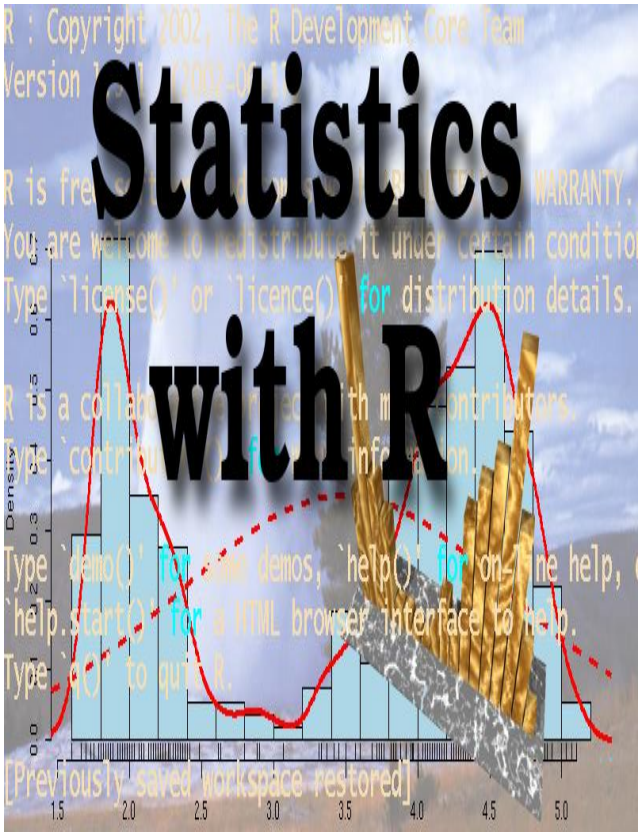


LAB MANUAL

Statistics With R programming

ABSTRACT

A Record of all the lab assessments and assignment done during the year 2018-19 in the subject of Statistics with R under the Guidance of Dr. Santanu Mandal. The document basically deals with how we understand the basics of R along with statistics and implement it in solving problems of probabilities, Bayes theorem , Random Sampling etc.

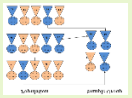
Tathagat Banerjee

17BCE7100 ~ (c1)

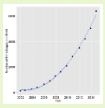
Table of Contents

Basics of Programming with R pg. 3

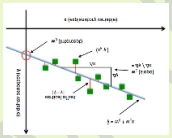

Computations with R
(Basic Statistical Analysis) pg.9

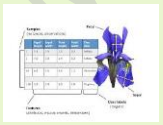

Using Datasets for Random Sampling with R pg.26


Vectors and Data Frame creations with R pg.32


Understanding HSAUR package in R pg.36


Probability , Matrices and Arrays in R pg.44


Linear Regression : A Case Study pg.48


Petal Dataset Analysis pg.50

CASE STUDY - Hypothesis pg.53

Basics of Programming with R

getwd()

[1] "C:/Users/17BCE7100/Downloads"

Q1.Simple Operations

(a)Enter the data {2,5,3,7,1,9,6}directly and store it in a variable x.

```
x<-c(2,5,3,7,1,9,6)
x
```

[1] 2 5 3 7 1 9 6

(b)Find the number of elements in x, i.e. in the data list

```
length(x)
```

[1] 7

(c)Find the last element of x.

```
x[7]
```

[1] 6

(d)Find the minimum element of x

min(x)

[1] 1

(e)Find the maximum element of x.

max(x)

[1] 9

Q2.Enter the data {1, 2, …. ,19,20} in a variable x.

x<-1:20

x

[1] 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20

a)Find the 3rd element in the data list.

x[3]

[1] 3

b)Find 3rd to 5th element in the data list.

x[3:5]

[1] 3 4 5

c)Find 2nd, 5th, 6th, and 12th element in the list.

x[c(2,5,6,12)]

[1] 2 5 6 12

d) Print the data as {20, 19, …, 2, 1} without again entering the data.

x[20:1]

[1] 20 19 18 17 16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1

Q3.

a)Create a data list (4, 4, 4, 4, 3, 3, 3, 5, 5, 5) using 'rep' function

x<-c(rep(4,4),rep(3,3),rep(5,3))

x

[1] 4 4 4 4 3 3 3 5 5 5

b)Create a list (4, 6, 3, 4, 6, 3, …, 4, 6, 3)where there 10 occurrences of 4, 6, and 3 in the given order

x<-rep(c(4,6,3),10)

x

[1] 4 6 3 4 6 3 4 6 3 4 6 3 4 6 3 4 6 3 4 6 3 4 6 3 4 6 3 4 6 3

c)Create a list (3, 1, 5, 3, 2, 3, 4, 5, 7,7, 7, 7, 7,7, 6, 5, 4, 3, 2, 1, 34, 21, 54) using one line command.

x<-c(3,1,5,3,2:5,rep(7,6),6:1,34,21,54)

x

[1] 3 1 5 3 2 3 4 5 7 7 7 7 7 7 6 5 4 3 2 1 34 21 54

d)First create a list (2, 1, 3, 4). Then append this list at the end with another list (5, 7, 12, 6, -8). Check whether the number of elements in the augmented list is 11.

```
y<-c(2,1,3,4)
y
[1] 2 1 3 4
y<-c(y,5,7,12,-8)
y
[1] 2 1 3 4 5 7 12 -8
```

Q4.

(a)Print all numbers starting with 3 and ending with 7 with an increment of 0:5. Store these numbers in y.

```
y<-seq(3,7,0.5)
y
[1] 3.0 3.5 4.0 4.5 5.0 5.5 6.0 6.5 7.0
```

(b)Print all even numbers between 2 and 14 (both inclusive)

```
y<-seq(2,14,2)
```

y

[1] 2 4 6 8 10 12 14

(c)Type 2*y and see what you get. Each element of y is multiplied by 2.

2*y

[1] 4 8 12 16 20 24 28

Q5.Few simple statistical measures:

(a)Enter data as 1,2, … ,10.

x<-(1:10)

x

[1] 1 2 3 4 5 6 7 8 9 10

(b)Find sum of the numbers.

sum(x)

[1] 55

(c)Find mean, median.

mean(x)

[1] 5.5

```
median(x)
```

```
[1] 5.5
```

(d)Find sum of squares of these values.

```
sum(x*x)
```

```
[1] 385
```

(e)Find the value of , This is known as mean deviation about mean (MD)

```
m<-mean(x)
```

```
s<-sum(x-m)
```

```
md<-s/10
```

```
md
```

```
[1] 0
```

(f)Check whether(MD) is less than or equal to standard deviation

```
sd(x)
```

```
[1] 3.02765
```

Computations with R (Basic Statistical Analysis)

Question 1

1. Few simple statistical measures:

a) Enter data as 1,2, ... ,10.

```
x <- c(1:10)
```

```
[1]1 2 3 4 5 6 7 8 9 10
```

b) Find sum of the numbers.

```
sum(x)
```

```
[1] 55
```

c) Find mean, median.

```
mean(x)
```

```
[1] 5.5
```

```
median(x)
```

```
[1] 5.5
```

d) Find sum of squares of these values.

```
sum(x*x)
```

```
[1] 385
```

e) Find the value of $1/n\sum |xi - \bar{x}|$, This is known as mean deviation about mean($MD_{\bar{x}}$).

```
mean(x - mean(x))
```

[1] 0

abs(x-mean(x))/length(x)

[1] 0.45 0.35 0.25 0.15 0.05 0.05 0.15 0.25 0.35 0.45

y <- abs(x-mean(x))/length(x)

f) Find standard deviation using formula.

sd(x)

[1] 3.02765

g) Check whether MD$\bar{x}$ is less than or equal to standard deviation.

y <- abs(x-mean(x))*abs(x-mean(x)>sd)

[1] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE [1] 3.02765

Question 2

2. Reading a data file and working with it:

a) Read the file first and store it in a.

a <- read.csv("house_data_1.csv", header = T)

a

	Price	Floor Area	Rooms	Age	Central Heating
1	52.00	1225	3	6.2	no
2	54.75	1230	3	7.5	no
3	57.50	1200	3	4.2	no
4	57.50	1000	2	8.8	no
5	59.75	1420	4	1.9	yes
6	62.50	1450	3	5.2	no
7	64.75	1380	4	6.6	yes
8	67.25	1510	4	2.3	no

9 67.50 1400 5 6.1 no

10 69.75 1550 6 9.2 no

11 70.00 1720 6 4.3 yes

12 75.50 1700 5 4.3 no

13 77.50 1660 6 1.0 yes

14 78.00 1800 7 7.0 yes

15 81.25 1830 6 3.6 yes

16 82.50 1790 6 1.7 yes

17 86.25 2010 6 1.2 yes

18 87.50 2000 6 0.0 yes

19 88.00 2100 8 2.3 yes

20 92.00 2240 7 0.7 yes

b) How many rows are there in this table? How many columns are there?

```
nrow(a)
[1] 20
ncol(a)
[1] 5
```

c) How to find the number of rows and number of columns by a single command?

```
x <- c(nrow(a), ncol(a))
x
[1] 20  5
```

d) What are the variables in the data file?

```
names(a)
[1] "Price"        "FloorArea"     "Rooms"        "Age"          "CentralHeating"
```

e) If the file is very large, naturally we cannot simply type `a', because it will cover the entire screen and we won't be able to understand anything. So how to see the

top or bottom few lines in this file?

head(a)

	Price	FloorArea	Rooms	Age	CentralHeating
1	52.00	1225	3	6.2	no
2	54.75	1230	3	7.5	no
3	57.50	1200	3	4.2	no
4	57.50	1000	2	8.8	no
5	59.75	1420	4	1.9	yes
6	62.50	1450	3	5.2	no

tail(a)

	Price	FloorArea	Rooms	Age	Central Heating
15	81.25	1830	6	3.6	yes
16	82.50	1790	6	1.7	yes
17	86.25	2010	6	1.2	yes
18	87.50	2000	6	0.0	yes
19	88.00	2100	8	2.3	yes
20	92.00	2240	7	0.7	yes

f) If the number of columns is too large, again we may face the same problem. So how to see the first 5 rows and first 3 columns?

a[1:5, 1:3]

	Price	FloorArea	Rooms
1	52.00	1225	3
2	54.75	1230	3
3	57.50	1200	3
4	57.50	1000	2
5	59.75	1420	4

g) How to get 1st, 3rd, 6th, and 10th row and 2nd, 4th, and 5th column?

a[c(1,3,6,10),c(2,4,5)]

	Floor	Area	Age	Central Heating
1	1225	6.2		no
3	1200	4.2		no
6	1450	5.2		no
10	1550	9.2		no

h) How to get values in a specific row or a column?

a[,4]

[1] 6.2 7.5 4.2 8.8 1.9 5.2 6.6 2.3 6.1 9.2 4.3 4.3 1.0 7.0 3.6 1.7 1.2 0.0 2.3 0.7

Question 3

3. Calculate simple statistical measures using the values in the data file.

house_data_1.csv

Price	FloorArea		Rooms	Age	CentralHeating
52	1225	3	6.2	no	
54.75	1230	3	7.5	no	
57.5	1200	3	4.2	no	
57.5	1000	2	8.8	no	
59.75	1420	4	1.9	yes	
62.5	1450	3	5.2	no	
64.75	1380	4	6.6	yes	
67.25	1510	4	2.3	no	
67.5	1400	5	6.1	no	
69.75	1550	6	9.2	no	
70	1720	6	4.3	yes	

75.5	1700	5	4.3	no
77.5	1660	6	1	yes
78	1800	7	7	yes
81.25	1830	6	3.6	yes
82.5	1790	6	1.7	yes
86.25	2010	6	1.2	yes
87.5	2000	6	0	yes
88	2100	8	2.3	yes
92	2240	7	0.7	yes

a) Find means, medians, standard deviations of Price, Floor Area, Rooms, and Age.

```
mean(a$Price)
[1] 71.5875
mean(a$FloorArea)
[1] 1610.75
mean(a$Rooms)
[1] 5
mean(a$Age)
[1] 4.205
median(a$Price)
[1] 69.875
median(a$FloorArea)
[1] 1605
median(a$Rooms)
[1] 5.5
median(a$Age)
```

[1] 4.25

sd(a$Price)

[1] 12.21094

sd(a$FloorArea)

[1] 331.9649

sd(a$Rooms)

[1] 1.65434

sd(a$Age)

[1] 2.786523

b) How many houses have central heating and how many don't have?

y<-subset(a, CentralHeating == "yes")

y

	Price	FloorArea	Rooms	Age	CentralHeating
5	59.75	1420	4	1.9	yes
7	64.75	1380	4	6.6	yes
11	70.00	1720	6	4.3	yes
13	77.50	1660	6	1.0	yes
14	78.00	1800	7	7.0	yes
15	81.25	1830	6	3.6	yes
16	82.50	1790	6	1.7	yes
17	86.25	2010	6	1.2	yes
18	87.50	2000	6	0.0	yes
19	88.00	2100	8	2.3	yes
20	92.00	2240	7	0.7	yes

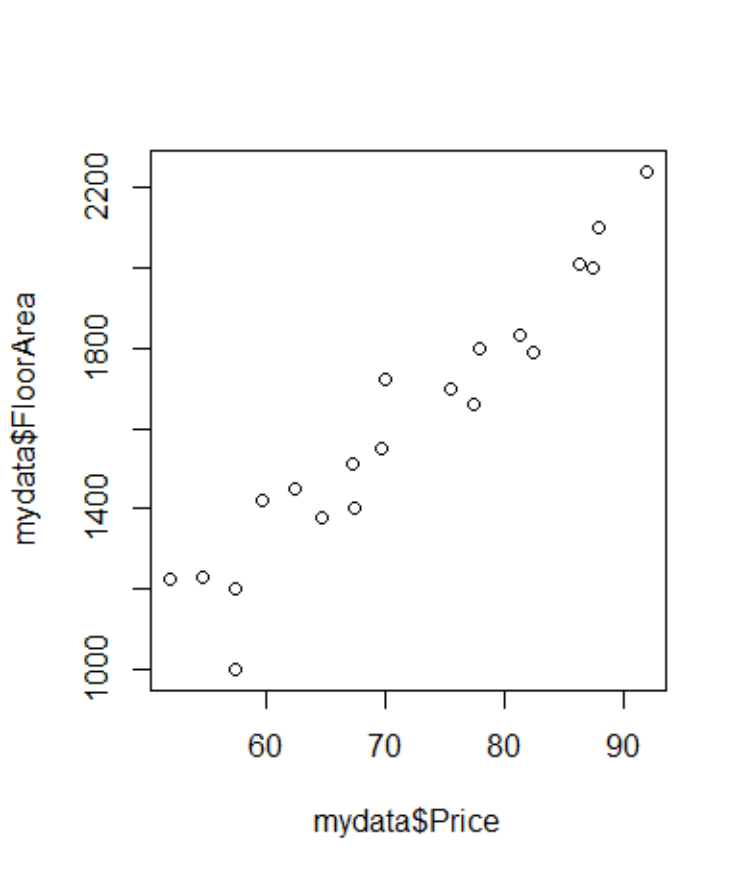
nrow(y)

[1] 11

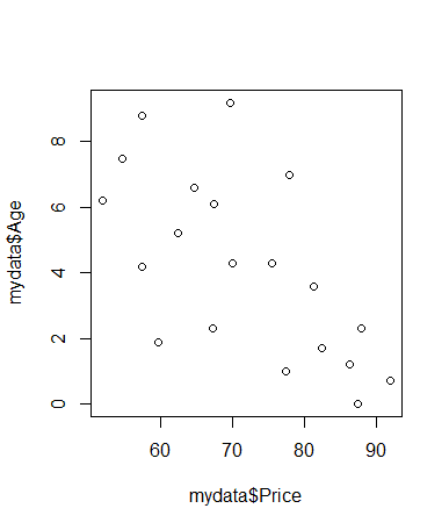
```
y<-subset(a, CentralHeating == "no")
y
   Price FloorArea Rooms Age CentralHeating
1  52.00      1225     3 6.2           no
2  54.75      1230     3 7.5           no
3  57.50      1200     3 4.2           no
4  57.50      1000     2 8.8           no
6  62.50      1450     3 5.2           no
8  67.25      1510     4 2.3           no
9  67.50      1400     5 6.1           no
10 69.75      1550     6 9.2           no
12 75.50      1700     5 4.3           no
nrow(y)
[1] 9
```

c) Plot Price vs. Floor, Price vs. Age, and Price vs. rooms, in separate graphs.

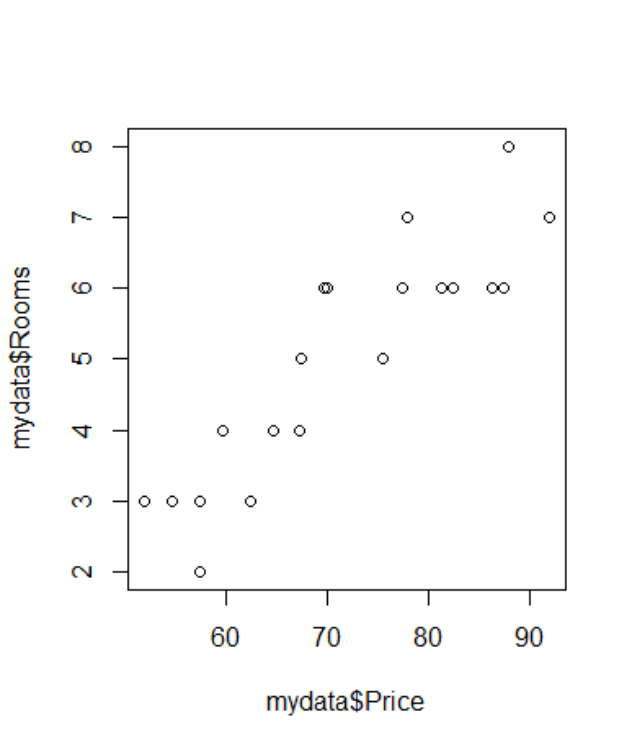
```
> plot(a$Price, a$FloorArea)
```



- plot(a$Price, a$Age)



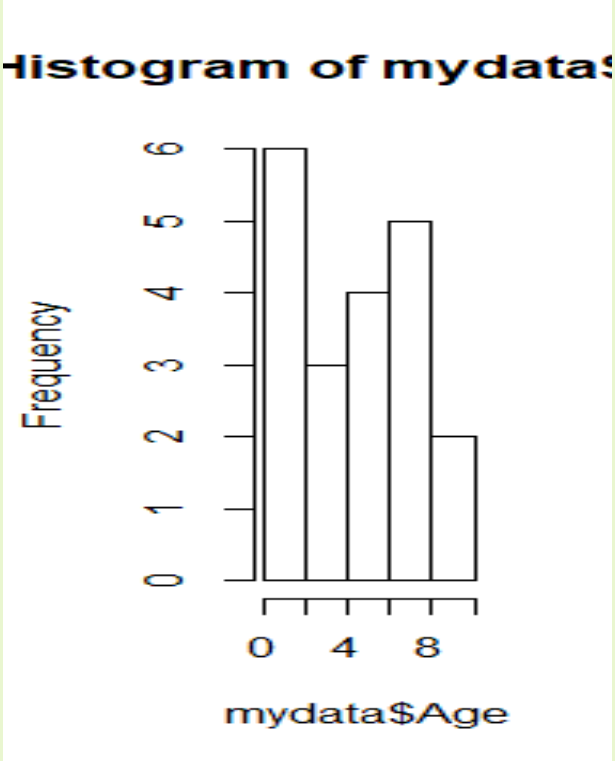
- plot(a$Price, a$Rooms)



d) Draw histograms of Prices,and Age.

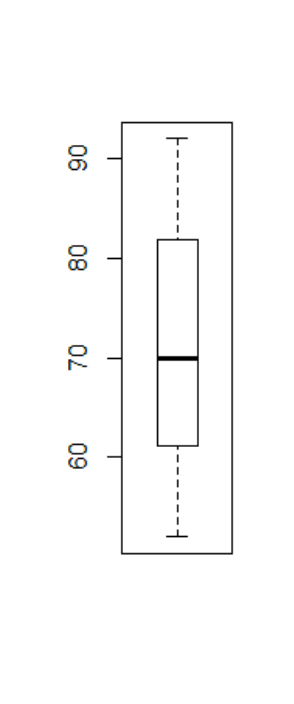
> hist(a$Price)

- hist(a$Age)

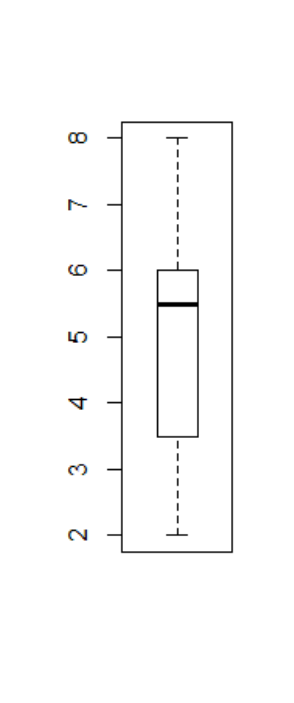


e) Draw box plots of Price, FloorArea and age

> boxplot(a$Price)

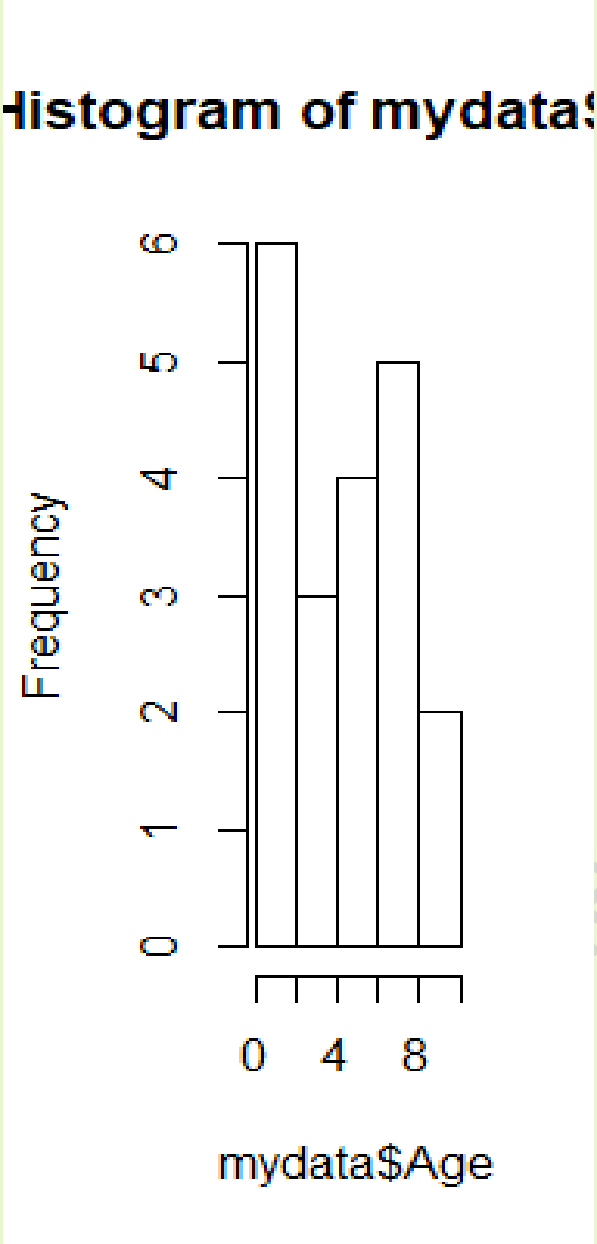


- boxplot(a$FloorArea)

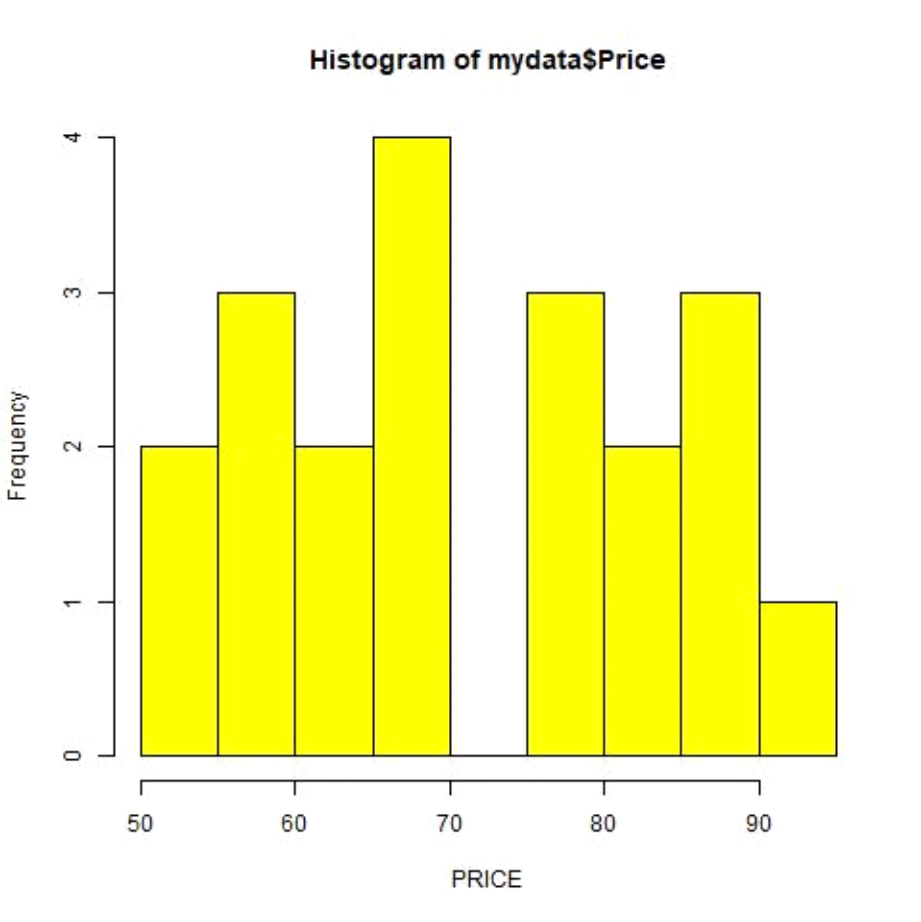


d) Draw the histogram of age ,price and floor area

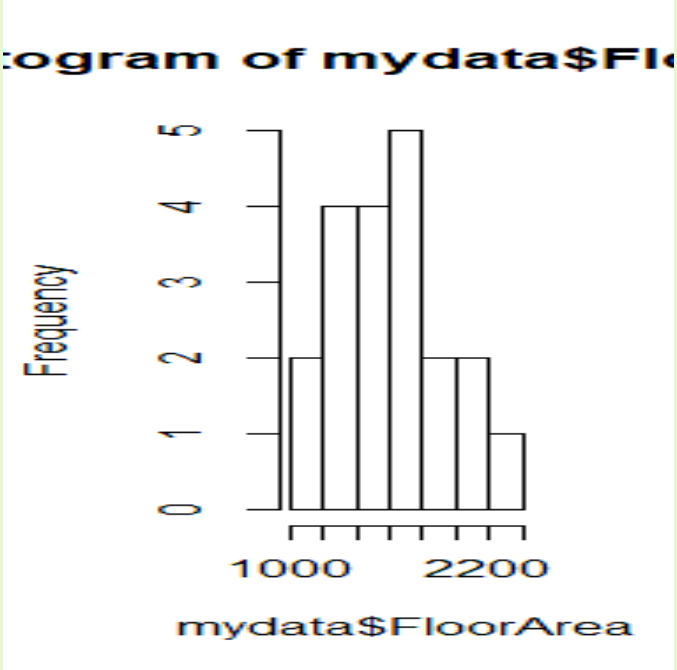
- hist(a$Age, freq=F)



- hist(a$Price, freq=F)

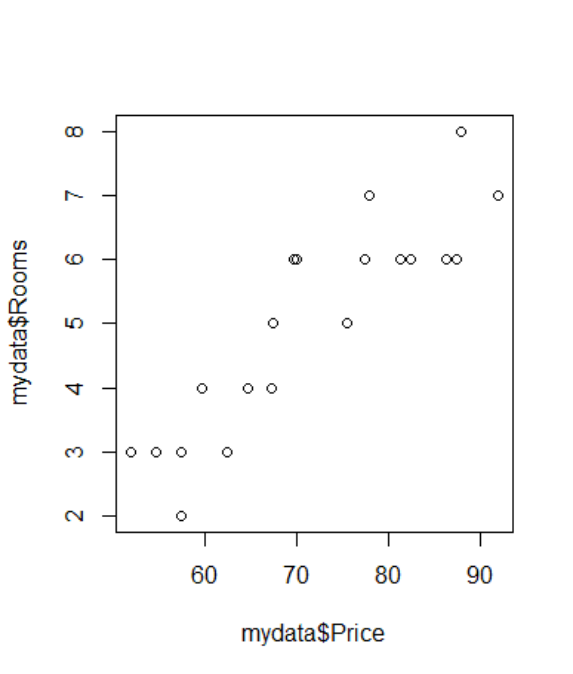


- hist(a$FloorArea, freq=F)

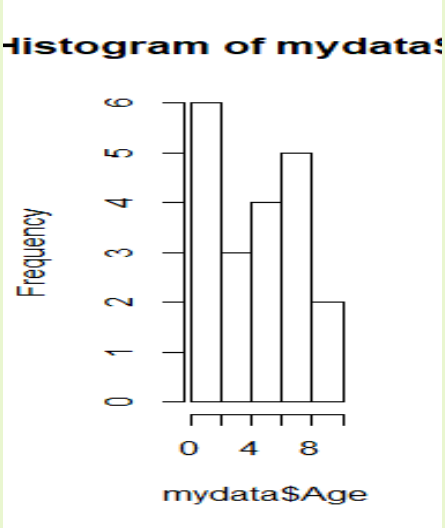


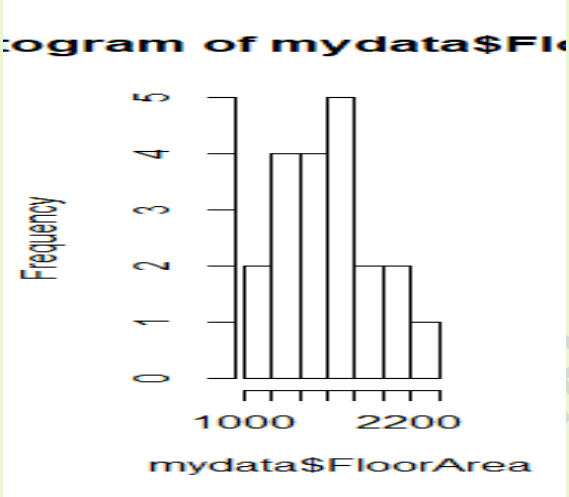
f) Draw all the graphs in (c), (d), and (e) in the same graph paper.

> plot(a$Price, a$FloorArea, col=3, pch=2)



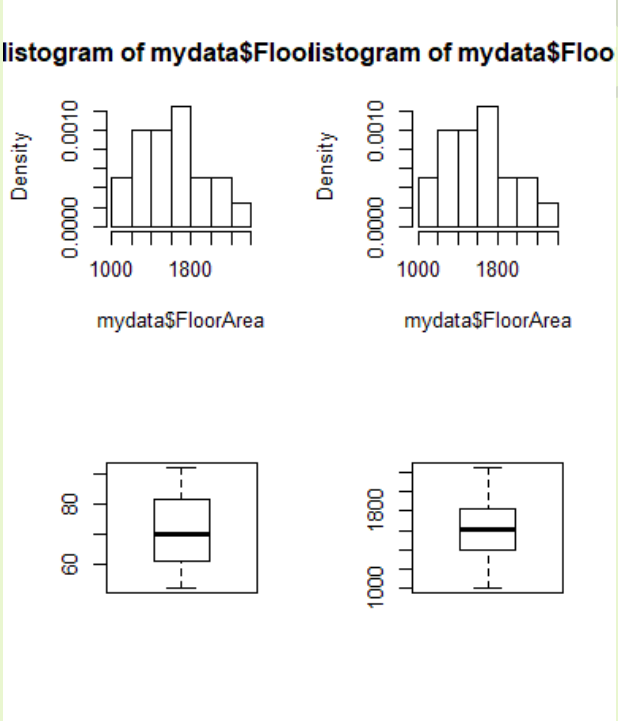
- hist(a$FloorArea, freq=F)

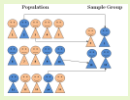




- hist(a$Age, freq=F)

- boxplot(a$Age)





Using Datasets for Random Sampling with R

1. Augmenting the file and saving the resultant file:

a) Calculate the value per square foot area of each apartment and store it in a vector named "PriceSqFt".

```
> a<-read.csv ("house_data_1.csv")
> a
   Price FloorArea Rooms Age CentralHeating
1  52.00      1225     3 6.2            no
2  54.75      1230     3 7.5            no
3  57.50      1200     3 4.2            no
4  57.50      1000     2 8.8            no
5  59.75      1420     4 1.9           yes
6  62.50      1450     3 5.2            no
7  64.75      1380     4 6.6           yes
8  67.25      1510     4 2.3            no
9  67.50      1400     5 6.1            no
10 69.75      1550     6 9.2            no
11 70.00      1720     6 4.3           yes
12 75.50      1700     5 4.3            no
13 77.50      1660     6 1.0           yes
14 78.00      1800     7 7.0           yes
15 81.25      1830     6 3.6           yes
16 82.50      1790     6 1.7           yes
17 86.25      2010     6 1.2           yes
18 87.50      2000     6 0.0           yes
19 88.00      2100     8 2.3           yes
20 92.00      2240     7 0.7           yes
> PriceSqFt<-c(sum(a$Price)/sum(a$FloorArea))
```

b) Place this vector after the last column in the data file.

```
> a<-cbind(a, PriceSqFt)
> a
   Price  FloorArea Rooms Age CentralHeating  PriceSqFt
1  52.00      1225     3         6.2            no        0.04444358
2  54.75      1230     3         7.5            no        0.04444358
3  57.50      1200     3         4.2            no        0.04444358
4  57.50      1000     2         8.8            no        0.04444358
5  59.75      1420     4         1.9           yes        0.04444358
6  62.50      1450     3         5.2            no        0.04444358
7  64.75      1380     4         6.6           yes        0.04444358
8  67.25      1510     4         2.3            no        0.04444358
9  67.50      1400     5         6.1            no        0.04444358
10 69.75      1550     6         9.2            no        0.04444358
11 70.00      1720     6         4.3           yes        0.04444358
12 75.50      1700     5         4.3            no        0.04444358
13 77.50      1660     6         1.0           yes        0.04444358
14 78.00      1800     7         7.0           yes        0.04444358
15 81.25      1830     6         3.6           yes        0.04444358
16 82.50      1790     6         1.7           yes        0.04444358
17 86.25      2010     6         1.2           yes        0.04444358
18 87.50      2000     6         0.0           yes        0.04444358
19 88.00      2100     8         2.3           yes        0.04444358
20 92.00      2240     7         0.7           yes        0.04444358
```

c) Save the augmented file under name “HouseInfo.txt”.

```
> write.table(a, "HouseInfo.txt", row.names=F, col.names=T, sep="\t")
> getwd()
[1] "C:/Users/user/Desktop/MAT1004/29_08_2018"
> dir()
[1] "Box plot.pdf"                    "Hands on exercise on R Day 2.pdf"
[3] "Hands on exercise on R Day 3.pdf" "house_data_1.csv"
[5] "HouseInfo.txt"                   "standard deviation formula.png"
[7] "Thumbs.db"
```

d) Read the file "HouseInfo.txt".

```
> b<-read.delim("HouseInfo.txt")
> b
   Price  FloorArea Rooms Age CentralHeating  PriceSqFt
```

1	52.00	1225	3	6.2	no	0.04444358
2	54.75	1230	3	7.5	no	0.04444358
3	57.50	1200	3	4.2	no	0.04444358
4	57.50	1000	2	8.8	no	0.04444358
5	59.75	1420	4	1.9	yes	0.04444358
6	62.50	1450	3	5.2	no	0.04444358
7	64.75	1380	4	6.6	yes	0.04444358
8	67.25	1510	4	2.3	no	0.04444358
9	67.50	1400	5	6.1	no	0.04444358
10	69.75	1550	6	9.2	no	0.04444358
11	70.00	1720	6	4.3	yes	0.04444358
12	75.50	1700	5	4.3	no	0.04444358
13	77.50	1660	6	1.0	yes	0.04444358
14	78.00	1800	7	7.0	yes	0.04444358
15	81.25	1830	6	3.6	yes	0.04444358
16	82.50	1790	6	1.7	yes	0.04444358
17	86.25	2010	6	1.2	yes	0.04444358
18	87.50	2000	6	0.0	yes	0.04444358
19	88.00	2100	8	2.3	yes	0.04444358
20	92.00	2240	7	0.7	yes	0.04444358

2. Matrices and arrays

a) Matrices and arrays are represented as vectors with dimensions: Create one matrix x with 1 to 12 numbers with 3X4 order.

```
> x<-1:12
> x
 [1]  1  2  3  4  5  6  7  8  9 10 11 12
> dim(x)<-c(3,4)
> x
     [,1] [,2] [,3] [,4]
[1,]    1    4    7   10
[2,]    2    5    8   11
[3,]    3    6    9   12
```

b) Create same matrix with *matrix* function

```
> matrix(1:12, nrow=3)
     [,1] [,2] [,3] [,4]
[1,]    1    4    7   10
```

```
[2,]    2    5    8   11
[3,]    3    6    9   12
> matrix(1:12, nrow=3, byrow=T)
     [,1] [,2] [,3] [,4]
[1,]    1    2    3    4
[2,]    5    6    7    8
[3,]    9   10   11   12
```

c) Give name of rows of this matrix with A,B,C.

```
> m <- matrix(1:12, nrow=3, byrow=T)
> rownames(m)<- c("A", "B", "C")
> m
  [,1] [,2] [,3] [,4]
A    1    2    3    4
B    5    6    7    8
C    9   10   11   12
```

OR

```
> rownames(m)<- LETTERS[1:3]
> m
  [,1] [,2] [,3] [,4]
A    1    2    3    4
B    5    6    7    8
C    9   10   11   12
> colnames(m)<-letters[1:4]
> m
   a  b  c  d
A  1  2  3  4
B  5  6  7  8
C  9 10 11 12
```

d) Transpose of the matrix

```
> t(m)
  A B  C
a 1 5  9
b 2 6 10
c 3 7 11
```

d 4 8 12

e) Use functions *cbind* and *rbind* separately to create different matrices.

```
> rbind(A=1:4,B=5:8,C=9:12)
  [,1] [,2] [,3] [,4]
A    1    2    3    4
B    5    6    7    8
C    9   10   11   12
> cbind(A=1:4,B=5:8,C=9:12)
     A B  C
[1,] 1 5  9
[2,] 2 6 10
[3,] 3 7 11
[4,] 4 8 12
```

f) Use arbitrary numbers to create matrix

```
> x<-c(2,5,4,9,3,8,1,2,0)
> matrix(x,nrow=3,byrow=T)
     [,1] [,2] [,3]
[1,]    2    5    4
[2,]    9    3    8
[3,]    1    2    0
> y<-cbind(A=c(3,6,7), B=c(4,7,5), C=c(1,1,2))
     A B C
[1,] 3 4 1
[2,] 6 7 1
[3,] 7 5 2
```

g) Verify matrix multiplication.

```
> x %*% y
       A  B  C
[1,] 67 76 13
[2,] 47 51 12
[3,] 60 72 12
```

3. Random sampling

a) In R, you can simulate these situations with the *sample* function. Pick five numbers at random from the set 1:40.

```
> sample(1:40, 5)
[1] 30 34 23 22 31

> sample(1:40, 5)
[1] 34 39  9 19 27
```

b) Notice that the default behaviour of *sample* is *sampling without replacement*. That is, the samples will not contain the same number twice, and size obviously cannot be bigger than the length of the vector to be sampled. If you want sampling with replacement, then you need to add the argument *replace=TRUE*. Sampling with replacement is suitable for modelling coin tosses or throws of a die. So, for instance, simulate 10 coin tosses.

```
> sample(1:40, 5, replace=T)
[1] 20 19  2 16 27
> sample(1:40, 5, replace=T)
[1] 24 14 11 38 30

> sample(c("H", "T"),10, replace=T)
 [1] "T" "H" "H" "T" "T" "H" "T" "T" "T" "T"
```

c) In fair coin-tossing, the probability of heads should equal the probability of tails, but the idea of a random event is not restricted to symmetric cases. It could be equally well applied to other cases, such as the successful outcome of a surgical procedure. Hopefully, there would be a better than 50% chance of this. Simulate data with nonequal probabilities for the outcomes (say, a 90% chance of success) by using the *prob* argument to sample.

```
> sample(c("Succ", "Fail"),10, replace=T, prob=c(0.90, 0.10))
 [1] "Succ" "Succ" "Succ" "Succ" "Succ" "Succ" "Succ" "Succ" "Succ"
"Succ"
> sample(c("Succ", "Fail"),10, replace=T, prob=c(0.90, 0.10))
 [1] "Succ" "Succ" "Succ" "Succ" "Succ" "Fail" "Succ" "Succ" "Succ"
"Succ"
```

d) The *choose* function can be used to calculate the following express.
40C5= 40!/35!*5!

```
> choose(40,5)
[1] 658008

> factorial(40)/(prod(factorial(5),factorial(35)))
[1] 658008
```

e) Find 5!

```
> factorial(5)
[1] 120
```

Vectors and Data Frame creations with R

1. Combine these individual vectors into a list mentioning 'before' and 'after' and store in 'mylist'.

```
> intake.pre<-c(5260,5470,5640,6180,6390,6515,6805,7515,7515,8230,8770)

>  intake.post<-c(3910,4220,3885,5160,5645,4680,5265,5975,6790,6900,7335)

 > mylist=list(before=intake.pre,after=intake.post)

 > mylist

 $`before`

 [1] 5260 5470 5640 6180 6390 6515 6805 7515

 [9] 7515 8230 8770


 $after

 [1] 3910 4220 3885 5160 5645 4680 5265 5975

 [9] 6790 6900 7335
```

2. Extract before components from mylist.

```
> bfre=mylist$before
> bfre
[1] 5260 5470 5640 6180 6390 6515 6805 7515
[9] 7515 8230 8770
```

3. Create data frames from pre-existing variables and store in d.

```
> d<-data.frame(intake.pre,intake.post)
> d
   intake.pre intake.post
1        5260        3910
2        5470        4220
3        5640        3885
4        6180        5160
5        6390        5645
6        6515        4680
7        6805        5265
8        7515        5975
9        7515        6790
10       8230        6900
11       8770        7335
```

4. Create data with pre intake from d.

```
> Data<-d$intake.pre
> Data
 [1] 5260 5470 5640 6180 6390 6515 6805
 [8] 7515 7515 8230 8770
```

5. Data for more than one woman, for instance nos. 3, 5, and 7.

```
> d[c(3,5,7),]
```

intake.pre intake.post

3 5640 3885

5 6390 5645

7 6805 5265

Conditional selection

6. Medicine intake above 7000 kJ pre-treatment. To combine several expressions, you can use the logical operators & (logical "and"), | (logical "or"), and ! (logical "not").

>d[d[,1]>7000,]

intake.pre intake.post

8 7515 5975

9 7515 6790

10 8230 6900

11 8770 7335

7. Find the post-treatment intake for women with a pre-treatment intake between 7000 and 8000 kJ.

> d[d[,1]>7000 & d[,1]<8000,2]

[1] 5975 6790

8. Logical expression

> d[,1]>7000 & d[,1]<8000

[1] FALSE FALSE FALSE FALSE FALSE FALSE

[7] FALSE TRUE TRUE FALSE FALSE

9. The "pre" measurement for woman no. 5

> d[5,1]

[1] 6390

10. All measurements for woman no. 5

> d[5,]

11. Extract all data for cases that satisfy some criterion, such as women with a pre-treatment intake above 7000 kJ.

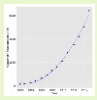
```
> d[d[,1]>7000,]
intake.pre intake.post
8       7515       5975
9       7515       6790
10      8230       6900
11      8770       7335
```

12. It is trivial to sort a vector. Just use the sort function. However, sorting a single vector is not always what is required. Often you need to sort a series of variables according to the values of some other variables — blood pressures sorted by sex and age, for instance.

```
> sort(d$intake.pre)
 [1] 5260 5470 5640 6180 6390 6515 6805 7515
 [9] 7515 8230 8770
> sort(d$intake.post)
 [1] 3885 3910 4220 4680 5160 5265 5645 5975
 [9] 6790 6900 7335
```

13. Compute an ordering of a variable

```
> order(d$intake.pre)
 [1]  1  2  3  4  5  6  7  8  9 10 11
> order(d$intake.post)
 [1]  3  1  2  6  4  7  5  8  9 10 11
```



Understanding HSAUR package in R

Day 5: R Lab Assignment

The data handling and manipulation techniques explained in this chapter will be illustrated by means of a data set of 2000 world leading companies, the Forbes 2000 list for the year 2004 collected by 'Forbes Magazine'.
The data function searches for data objects of the specified name ("Forbes2000") in the package specified via the package("HSAUR").

1. Install this package.

install.packages("HSAUR")

2. One can imagine, printing a list of 2000 companies

library("HSAUR")

Forbes2000

3. It is more useful to look at a description of their structure found
rank: the ranking of the company,
name: the name of the company,
country: the country the company is situated in,
category: a category describing the products the company produces,
sales: the amount of sales of the company in billion US dollars,
profits: the profit of the company in billion US dollars,
assets: the assets of the company in billion US dollars,
marketvalue: the market value of the company in billion US dollars.

> str(Forbes2000)

'data.frame': 2000 obs. of 8 variables:

$ rank : int 1 2 3 4 5 6 7 8 9 10 ...

$ name : chr "Citigroup" "General Electric" "American Intl Group" "ExxonMobil" ...

$ country : Factor w/ 61 levels "Africa","Australia",..: 60 60 60 60 56 60 56 28 60 60 ...

$ category : Factor w/ 27 levels "Aerospace & defense",..: 2 6 16 19 19 2 2 8 9 20 ...

$ sales : num 94.7 134.2 76.7 222.9 232.6 ...

$ profits : num 17.85 15.59 6.46 20.96 10.27 ...

$ assets : num 1264 627 648 167 178 ...

$ marketvalue: num 255 329 195 277 174 ...

4. A similar but more detailed description is available from the help page for the Forbes2000 object:

```
> help("Forbes2000")
```

or

```
> ?Forbes2000
```

5. The name of the class of an object can be determined

```
> class(Forbes2000)
[1] "data.frame"
```

6. The dimensions of such a table can be extracted

```
> dim(Forbes2000)
[1] 2000    8
```

7. The variable names are accessible

```
> names(Forbes2000)
[1] "rank"        "name"        "country"     "category"    "sales"
[6] "profits"     "assets"      "marketvalue"
```

8. Different levels, i.e., business categories, which can be extracted

```
> levels(Forbes2000[, "category"])
 [1] "Aerospace & defense"            "Banking"
 [3] "Business services & supplies"   "Capital goods"
 [5] "Chemicals"                      "Conglomerates"
 [7] "Construction"                   "Consumer durables"
 [9] "Diversified financials"         "Drugs & biotechnology"
[11] "Food drink & tobacco"           "Food markets"
[13] "Health care equipment & services" "Hotels restaurants & leisure"
[15] "Household & personal products"  "Insurance"
[17] "Materials"                      "Media"
[19] "Oil & gas operations"           "Retailing"
[21] "Semiconductors"                 "Software & services"
[23] "Technology hardware & equipment"  "Telecommunications services"
[25] "Trading companies"              "Transportation"
[27] "Utilities"
```

9. As a simple summary statistic, the frequencies of the levels of such a factor variable can be found

```
>table(Forbes2000[,"category"])
```

10. The summary method can be applied to a numeric vector to give a set of useful summary statistics namely the minimum, maximum, mean, median and the 25% and 75% quartiles; Find summery for sales

```
summary(Forbes2000)
     rank           name                country
```

```
Min.   :   1.0   Length:2000        United States :751
1st Qu.: 500.8   Class :character   Japan         :316
Median :1000.5   Mode  :character   United Kingdom:137
Mean   :1000.5                      Germany       : 65
3rd Qu.:1500.2                      France        : 63
Max.   :2000.0                      Canada        : 56
                                    (Other)       :612
               category        sales           profits
Banking              : 313   Min.   :  0.010   Min.   :-25.8300
Diversified financials: 158   1st Qu.:  2.018   1st Qu.:  0.0800
Insurance            : 112   Median :  4.365   Median :  0.2000
Utilities            : 110   Mean   :  9.697   Mean   :  0.3811
Materials            :  97   3rd Qu.:  9.547   3rd Qu.:  0.4400
Oil & gas operations :  90   Max.   :256.330   Max.   : 20.9600
(Other)              :1120                     NA's   :5
    assets         marketvalue
Min.   :   0.270   Min.   :  0.02
1st Qu.:   4.025   1st Qu.:  2.72
Median :   9.345   Median :  5.15
Mean   :  34.042   Mean   : 11.88
3rd Qu.:  22.793   3rd Qu.: 10.60
Max.   :1264.030   Max.   :328.54
```

11. Internally, a data.frame is a list of vectors of a common length n, the number of rows of the table. Each of those vectors represents the measurements of one variable and we have seen that we can access such a variable by its name, for example the names of the companies.

12. The top three companies can be extracted utilising an integer vector of the numbers one to three.

```
> companies<-Forbes2000[,"name"]
> companies[1:3]
[1] "Citigroup"          "General Electric"    "American Intl Group"
```

13. Extract the variables name, sales, profits and assets for the three largest companies.

```
> Forbes2000[c(1:3),c("name","sales","profits","assets")]
                 name  sales profits  assets
```

1 Citigroup 94.71 17.85 1264.03
2 General Electric 134.19 15.59 626.93
3 American Intl Group 76.66 6.46 647.664

14. Compute the ordering of the Top 3 companies' sales

```
> order_sales<-order(Forbes2000$sales)
> companies[order_sales[1:3]]
[1] "Custodia Holding"       "Central European Media" "Minara Resources"
```

15. The indices of the three top sellers are the elements 1998, 1999 and 2000

```
> Forbes2000[order_sales[c(1998:2000)],]
  rank          name        country             category  sales profits
4    4     ExxonMobil  United States Oil & gas operations 222.88   20.96
5    5             BP United Kingdom Oil & gas operations 232.57   10.27
10  10 Wal-Mart Stores  United States           Retailing 256.33    9.05
  assets marketvalue
4 166.99      277.02
5 177.57      173.54
10 104.91      243.74
```

16. The companies with assets of more than 1000 billion US dollars

```
> companies[Forbes2000$assets>1000]
[1] "Citigroup"       "Fannie Mae"      "Mizuho Financial"
```

17. In fact, for some of the companies the measurement of the profits variable are missing. In R, missing values are treated by a special symbol, NA, indicating that this measurement is not available. The observations with profit information missing can be obtained.

```
> companies[is.na(Forbes2000$profits)]
[1] "AMP"                 "HHG"                 "NTL"                 "US Airways Group"
"Laidlaw International"
```

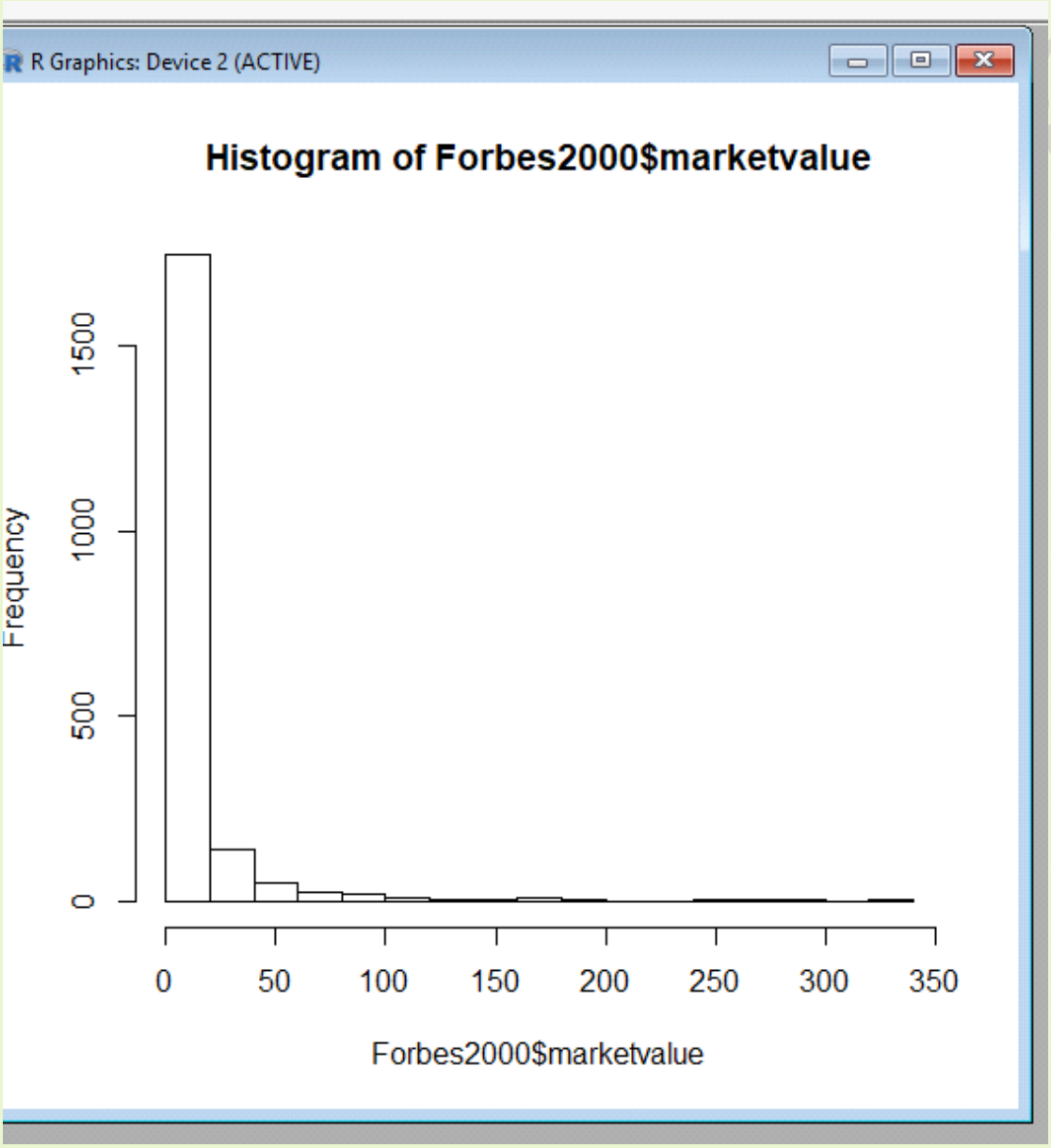
18. We can select a subset of the Forbes 2000 list consisting of all companies situated in the United Kingdom

```
> companies[Forbes2000$country == "United Kingdom"]
  [1] "BP"                    "HSBC Group"            "Royal Bank of Scotland"   "Barclays"
"HBOS"                  "Lloyds TSB Group"
  [7] "GlaxoSmithKline"       "BT Group"              "Prudential"               "AstraZeneca"
"Tesco"                 "Anglo American"
 [13] "British Amer Tobacco"  "National Grid Transco" "Standard Chartered Group"
"Centrica"              "J Sainsbury"           "Scottish Power"
 [19] "Cadbury Schweppes"    "Compass Group"         "Marks & Spencer"          "Old
Mutual"            "Kingfisher"            "Vodafone"
 [25] "GUS"                  "Alliance & Leicester"  "Aviva"                   "Diageo"
"Imperial Tobacco Group" "Northern Rock"
 [31] "Wolseley"             "BG Group"              "SABMiller"                "Legal & General
Group"  "Abbey National"        "Scottish & Southern"
 [37] "Reckitt Benckiser"    "Assoc British Foods"   "Boots"                    "BAA"
"BOC Group"             "WPP"
 [43] "Allied Domecq"        "Safeway Plc"           "Hanson"                   "British Airways"
"Scottish & Newcastle"  "Imperial Chemical Inds"
 [49] "BAE Systems"          "Royal & Sun Alliance"  "Dixons Group"             "mmO2"
"Hilton Group"          "United Utilities"
 [55] "British Sky Broadcasting" "Bradford & Bingley"    "Gallaher Group"       "Land
Securities Group"  "Rolls-Royce"           "Friends Provident"
 [61] "Wm Morrison Supermarkets" "Pearson"               "Willis Group Holdings"  "Smiths
Group"          "Alliance UniChem"      "Cable & Wireless"
 [67] "Rentokil Initial"     "Man Group Plc"         "British Land"             "Next"
"InterContinental Hotels"  "Exel"
 [73] "RMC Group"            "Corus Group"           "Severn Trent"             "GKN"
"Tomkins"               "Amersham"
 [79] "Reuters Group"        "Whitbread Holdings"    "Johnson Matthey"          "HHG"
"EMI Group"             "Smith & Nephew"
 [85] "George Wimpey"        "Bunzl"                 "Xstrata"                  "Taylor Woodrow"
"Rexam"                 "Liberty International"
 [91] "Tate & Lyle Group"    "Invensys"              "Kesa Electricals"         "Barratt
Developments"  "Kelda Group"           "Mitchells & Butlers"
 [97] "Enterprise Inns"      "Persimmon"             "Slough Estates"           "Rank Group"
"ITV"                   "Amdocs"
[103] "Amvescap"             "International Power"   "Signet Group"             "Daily Mail &
General"  "Peninsular & Oriental"  "Hammerson"
[109] "Inchcape"             "William Hill Org"      "Sage Group"               "Pilkington"
"Hays"                  "Canary Wharf Group"
[115] "FirstGroup"           "Provident Financial Plc"  "Schroders"            "EMAP"
"Shire Pharmaceuticals"  "Travis Perkins"
[121] "AWG"                  "Britannic Group"       "Marconi"                  "BPB"
"Big Food Group"        "LogicaCMG"
[127] "Somerfield"           "Arriva"                "MyTravel Group"           "Northumbrian
Water"      "Balfour Beatty"        "Antofagasta"
```

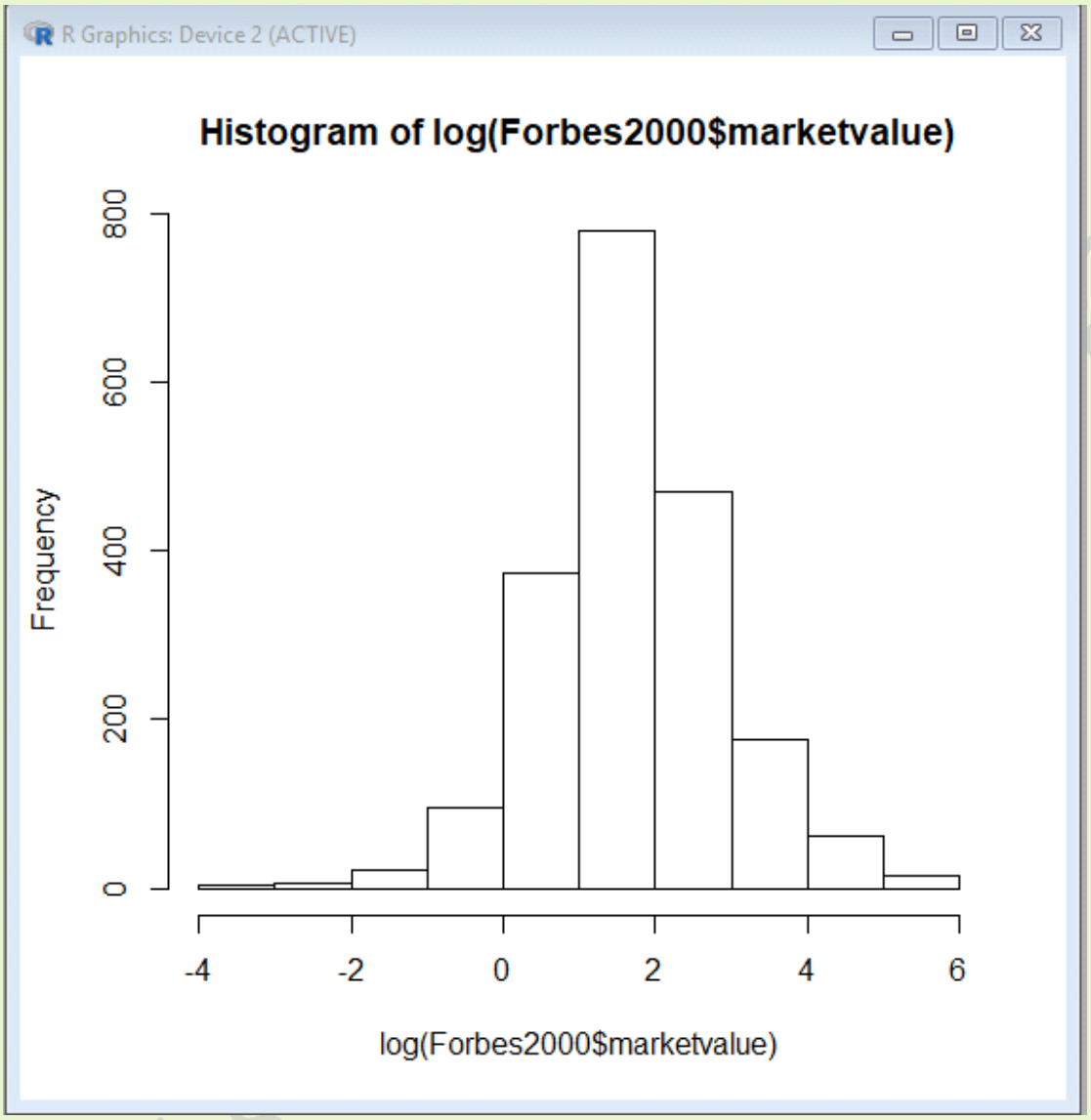
[133] "Associated British Ports" "Berkeley Group" "ICAP" "Punch Taverns" "AMEC"

19. Draw histograms of the market value and the logarithm of the market value for the companies contained in the Forbes 2000 list.

> hist(Forbes2000$marketvalue)



> hist(log(Forbes2000$marketvalue))


R Graphics: Device 2 (ACTIVE)
Histogram of log(Forbes2000$marketvalue)
Frequency
0
200
400
600
800
-4
-2
0
2
4
6
log(Forbes2000$marketvalue)

 Probability , Matrices and Arrays in R

Q1. Matrices and arrays
a) Matrices and arrays are represented as vectors with dimensions:
Create one matrix x with 1 to 12 numbers with 3X4 order.
Ans:

```
> x <- 1:12
> x
 [1]  1  2  3  4  5  6  7  8  9 10 11 12
> dim(x) <- c(3,4)
> x
     [,1] [,2] [,3] [,4]
[1,]    1    4    7   10
[2,]    2    5    8   11
[3,]    3    6    9   12
```

b) Create same matrix with matrix function.
Ans:

```
> matrix(1:12, nrow=3, ncol=4)
     [,1] [,2] [,3] [,4]
[1,]    1    4    7   10
[2,]    2    5    8   11
[3,]    3    6    9   12
> matrix(1:12, nrow=3, ncol=4, byrow=T)
     [,1] [,2] [,3] [,4]
[1,]    1    2    3    4
[2,]    5    6    7    8
[3,]    9   10   11   12
```

c) Give name of rows of this matrix with A,B,C.
Ans:

```
> rownames(x) <- LETTERS[1:3]
> x
  [,1] [,2] [,3] [,4]
A    1    4    7   10
B    2    5    8   11
C    3    6    9   12
```

```
> colnames(x) <- letters[1:4]
> x
  a b c  d
A 1 4 7 10
B 2 5 8 11
C 3 6 9 12
```

d) Transpose of the matrix.

Ans:

```
> t(x)
   A  B  C
a  1  2  3
b  4  5  6
c  7  8  9
d 10 11 12
```

e) Use functions cbind and rbind separately to create different matrices.

Ans:

```
> cbind(A=1:4, B=5:8, C=13:16)
     A B  C
[1,] 1 5 13
[2,] 2 6 14
[3,] 3 7 15
[4,] 4 8 16
```

f) Use arbitrary numbers to create matrix.

Ans:

```
> rbind(a=c(2,4,6,3), b=c(43,67,54,12), c=c(21,59,89,66), d=c(43,8,1,7))
  [,1] [,2] [,3] [,4]
a    2    4    6    3
b   43   67   54   12
c   21   59   89   66
d   43    8    1    7
```

g) Verify matrix multiplication.

Ans:

```
> P <- matrix(1:16, nrow=4, ncol=4)
> Q <- matrix(1:8, nrow=4, ncol=2)
> P
     [,1] [,2] [,3] [,4]
[1,]    1    5    9   13
[2,]    2    6   10   14
[3,]    3    7   11   15
[4,]    4    8   12   16
> Q
```

```
     [,1] [,2]
[1,]    1    5
[2,]    2    6
[3,]    3    7
[4,]    4    8
> P%*%Q
     [,1] [,2]
[1,]   90  202
[2,]  100  228
[3,]  110  254
[4,]  120  280
```

Q2. Random sampling

a) In R, you can simulate these situations with the sample function. Pick five numbers at random from the set 1:40.

Ans:

```
> sample(1:40, 5)
[1] 33 24 19  6 20
> sample(1:40, 5)
[1]  8 25 37 16 38
> sample(1:40, 5)
[1] 14 17 21 20  8
```

b) Notice that the default behaviour of sample is sampling without replacement. That is, the samples will not contain the same number twice, and size obviously cannot be bigger than the length of the vector to be sampled. If you want sampling with replacement, then you need to add the argument replace=TRUE.

Sampling with replacement is suitable for modelling coin tosses or throws of a die. So, for instance, simulate 10 coin tosses.

Ans:

```
> sample(c("H","T"), replace=T)
[1] "T" "H"
> sample(c("H","T"), 10, replace=T)
 [1] "H" "T" "T" "H" "H" "H" "H" "H" "H" "T"
> sample(c("H","T"), 10, replace=T)
 [1] "H" "T" "T" "H" "H" "T" "H" "H" "H" "T"
```

c) In fair coin-tossing, the probability of heads should equal the probability of tails, but the idea of a random event is not restricted to symmetric cases. It could be equally well applied to other cases, such as the successful outcome of a surgical procedure. Hopefully, there would be a better than 50% chance of this. Simulate data with non-equal probabilities for the outcomes (say, a 90% chance of success) by using the prob argument to sample.

Ans:

```
> sample(c("H","T"), 10, replace=T, prob=c(30,70))
 [1] "T" "T" "T" "T" "T" "H" "H" "H" "T" "T"
> sample(c("H","T"), 7, replace=T, prob=c(23,77))
```

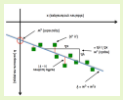
[1] "T" "T" "T" "T" "T" "H" "T"
> sample(c("H","T"), 7, replace=T, prob=c(0.9, 0.1))
[1] "H" "H" "H" "H" "H" "H" "H"

d) The choose function can be used to calculate the following express.

$$\binom{40}{5} = \frac{40!}{5!35!}$$

Ans:
> choose(40,5)
[1] 658008

e) Find 5!
Ans:
> factorial(5)
[1] 120
> gamma(5+1)
[1] 120
> prod(5:1)
[1] 120



Linear Regression : A Case Study

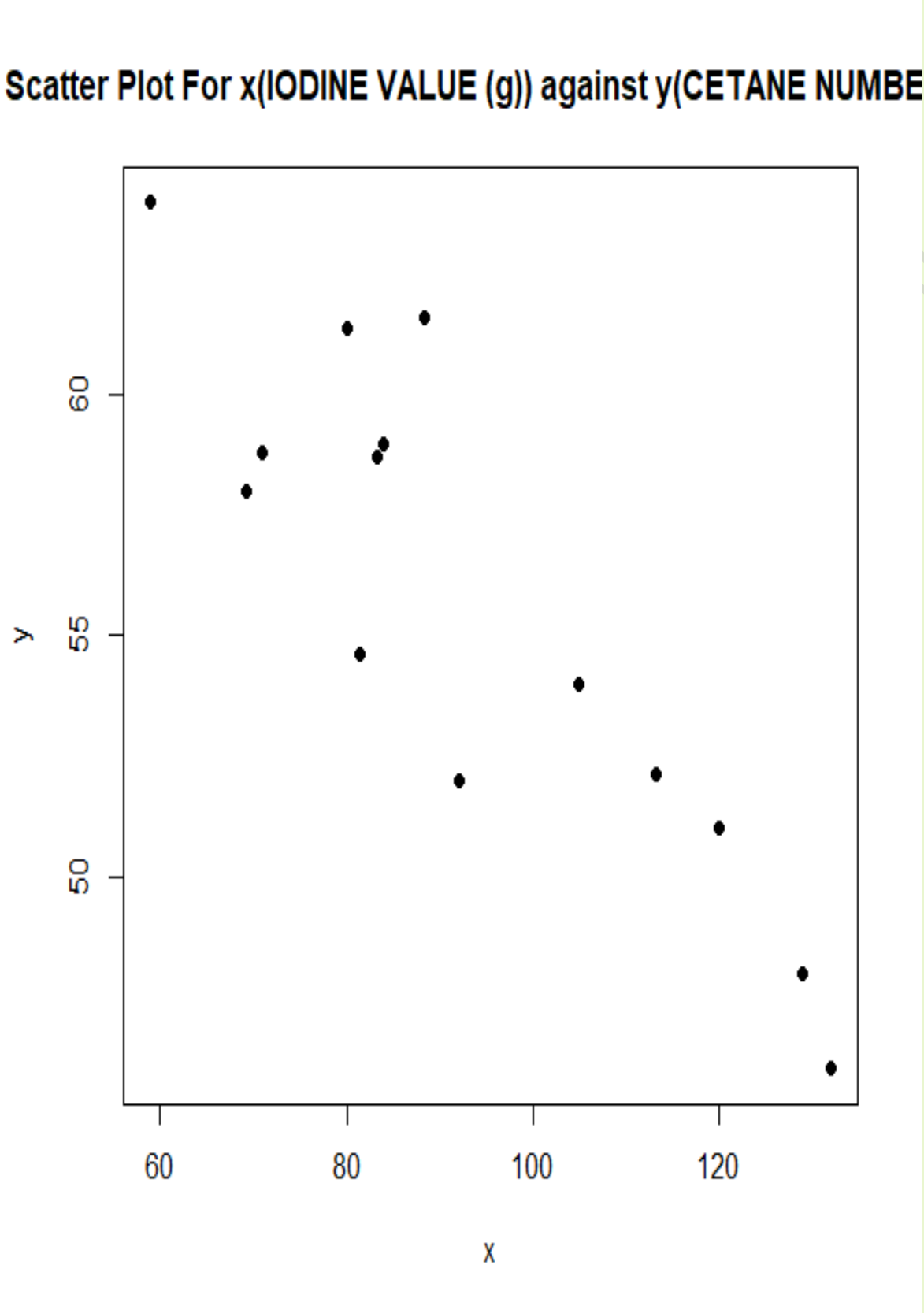
The cetane number is a critical property in specifying the ignition quality of a fuel used in a diesel engine. Determination of this number for a biodiesel fuel is expensive and time-consuming. The article "Relating the Cetane Number of Biodiesel Fuels to Their Fatty Acid Composition: A Critical Study" (*J. of Automobile Engr.*, 2009: 565–583) included the following data on x = iodine value (g) and y = cetane number for a sample of 14 biofuels. The iodine value is the amount of iodine necessary to saturate a sample of 100 g of oil. The article's authors fit the simple linear regression model to this data, so let's follow their lead.

x	132.0	129.0	120.0	113.2	105.0	92.0	84.0	83.2	88.4	59.0	80.0	81.5	71.0	69.2
y	46.0	48.0	51.0	52.1	54.0	52.0	59.0	58.7	61.6	64.0	61.4	54.6	58.8	58.0

Code-

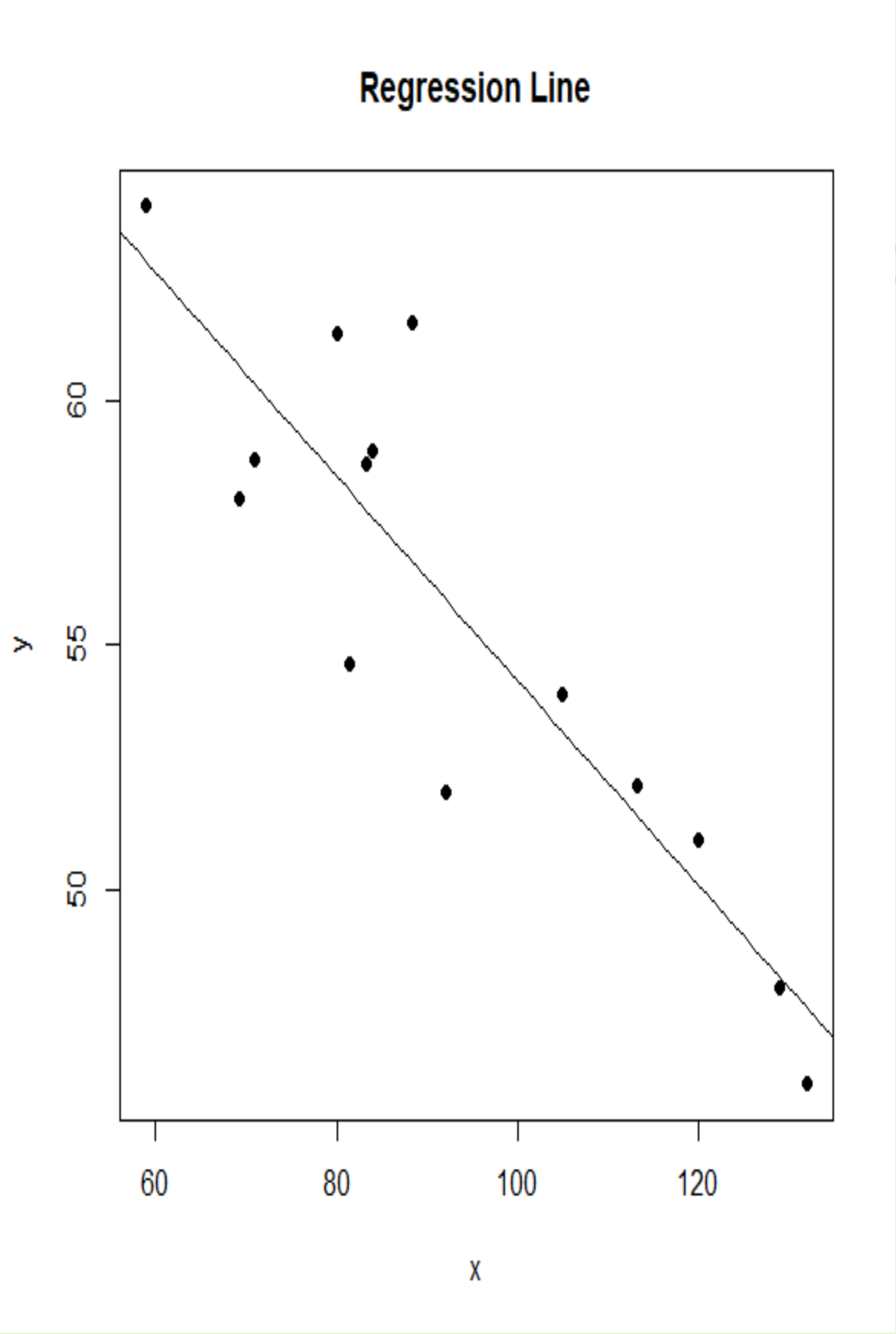
Q.1-Scatter Plot

```
y<-c(46.0,48.0,51.0,52.1,54.0,52.0,59.0,58.7,61.6,64.0,61.4,
54.6,58.8,58.0)
> y
 [1] 46.0 48.0 51.0 52.1 54.0 52.0 59.0 58.7 61.6 64.0 61.4
54.6 58.8 58.0
> x <- c(132.0,129.0,120.0,113.2,105.0,92.0,84.0,83.2,88.4,5
9.0,80.0,81.5,71.0,69.2)
> x
 [1] 132.0 129.0 120.0 113.2 105.0  92.0  84.0  83.2  88.4
59.0  80.0  81.5  71.0  69.2
>plot(x,y,pch=19,main="Scatter Plot For x(IODINE VALUE (g))
against y(CETANE NUMBERS)")
```


Scatter Plot For x(IODINE VALUE (g)) against y(CETANE NUMBE
y
50
55
60
60
80
100
120
X

Q.2-Find True Regression Line.

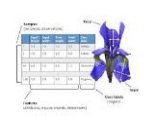
```
y<-
c(46.0,48.0,51.0,52.1,54.0,52.0,59.0,58.7,61.6,64.0,61.4,54.
6,58.8,58.0)

x <-
c(132.0,129.0,120.0,113.2,105.0,92.0,84.0,83.2,88.4,59.0,80.
0,81.5,71.0,69.2)

sxx<-sum(x*x)-((sum(x)*sum(x)/length(x)))

sxy<-sum(x*y)-((sum(x)*sum(y))/length(x))

b1=sxy/sxx

b0=mean(y)-b1*mean(x)

x1<-c(0:200)

y1<-b1*x1 + b0

lines(x1,y1,pch=".")

plot(x,y,pch=19,main="Regression Line")
```


Regression Line
y
60
55
50
60
80
100
120
x

Q.3-Predict the value of y for given x=90 for the following problem.

```
x<-90
yp<-b0+b1*x
```

```
> x<-90
> yp<-b0+b1*x
> yp
[1] 56.36756
```



PETAL DATASET ANALYSIS

Experiment with 'iris' data.

1. Analyze how petal width depends on petal length using linear model function.
2. Plot the observed data along with the regression line in one graph.
3. Calculate multiple correlation co-e coefficients for all lengths and widths.

Q.1-Analyze how petal width depends on petal length using linear model function.

```
linear_model<-lm(Petal.Width~Petal.Length,data=iris)
linear_model
```

Call:

lm(formula = Petal.Width ~ Petal.Length, data = iris)

Coefficients:

(Intercept)	*Petal.Length*
-0.3631	*0.4158*

Here we have used Linear Model function to analyze the dependency of Petal width on petal length.

The **lm ()** commands provides us with two values-

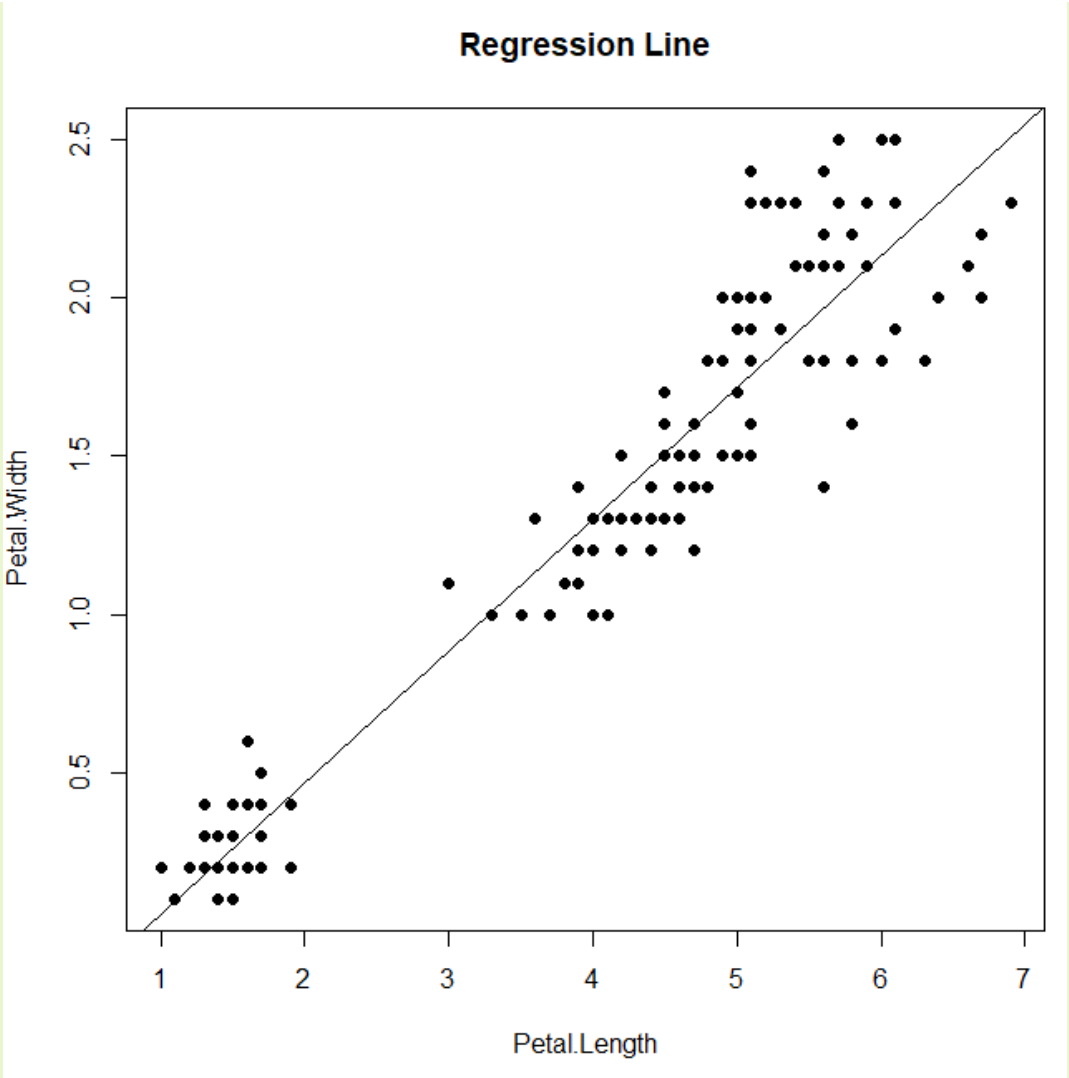
Intercept (-0.3631) - The value (expected) of petal width for petal length=0.

Petal. Length (0.4158)-The slope of the line plotted between petal width and petal length.

Using these values we can plot the **regression line** for these values as in the following

Q.2-Plot the observed data along with the regression line in one graph.

```
plot(iris$Petal.Width~iris$Sepal.Length,pch=19,main="Regression Line")
abline(linear_model)
```



Q.3-Calculate multiple correlation co-e coefficients for all lengths and widths.

```
cor(iris[,c(1,2,3,4)])
```

```
             Sepal.Length Sepal.Width Petal.Length Petal.Width
Sepal.Length    1.0000000  -0.1175698    0.8717538   0.8179411
Sepal.Width    -0.1175698   1.0000000   -0.4284401  -0.3661259
Petal.Length    0.8717538  -0.4284401    1.0000000   0.9628654
Petal.Width     0.8179411  -0.3661259    0.9628654   1.0000000
```

CASE STUDY – HYPOTHESIS

Two new methods for producing a tire have been proposed. To ascertain which is superior, a tire manufacturer produces a sample of 10 tires using the ?rst method and a sample of 8 using the second. The ?rst set is to be road tested at location A and the second at location B. It is known from past experience that the lifetime of a tire that is road tested at one of these locations is normally distributed with a mean life due to the tire but with a variance due (for the most part) to the location. Speci?cally, it is known that the lifetimes of tires tested at location A are normal with standard deviation equal to 4,000 kilometers, whereas those tested at location B are normal with s = 6,000 kilometers. If the manufacturer is interested in testing the hypothesis that there is no appreciable difference

in the mean life of tires produced by either method, what conclusion should be drawn at

the 5 percent level of signi?cance if the resulting data are as given in Table 8.3?

TABLE 8.3

Tire Lives in Units of 100 Kilometers

Tires Tested at A	Tires Tested at B
61.1	62.2
58.2	56.6
62.3	66.4
64	56.2
59.7	57.4
66.2	58.4
57.8	57.6
61.4	65.4
62.2	
63.6	

Tires Lives in units of 100 kilometers								
Tires Tested at A		Tires Tested atB				z-Test: Two Sample for Means		
61.1		62.2		61.65				
58.2		56.6		60.025			*Variable 1*	*Variable 2*
62.3		66.4		6.849444		Mean	61.65	60.025
64		56.2		16.57643		Known Variance	6.848	16.576
59.7		57.4				Observations	10	8
66.2		58.4				Hypothesized Mean Differen	1.625	
57.8		57.6				z	-4.3E-15	
61.4		65.4				P(Z<=z) one-tail	0.5	
62.2						z Critical one-tail	1.644854	
63.6						P(Z<=z) two-tail	1	
						z Critical two-tail	1.959964	