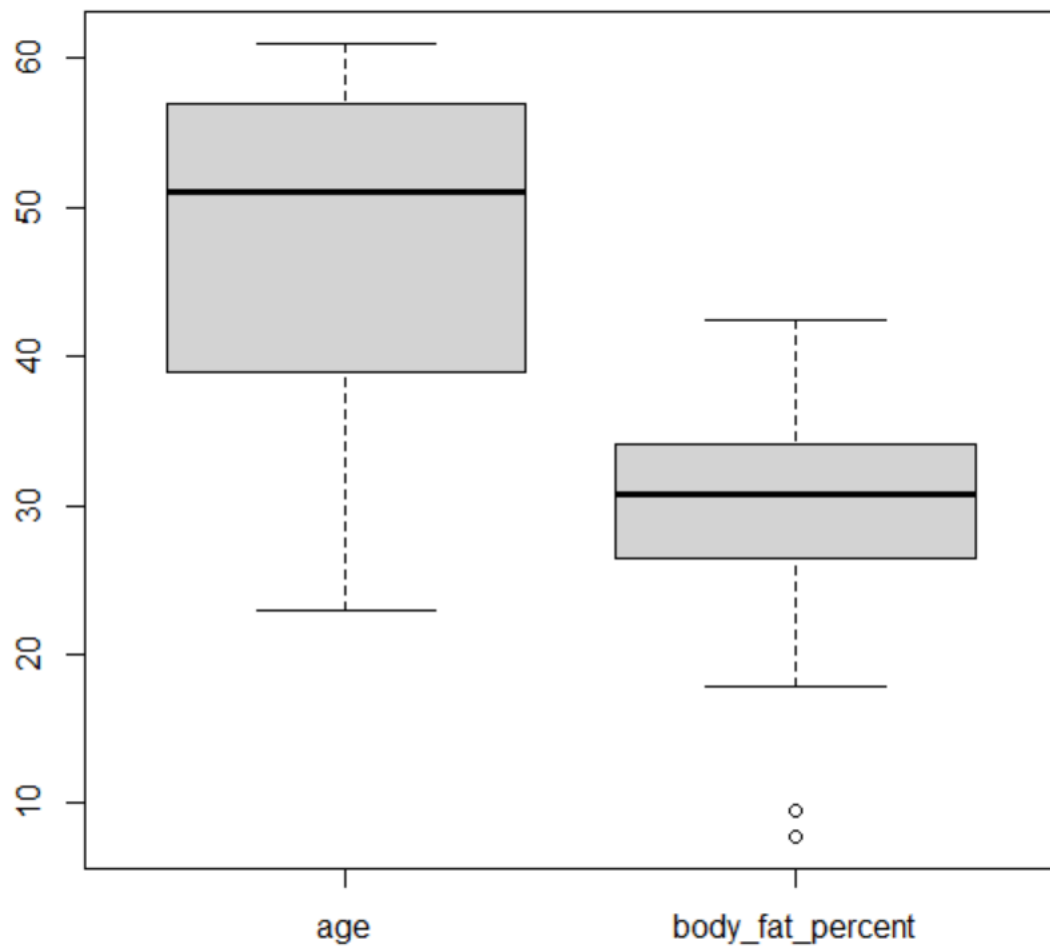
##### **#5a**

```
> mean(age)
[1] 46.44444
> mean(body_fat_percent)
[1] 28.78333
> median(age)
[1] 51
> median(body_fat_percent)
[1] 30.7
> sd(age)
[1] 13.21862
> sd(body_fat_percent)
[1] 9.254395
```

##### **#5.b**

BOXPLOT-

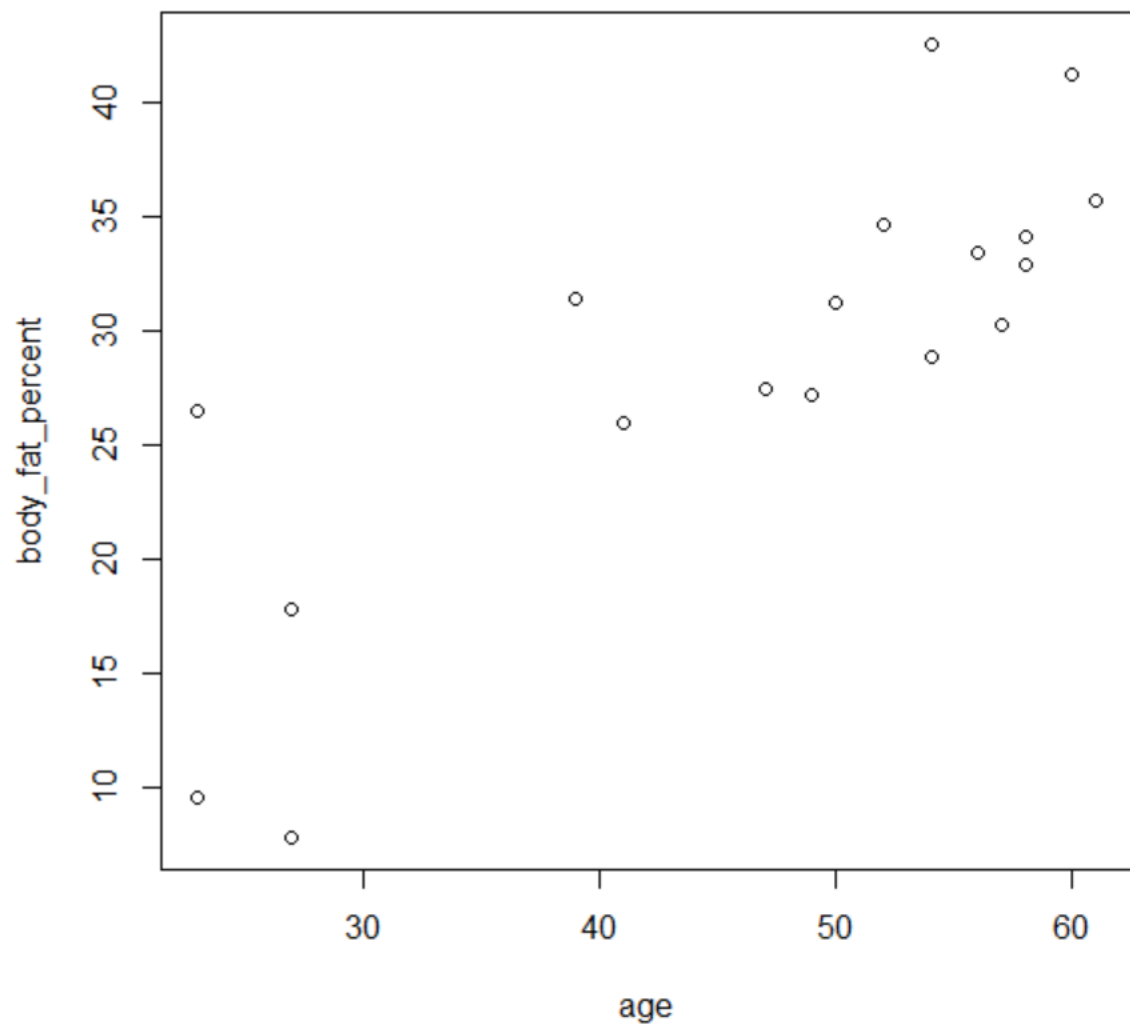




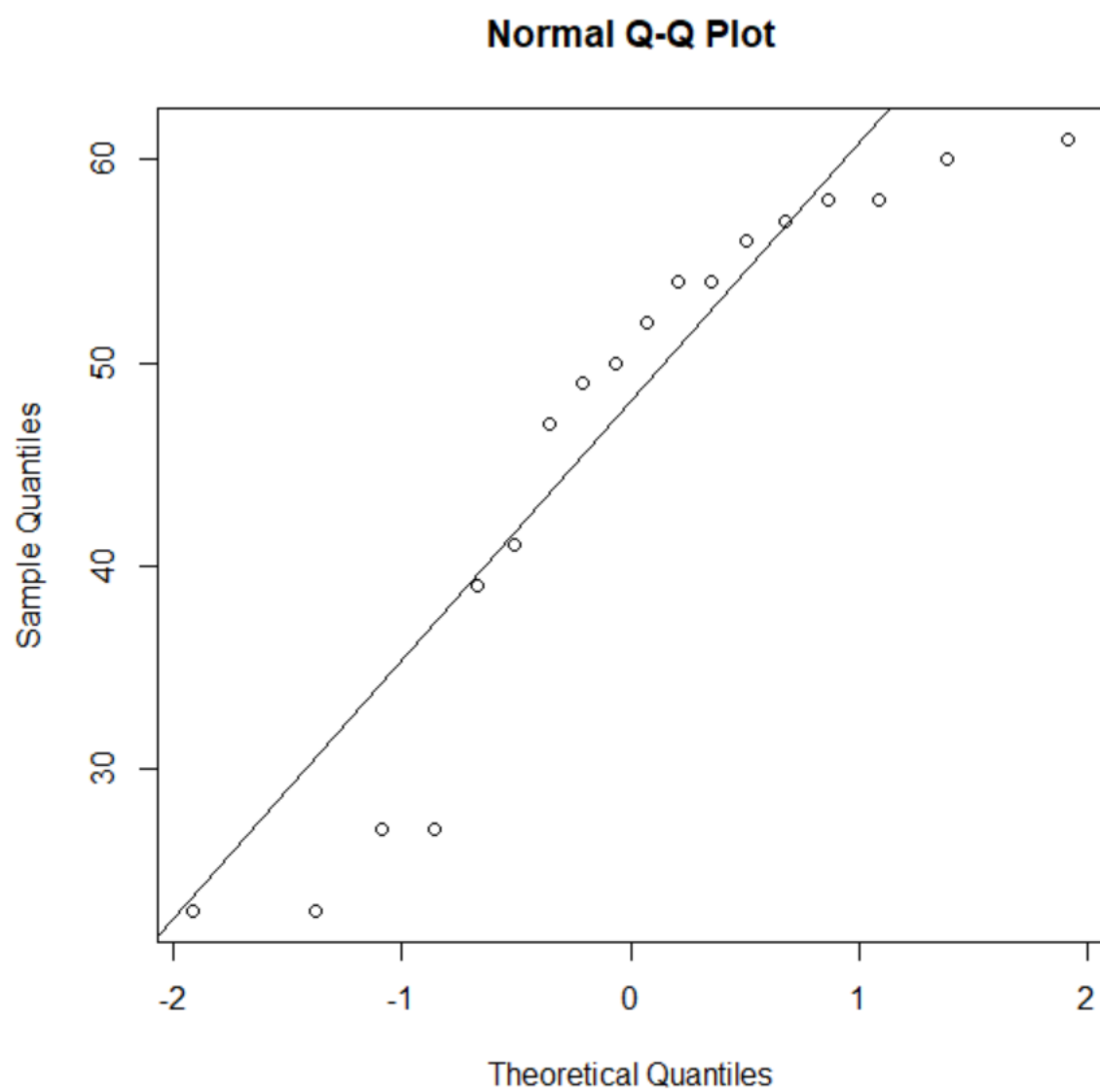
SCATTER PLOT-

#5c

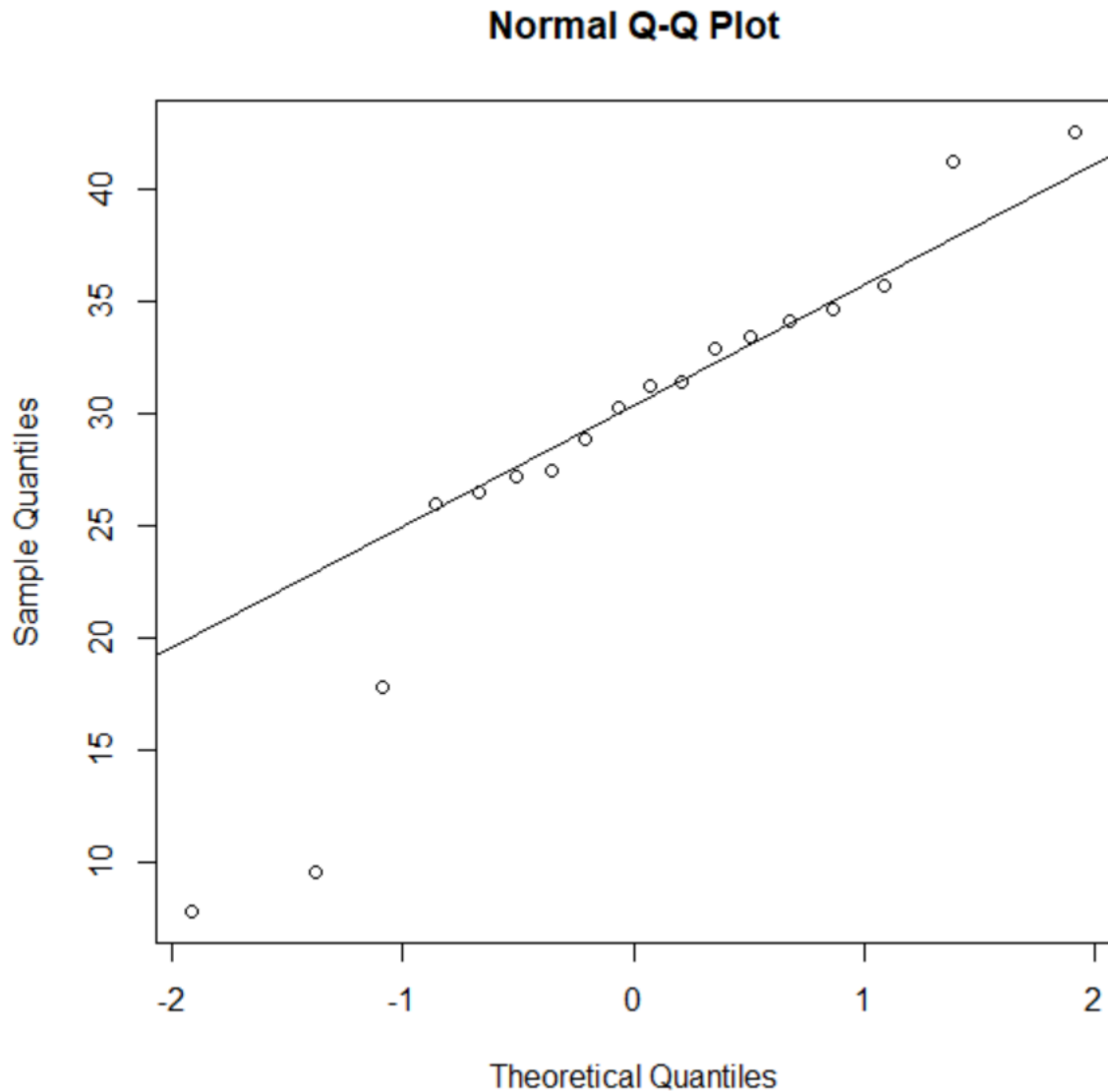
QQ



QQ PLOT FOR AGE-



QQPLOT FOR BODY FAT PERCENT-



6. Suppose that a hospital tested the age and body fat data for 18 randomly selected adults with the following results:

- (i) Use min-max normalization to transform the value 35 for age onto the range [0.0, 1.0].
- (ii) Use z-score normalization to transform the value 35 for age, where the standard deviation of age is 12.94 years.
- (iii) Use normalization by decimal scaling to transform the value 35 for age. Perform the above functions using R – tool

#### **CODING-**

```
age <- c(23,23,27,27,39,41,47,49,50,52,54,54,56,57,58,58,60,61)
new_age<-c()
for(i in age){
```

```
if(i<=35){  
  new_age=append(new_age,i)  
}  
}  
print(new_age)
```

#### **#6a**

#min max normalization

```
min<-min(new_age)  
max<-max(new_age)  
for (i in new_age)  
{  
  result1=i-min  
  result2=max-min  
  result3=result1/result2  
  print(result3)  
}
```

#### **#6b**

#z score normalization

```
mean1<-mean(new_age)  
for (i in new_age)  
{  
  result1=i-mean1  
  result2=result1/12.94  
  print(result2)  
}
```

#### **#6c**

#decimal scaling

```
n=200  
j=nchar(y)  
scaling=n/10^j  
print(scaling)
```

## OUTPUT-

6.a MIN MAX NORMALIZATION

[1] 0

[1] 0

[1] 1

[1] 1

6.b Z SCORE NORMALIZATION

[1] -0.8660254

[1] -0.8660254

[1] 0.8660254

[1] 0.8660254

6.c DECIMAL SCALING

[1] 0.2

**7.The following values are the number of pencils available in the different boxes. Create a vector and find out the mean, median and mode values of set of pencils in the given data.**

Box1	Box2	Box3	Box4	Box5	Box6	Box7	Box8	Box9	Box 10
9	25	23	12	11	6	7	8	9	10

## CODING-

```
box_no=c("box1","box2","box3","box4","box5","box6","box7","box8","box9","box10")
```

```
pencil=c(9,25,23,12,11,6,7,8,9,10)
```

```
df<-data.frame(box_no,pencil)
```

```
#dataframe
```

```
print(df)
```

```
#mean
```

```
mean(pencil)
```

```
#median
median(pencil)

#mode
mode=names(which.max(table(pencil)))
print(mode)
```

### OUTPUT-

```
> data.frame(box_NO,pencil)
```

```
  box_NO pencil
```

```
1 box1    9
```

```
2 box2   25
```

```
3 box3   23
```

```
4 box4   12
```

```
5 box5   11
```

```
6 box6    6
```

```
7 box7    7
```

```
8 box8    8
```

```
9 box9    9
```

```
10 box10  10
```

```
> mean(pencil)
```

```
[1] 12
```

```
> median(pencil)
```

```
[1] 9.5
```

```
> print(mode)
```

```
[1] "9"
```

**8. the following table would be plotted as (x,y) points, with the first column being the x values as number of mobile phones sold and the second column being the y values as money. To use the scatter plot for how many mobile phones sold.**

**x :4 1 5 7 10 2 50 25 90 36**

**y :12 5 13 19 31 7 153 72 275 110**

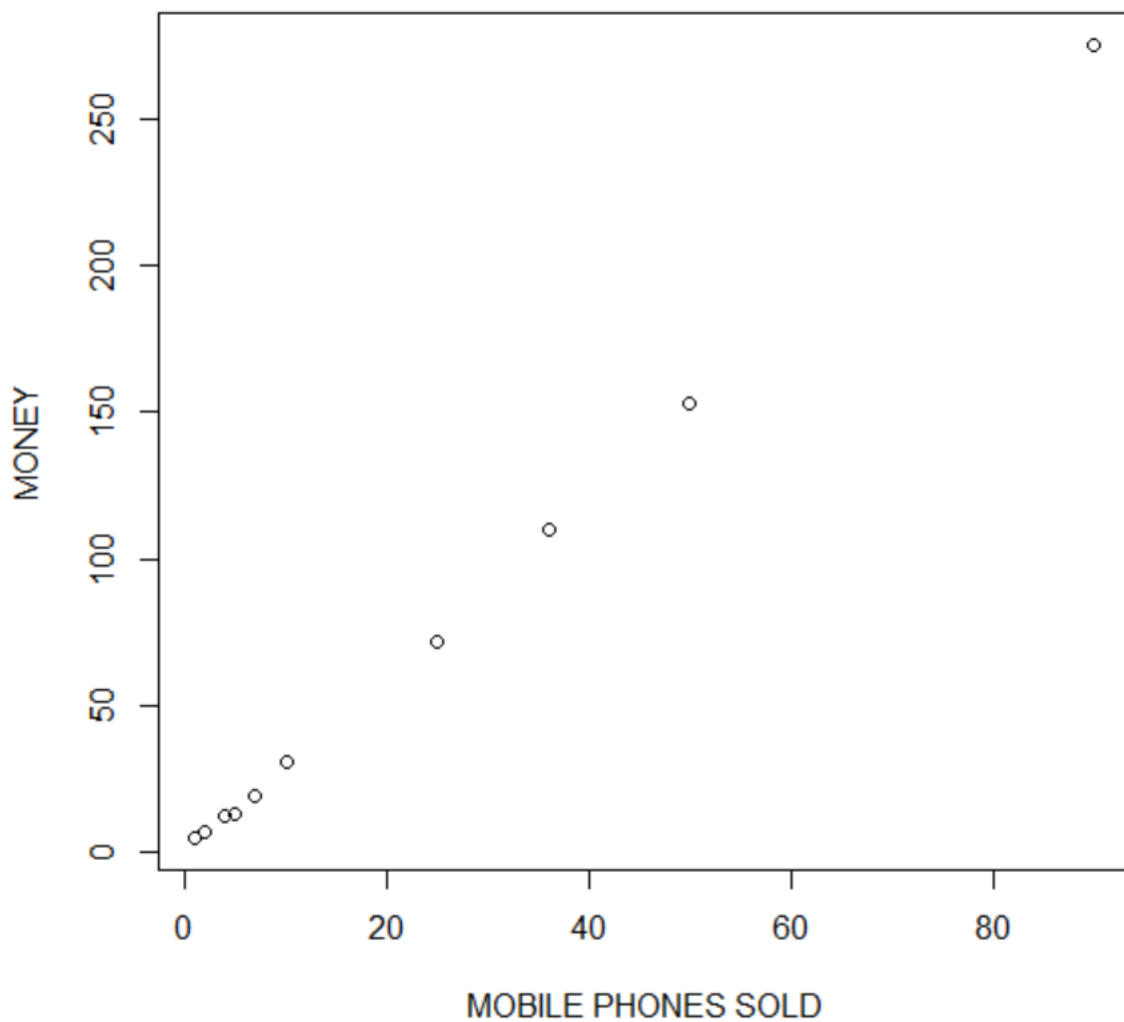
CODING-

```
x<-c(4, 1, 5, 7, 10, 2, 50, 25, 90, 36)
```

```
y<-c(12,5, 13, 19, 31, 7, 153, 72, 275, 110)
```

```
plot(x,y,xlab='MOBILE PHONES SOLD',ylab='MONEY')
```

OUTPUT-





**9. Implement of the R script using marks scored by a student in his model exam has been sorted as follows: 55, 60, 71, 63, 55, 65, 50, 55,58,59,61,63,65,67,71,72,75. Partition them into three bins by each of the following methods. Plot the data points using histogram.**

**(a) equal-frequency (equi-depth) partitioning (b) equal-width partitioning**

**CODING-**

```
marks<-c(55, 60, 71, 63, 55, 65, 50, 55,58,59,61,63,65,67,71,72,75)
```

```
binning1=c()
```

```
binning2=c()
```

```
binning3=c()
```

```
class=6
```

```
#binning partition
```

```
for(a in marks[1:class]){
```

```
  binning1=append(binning1,a)
```

```
}
```

```
range1=range+1
```

```
range2=range*2
```

```
for(b in marks[range1:range2])
```

```
{
```

```
  binning2=append(binning2,b)
```

```
}
```

```
range3=range2+1
```

```
range4=range*3
```

```
for(c in marks[range3:range4])
```

```
{
```

```
    binning3=append(binning3,c)
}
```

```
print(binning1)
```

```
print(binning2)
```

```
print(binning3)
```

### **#histogram**

```
hist(binning1)
```

```
hist(binning2)
```

```
hist(binning3)
```

### **#9a**

#### **#equal-frequency**

```
freq=length(marks)/range
```

```
print(freq)
```

### **#9b**

#### **#equal-width**

```
min<-min(marks)
```

```
max<-max(marks)
```

```
result<-max-min
```

```
width<-result/range
```

```
cat("width is",width)
```

```
bin1=width+min
```

```
print(bin1)
```

```
bin2=2*width+min
```

```
print(bin2)
```

```
bin3=3*width+min
```

```
print(bin3)
```

OUTPUT-

```
> print(binning1)
```

```
[1] 55 60 71 63 55 65
```

```
> print(binning2)
```

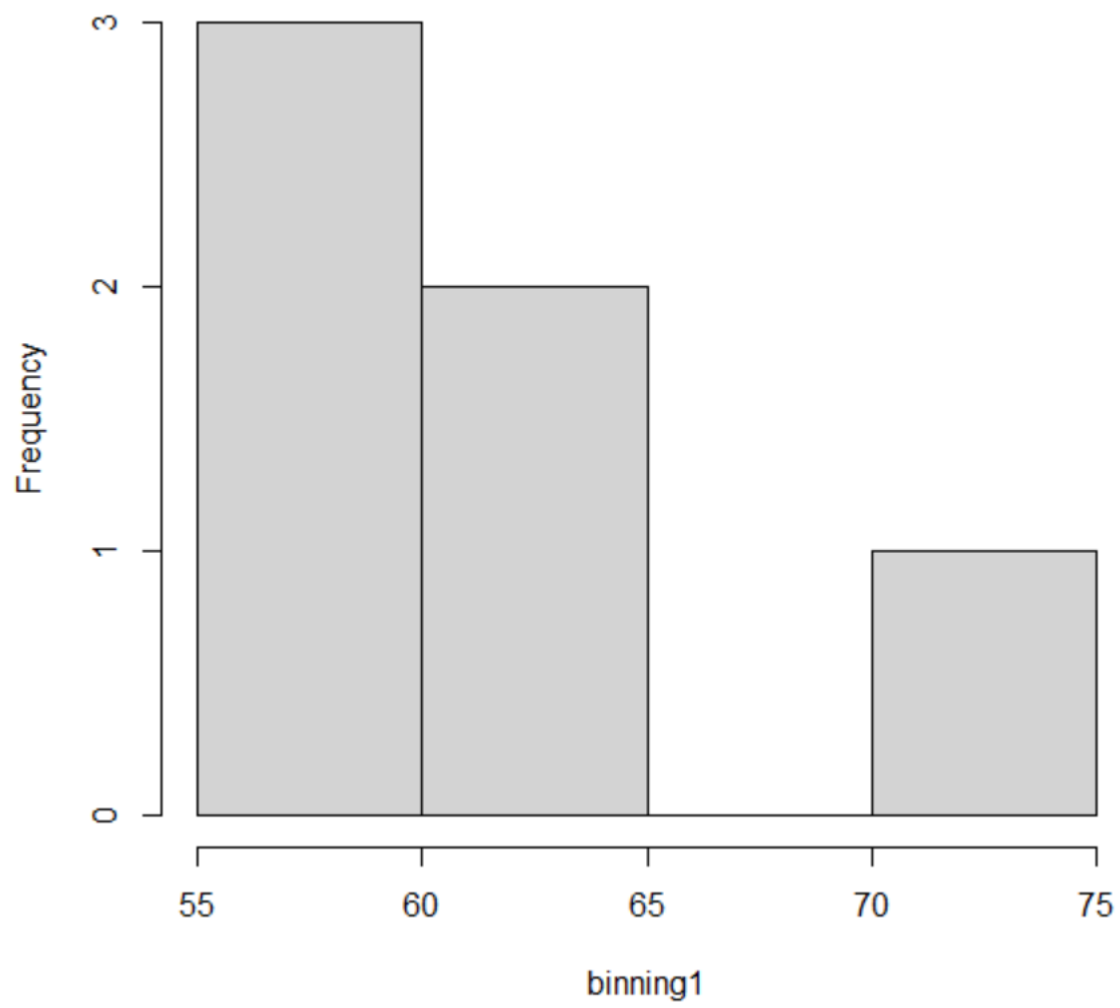
```
[1] 50 55 58 59 61 63
```

```
> print(binning3)
```

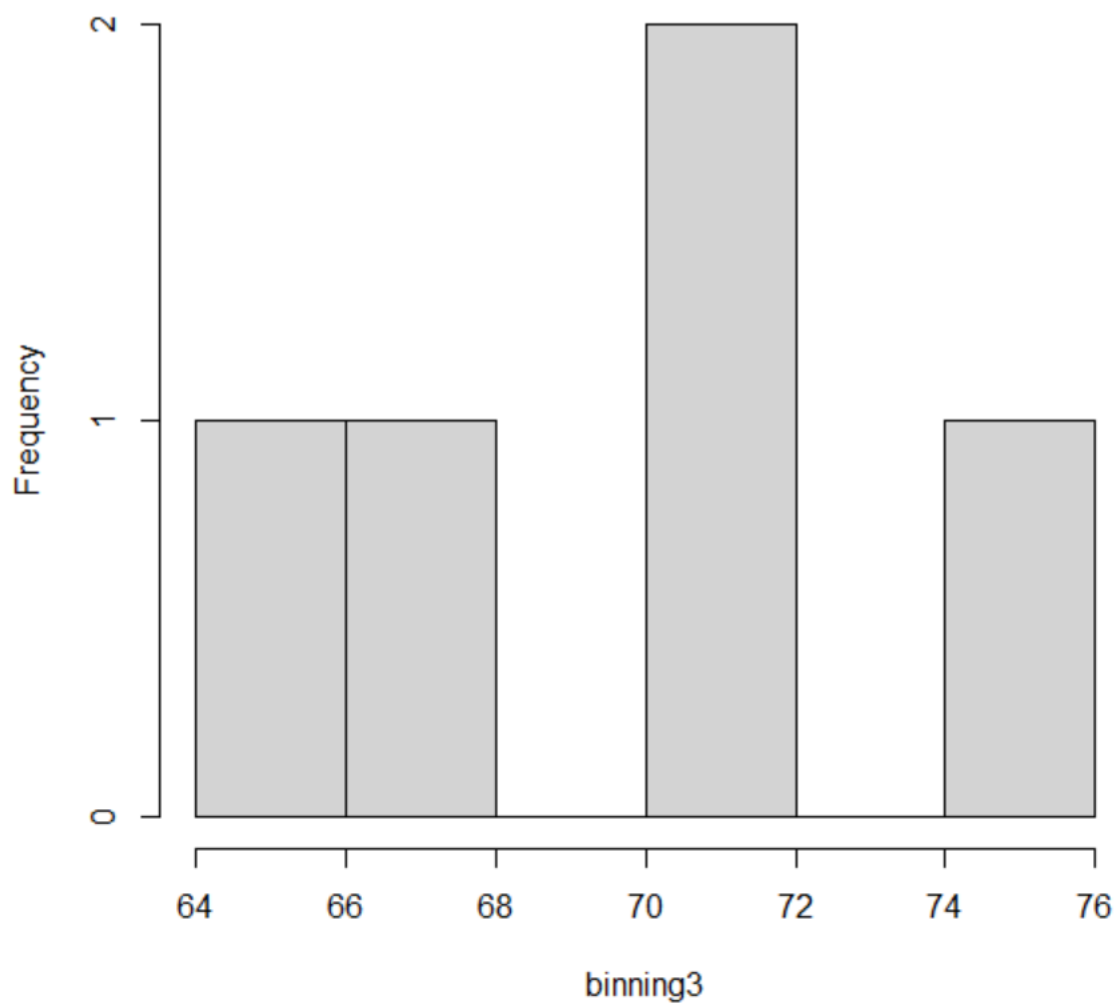
```
[1] 65 67 71 72 75 NA
```

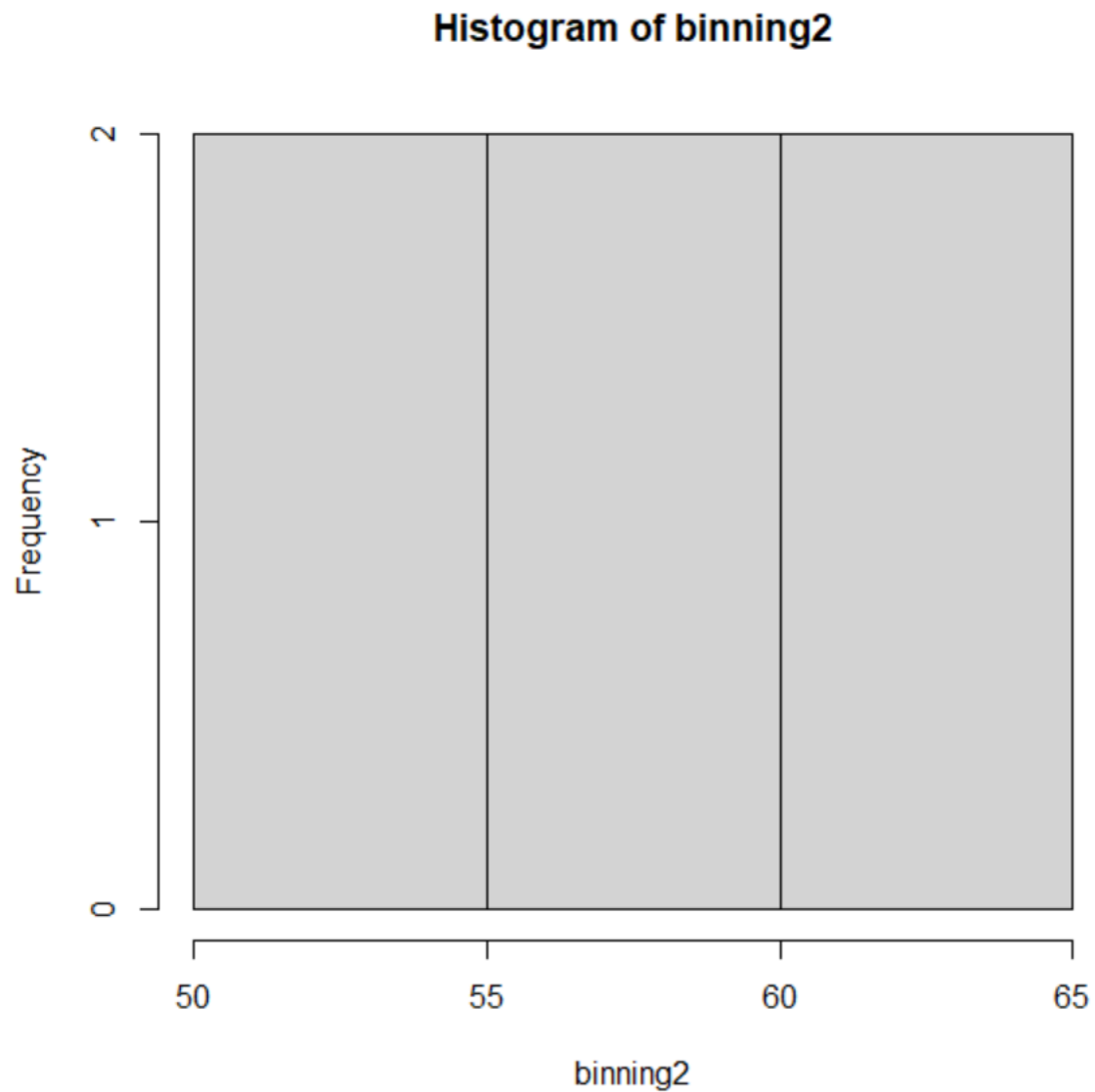
HISTOGRAM-

**Histogram of binning1**



Histogram of binning3





**#9a**

#equal frequency

```
> print(freq)
```

```
[1] 2.833333
```

**#9b**

#equal width

```
width is 4.166667> bin1=width+min
```

```
> print(bin1)
```

```
[1] 54.16667
```

```
> bin2=2*width+min
```

```
> print(bin2)
```

```
[1] 58.33333
```

```
> bin3=3*width+min
```

```
> print(bin3)
```

```
[1] 62.5
```

**10. Suppose that the speed car is mentioned in different driving style.**

**Regular 78.3 81.8 82 74.2 83.4 84.5 82.9 77.5 80.9 70.6 Speed**

**Calculate the Inter quantile and standard deviation of the given data.**

**CODING-**

```
speed<-c(78.3 ,81.8 ,82 ,74.2 ,83.4 ,84.5 ,82.9 ,77.5 ,80.9 ,70.6 )
```

```
#interquartile
```

```
IQR(speed)
```

```
#standard deviation
```

```
sd(speed)
```

**OUTPUT-**

```
> IQR(speed)
```

```
[1] 4.975
```

```
> sd(speed)
```

```
[1] 4.445835
```

**11. Suppose that the data for analysis includes the attribute age.**

**The age values for the data tuples are (in increasing order) 13, 15, 16, 16, 19, 20, 20, 21, 22, 22, 25, 25, 25, 25, 30, 33, 33, 35, 35, 35, 35, 36, 40, 45, 46, 52, 70.**

**Can you find (roughly) the first quartile (Q1) and the third quartile (Q3) of the data?**

**CODING-x**

OUTPUT-

```
> quantile(marks)
```

```
0% 25% 50% 75% 100%
```

```
13.0 20.5 25.0 35.0 70.0
```

12.Covariance and correlation

Children of three ages are asked to indicate their preference for three photographs of adults. Do the data suggest that there is a significant relationship between age and photograph preference? What is wrong with this study?

Age of child	Photograph:		
	A	B	C
5-6 years:	18	22	20
7-8 years:	2	28	40
9-10 years:	20	10	40

(i) Use `cov()` to calculate the sample covariance between B and C.

(ii) Use another call to `cov()` to calculate the sample covariance matrix for the preferences.

(iii) Use `cor()` to calculate the sample correlation between B and C.

(iv) Use another call to `cor()` to calculate the sample correlation matrix for the preferences.

CODE:

```
(i) b<-c(22, 28, 10)
```

```
c<-c(20, 40, 40)
```

```
cov(b,c)
```

```
(ii) a<-c(18, 2, 20)
```

```
b<-c(22, 28, 10)
```

```
c<-c(20, 40, 40)
```

```
pre<-cbind(a,b,c)
```



```
cov(pre)
```

```
(iii).b<-c(22, 28, 10)
c<-c(20, 40, 40)
cor(b,c)
```

```
(iv)a<-c(18, 2, 20)
b<-c(22, 28, 10)
c<-c(20, 40, 40)
pre<-cbind(a,b,c)
cor(pre)
```

OUTPUT:

```
> b<-c(22, 28, 10)
> c<-c(20, 40, 40)
> cov(b,c)
[1] -20
> a<-c(18, 2, 20)
> b<-c(22, 28, 10)
> c<-c(20, 40, 40)
> pre<-cbind(a,b,c)
> cov(pre)
      a      b      c
a 97.33333 -74 -46.66667
b -74.00000 84 -20.00000
c -46.66667 -20 133.33333
> b<-c(22, 28, 10)
> c<-c(20, 40, 40)
> cor(b,c)
[1] -0.1889822
> a<-c(18, 2, 20)
> b<-c(22, 28, 10)
> c<-c(20, 40, 40)
> pre<-cbind(a,b,c)
> cor(pre)
      a      b      c
a 1.0000000 -0.8183918 -0.4096440
b -0.8183918 1.0000000 -0.1889822
c -0.4096440 -0.1889822 1.0000000
```

13. Imagine that you have selected data from the All Electronics data warehouse for analysis. The data set will be huge! The following data are a list of All Electronics prices for commonly sold items (rounded to the nearest dollar). The numbers have been sorted: 1, 1, 5, 5, 5, 5, 5, 8, 8, 10, 10, 10, 10, 12, 14, 14, 14, 15, 15, 15, 15, 15, 15, 18, 18, 18, 18, 18,

18, 18, 18, 20, 20, 20, 20, 20, 20, 20, 21, 21, 21, 21, 25, 25, 25, 25, 25, 28, 28, 30,
---

30, 30.
---------

(i) Partition the dataset using an equal-frequency partitioning method with bin equal to 3 (ii) apply data smoothing using bin means and bin boundary.  
(iii) Plot Histogram for the above frequency division

CODE:

```
data<-c(1, 1, 5, 5, 5, 5, 8, 8, 10, 10, 10, 10, 12, 14, 14, 14, 15, 15, 15, 15, 15, 15, 18, 18, 18, 18, 18,
18, 18, 18, 20, 20, 20, 20, 20, 20, 20, 20, 21, 21, 21, 21, 25, 25, 25, 25, 25, 28, 28, 30, 30, 30)
bin<-length(data)/3
bins<-cut(data, breaks = c(-Inf, quantile(data, probs = seq(0,1, 1/3)),Inf), include.lowest = TRUE)
tapply(data,bins,mean)
tapply(data, bins,function(x) c(min(x),max(x)))

hist(data,breaks = 3, main = "Histogram", xlab = "prices")
```

OUTPUT:

```

R 4.2.2 > console
> data<-c(1, 1, 5, 5, 5, 5, 5, 8, 8, 10, 10, 10, 10, 12, 14, 14, 14, 15, 15, 15, 15, 15, 15, 18, 1
8, 18, 18, 18, 18, 18, 20, 20, 20, 20, 20, 20, 20, 21, 21, 21, 21, 25, 25, 25, 25, 25, 28, 28,
30, 30, 30)
> bin<-length(data)/3
> bins<-cut(data, breaks = c(-Inf, quantile(data, probs = seq(0,1, 1/3)),Inf), include.lowest = TRU
E)
> tapply(data,bins,mean)
[-Inf,1] (1,15] (15,20] (20,30] (30, Inf]
1.00000 10.71429 18.93333 25.35714 NA
> tapply(data, bins,function(x) c(min(x),max(x)))
$[-Inf,1]
[1] 1 1

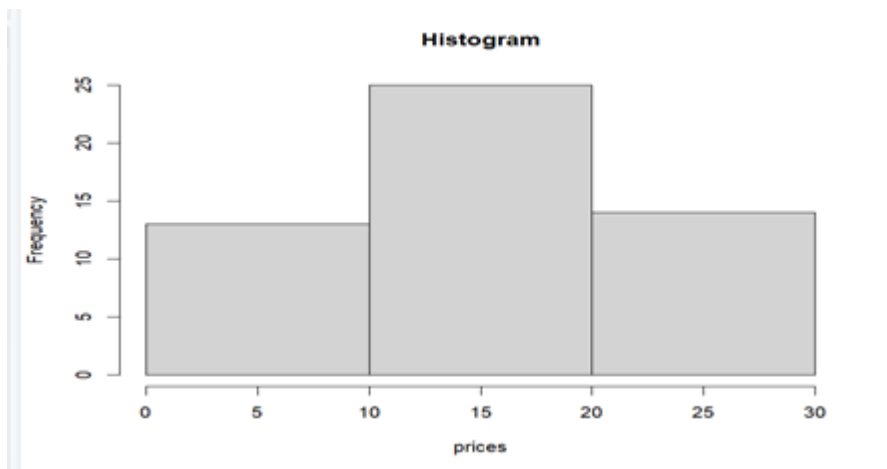
$ (1,15]
[1] 5 15

$ (15,20]
[1] 18 20

$ (20,30]
[1] 21 30

$ (30, Inf]
NULL
>
> hist(data,breaks = 3, main = "Histogram", xlab = "prices")
>

```



14. Two Maths teachers are comparing how their Year 9 classes performed in the end of year exams. Their results are as follows:

Class A: 76, 35, 47, 64, 95, 66, 89, 36, 84

Class B: 51, 56, 84, 60, 59, 70, 63, 66, 50

(i) Find which class had scored higher mean, median and range.

(ii) Plot above in boxplot and give the inferences

Class B: 51, 56, 84, 60, 59, 70, 63, 66, 50

CODE:

```
A <- c(76, 35, 47, 64, 95, 66, 89, 36, 84)
```

```
B <- c(51, 56, 84, 60, 59, 70, 63, 66, 50)
```

```
mean_A <- mean(A)
```

```
median_A <- median(A)
```

```
range_A <- max(A) - min(A)
```

```
mean_B <- mean(B)
```

```
median_B <- median(B)
```

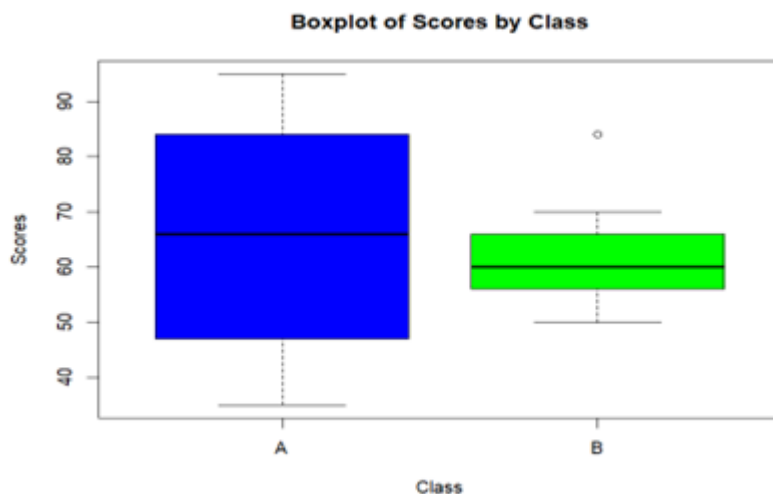
```
range_B <- max(B) - min(B)
```

```
combined_data <- data.frame(Class = c(rep("A", length(A)), rep("B", length(B))), Score = c(A, B))  
boxplot(Score ~ Class, data = combined_data, col = c("blue", "green"), xlab = "Class", ylab = "Scores",  
main = "Boxplot of Scores by Class")  
(II)
```

```
combined_data <- data.frame(Class = c(rep("A", length(A)), rep("B", length(B))), Score = c(A, B))
```

```
boxplot(Score ~ Class, data = combined_data, col = c("blue", "green"), xlab = "Class", ylab = "Scores",  
main = "Boxplot of Scores by Class")
```

OUTPUT:



15. Let us consider one example to make the calculation method clear. Assume that the minimum and maximum values for the feature  $F$  are \$50,000 and \$100,000 correspondingly. It needs to range  $F$  from 0 to 1. In accordance with min-max normalization,  $v = \$80$ ,

b) Use the two methods below to normalize the following group of data: 200, 300, 400, 600, 1000

(a) min-max normalization by setting min = 0 and max = 1

(b) z-score normalization

CODE:

```
data <- c(200, 300, 400, 600, 1000)
```

```
min_value <- 50000
```

```
max_value <- 100000
```

```
v <- 80
```

```
min_max_normalized <- (v - min_value) / (max_value - min_value)
```

```
min_max_normalized
```

```
mean_value <- mean(data)
```

```
standard_deviation <- sd(data)
```

```
z_score_normalized <- (v - mean_value) / standard_deviation
z_score_normalized
```

OUTPUT:

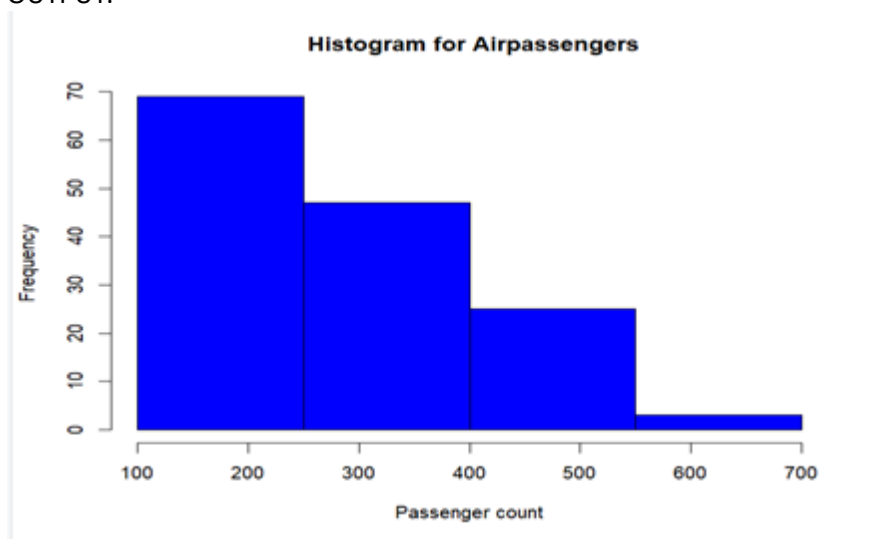
```
> data <- c(200, 300, 400, 600, 1000)
>
> min_value <- 50000
> max_value <- 100000
> v <- 80
> min_max_normalized <- (v - min_value) / (max_value - min_value)
> min_max_normalized
[1] -0.9984
> mean_value <- mean(data)
> standard_deviation <- sd(data)
> z_score_normalized <- (v - mean_value) / standard_deviation
> z_score_normalized
[1] -1.328157
```

16. Make a histogram for the "AirPassengers" dataset, start at 100 on the x-axis, and from values 200 to 700, make the bins 150 wide

CODE:

```
data("AirPassengers")
hist(AirPassengers, breaks = seq(100, 700, by = 150), col = "blue", main = "Histogram for Airpassengers", xlab = "Passenger count", ylab = "Frequency")
```

OUTPUT:



17. Obtain Multiple Lines in Line Chart using a single Plot Function in R. Use attributes "mpg" and "qsec" of the dataset "mtcars"

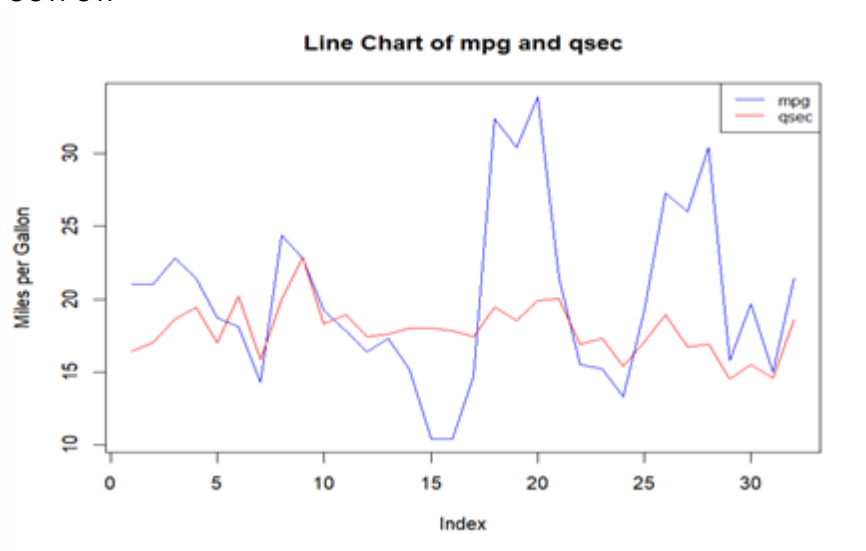
CODE:

```
data("mtcars")
```

```
plot(mtcars$mpg, type = "l", col = "blue", xlab = "Index", ylab = "Miles per Gallon", main = "Line Chart of mpg and qsec")
```

```
lines(mtcars$qsec, col = "red")
legend("topright", legend = c("mpg", "qsec"), col = c("blue", "red"), lty = 1, cex = 0.8)
```

OUTPUT:



18.Download the Dataset "water" From R dataset Link.Find out whether there is a linear relation between attributes "mortality" and "hardness" by plot function.Fit the Data into the Linear Regression model.Predict the mortality for the hardness=88.

CODE:

```
data("iris")
str(iris)
plot(iris$Sepal.Length, iris$Petal.Length, main = "Scatter plot of Sepal.Length vs. Petal.Length", xlab =
"Sepal.Length", ylab = "Petal.Length", col = "blue", pch = 16)
model <- lm(Petal.Length ~ Sepal.Length, data = iris)
abline(model, col = "red")
new_data <- data.frame(Sepal.Length = 5.5)
predicted_Petal_Length <- predict(model, newdata = new_data)
predicted_Petal_Length
```

OUTPUT:

```

> str(iris)
'data.frame': 150 obs. of 5 variables:
 $ Sepal.Length: num 5.1 4.9 4.7 4.6 5 5.4 4.6 5 4.4 4.9 ...
 $ Sepal.Width : num 3.5 3 3.2 3.1 3.6 3.9 3.4 3.4 2.9 3.1 ...
 $ Petal.Length: num 1.4 1.4 1.3 1.5 1.4 1.7 1.4 1.5 1.4 1.5 ...
 $ Petal.Width : num 0.2 0.2 0.2 0.2 0.2 0.4 0.3 0.2 0.2 0.1 ...
 $ Species : Factor w/ 3 levels "setosa","versicolor"...: 1 1 1 1 1 1 1 1 1 1 ...
> plot(iris$Sepal.Length, iris$Petal.Length, main = "Scatter plot of Sepal.Length vs. Petal.Length",
+ xlab = "Sepal.Length", ylab = "Petal.Length", col = "blue", pch = 16)
> model <- lm(Petal.Length ~ Sepal.Length, data = iris)
> abline(model, col = "red")
> new_data <- data.frame(Sepal.Length = 5.5)
> predicted_Petal_Length <- predict(model, newdata = new_data)
> predicted_Petal_Length
1
3.119938
>

```

19. Create a Boxplot graph for the relation between "mpg" (miles per gallon) and "cyl" (number of cylinders) for the dataset "mtcars" available in R Environment.

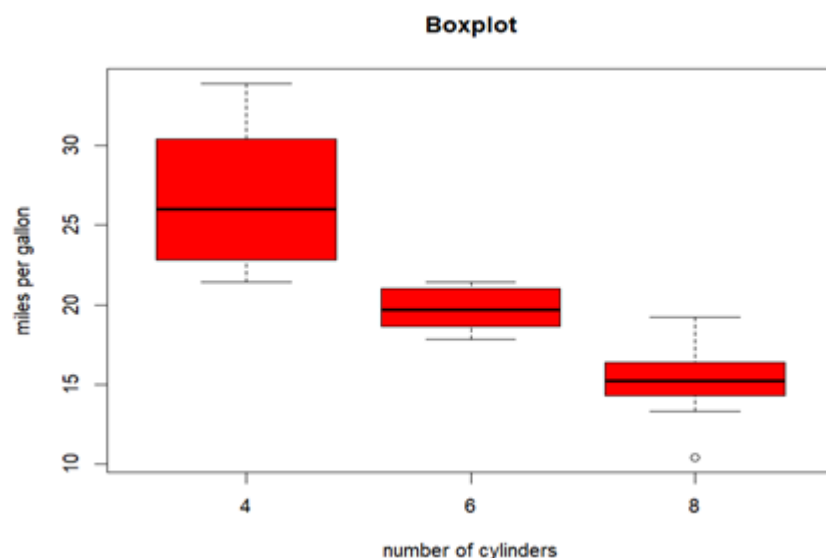
CODE:

```

data("mtcars")
boxplot(mpg ~ cyl, data = mtcars, main = "Boxplot", xlab = "number of cylinders", ylab = "miles per gallon", col = "red")

```

OUTPUT:



20. Assume the Tennis coach wants to determine if any of his team players are scoring outliers. To visualize the distribution of points scored by his players, then how can he decide to develop the box plot? Give suitable example using Boxplot visualization

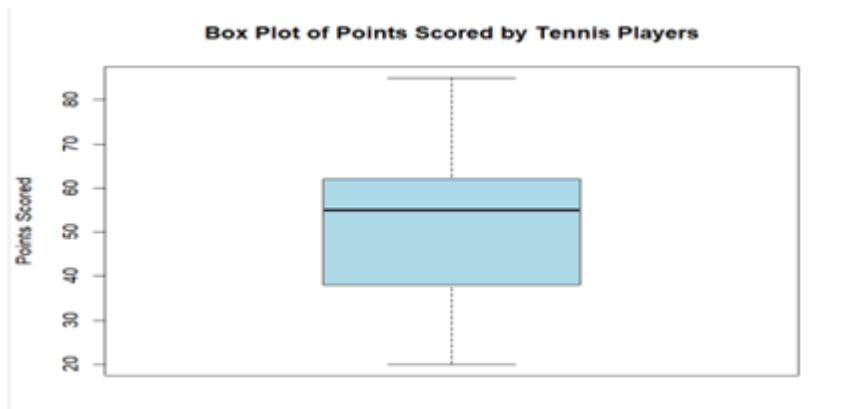
technique.

CODE:

```
score <- c(20, 25, 30, 32, 35, 38, 40, 45, 50, 52, 55, 56, 58, 59, 60, 62, 65, 70, 75, 80, 85)
```

```
boxplot(score, col = "lightblue", main = "Box Plot of Points Scored by Tennis Players", ylab = "Points Scored")
```

OUTPUT:



21. Implement using R language in which age group of people are affected by blood pressure based on the diabetes dataset show it using scatterplot and bar chart (that is Blood Pressure vs Age using dataset "diabetes.csv")

CODE: w

```
dia<-read.csv("C:/Users/haris/Downloads/diabetes.csv")
```

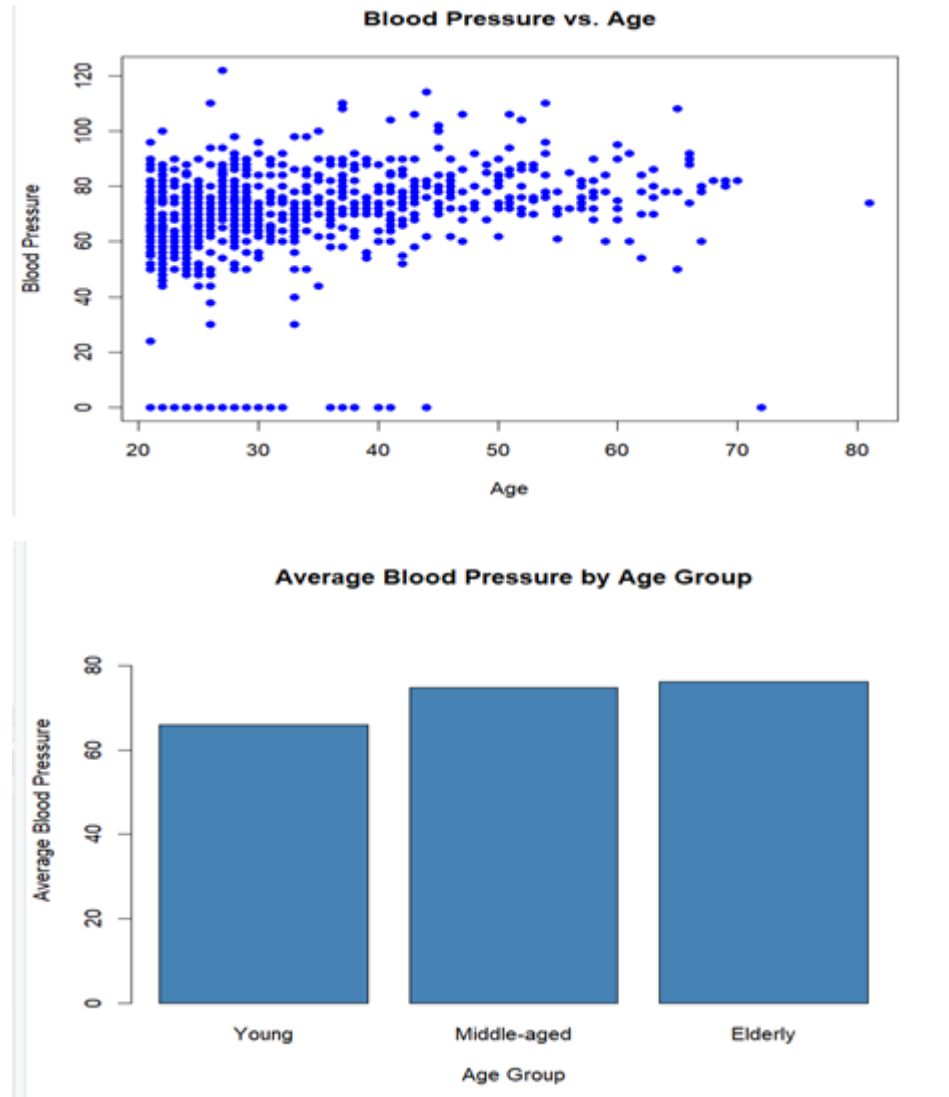
```
View(dia)
```

```
plot(dia$Age, dia$BloodPressure, xlab = "Age", ylab = "Blood Pressure", main = "Blood Pressure vs. Age", col = "blue", pch = 16)
```

```
barplot(dia$Age, dia$Blood_Pressure)
```



OUTPUT:



22. Consider the data set and perform the Apriori Algorithm and FP algorithm support:3 and confidence=50%

Customer ID	Transaction ID	Items Bought
1	0001	{a, d, e}
1	0024	{a, b, c, e}
2	0012	{a, b, d, e}
2	0031	{a, c, d, e}
3	0015	{b, c, e}
3	0022	{b, d, e}
4	0029	{c, d}
4	0040	{a, b, c}
5	0033	{a, d, e}
5	0038	{a, b, e}

## Input:

@relation dataset

@attribute a{true,false}

@attribute b{true,false}

@attribute c{true,false}

@attribute d{true,false}

@attribute e{true,false}

@data

true false false true true

true true true false true

true true false true true

true false true true true

false true true false true

false true false true true

false false true true false

true true true false false

true false false true true

true true false false true

## output:

### FPGROWTH:

The screenshot shows the Weka Associator application window. The 'Choose' button is selected, and the command 'FPGrowth -P 2 -I -1 -N 10 -T 0 -C 0.9 -D 0.05 -U 1.0 -M 0.3' is entered. The 'Start' button is highlighted. The 'Associator output' pane displays the following information:

```
=== Run information ===  
  
Scheme:      weka.associations.FPGrowth -P 2 -I -1 -N 10 -T 0 -C 0.9 -D 0.05 -U 1.0 -M 0.3  
Relation:    dataset  
Instances:   10  
Attributes:  5  
             a  
             b  
             c  
             d  
             e  
  
=== Associator model (full training set) ===  
  
No rules found!
```

## APRIORI ALGORITHM:

The screenshot shows the 'Associator' software interface. At the top, there's a 'Choose' button and a text field containing 'Apriori -N 10 -T 0 -C 0.5 -D 0.05 -U 1.0 -M 0.3 -S -1.0 -c -1'. Below this are 'Start' and 'Stop' buttons. On the left, a 'Result list (right-click for ...)' shows two entries: '21:19:47 - FPGrowth' and '21:38:17 - Apriori', with the latter selected. The main area, titled 'Associator output', displays the following text:

```
d
e
=== Associator model (full training set) ===

Apriori
=====

Minimum support: 0.45 (4 instances)
Minimum metric <confidence>: 0.5
Number of cycles performed: 11

Generated sets of large itemsets:

Size of set of large itemsets L(1): 8
Size of set of large itemsets L(2): 10
Size of set of large itemsets L(3): 3

Best rules found:

1. c=false 5 ==> e=true 5    <conf:(1)> lift:(1.25) lev:(0.1) [0] conv:(1)
2. d=false 4 ==> b=true 4    <conf:(1)> lift:(1.67) lev:(0.16) [1] conv:(1.6)
3. b=false 4 ==> d=true 4    <conf:(1)> lift:(1.67) lev:(0.16) [1] conv:(1.6)
4. a=true c=false 4 ==> e=true 4    <conf:(1)> lift:(1.25) lev:(0.08) [0] conv:(0.8)
5. a=true d=true 4 ==> e=true 4    <conf:(1)> lift:(1.25) lev:(0.08) [0] conv:(0.8)
6. c=false d=true 4 ==> e=true 4    <conf:(1)> lift:(1.25) lev:(0.08) [0] conv:(0.8)
7. a=true 7 ==> e=true 6    <conf:(0.86)> lift:(1.07) lev:(0.04) [0] conv:(0.7)
8. b=true 6 ==> e=true 5    <conf:(0.83)> lift:(1.04) lev:(0.02) [0] conv:(0.6)
9. d=true 6 ==> e=true 5    <conf:(0.83)> lift:(1.04) lev:(0.02) [0] conv:(0.6)
10. c=false 5 ==> a=true 4    <conf:(0.8)> lift:(1.14) lev:(0.05) [0] conv:(0.75)
```

23. Consider the data set and perform the Apriori Algorithm and FP algorithm support:3 and confidence=50%

Consider the market basket transactions shown in the above table.

- (a) What is the maximum number of association rules that can be extracted from this data (including rules that have zero support)?
- (b) What is the maximum size of frequent itemsets that can be extracted (assuming minsup > 0)?

Transaction ID	Items Bought
1	{Milk, Beer, Diapers}
2	{Bread, Butter, Milk}
3	{Milk, Diapers, Cookies}
4	{Bread, Butter, Cookies}
5	{Beer, Cookies, Diapers}
6	{Milk, Diapers, Bread, Butter}
7	{Bread, Butter, Diapers}
8	{Beer, Diapers}
9	{Milk, Diapers, Bread, Butter}
10	{Beer, Cookies}

## Apriori algorithm:

Associator

Choose **Apriori** -N 10 -T 0 -C 0.5 -D 0.05 -U 1.0 -M 0.3 -S 1.0 -c -1

Start Stop

Result list (right-click for ...)

10:05:17 - Apriori

Associator output

```

=====
beer
diapers
bread
butter
cookies

=== Associator model (full training set) ===

Apriori
=====

Minimum support: 0.55 (5 instances)
Minimum metric <confidence>: 0.5
Number of cycles performed: 5

Generated sets of large itemsets:

Size of set of large itemsets L(1): 9
Size of set of large itemsets L(2): 5
Size of set of large itemsets L(3): 1

Best rules found:

1. bread=T 5 ==> beer=F 5 <conf: (1)> lift: (1.67) lev: (0.2) [2] conv: (2)
2. butter=T 5 ==> beer=F 5 <conf: (1)> lift: (1.67) lev: (0.2) [2] conv: (2)
3. butter=T 5 ==> bread=T 5 <conf: (1)> lift: (2) lev: (0.25) [2] conv: (2.5)
4. bread=T 5 ==> butter=T 5 <conf: (1)> lift: (2) lev: (0.25) [2] conv: (2.5)
5. butter=T 5 ==> bread=F 5 <conf: (1)> lift: (2) lev: (0.25) [2] conv: (2.5)
6. bread=F 5 ==> butter=F 5 <conf: (1)> lift: (2) lev: (0.25) [2] conv: (2.5)
7. bread=T butter=T 5 ==> beer=F 5 <conf: (1)> lift: (1.67) lev: (0.2) [2] conv: (2)
8. beer=T butter=T 5 ==> bread=T 5 <conf: (1)> lift: (2) lev: (0.25) [2] conv: (2.5)
9. beer=F bread=T 5 ==> butter=T 5 <conf: (1)> lift: (2) lev: (0.25) [2] conv: (2.5)
10. butter=T 5 ==> beer=F bread=T 5 <conf: (1)> lift: (2) lev: (0.25) [2] conv: (2.5)

```

## Fp growth algorithm:

Associator

Choose **FPGrowth** -P 2 -I -1 -N 10 -T 0 -C 0.9 -D 0.05 -U 1.0 -M 0.3

Start Stop

Result list (right-click for ...)

10:05:17 - Apriori

10:06:24 - FPGrowth

Associator output

```

=== Run information ===

Scheme:      weka.associations.FPGrowth -P 2 -I -1 -N 10 -T 0 -C 0.9 -D 0.05 -U 1.0 -M 0.3
Relation:    items
Instances:   10
Attributes:  6
             milk
             beer
             diapers
             bread
             butter
             cookies

=== Associator model (full training set) ===

FPGrowth found 4 rules (displaying top 4)

1. [butter=F]: 5 ==> [bread=F]: 5 <conf: (1)> lift: (2) lev: (0.25) conv: (2.5)
2. [bread=F]: 5 ==> [butter=F]: 5 <conf: (1)> lift: (2) lev: (0.25) conv: (2.5)
3. [milk=F, butter=F]: 3 ==> [bread=F]: 3 <conf: (1)> lift: (2) lev: (0.15) conv: (1.5)
4. [milk=F, bread=F]: 3 ==> [butter=F]: 3 <conf: (1)> lift: (2) lev: (0.15) conv: (1.5)

```

## 24. Bayes classification and decision tree (using training and test data)

RID	age	income	student	credit_rating	Class: buys_computer
1	<=30	high	no	fair	no
2	<=30	high	no	excellent	no
3	31 ... 40	high	no	fair	yes
4	>40	medium	no	fair	yes
5	>40	low	yes	fair	yes
6	>40	low	yes	excellent	no
7	31 ... 40	low	yes	excellent	yes
8	<=30	medium	no	fair	no
9	<=30	low	yes	fair	yes
10	>40	medium	yes	fair	yes
11	<=30	medium	yes	excellent	yes
12	31 ... 40	medium	no	excellent	yes
13	31 ... 40	high	yes	fair	yes
14	>40	medium	no	excellent	no

## Input:

@relation decision\_tree

@attribute age{young,middle,old}

@attribute income{low,medium,high}

@attribute student{yes,no}

@attribute Credit\_rating{fair,excellent}

@attribute class{yes,no}

@data

young high no fair no

young high no excellent no

middle high no fair yes

old medium no fair yes

old low yes fair yes

old low yes excellent no

middle low yes excellent yes

young medium no fair no

young low yes fair yes

old medium yes fair yes

young medium yes excellent yes

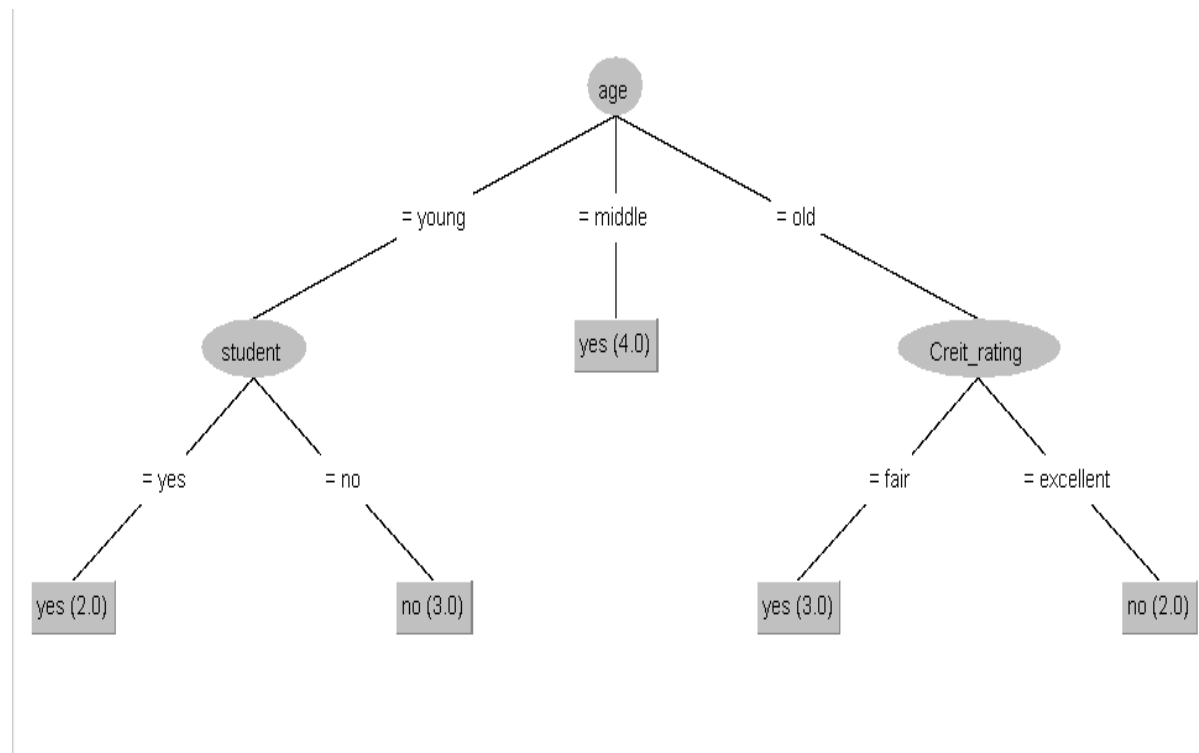
middle medium no excellent yes

middle high yes fair yes

old medium no excellent no

output:

tree:



Preprocess Classify Cluster Associate Select attributes Visualize

Classifier

Choose **J48 -C 0.25 -M 2**

Test options

☐ Use training set

☐ Supplied test set

☒ Cross-validation Folds

☐ Percentage split %

(Nom) class

Result list (right-click for options)

21:41:41 - trees.J48

Classifier output

Number of Leaves : 5

Size of the tree : 8

Time taken to build model: 0.01 seconds

=== Stratified cross-validation ===

=== Summary ===

Correctly Classified Instances	7	50	%
Incorrectly Classified Instances	7	50	%
Kappa statistic	-0.0426		
Mean absolute error	0.4167		
Root mean squared error	0.5984		
Relative absolute error	87.5	%	
Root relative squared error	121.2987	%	
Total Number of Instances	14		

=== Detailed Accuracy By Class ===

	TP Rate	FP Rate	Precision	Recall	F-Measure	MCC	ROC Area	PRC Area	Class
	0.556	0.600	0.625	0.556	0.588	-0.043	0.633	0.758	yes
	0.400	0.444	0.333	0.400	0.364	-0.043	0.633	0.457	no
Weighted Avg.	0.500	0.544	0.521	0.500	0.508	-0.043	0.633	0.650	

=== Confusion Matrix ===

```

a b  <-- classified as
5 4 | a = yes
3 2 | b = no

```

25. Analysis the dataset “diabetes. csv” how the diabetes trend is for different age people, using linear regression and multiple regression.

Input:

```
data<-read.csv("C:/Users/Hari Naidu/Desktop/POM/download papers/diabetes.csv")
```

```
data
```

```
relation<-lm(data$Age~data$Outcome)
```

```
relation
```

```
relation<-lm(data$Age~data$Outcome+data$BMI)
```

```
relation
```

output:

```

[ reached 'max' / getoption("max.print") -- omitted 657 rows ]
> relation<-lm(data$Age~data$Outcome+data$BMI)
> relation

Call:
lm(formula = data$Age ~ data$Outcome + data$BMI)

Coefficients:
(Intercept)  data$Outcome  data$BMI
    32.84734      6.14177     -0.05469

> relation<-lm(data$Age~data$Outcome)
> relation

Call:
lm(formula = data$Age ~ data$Outcome)

Coefficients:
(Intercept)  data$Outcome
    31.190      5.877

```

26. Implement using WEKA for the given Suppose a database has five transactions. Let min sup= 50%(2) and min con f= 80%.

Transactions	Items
T1	(M, O, N, K, E, Y)
T2	(D, O, N, K, E, Y)
T3	(M, A, K, E)
T4	(M, U, C, K, Y)
T5	(C, O, O, K, I, E)

- Find all frequent item sets using Apriori algorithm
- Also draw FP-Growth Tree

The screenshot shows the WEKA software interface with the Apriori algorithm selected. The 'Choose' dropdown is set to 'Apriori'. The 'Start' button is highlighted. The 'Result list' on the left shows '17:11:38 - Apriori'. The main 'Associator output' window displays the following text:

```

y
d
a
u
c
i
=== Associator model (full training set) ===

Apriori
=====
Minimum support: 0.05 (4 instances)
Minimum metric <confidence>: 0.8
Number of cycles performed: 3

Generated sets of large itemsets:

Size of set of large itemsets L(1): 6
Size of set of large itemsets L(2): 6
Size of set of large itemsets L(3): 1

Best rules found:

1. e=T 4 ==> k=T 4 <conf: (1)> lift: (1) lev: (0) [0] conv: (0)
2. d=F 4 ==> k=T 4 <conf: (1)> lift: (1) lev: (0) [0] conv: (0)
3. a=F 4 ==> k=T 4 <conf: (1)> lift: (1) lev: (0) [0] conv: (0)
4. u=F 4 ==> k=T 4 <conf: (1)> lift: (1) lev: (0) [0] conv: (0)
5. i=F 4 ==> k=T 4 <conf: (1)> lift: (1) lev: (0) [0] conv: (0)
6. u=F 4 ==> e=T 4 <conf: (1)> lift: (1.25) lev: (0.16) [0] conv: (0.8)
7. e=T 4 ==> u=F 4 <conf: (1)> lift: (1.25) lev: (0.16) [0] conv: (0.8)
8. e=T u=F 4 ==> k=T 4 <conf: (1)> lift: (1) lev: (0) [0] conv: (0)
9. k=T u=F 4 ==> e=T 4 <conf: (1)> lift: (1.25) lev: (0.16) [0] conv: (0.8)
10. k=T e=T 4 ==> u=F 4 <conf: (1)> lift: (1.25) lev: (0.16) [0] conv: (0.8)

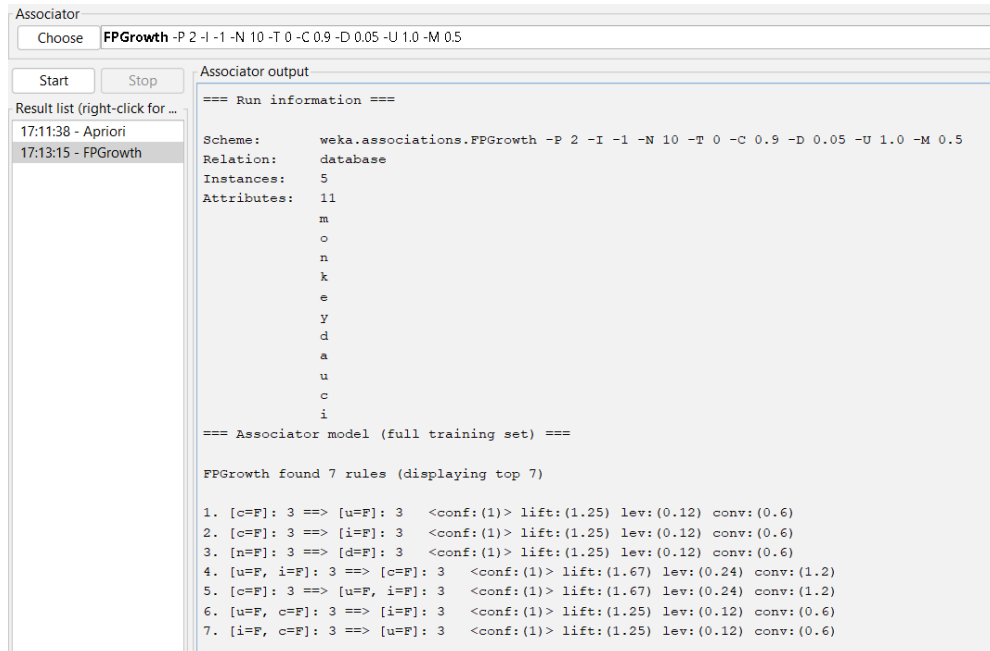
```

Input:



Apriori algorithm:

Fpgrowth algorithm:



27. Prediction of Categorical Data using Decision Tree Algorithm through WEKA using any datasets. a) Tree b) Preprocess c) Logistic

Output:

Tree:

Preprocess **Classify** Cluster Associate Select attributes Visualize

Classifier

Choose **J48 -C 0.25 -M 2**

Test options

☐ Use training set

☐ Supplied test set

☒ Cross-validation Folds

☐ Percentage split %

(Nom) class

Result list (right-click for options)

- 12:51:54 - bayes.NaiveBayes
- 12:58:01 - bayes.NaiveBayes
- 13:06:38 - trees.J48
- 13:09:14 - functions.Logistic
- 13:10:16 - functions.Logistic
- 13:11:08 - trees.J48**

Classifier output

Number of Leaves : 20

Size of the tree : 39

Time taken to build model: 0.01 seconds

=== Stratified cross-validation ===

=== Summary ===

Correctly Classified Instances	567	73.8281 %
Incorrectly Classified Instances	201	26.1719 %
Kappa statistic	0.4164	
Mean absolute error	0.3158	
Root mean squared error	0.4463	
Relative absolute error	69.4841 %	
Root relative squared error	93.6293 %	
Total Number of Instances	768	

=== Detailed Accuracy By Class ===

	TP Rate	FP Rate	Precision	Recall	F-Measure	MCC	ROC Area	PRC Area	Class
	0.814	0.403	0.790	0.814	0.802	0.417	0.751	0.811	tested_neg
	0.597	0.186	0.632	0.597	0.614	0.417	0.751	0.572	tested_pos
Weighted Avg.	0.738	0.327	0.735	0.738	0.736	0.417	0.751	0.727	

=== Confusion Matrix ===

```

a  b  <-- classified as
407 93 | a = tested_negative
108 160 | b = tested_positive

```

Preprocessor:

Weka Explorer

Preprocess **Classify** Cluster Associate Select attributes Visualize

Filter

Choose **None**

Current relation

Relation: pima\_diabetes

Instances: 768

Attributes: 9

Sum of weights: 768

Attributes

No.	Name
1	<input checked="" type="checkbox"/> preg
2	<input type="checkbox"/> plas
3	<input type="checkbox"/> pres
4	<input type="checkbox"/> skin
5	<input type="checkbox"/> insu
6	<input type="checkbox"/> mass
7	<input type="checkbox"/> pedi
8	<input type="checkbox"/> age
9	<input type="checkbox"/> class

Selected attribute

Name: preg

Missing: 0 (0%)

Distinct: 17

Type: Numeric

Unique: 2 (0%)

Statistic	Value
Minimum	0
Maximum	17
Mean	3.845
StdDev	3.37

Class: class (Nom)

Logistic:

Preprocess Classify Cluster Associate Select attributes Visualize

Classifier  
Choose **Logistic -R 1.0E-8 -M -1 -num-decimal-places 4**

Test options  
☐ Use training set  
☐ Supplied test set Set...  
☒ Cross-validation Folds 10  
☐ Percentage split % 66  
 More options...

(Nom) class  
 Start Stop

Result list (right-click for options)  
 12:51:54 - ba... Starts the classification  
 12:58:01 - bayes.NaiveBayes  
 13:06:38 - trees.J48  
 13:09:14 - functions.Logistic  
**13:10:16 - functions.Logistic**

Classifier output

```

mass          0.9142
pedi          0.3886
age           0.9852

Time taken to build model: 0.01 seconds

=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances      593           77.2135 %
Incorrectly Classified Instances    175           22.7865 %
Kappa statistic                    0.4734
Mean absolute error                 0.3094
Root mean squared error             0.3954
Relative absolute error             68.0818 %
Root relative squared error         82.9651 %
Total Number of Instances          768

=== Detailed Accuracy By Class ===

                TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                0.880    0.429    0.793     0.880    0.834      0.480    0.832     0.892    tested_neg.
                0.571    0.120    0.718     0.571    0.636      0.480    0.832     0.715    tested_pos.
Weighted Avg.   0.772    0.321    0.767     0.772    0.765      0.480    0.832     0.831

=== Confusion Matrix ===

  a    b  <-- classified as
440  60 |  a = tested_negative
115 153 |  b = tested_positive

```

28.Create the dataset using ARFF file format:

Transaction ID	Items
T1	Hot Dogs, Buns, Ketchup
T2	Hot Dogs, Buns
T3	Hot Dogs, Coke, Chips
T4	Chips, Coke
T5	Chips, Ketchup
T6	Hot Dogs, Coke, Chips

a.Find the **frequent itemsets** and generate **association rules** on this. Assume that minimum support threshold ( $s = 33.33\%$ ) and minimum confident threshold ( $c = 60\%$ ).

b.List the various rule generated by apriori and FP tree algorithm ,mention wheather accepted or rejected.

Input:

@relation hotdogs

@attribute hotdogs {t,f}

@attribute buns {t,f}

@attribute ketchup{t,f}

@attribute coke{t,f}

@attribute chips{t,f}

@data

t t t f f

t t f f f

t f f t t

f f f t t

f f t f t

t f f t t

output:

apriori algorithm:

The screenshot shows the 'Associator' application window. The 'Choose' dropdown is set to 'Apriori' with parameters '-N 10 -T 0 -C 0.6 -D 0.05 -U 1.0 -M 0.2 -S -1.0 -c -1'. The 'Start' button is highlighted. On the left, a 'Result list' shows several entries, with '13:23:39 - Apriori' selected. The main 'Associator output' pane displays the following text:

```
==== Associator model (full training set) ====

Apriori
=====

Minimum support: 0.55 (3 instances)
Minimum metric <confidence>: 0.6
Number of cycles performed: 9

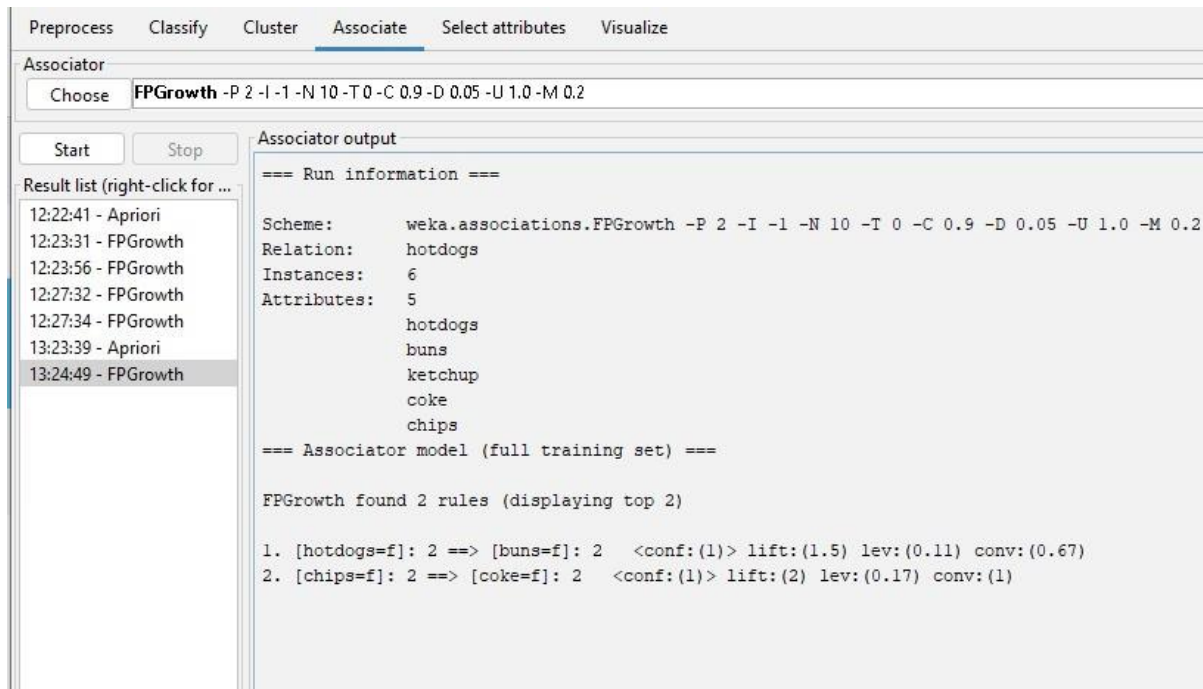
Generated sets of large itemsets:

Size of set of large itemsets L(1): 6
Size of set of large itemsets L(2): 7
Size of set of large itemsets L(3): 4
Size of set of large itemsets L(4): 1

Best rules found:

1. chips=t 4 ==> buns=f 4    <conf:(1)> lift:(1.5) lev:(0.22) [1] conv:(1.33)
2. buns=f 4 ==> chips=t 4    <conf:(1)> lift:(1.5) lev:(0.22) [1] conv:(1.33)
3. coke=t 3 ==> buns=f 3     <conf:(1)> lift:(1.5) lev:(0.17) [1] conv:(1)
4. coke=t 3 ==> ketchup=f 3  <conf:(1)> lift:(1.5) lev:(0.17) [1] conv:(1)
5. coke=t 3 ==> chips=t 3    <conf:(1)> lift:(1.5) lev:(0.17) [1] conv:(1)
6. ketchup=f coke=t 3 ==> buns=f 3  <conf:(1)> lift:(1.5) lev:(0.17) [1] conv:(1)
7. buns=f coke=t 3 ==> ketchup=f 3  <conf:(1)> lift:(1.5) lev:(0.17) [1] conv:(1)
8. buns=f ketchup=f 3 ==> coke=t 3  <conf:(1)> lift:(2) lev:(0.25) [1] conv:(1.5)
9. coke=t 3 ==> buns=f ketchup=f 3  <conf:(1)> lift:(2) lev:(0.25) [1] conv:(1.5)
10. ketchup=f chips=t 3 ==> buns=f 3  <conf:(1)> lift:(1.5) lev:(0.17) [1] conv:(1)
```

Fp growth:



29. Prediction of Categorical Data using Rule base classification and decision tree classification through WEKA using any datasets. Compare the accuracy using two algorithm and plot the graph

Input:

@relation decision\_tree

@attribute age{young,middle,old}

@attribute income{low,medium,high}

@attribute student{yes,no}

@attribute Creit\_rating{fair,excellent}

@attribute class{yes,no}

@data

young high no fair no

young high no excellent no

middle high no fair yes

old medium no fair yes

old low yes fair yes  
old low yes excellent no  
middle low yes excellent yes  
young medium no fair no  
young low yes fair yes  
old medium yes fair yes  
young medium yes excellent yes  
middle medium no excellent yes  
middle high yes fair yes  
old medium no excellent no

Output:

Rule based classification:

Classifier

Choose JRip -F 3 -N 2.0 -O 2 -S 1

Test options

☐ Use training set  
☐ Supplied test set 

Set...

  
☒ Cross-validation Folds 

10

  
☐ Percentage split % 

66

Train on a percentage of the data and test on the remainder

Classifier output

(age = old) and (Creit\_rating = excellent) => class=no (2.0/0.0)  
=> class=yes (9.0/0.0)

Number of Rules : 3

model: 0 seconds

(Nom) class

Start Stop

Result list (right-click for options)

21:41:41 - trees.J48

22:07:46 - rules.JRip

=== Stratified cross-validation ===

=== Summary ===

Correctly Classified Instances	11	78.5714 %
Incorrectly Classified Instances	3	21.4286 %
Kappa statistic	0.4615	
Mean absolute error	0.2143	
Root mean squared error	0.3928	
Relative absolute error	45 %	
Root relative squared error	79.6242 %	
Total Number of Instances	14	

=== Detailed Accuracy By Class ===

	TP Rate	FP Rate	Precision	Recall	F-Measure	MCC	ROC Area	PRC Area	Class
	1.000	0.600	0.750	1.000	0.857	0.548	0.900	0.928	yes
	0.400	0.000	1.000	0.400	0.571	0.548	0.900	0.800	no
Weighted Avg.	0.786	0.386	0.839	0.786	0.755	0.548	0.900	0.882	

=== Confusion Matrix ===

```

a b  <-- classified as
9 0 | a = yes
3 2 | b = no

```

Decision tree:

Classifier

ChooseJ48 -C 0.25 -M 2

Test options

☐ Use training set

☐ Supplied test set

☒ Cross-validation

☐ Percentage split

Set...

Folds10

%66

More options...

(Nom) class

Start

Stop

Result list (right-click for options)

21:41:41 - trees.J48

22:07:46 - rules.JRip

22:09:24 - rules.JRip

22:09:47 - trees.J48

Classifier output

Number of Leaves : 5

Size of the tree : 8

Time taken to build model: 0 seconds

=== Stratified cross-validation ===

=== Summary ===

Correctly Classified Instances750%

Incorrectly Classified Instances750%

Kappa statistic-0.0426

Mean absolute error0.4167

Root mean squared error0.5984

Relative absolute error87.5%

Root relative squared error121.2987%

Total Number of Instances14

=== Detailed Accuracy By Class ===

	TP Rate	FP Rate	Precision	Recall	F-Measure	MCC	ROC Area	PRC Area	Class
	0.556	0.600	0.625	0.556	0.588	-0.043	0.633	0.758	yes
	0.400	0.444	0.333	0.400	0.364	-0.043	0.633	0.457	no
Weighted Avg.	0.500	0.544	0.521	0.500	0.508	-0.043	0.633	0.650	

=== Confusion Matrix ===

a b

<-- classified as

5 4 | a = yes

3 2 | b = no