Day 1 – R Programming

```
> #install.packages('caret')
> num = 10
> num
[1] 10
> library('caret')
> x = 10.2
> y <- 10
> z = "Hello"
> x
[1] 10.2
> y
[1] 10
> z
[1] "Hello"
> as.integer(x)
[1] 10
> a = 1 + 10i
> a
[1] 1+10i
> sqrt(144)
[1] 12
> a = 5; b = 15
> out = a > b
> out
[1] FALSE
> age <- c(21, 25, 28, 30, 20, 26)
> age
[1] 21 25 28 30 20 26
> id = c(1:10)  #range values from 1-10
> id
 [1]  1  2  3  4  5  6  7  8  9 10
> seq(1, 20)
 [1]  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20
> seq(2, 20, 2)   #range values from 2 to 20 with offset 2
 [1]  2  4  6  8 10 12 14 16 18 20
> loan_default <- c(TRUE, FALSE, FALSE, TRUE, TRUE)
> loan_default
[1]  TRUE FALSE FALSE  TRUE  TRUE
> place_names <- c("China", "India", "Denmark", "UK", "Finland")
> place_names
[1] "China"   "India"   "Denmark" "UK"      "Finland"
> class(loan_default)
[1] "logical"
```

```
> class(age)
[1] "numeric"
> class(z)
[1] "character"
> num_as_str <- c("10", "30", "40", "50")
> class(num_as_str)
[1] "character"
> numbers <- as.integer(num_as_str)
> class(numbers)
[1] "integer"
> mean(numbers)
[1] 32.5
> max(age)
[1] 30
> min(numbers)
[1] 10
> median(age)
[1] 25.5
> range(numbers)
[1] 10 50
> var(age)
[1] 15.2
> sort(age)
[1] 20 21 25 26 28 30
> sort(age, decreasing = TRUE)
[1] 30 28 26 25 21 20
> random_ele <- c(15, 2.5, TRUE, "Hello")
> random_ele
[1] "15"    "2.5"   "TRUE"  "Hello"
> class(random_ele)
[1] "character"
> mat <- c(1:16)
> mat <- matrix(mat, ncol=4)
> mat
     [,1] [,2] [,3] [,4]
[1,]    1    5    9   13
[2,]    2    6   10   14
[3,]    3    7   11   15
[4,]    4    8   12   16
> mat1 <- c(1:16)
> mat1 <- matrix(mat1, ncol = 4, byrow = T)
> mat1
     [,1] [,2] [,3] [,4]
[1,]    1    2    3    4
```

```
[2,]    5    6    7    8
[3,]    9   10   11   12
[4,]   13   14   15   16
> matrix(c(56, 72, 25, 14, 87, 99), ncol = 3, byrow = T)
     [,1] [,2] [,3]
[1,]   56   72   25
[2,]   14   87   99
> mat1[2,]
[1] 5 6 7 8
> mat1[2,2]
[1] 6
> mat1[,4]
[1]  4  8 12 16
> matr = matrix(c(5:16), nrow = 3, byrow = TRUE)
> column.names <- c("COL1", "COL2", "COL3")
> row.names <- c("ROW1", "ROW2", "ROW3")
> column.names <- c("COL1", "COL2", "COL3", "COL4")
> result <- matrix(c(5:16), nrow = 3, byrow = TRUE, dimnames = list(row.names, column.names))
> result
     COL1 COL2 COL3 COL4
ROW1    5    6    7    8
ROW2    9   10   11   12
ROW3   13   14   15   16
> employee = list(1, c("John", "Rose"), c(12000, 15000))
> employee
[[1]]
[1] 1

[[2]]
[1] "John" "Rose"

[[3]]
[1] 12000 15000

> employee[[1]]
[1] 1
> employee[[2]]
[1] "John" "Rose"
> employee[[3]]
[1] 12000 15000
> employee = list(EmpID=1, EmpName=c("John", "Rose"), basic_pay=c(12000, 15000))
> employee
$EmpID
[1] 1
```

```
$EmpName
[1] "John" "Rose"

$basic_pay
[1] 12000 15000

> employee$EmpName
[1] "John" "Rose"
> list_of_expenses <- list(100, 150, 350, 50)
> class((list_of_expenses))
[1] "list"
> expenses <- unlist(list_of_expenses)
> class(expenses)
[1] "numeric"
> length(expenses)
[1] 4
> days_from_purchase <- c(10, 15, 20, 25)
> days_from_purchase
[1] 10 15 20 25
> ctf <- as.factor(days_from_purchase)
> typeof(ctf)
[1] "integer"
> class(ctf)
[1] "factor"
> age <- c(21, 42, 28, 31, 19)
> names <- c("John", "Sachin", "Rahul", "Ravi", "Sameer")
> salary <- c(12000, 20000, 25000, 16000, 28000)
> ownhouse <- c(TRUE, FALSE, TRUE, TRUE, FALSE)
> mydf <- data.frame(names, age, salary, ownhouse)
> mydf
   names age salary ownhouse
1   John  21  12000     TRUE
2 Sachin  42  20000    FALSE
3  Rahul  28  25000     TRUE
4   Ravi  31  16000     TRUE
5 Sameer  19  28000    FALSE
> stock_price <- c(110.55, 102.50, 145.90, 130.70, 160.45, 112.80)
> stock_mat <- matrix(stock_price, ncol = 2, byrow = T)
> stock_df = data.frame(stock_mat)
> stock_df
      X1    X2
1 110.55 102.5
2 145.90 130.7
```

```
3 160.45 112.8
> colnames(stock_df) <- c("Open Price", "Close Price")
> letters[1:10]
 [1] "a" "b" "c" "d" "e" "f" "g" "h" "i" "j"
> letters[1:26]
 [1] "a" "b" "c" "d" "e" "f" "g" "h" "i" "j" "k" "l" "m" "n" "o" "p" "q" "r" "s" "t" "u" "v" "w" "x" "y" "z"
> rownames(stock_df) <- letters[1:3]
> stock_df
  Open Price Close Price
a     110.55       102.5
b     145.90       130.7
c     160.45       112.8
> stock_df$`Close Price`
[1] 102.5 130.7 112.8
```

Day 2 – R Programming

```
> X <- matrix(c(50, 70, 40, 90, 60, 80, 50, 90, 100, 50, 30, 70), nrow = 3)
> X
     [,1] [,2] [,3] [,4]
[1,]   50   90   50   50
[2,]   70   60   90   30
[3,]   40   80  100   70
> rowSums(X)
[1] 240 250 290
> colSums(X)
[1] 160 230 240 150
> X <- rbind(X, apply(X, 2, mean))  #Add a row and apply mean function columnswise - 2, for rowwise its 
1
> X
         [,1]     [,2] [,3] [,4]
[1,] 50.00000 90.00000   50   50
[2,] 70.00000 60.00000   90   30
[3,] 40.00000 80.00000  100   70
[4,] 53.33333 76.66667   80   50
> X <- cbind(X, apply(X, 1, var))   #Add a column and apply variance function rowwise - 1
> X
         [,1]     [,2] [,3] [,4]     [,5]
[1,] 50.00000 90.00000   50   50 400.0000
[2,] 70.00000 60.00000   90   30 625.0000
[3,] 40.00000 80.00000  100   70 625.0000
[4,] 53.33333 76.66667   80   50 240.7407
> X <- matrix(c(50, 70, 40, 90, 60, 80, 50, 90, 100, 50, 30, 70), nrow = 3)
> X <- cbind(X, apply(X, 1, sd))    #Add a column and apply standard deviation function rowwise - 1
> X
```

```
     [,1] [,2] [,3] [,4] [,5]
[1,]   50   90   50   50   20
[2,]   70   60   90   30   25
[3,]   40   80  100   70   25
> X <- rbind(X, apply(X, 2, max))   #Add a row and apply maximum function columnswise - 2, for rowwise
its 1
> X
     [,1] [,2] [,3] [,4] [,5]
[1,]   50   90   50   50   20
[2,]   70   60   90   30   25
[3,]   40   80  100   70   25
[4,]   70   90  100   70   25
> stock_df[[1]]   #1st column
[1] 110.55 145.90 160.45
> stock_df[[2]]   #2nd column
[1] 102.5 130.7 112.8
> stock_df
  Open Price Close Price BuyOrSell
a     110.55       102.5      Sell
b     145.90       130.7      Sell
c     160.45       112.8      Sell
> stock_df[1:2, 2]
[1] 102.5 130.7
> stock_df[1:3, 1:2]
  Open Price Close Price
a     110.55       102.5
b     145.90       130.7
c     160.45       112.8
> stock_df[, 1:2]
  Open Price Close Price
a     110.55       102.5
b     145.90       130.7
c     160.45       112.8
> stock_df[c(1, 3), 1:2]
  Open Price Close Price
a     110.55       102.5
c     160.45       112.8
> stock_df[-1, 1]
[1] 145.90 160.45
> stock_df[-c(1, 3), 1:2]
  Open Price Close Price
b      145.9       130.7
> v_sub <- stock_df[1:3, 2]
> v_sub
```

```
[1] 102.5 130.7 112.8
> df_subsetdata <- stock_df[1:3, 2, drop=F]
> df_subsetdata
  Close Price
a       102.5
b       130.7
c       112.8
> class(v_sub)
[1] "numeric"
> class(df_subsetdata)
[1] "data.frame"
> setwd("C:/zubeda/PGA02_Zubu/R Programming")  #Set current working directory
> housing_df <- read.csv("Housing.csv")
> housing_df
     price  area bedrooms bathrooms stories mainroad guestroom basement
1  13300000  7420        4         2       3      yes        no       no
2  12250000  8960        4         4       4      yes        no       no
3  12250000  9960        3         2       2      yes        no      yes
4  12215000  7500        4         2       2      yes        no      yes
5  11410000  7420        4         1       2      yes       yes      yes
6  10850000  7500        3         3       1      yes        no      yes
7  10150000  8580        4         3       4      yes        no       no
8  10150000 16200        5         3       2      yes        no       no
9   9870000  8100        4         1       2      yes       yes      yes
10  9800000  5750        3         2       4      yes       yes       no
11  9800000 13200        3         1       2      yes        no      yes
12  9681000  6000        4         3       2      yes       yes      yes
13  9310000  6550        4         2       2      yes        no       no
14  9240000  3500        4         2       2      yes        no       no
15  9240000  7800        3         2       2      yes        no       no
16  9100000  6000        4         1       2      yes        no      yes
17  9100000  6600        4         2       2      yes       yes      yes
18  8960000  8500        3         2       4      yes        no       no
19  8890000  4600        3         2       2      yes       yes       no
20  8855000  6420        3         2       2      yes        no       no
21  8750000  4320        3         1       2      yes        no      yes
22  8680000  7155        3         2       1      yes       yes      yes
23  8645000  8050        3         1       1      yes       yes      yes
24  8645000  4560        3         2       2      yes       yes      yes
25  8575000  8800        3         2       2      yes        no       no
26  8540000  6540        4         2       2      yes       yes      yes
27  8463000  6000        3         2       4      yes       yes      yes
28  8400000  8875        3         1       1      yes        no       no
29  8400000  7950        5         2       2      yes        no      yes
```

30	8400000	5500	4	2	2	yes	no	yes
31	8400000	7475	3	2	4	yes	no	no
32	8400000	7000	3	1	4	yes	no	no
33	8295000	4880	4	2	2	yes	no	no
34	8190000	5960	3	3	2	yes	yes	yes
35	8120000	6840	5	1	2	yes	yes	yes
36	8080940	7000	3	2	4	yes	no	no
37	8043000	7482	3	2	3	yes	no	no
38	7980000	9000	4	2	4	yes	no	no
39	7962500	6000	3	1	4	yes	yes	no
40	7910000	6000	4	2	4	yes	no	no
41	7875000	6550	3	1	2	yes	no	yes
42	7840000	6360	3	2	4	yes	no	no
43	7700000	6480	3	2	4	yes	no	no
44	7700000	6000	4	2	4	yes	no	no
45	7560000	6000	4	2	4	yes	no	no
46	7560000	6000	3	2	3	yes	no	no
47	7525000	6000	3	2	4	yes	no	no
48	7490000	6600	3	1	4	yes	no	no
49	7455000	4300	3	2	2	yes	no	yes
50	7420000	7440	3	2	1	yes	yes	yes
51	7420000	7440	3	2	4	yes	no	no
52	7420000	6325	3	1	4	yes	no	no
53	7350000	6000	4	2	4	yes	yes	no
54	7350000	5150	3	2	4	yes	no	no
55	7350000	6000	3	2	2	yes	yes	no
56	7350000	6000	3	1	2	yes	no	no
57	7343000	11440	4	1	2	yes	no	yes
58	7245000	9000	4	2	4	yes	yes	no
59	7210000	7680	4	2	4	yes	yes	no
60	7210000	6000	3	2	4	yes	yes	no
61	7140000	6000	3	2	2	yes	yes	no
62	7070000	8880	2	1	1	yes	no	no
63	7070000	6240	4	2	2	yes	no	no
64	7035000	6360	4	2	3	yes	no	no
65	7000000	11175	3	1	1	yes	no	yes
66	6930000	8880	3	2	2	yes	no	yes
67	6930000	13200	2	1	1	yes	no	yes
68	6895000	7700	3	2	1	yes	no	no
69	6860000	6000	3	1	1	yes	no	no
70	6790000	12090	4	2	2	yes	no	no
71	6790000	4000	3	2	2	yes	no	yes
72	6755000	6000	4	2	4	yes	no	no
73	6720000	5020	3	1	4	yes	no	no

74	6685000	6600	2	2	4	yes	no	yes
75	6650000	4040	3	1	2	yes	no	yes
76	6650000	4260	4	2	2	yes	no	no

	hotwaterheating	airconditioning	parking	prefarea	furnishingstatus
1	no	yes	2	yes	furnished
2	no	yes	3	no	furnished
3	no	no	2	yes	semi-furnished
4	no	yes	3	yes	furnished
5	no	yes	2	no	furnished
6	no	yes	2	yes	semi-furnished
7	no	yes	2	yes	semi-furnished
8	no	no	0	no	unfurnished
9	no	yes	2	yes	furnished
10	no	yes	1	yes	unfurnished
11	no	yes	2	yes	furnished
12	yes	no	2	no	semi-furnished
13	no	yes	1	yes	semi-furnished
14	yes	no	2	no	furnished
15	no	no	0	yes	semi-furnished
16	no	no	2	no	semi-furnished
17	no	yes	1	yes	unfurnished
18	no	yes	2	no	furnished
19	no	yes	2	no	furnished
20	no	yes	1	yes	semi-furnished
21	yes	no	2	no	semi-furnished
22	no	yes	2	no	unfurnished
23	no	yes	1	no	furnished
24	no	yes	1	no	furnished
25	no	yes	2	no	furnished
26	no	yes	2	yes	furnished
27	no	yes	0	yes	semi-furnished
28	no	no	1	no	semi-furnished
29	yes	no	2	no	unfurnished
30	no	yes	1	yes	semi-furnished
31	no	yes	2	no	unfurnished
32	no	yes	2	no	semi-furnished
33	no	yes	1	yes	furnished
34	no	no	1	no	unfurnished
35	no	yes	1	no	furnished
36	no	yes	2	no	furnished
37	yes	no	1	yes	furnished
38	no	yes	2	no	furnished
39	no	yes	2	no	unfurnished
40	no	yes	1	no	semi-furnished

```
41           no          yes     0     yes       furnished
42           no          yes     0     yes       furnished
43           no          yes     2      no     unfurnished
44           no           no     2      no   semi-furnished
45           no          yes     1      no       furnished
46           no          yes     0      no   semi-furnished
47           no          yes     1      no       furnished
48           no          yes     3     yes       furnished
49           no           no     1      no     unfurnished
50           no          yes     0     yes   semi-furnished
51           no           no     1     yes     unfurnished
52           no          yes     1      no     unfurnished
53           no          yes     1      no       furnished
54           no          yes     2      no   semi-furnished
55           no          yes     1      no   semi-furnished
56           no          yes     1      no     unfurnished
57           no           no     1     yes   semi-furnished
58           no          yes     1     yes       furnished
59           no          yes     1      no   semi-furnished
60           no          yes     1      no       furnished
61           no           no     1      no   semi-furnished
62           no          yes     1      no   semi-furnished
63           no          yes     1      no       furnished
64           no          yes     2     yes       furnished
65           no          yes     1     yes       furnished
66           no          yes     1      no       furnished
67          yes           no     1      no       furnished
68           no           no     2      no     unfurnished
69           no          yes     1      no       furnished
70           no           no     2     yes       furnished
71           no          yes     0     yes   semi-furnished
72           no          yes     0      no     unfurnished
73           no          yes     0     yes     unfurnished
74           no           no     0     yes       furnished
75          yes           no     1      no       furnished
76          yes           no     0      no   semi-furnished
 [ reached 'max' / getOption("max.print") -- omitted 469 rows ]
> dim(housing_df)  #no. of rows, no. of columns
[1] 545  13
> filter_df <- housing_df[housing_df$price > 10000000, ]
> filter_df
     price  area bedrooms bathrooms stories mainroad guestroom basement
1 13300000  7420        4         2       3      yes        no       no
2 12250000  8960        4         4       4      yes        no       no
```

```
3 12250000  9960       3       2      2     yes       no     yes
4 12215000  7500       4       2      2     yes       no     yes
5 11410000  7420       4       1      2     yes      yes     yes
6 10850000  7500       3       3      1     yes       no     yes
7 10150000  8580       4       3      4     yes       no      no
8 10150000 16200       5       3      2     yes       no      no
  hotwaterheating airconditioning parking prefarea furnishingstatus
1              no             yes       2      yes        furnished
2              no             yes       3       no        furnished
3              no              no       2      yes   semi-furnished
4              no             yes       3      yes        furnished
5              no             yes       2       no        furnished
6              no             yes       2      yes   semi-furnished
7              no             yes       2      yes   semi-furnished
8              no              no       0       no      unfurnished
> filt_df <- housing_df[housing_df$area > 6000, ]
> filt_df
       price  area bedrooms bathrooms stories mainroad guestroom basement
1   13300000  7420        4         2       3      yes        no       no
2   12250000  8960        4         4       4      yes        no       no
3   12250000  9960        3         2       2      yes        no      yes
4   12215000  7500        4         2       2      yes        no      yes
5   11410000  7420        4         1       2      yes       yes      yes
6   10850000  7500        3         3       1      yes        no      yes
7   10150000  8580        4         3       4      yes        no       no
8   10150000 16200        5         3       2      yes        no       no
9    9870000  8100        4         1       2      yes       yes      yes
11   9800000 13200        3         1       2      yes        no      yes
13   9310000  6550        4         2       2      yes        no       no
15   9240000  7800        3         2       2      yes        no       no
17   9100000  6600        4         2       2      yes       yes      yes
18   8960000  8500        3         2       4      yes        no       no
20   8855000  6420        3         2       2      yes        no       no
22   8680000  7155        3         2       1      yes       yes      yes
23   8645000  8050        3         1       1      yes       yes      yes
25   8575000  8800        3         2       2      yes        no       no
26   8540000  6540        4         2       2      yes       yes      yes
28   8400000  8875        3         1       1      yes        no       no
29   8400000  7950        5         2       2      yes        no      yes
31   8400000  7475        3         2       4      yes        no       no
32   8400000  7000        3         1       4      yes        no       no
35   8120000  6840        5         1       2      yes       yes      yes
36   8080940  7000        3         2       4      yes        no       no
37   8043000  7482        3         2       3      yes        no       no
```

38	7980000	9000	4	2	4	yes	no	no
41	7875000	6550	3	1	2	yes	no	yes
42	7840000	6360	3	2	4	yes	no	no
43	7700000	6480	3	2	4	yes	no	no
48	7490000	6600	3	1	4	yes	no	no
50	7420000	7440	3	2	1	yes	yes	yes
51	7420000	7440	3	2	4	yes	no	no
52	7420000	6325	3	1	4	yes	no	no
57	7343000	11440	4	1	2	yes	no	yes
58	7245000	9000	4	2	4	yes	yes	no
59	7210000	7680	4	2	4	yes	yes	no
62	7070000	8880	2	1	1	yes	no	no
63	7070000	6240	4	2	2	yes	no	no
64	7035000	6360	4	2	3	yes	no	no
65	7000000	11175	3	1	1	yes	no	yes
66	6930000	8880	3	2	2	yes	no	yes
67	6930000	13200	2	1	1	yes	no	yes
68	6895000	7700	3	2	1	yes	no	no
70	6790000	12090	4	2	2	yes	no	no
74	6685000	6600	2	2	4	yes	no	yes
77	6650000	6420	3	2	3	yes	no	no
78	6650000	6500	3	2	3	yes	no	no
83	6615000	10500	3	2	1	yes	no	yes
86	6510000	8250	3	2	3	yes	no	no
87	6510000	6670	3	1	3	yes	no	yes
89	6475000	7410	3	1	1	yes	yes	yes
90	6440000	8580	5	3	2	yes	no	no
92	6419000	6750	2	1	1	yes	yes	yes
94	6300000	7200	3	2	1	yes	no	yes
97	6300000	9000	3	1	1	yes	no	yes
98	6300000	6400	3	1	1	yes	yes	yes
99	6293000	6600	3	2	3	yes	no	no
101	6230000	6600	3	2	1	yes	no	yes
104	6195000	6350	3	2	3	yes	yes	no
108	6125000	6420	3	1	3	yes	no	yes
110	6090000	6615	4	2	2	yes	yes	no
111	6090000	6600	3	1	1	yes	yes	yes
112	6090000	8372	3	1	3	yes	no	no
114	6083000	9620	3	1	1	yes	no	yes
115	6020000	6800	2	1	1	yes	yes	yes
116	6020000	8000	3	1	1	yes	yes	yes
117	6020000	6900	3	2	1	yes	yes	yes
119	5950000	6420	3	1	1	yes	no	yes
120	5950000	7020	3	1	1	yes	no	yes

121	5950000	6540	3	1	1	yes	yes	yes
122	5950000	7231	3	1	2	yes	yes	yes
123	5950000	6254	4	2	1	yes	no	yes
124	5950000	7320	4	2	2	yes	no	no
125	5950000	6525	3	2	4	yes	no	no
126	5943000	15600	3	1	1	yes	no	no

	hotwaterheating	airconditioning	parking	prefarea	furnishingstatus
1	no	yes	2	yes	furnished
2	no	yes	3	no	furnished
3	no	no	2	yes	semi-furnished
4	no	yes	3	yes	furnished
5	no	yes	2	no	furnished
6	no	yes	2	yes	semi-furnished
7	no	yes	2	yes	semi-furnished
8	no	no	0	no	unfurnished
9	no	yes	2	yes	furnished
11	no	yes	2	yes	furnished
13	no	yes	1	yes	semi-furnished
15	no	no	0	yes	semi-furnished
17	no	yes	1	yes	unfurnished
18	no	yes	2	no	furnished
20	no	yes	1	yes	semi-furnished
22	no	yes	2	no	unfurnished
23	no	yes	1	no	furnished
25	no	yes	2	no	furnished
26	no	yes	2	yes	furnished
28	no	no	1	no	semi-furnished
29	yes	no	2	no	unfurnished
31	no	yes	2	no	unfurnished
32	no	yes	2	no	semi-furnished
35	no	yes	1	no	furnished
36	no	yes	2	no	furnished
37	yes	no	1	yes	furnished
38	no	yes	2	no	furnished
41	no	yes	0	yes	furnished
42	no	yes	0	yes	furnished
43	no	yes	2	no	unfurnished
48	no	yes	3	yes	furnished
50	no	yes	0	yes	semi-furnished
51	no	no	1	yes	unfurnished
52	no	yes	1	no	unfurnished
57	no	no	1	yes	semi-furnished
58	no	yes	1	yes	furnished
59	no	yes	1	no	semi-furnished

```
62   no  yes  1  no  semi-furnished
63   no  yes  1  no  furnished
64   no  yes  2  yes  furnished
65   no  yes  1  yes  furnished
66   no  yes  1  no  furnished
67   yes  no  1  no  furnished
68   no  no  2  no  unfurnished
70   no  no  2  yes  furnished
74   no  no  0  yes  furnished
77   no  yes  0  yes  furnished
78   no  yes  0  yes  furnished
83   no  yes  1  yes  furnished
86   no  yes  0  no  furnished
87   no  no  0  yes  unfurnished
89   no  yes  2  yes  unfurnished
90   no  no  2  no  furnished
92   no  no  2  yes  furnished
94   no  yes  3  no  semi-furnished
97   no  no  1  yes  furnished
98   no  yes  1  yes  semi-furnished
99   no  yes  0  yes  unfurnished
101  no  yes  0  yes  unfurnished
104  no  yes  0  no  furnished
108  no  no  0  yes  unfurnished
110  yes  no  1  no  semi-furnished
111  no  no  2  yes  semi-furnished
112  no  yes  2  no  unfurnished
114  no  no  2  yes  furnished
115  no  no  2  no  furnished
116  no  yes  2  yes  semi-furnished
117  no  no  0  yes  unfurnished
119  no  yes  0  yes  furnished
120  no  yes  2  yes  semi-furnished
121  no  no  2  yes  furnished
122  no  yes  0  yes  semi-furnished
123  no  no  1  yes  semi-furnished
124  no  no  0  no  furnished
125  no  no  1  no  furnished
126  no  yes  2  no  semi-furnished
 [ reached 'max' / getOption("max.print") -- omitted 81 rows ]
> price <- 5
> if(price > 5) {
+   print("Sell the stock")
+ } else {
```

```
+   print("Buy the stock")
+ }
[1] "Buy the stock"
> source("Conditional.R")
[1] "Buy the stock"
> stock_df
  Open Price Close Price BuyOrSell
a     110.55       102.5      Sell
b     145.90       130.7      Sell
c     160.45       112.8      Sell
> stock_df$BuyOrSell <- ifelse(stock_df$`Close Price` < 80, "Buy", "Sell")
> stock_df
  Open Price Close Price BuyOrSell
a     110.55       102.5      Sell
b     145.90       130.7      Sell
c     160.45       112.8      Sell
> for (x in 1:10) { print(x ^ 2) } #i raised to 2
[1] 1
[1] 4
[1] 9
[1] 16
[1] 25
[1] 36
[1] 49
[1] 64
[1] 81
[1] 100
> mtcars   #inbuilt dataset
                   mpg cyl  disp  hp drat    wt  qsec vs am gear carb
Mazda RX4         21.0   6 160.0 110 3.90 2.620 16.46  0  1    4    4
Mazda RX4 Wag     21.0   6 160.0 110 3.90 2.875 17.02  0  1    4    4
Datsun 710        22.8   4 108.0  93 3.85 2.320 18.61  1  1    4    1
Hornet 4 Drive    21.4   6 258.0 110 3.08 3.215 19.44  1  0    3    1
Hornet Sportabout 18.7   8 360.0 175 3.15 3.440 17.02  0  0    3    2
Valiant           18.1   6 225.0 105 2.76 3.460 20.22  1  0    3    1
Duster 360        14.3   8 360.0 245 3.21 3.570 15.84  0  0    3    4
Merc 240D         24.4   4 146.7  62 3.69 3.190 20.00  1  0    4    2
Merc 230          22.8   4 140.8  95 3.92 3.150 22.90  1  0    4    2
Merc 280          19.2   6 167.6 123 3.92 3.440 18.30  1  0    4    4
Merc 280C         17.8   6 167.6 123 3.92 3.440 18.90  1  0    4    4
Merc 450SE        16.4   8 275.8 180 3.07 4.070 17.40  0  0    3    3
Merc 450SL        17.3   8 275.8 180 3.07 3.730 17.60  0  0    3    3
Merc 450SLC       15.2   8 275.8 180 3.07 3.780 18.00  0  0    3    3
Cadillac Fleetwood 10.4   8 472.0 205 2.93 5.250 17.98  0  0    3    4
```

```
Lincoln Continental 10.4   8 460.0 215 3.00 5.424 17.82  0  0    3    4
Chrysler Imperial   14.7   8 440.0 230 3.23 5.345 17.42  0  0    3    4
Fiat 128            32.4   4  78.7  66 4.08 2.200 19.47  1  1    4    1
Honda Civic         30.4   4  75.7  52 4.93 1.615 18.52  1  1    4    2
Toyota Corolla      33.9   4  71.1  65 4.22 1.835 19.90  1  1    4    1
Toyota Corona       21.5   4 120.1  97 3.70 2.465 20.01  1  0    3    1
Dodge Challenger    15.5   8 318.0 150 2.76 3.520 16.87  0  0    3    2
AMC Javelin         15.2   8 304.0 150 3.15 3.435 17.30  0  0    3    2
Camaro Z28          13.3   8 350.0 245 3.73 3.840 15.41  0  0    3    4
Pontiac Firebird    19.2   8 400.0 175 3.08 3.845 17.05  0  0    3    2
Fiat X1-9           27.3   4  79.0  66 4.08 1.935 18.90  1  1    4    1
Porsche 914-2       26.0   4 120.3  91 4.43 2.140 16.70  0  1    5    2
Lotus Europa        30.4   4  95.1 113 3.77 1.513 16.90  1  1    5    2
Ford Pantera L      15.8   8 351.0 264 4.22 3.170 14.50  0  1    5    4
Ferrari Dino        19.7   6 145.0 175 3.62 2.770 15.50  0  1    5    6
Maserati Bora       15.0   8 301.0 335 3.54 3.570 14.60  0  1    5    8
Volvo 142E          21.4   4 121.0 109 4.11 2.780 18.60  1  1    4    2
> iris     #inbuilt dataset
   Sepal.Length Sepal.Width Petal.Length Petal.Width    Species
1          5.1         3.5          1.4         0.2     setosa
2          4.9         3.0          1.4         0.2     setosa
3          4.7         3.2          1.3         0.2     setosa
4          4.6         3.1          1.5         0.2     setosa
5          5.0         3.6          1.4         0.2     setosa
6          5.4         3.9          1.7         0.4     setosa
7          4.6         3.4          1.4         0.3     setosa
8          5.0         3.4          1.5         0.2     setosa
9          4.4         2.9          1.4         0.2     setosa
10         4.9         3.1          1.5         0.1     setosa
11         5.4         3.7          1.5         0.2     setosa
12         4.8         3.4          1.6         0.2     setosa
13         4.8         3.0          1.4         0.1     setosa
14         4.3         3.0          1.1         0.1     setosa
15         5.8         4.0          1.2         0.2     setosa
16         5.7         4.4          1.5         0.4     setosa
17         5.4         3.9          1.3         0.4     setosa
18         5.1         3.5          1.4         0.3     setosa
19         5.7         3.8          1.7         0.3     setosa
20         5.1         3.8          1.5         0.3     setosa
21         5.4         3.4          1.7         0.2     setosa
22         5.1         3.7          1.5         0.4     setosa
23         4.6         3.6          1.0         0.2     setosa
24         5.1         3.3          1.7         0.5     setosa
25         4.8         3.4          1.9         0.2     setosa
```

26	5.0	3.0	1.6	0.2	setosa
27	5.0	3.4	1.6	0.4	setosa
28	5.2	3.5	1.5	0.2	setosa
29	5.2	3.4	1.4	0.2	setosa
30	4.7	3.2	1.6	0.2	setosa
31	4.8	3.1	1.6	0.2	setosa
32	5.4	3.4	1.5	0.4	setosa
33	5.2	4.1	1.5	0.1	setosa
34	5.5	4.2	1.4	0.2	setosa
35	4.9	3.1	1.5	0.2	setosa
36	5.0	3.2	1.2	0.2	setosa
37	5.5	3.5	1.3	0.2	setosa
38	4.9	3.6	1.4	0.1	setosa
39	4.4	3.0	1.3	0.2	setosa
40	5.1	3.4	1.5	0.2	setosa
41	5.0	3.5	1.3	0.3	setosa
42	4.5	2.3	1.3	0.3	setosa
43	4.4	3.2	1.3	0.2	setosa
44	5.0	3.5	1.6	0.6	setosa
45	5.1	3.8	1.9	0.4	setosa
46	4.8	3.0	1.4	0.3	setosa
47	5.1	3.8	1.6	0.2	setosa
48	4.6	3.2	1.4	0.2	setosa
49	5.3	3.7	1.5	0.2	setosa
50	5.0	3.3	1.4	0.2	setosa
51	7.0	3.2	4.7	1.4	versicolor
52	6.4	3.2	4.5	1.5	versicolor
53	6.9	3.1	4.9	1.5	versicolor
54	5.5	2.3	4.0	1.3	versicolor
55	6.5	2.8	4.6	1.5	versicolor
56	5.7	2.8	4.5	1.3	versicolor
57	6.3	3.3	4.7	1.6	versicolor
58	4.9	2.4	3.3	1.0	versicolor
59	6.6	2.9	4.6	1.3	versicolor
60	5.2	2.7	3.9	1.4	versicolor
61	5.0	2.0	3.5	1.0	versicolor
62	5.9	3.0	4.2	1.5	versicolor
63	6.0	2.2	4.0	1.0	versicolor
64	6.1	2.9	4.7	1.4	versicolor
65	5.6	2.9	3.6	1.3	versicolor
66	6.7	3.1	4.4	1.4	versicolor
67	5.6	3.0	4.5	1.5	versicolor
68	5.8	2.7	4.1	1.0	versicolor
69	6.2	2.2	4.5	1.5	versicolor

70	5.6	2.5	3.9	1.1	versicolor
71	5.9	3.2	4.8	1.8	versicolor
72	6.1	2.8	4.0	1.3	versicolor
73	6.3	2.5	4.9	1.5	versicolor
74	6.1	2.8	4.7	1.2	versicolor
75	6.4	2.9	4.3	1.3	versicolor
76	6.6	3.0	4.4	1.4	versicolor
77	6.8	2.8	4.8	1.4	versicolor
78	6.7	3.0	5.0	1.7	versicolor
79	6.0	2.9	4.5	1.5	versicolor
80	5.7	2.6	3.5	1.0	versicolor
81	5.5	2.4	3.8	1.1	versicolor
82	5.5	2.4	3.7	1.0	versicolor
83	5.8	2.7	3.9	1.2	versicolor
84	6.0	2.7	5.1	1.6	versicolor
85	5.4	3.0	4.5	1.5	versicolor
86	6.0	3.4	4.5	1.6	versicolor
87	6.7	3.1	4.7	1.5	versicolor
88	6.3	2.3	4.4	1.3	versicolor
89	5.6	3.0	4.1	1.3	versicolor
90	5.5	2.5	4.0	1.3	versicolor
91	5.5	2.6	4.4	1.2	versicolor
92	6.1	3.0	4.6	1.4	versicolor
93	5.8	2.6	4.0	1.2	versicolor
94	5.0	2.3	3.3	1.0	versicolor
95	5.6	2.7	4.2	1.3	versicolor
96	5.7	3.0	4.2	1.2	versicolor
97	5.7	2.9	4.2	1.3	versicolor
98	6.2	2.9	4.3	1.3	versicolor
99	5.1	2.5	3.0	1.1	versicolor
100	5.7	2.8	4.1	1.3	versicolor
101	6.3	3.3	6.0	2.5	virginica
102	5.8	2.7	5.1	1.9	virginica
103	7.1	3.0	5.9	2.1	virginica
104	6.3	2.9	5.6	1.8	virginica
105	6.5	3.0	5.8	2.2	virginica
106	7.6	3.0	6.6	2.1	virginica
107	4.9	2.5	4.5	1.7	virginica
108	7.3	2.9	6.3	1.8	virginica
109	6.7	2.5	5.8	1.8	virginica
110	7.2	3.6	6.1	2.5	virginica
111	6.5	3.2	5.1	2.0	virginica
112	6.4	2.7	5.3	1.9	virginica
113	6.8	3.0	5.5	2.1	virginica

```
114        5.7        2.5        5.0        2.0  virginica
115        5.8        2.8        5.1        2.4  virginica
116        6.4        3.2        5.3        2.3  virginica
117        6.5        3.0        5.5        1.8  virginica
118        7.7        3.8        6.7        2.2  virginica
119        7.7        2.6        6.9        2.3  virginica
120        6.0        2.2        5.0        1.5  virginica
121        6.9        3.2        5.7        2.3  virginica
122        5.6        2.8        4.9        2.0  virginica
123        7.7        2.8        6.7        2.0  virginica
124        6.3        2.7        4.9        1.8  virginica
125        6.7        3.3        5.7        2.1  virginica
126        7.2        3.2        6.0        1.8  virginica
127        6.2        2.8        4.8        1.8  virginica
128        6.1        3.0        4.9        1.8  virginica
129        6.4        2.8        5.6        2.1  virginica
130        7.2        3.0        5.8        1.6  virginica
131        7.4        2.8        6.1        1.9  virginica
132        7.9        3.8        6.4        2.0  virginica
133        6.4        2.8        5.6        2.2  virginica
134        6.3        2.8        5.1        1.5  virginica
135        6.1        2.6        5.6        1.4  virginica
136        7.7        3.0        6.1        2.3  virginica
137        6.3        3.4        5.6        2.4  virginica
138        6.4        3.1        5.5        1.8  virginica
139        6.0        3.0        4.8        1.8  virginica
140        6.9        3.1        5.4        2.1  virginica
141        6.7        3.1        5.6        2.4  virginica
142        6.9        3.1        5.1        2.3  virginica
143        5.8        2.7        5.1        1.9  virginica
144        6.8        3.2        5.9        2.3  virginica
145        6.7        3.3        5.7        2.5  virginica
146        6.7        3.0        5.2        2.3  virginica
147        6.3        2.5        5.0        1.9  virginica
148        6.5        3.0        5.2        2.0  virginica
149        6.2        3.4        5.4        2.3  virginica
150        5.9        3.0        5.1        1.8  virginica
> names(mtcars)  #variable/column names
 [1] "mpg"  "cyl"  "disp" "hp"   "drat" "wt"   "qsec" "vs"   "am"   "gear" "carb"
> for (c in names(mtcars)) { print(c) }
[1] "mpg"
[1] "cyl"
[1] "disp"
[1] "hp"
```

```
[1] "drat"
[1] "wt"
[1] "qsec"
[1] "vs"
[1] "am"
[1] "gear"
[1] "carb"
> price <- 12.99
> while (price < 15) {
+   price <- price + 1
+   print(price)
+ }
[1] 13.99
[1] 14.99
[1] 15.99
> check_price <- function(x) {
+   if(x > 110) {
+     print("Price beyond threshold")
+   } else {
+     print("Price within threshold")
+   }
+ }
> check_price(200)
[1] "Price beyond threshold"
> myvect <- c(10, 20, 30, NA, 60, 80)
> mean(myvect)
[1] NA
> sd(myvect)
[1] NA
> min(myvect)
[1] NA
> mean(myvect, na.rm = TRUE)
[1] 40
> stock_price <- c(10, 5, 20, 15, 12, 22)
> matrix_form <- matrix(stock_price, ncol = 2, byrow = TRUE)
> matrix_form
     [,1] [,2]
[1,]   10    5
[2,]   20   15
[3,]   12   22
> apply(matrix_form, 1, sum)
[1] 15 35 34
> apply(matrix_form, 2, sum)
[1] 42 42
```

```
> lapply(1:3, function(x) x ^ 2)  #Returns list
[[1]]
[1] 1

[[2]]
[1] 4

[[3]]
[1] 9

> sapply(1:3, function(x) x ^ 2)  #Returns vector
[1] 1 4 9
> l <- lapply(1:3, function(x) x ^ 2)
> class(l)
[1] "list"
> s <- sapply(1:3, function(x) x ^ 2)
> class(s)
[1] "numeric"
> #Initial Date: 1/1/1970
> purchase_on <- 365
> class(purchase_on) <- "Date"  #Convert to Date & Adds 365 days to the default date
> purchase_on
[1] "1971-01-01"
> purchase_on <- -10
> class(purchase_on) <- "Date"  #Convert to Date & Subtracts 10 days from the default date
> purchase_on
[1] "1969-12-22"
> purchase_date <- as.Date(365, origin=as.Date("2015-03-31"))  #365 days added to origin date
> purchase_date
[1] "2016-03-30"
> sale_date <- as.Date(-10, origin=as.Date("2015-02-10"))      #10 days subtracted from origin date
> sale_date
[1] "2015-01-31"
> format(sale_date, "%Y")
[1] "2015"
> format(sale_date, "%m")
[1] "01"
> format(sale_date, "%b")
[1] "Jan"
> format(sale_date, "%B")
[1] "January"
> Sys.Date()
[1] "2022-02-15"
> format(Sys.Date(), "%d/%m/%Y")
```

```
[1] "15/02/2022"
> as.Date("2021/02/04", format="%Y/%m/%d")   #convert a format of date to date type
[1] "2021-02-04"
> as.Date(purchase_date) > as.Date(sale_date)
[1] TRUE
> as.Date(purchase_date) < as.Date(sale_date)
[1] FALSE
> first_date <- "2020-05-16"
> second_date <- "2020-12-24"
> as.Date(first_date) > as.Date(second_date)
[1] FALSE
> as.Date(first_date) < as.Date(second_date)
[1] TRUE
> dim(housing_df)
[1] 545  13
> str(housing_df)
'data.frame':	545 obs. of  13 variables:
 $ price           : int  13300000 12250000 12250000 12215000 11410000 10850000 10150000 10150000
9870000 9800000 ...
 $ area            : int  7420 8960 9960 7500 7420 7500 8580 16200 8100 5750 ...
 $ bedrooms        : int  4 4 3 4 4 3 4 5 4 3 ...
 $ bathrooms       : int  2 4 2 2 1 3 3 3 1 2 ...
 $ stories         : int  3 4 2 2 2 1 4 2 2 4 ...
 $ mainroad        : chr  "yes" "yes" "yes" "yes" ...
 $ guestroom       : chr  "no" "no" "no" "no" ...
 $ basement        : chr  "no" "no" "yes" "yes" ...
 $ hotwaterheating : chr  "no" "no" "no" "no" ...
 $ airconditioning : chr  "yes" "yes" "no" "yes" ...
 $ parking         : int  2 3 2 3 2 2 2 0 2 1 ...
 $ prefarea        : chr  "yes" "no" "yes" "yes" ...
 $ furnishingstatus: chr  "furnished" "furnished" "semi-furnished" "furnished" ...
> summary(housing_df)
     price              area          bedrooms       bathrooms    
 Min.   : 1750000   Min.   : 1650   Min.   :1.000   Min.   :1.000  
 1st Qu.: 3430000   1st Qu.: 3600   1st Qu.:2.000   1st Qu.:1.000  
 Median : 4340000   Median : 4600   Median :3.000   Median :1.000  
 Mean   : 4766729   Mean   : 5151   Mean   :2.965   Mean   :1.286  
 3rd Qu.: 5740000   3rd Qu.: 6360   3rd Qu.:3.000   3rd Qu.:2.000  
 Max.   :13300000   Max.   :16200   Max.   :6.000   Max.   :4.000  
    stories        mainroad          guestroom           basement        
 Min.   :1.000   Length:545         Length:545         Length:545        
 1st Qu.:1.000   Class :character   Class :character   Class :character  
 Median :2.000   Mode  :character   Mode  :character   Mode  :character  
 Mean   :1.806
```

```
 3rd Qu.:2.000
 Max.   :4.000
 hotwaterheating    airconditioning      parking        prefarea
 Length:545         Length:545         Min.   :0.0000   Length:545
 Class :character   Class :character   1st Qu.:0.0000   Class :character
 Mode  :character   Mode  :character   Median :0.0000   Mode  :character
                                       Mean   :0.6936
                                       3rd Qu.:1.0000
                                       Max.   :3.0000
 furnishingstatus
 Length:545
 Class :character
 Mode  :character
```

Day 3 – R Programming

```
> ages <- c(34, 45, 26, 32, 21)
> location <- c("Urban", "Rural", "Urban", "Rural", "Urban")
> tapply(ages, location, mean)    #location wise age mean
Rural Urban
 38.5  27.0
> #history()          #get previous command
> setwd("C:/zubeda/PGA02_Zubu/R Programming")  #Set current working directory
> housing_df = read.csv("Housing.csv")
> housing_df
     price  area bedrooms bathrooms stories mainroad guestroom basement hotwaterheating
airconditioning parking
1  13300000  7420        4         2       3      yes        no       no              no
yes       2
2  12250000  8960        4         4       4      yes        no       no              no
yes       3
3  12250000  9960        3         2       2      yes        no      yes              no
no       2
4  12215000  7500        4         2       2      yes        no      yes              no
yes       3
5  11410000  7420        4         1       2      yes       yes      yes              no
yes       2
6  10850000  7500        3         3       1      yes        no      yes              no
yes       2
7  10150000  8580        4         3       4      yes        no       no              no
yes       2
8  10150000 16200        5         3       2      yes        no       no              no
no       0
9   9870000  8100        4         1       2      yes       yes      yes              no
yes       2
10  9800000  5750        3         2       4      yes       yes       no              no
yes       1
11  9800000 13200        3         1       2      yes        no      yes              no
yes       2
12  9681000  6000        4         3       2      yes       yes      yes             yes
no       2
13  9310000  6550        4         2       2      yes        no       no              no
yes       1
14  9240000  3500        4         2       2      yes        no       no             yes
no       2
15  9240000  7800        3         2       2      yes        no       no              no
no       0
16  9100000  6000        4         1       2      yes        no      yes              no
no       2
17  9100000  6600        4         2       2      yes       yes      yes              no
yes       1
18  8960000  8500        3         2       4      yes        no       no              no
yes       2
```

19	8890000	4600	3	2	2	yes	yes	no	no	yes	2
20	8855000	6420	3	2	2	yes	no	no	no	yes	1
21	8750000	4320	3	1	2	yes	no	yes	yes	no	2
22	8680000	7155	3	2	1	yes	yes	yes	no	yes	2
23	8645000	8050	3	1	1	yes	yes	yes	no	yes	1
24	8645000	4560	3	2	2	yes	yes	yes	no	yes	1
25	8575000	8800	3	2	2	yes	no	no	no	yes	2
26	8540000	6540	4	2	2	yes	yes	yes	no	yes	2
27	8463000	6000	3	2	4	yes	yes	yes	no	yes	0
28	8400000	8875	3	1	1	yes	no	no	no	no	1
29	8400000	7950	5	2	2	yes	no	yes	yes	no	2
30	8400000	5500	4	2	2	yes	no	yes	no	yes	1
31	8400000	7475	3	2	4	yes	no	no	no	yes	2
32	8400000	7000	3	1	4	yes	no	no	no	yes	2
33	8295000	4880	4	2	2	yes	no	no	no	yes	1
34	8190000	5960	3	3	2	yes	yes	yes	no	no	1
35	8120000	6840	5	1	2	yes	yes	yes	no	yes	1
36	8080940	7000	3	2	4	yes	no	no	no	yes	2
37	8043000	7482	3	2	3	yes	no	no	yes	no	1
38	7980000	9000	4	2	4	yes	no	no	no	yes	2
39	7962500	6000	3	1	4	yes	yes	no	no	yes	2
40	7910000	6000	4	2	4	yes	no	no	no	yes	1
41	7875000	6550	3	1	2	yes	no	yes	no	yes	0
42	7840000	6360	3	2	4	yes	no	no	no	yes	0
43	7700000	6480	3	2	4	yes	no	no	no	yes	2
44	7700000	6000	4	2	4	yes	no	no	no	no	2
45	7560000	6000	4	2	4	yes	no	no	no	yes	1
46	7560000	6000	3	2	3	yes	no	no	no	yes	0
47	7525000	6000	3	2	4	yes	no	no	no	yes	1
48	7490000	6600	3	1	4	yes	no	no	no	yes	3
49	7455000	4300	3	2	2	yes	no	yes	no	no	1
50	7420000	7440	3	2	1	yes	yes	yes	no	yes	0
51	7420000	7440	3	2	4	yes	no	no	no	no	1
52	7420000	6325	3	1	4	yes	no	no	no	yes	1
53	7350000	6000	4	2	4	yes	yes	no	no	yes	1
54	7350000	5150	3	2	4	yes	no	no	no	yes	2
55	7350000	6000	3	2	2	yes	yes	no	no	yes	1
56	7350000	6000	3	1	2	yes	no	no	no	yes	1
57	7343000	11440	4	1	2	yes	no	yes	no	no	1
58	7245000	9000	4	2	4	yes	yes	no	no	yes	1
59	7210000	7680	4	2	4	yes	yes	no	no	yes	1
60	7210000	6000	3	2	4	yes	yes	no	no	yes	1
61	7140000	6000	3	2	2	yes	yes	no	no	no	1
62	7070000	8880	2	1	1	yes	no	no	no	yes	1

63	7070000	6240	4	2	2	yes	no	no	no	yes	1
64	7035000	6360	4	2	3	yes	no	no	no	yes	2
65	7000000	11175	3	1	1	yes	no	yes	no	yes	1
66	6930000	8880	3	2	2	yes	no	yes	no	yes	1
67	6930000	13200	2	1	1	yes	no	yes	yes	no	1
68	6895000	7700	3	2	1	yes	no	no	no	no	2
69	6860000	6000	3	1	1	yes	no	no	no	yes	1
70	6790000	12090	4	2	2	yes	no	no	no	no	2
71	6790000	4000	3	2	2	yes	no	yes	no	yes	0
72	6755000	6000	4	2	4	yes	no	no	no	yes	0
73	6720000	5020	3	1	4	yes	no	no	no	yes	0
74	6685000	6600	2	2	4	yes	no	yes	no	no	0
75	6650000	4040	3	1	2	yes	no	yes	yes	no	1
76	6650000	4260	4	2	2	yes	no	no	yes	no	0

	prefarea	furnishingstatus
1	yes	furnished
2	no	furnished
3	yes	semi-furnished
4	yes	furnished
5	no	furnished
6	yes	semi-furnished
7	yes	semi-furnished
8	no	unfurnished
9	yes	furnished
10	yes	unfurnished
11	yes	furnished
12	no	semi-furnished
13	yes	semi-furnished
14	no	furnished
15	yes	semi-furnished
16	no	semi-furnished
17	yes	unfurnished
18	no	furnished
19	no	furnished
20	yes	semi-furnished
21	no	semi-furnished
22	no	unfurnished
23	no	furnished
24	no	furnished
25	no	furnished
26	yes	furnished
27	yes	semi-furnished
28	no	semi-furnished
29	no	unfurnished

30	yes	semi-furnished
31	no	unfurnished
32	no	semi-furnished
33	yes	furnished
34	no	unfurnished
35	no	furnished
36	no	furnished
37	yes	furnished
38	no	furnished
39	no	unfurnished
40	no	semi-furnished
41	yes	furnished
42	yes	furnished
43	no	unfurnished
44	no	semi-furnished
45	no	furnished
46	no	semi-furnished
47	no	furnished
48	yes	furnished
49	no	unfurnished
50	yes	semi-furnished
51	yes	unfurnished
52	no	unfurnished
53	no	furnished
54	no	semi-furnished
55	no	semi-furnished
56	no	unfurnished
57	yes	semi-furnished
58	yes	furnished
59	no	semi-furnished
60	no	furnished
61	no	semi-furnished
62	no	semi-furnished
63	no	furnished
64	yes	furnished
65	yes	furnished
66	no	furnished
67	no	furnished
68	no	unfurnished
69	no	furnished
70	yes	furnished
71	yes	semi-furnished
72	no	unfurnished
73	yes	unfurnished

```
74      yes       furnished
75       no       furnished
76       no   semi-furnished
 [ reached 'max' / getOption("max.print") -- omitted 469 rows ]
> dev.off()           #clear plot window
null device
          1
> par(mfrow=c(2,1))    #subplots/partions of 2 rows, 1 col
> #Univariate Analysis
> hist(housing_df$area, col = "orange")
> boxplot(housing_df$area, col = "light blue")
> dev.off()
null device
          1
> boxplot(housing_df$area, horizontal = T, col = "light blue")
> dev.off()
null device
          1
> summary(mtcars)
     mpg             cyl             disp             hp             drat             wt             qsec
 Min.   :10.40   Min.   :4.000   Min.   : 71.1   Min.   : 52.0   Min.   :2.760   Min.   :1.513   Min.   :14.50
 1st Qu.:15.43   1st Qu.:4.000   1st Qu.:120.8   1st Qu.: 96.5   1st Qu.:3.080   1st Qu.:2.581   1st Qu.:16.89
 Median :19.20   Median :6.000   Median :196.3   Median :123.0   Median :3.695   Median :3.325
Median :17.71
 Mean   :20.09   Mean   :6.188   Mean   :230.7   Mean   :146.7   Mean   :3.597   Mean   :3.217   Mean
:17.85
 3rd Qu.:22.80   3rd Qu.:8.000   3rd Qu.:326.0   3rd Qu.:180.0   3rd Qu.:3.920   3rd Qu.:3.610   3rd
Qu.:18.90
 Max.   :33.90   Max.   :8.000   Max.   :472.0   Max.   :335.0   Max.   :4.930   Max.   :5.424   Max.   :22.90
      vs               am              gear            carb
 Min.   :0.0000   Min.   :0.0000   Min.   :3.000   Min.   :1.000
 1st Qu.:0.0000   1st Qu.:0.0000   1st Qu.:3.000   1st Qu.:2.000
 Median :0.0000   Median :0.0000   Median :4.000   Median :2.000
 Mean   :0.4375   Mean   :0.4062   Mean   :3.688   Mean   :2.812
 3rd Qu.:1.0000   3rd Qu.:1.0000   3rd Qu.:4.000   3rd Qu.:4.000
 Max.   :1.0000   Max.   :1.0000   Max.   :5.000   Max.   :8.000
> #Bivariate Analysis
> table(mtcars$vs, mtcars$gear)   #Frequency table/Cross table

     3  4  5
  0 12  2  4
  1  3 10  1
> #row index - vs, col index - gear
> df_numeric_vars <- Filter(is.numeric, housing_df)   #Filter(condition, df)
```

```
> names(df_numeric_vars)
[1] "price"     "area"      "bedrooms"  "bathrooms" "stories"   "parking"
> df_categorical_vars <- Filter(is.factor, housing_df)
> names(df_categorical_vars)
character(0)
> rownames(mtcars)
 [1] "Mazda RX4"           "Mazda RX4 Wag"       "Datsun 710"          "Hornet 4 Drive"
 [5] "Hornet Sportabout"   "Valiant"             "Duster 360"          "Merc 240D"
 [9] "Merc 230"            "Merc 280"            "Merc 280C"           "Merc 450SE"
[13] "Merc 450SL"          "Merc 450SLC"         "Cadillac Fleetwood"  "Lincoln Continental"
[17] "Chrysler Imperial"   "Fiat 128"            "Honda Civic"         "Toyota Corolla"
[21] "Toyota Corona"       "Dodge Challenger"    "AMC Javelin"         "Camaro Z28"
[25] "Pontiac Firebird"    "Fiat X1-9"           "Porsche 914-2"       "Lotus Europa"
[29] "Ford Pantera L"      "Ferrari Dino"        "Maserati Bora"       "Volvo 142E"
> #?data/fn/keyword - get help documentation internally
> #??data/fn/keyword - get help documentation online
> ?mtcars
> ?iris
> counts <- table(mtcars$vs, mtcars$gear)
> #Side by Side barplot
> barplot(counts, main="Car Distribution by Gears and VS", xlab="Number of Gears", ylab="Frequency",
col=c("darkblue", "red"), legend=rownames(counts), beside=TRUE)
> dev.off()
null device
          1
> #Stacked barplot
> barplot(counts, main="Car Distribution by Gears and VS", xlab="Number of Gears", ylab="Frequency",
col=c("darkblue", "red"), legend=rownames(counts), names.arg=c("3", "4", "5"))
> #names.arg - label appear at the bottom of each bar
> nas <- sapply(housing_df, function(X) sum(is.na(x)))   #Missing value checking
> nas
           price             area         bedrooms        bathrooms          stories         mainroad
               0                0                0                0                0                0
       guestroom         basement  hotwaterheating  airconditioning          parking         prefarea
               0                0                0                0                0                0
furnishingstatus
               0
> missing_percent <- (nas * 100) / (nrow(housing_df))
> missing_percent
           price             area         bedrooms        bathrooms          stories         mainroad
               0                0                0                0                0                0
       guestroom         basement  hotwaterheating  airconditioning          parking         prefarea
               0                0                0                0                0                0
furnishingstatus
```

```
0
> colnames(mtcars)
 [1] "mpg"  "cyl"  "disp" "hp"   "drat" "wt"   "qsec" "vs"   "am"   "gear" "carb"
> names(mtcars)
 [1] "mpg"  "cyl"  "disp" "hp"   "drat" "wt"   "qsec" "vs"   "am"   "gear" "carb"
> dev.off()
null device
          1
> library(dplyr)
> library(ggplot2)
> data.frame(missing_percent, variable=colnames(housing_df))%>%  #redirection operator/pipe operator for chaining commands with dependency, passing output of one to another
+   ggplot(aes(variable, missing_percent)) +
+   geom_bar(stat="identity") +  #height of bars to represent values in the data
+   labs(x="Features", y="Percent of Missing values") +
+   theme(axis.text.x=element_text(angle=90, hjust=1))
> #aes(reorder(variable col, - or + the variable to be sorted)) sorts output in asc or desc order
> paste("Hello", "Everybody")   #Concats elements seperated by spaces
[1] "Hello Everybody"
> paste("A", "1", sep="")       #Concats elements with no spaces
[1] "A1"
> x <- c(32, 12, 30, 45)
> labels <- c("Mumbai", "Chennai", "Pune", "Banglore")
> pct <- round(x / sum(x) * 100)
> lbls <- paste(labels, pct)
> lbls <- paste(lbls, "%", sep="")
> pct
[1] 27 10 25 38
> lbls
[1] "Mumbai 27%"   "Chennai 10%"  "Pune 25%"     "Banglore 38%"
> pie(x, labels=lbls, col=rainbow(length(lbls)), main="City Pie Chart")   #rainbow(length) will generate 4 hexdecimal values
> legend("topright", c("Mumbai", "Chennai", "Pune", "Banglore"), cex=0.5, fill=rainbow(length(x)))
#cex=Controls zoom of the font
> legend("topright", c("Mumbai", "Chennai", "Pune", "Banglore"), cex=1, fill=rainbow(length(x)))
> #install.packages("Quandl")
> library("Quandl")
```

Day 4 – R Programming

```
> dev.off()
null device
          1
> setwd("C:/zubeda/PGA02_Zubu/R Programming")
> library("plyr")
```

```
> library("ggplot2")
> df_AP <- read.csv("ADANIPORTS.csv")
> edit(df_AP)
       Date     Symbol Series Prev.Close    Open    High     Low    Last   Close    VWAP   Volume        Turnover
1  2007-11-27 MUNDRAPORT     EQ     440.00  770.00 1050.00  770.00  959.00  962.90  984.72 27294366
2687719053785000
2  2007-11-28 MUNDRAPORT     EQ     962.90  984.00  990.00  874.00  885.00  893.90  941.38  4581338
431276530165000
3  2007-11-29 MUNDRAPORT     EQ     893.90  909.00  914.75  841.00  887.00  884.20  888.09  5124121
455065846265000
4  2007-11-30 MUNDRAPORT     EQ     884.20  890.00  958.00  890.00  929.00  921.55  929.17  4609762
428325662830000
5  2007-12-03 MUNDRAPORT     EQ     921.55  939.75  995.00  922.00  980.00  969.30  965.65  2977470
287519974300000
6  2007-12-04 MUNDRAPORT     EQ     969.30  985.00 1056.00  976.00 1049.00 1041.45 1015.39
4849250  492386736075000
7  2007-12-05 MUNDRAPORT     EQ    1041.45 1061.00 1099.50 1050.00 1084.00 1082.45 1082.79
2848209  308400973015000
8  2007-12-06 MUNDRAPORT     EQ    1082.45 1089.00 1109.70 1051.00 1090.10 1081.30 1087.03
1749516  190177114020000
9  2007-12-07 MUNDRAPORT     EQ    1081.30 1100.00 1134.00 1078.00 1100.00 1102.40 1106.57
2247904  248746530710000
10 2007-12-10 MUNDRAPORT     EQ    1102.40 1110.00 1110.00 1061.10 1073.55 1075.40 1080.38
1012350  109372679360000
11 2007-12-11 MUNDRAPORT     EQ    1075.40 1081.00 1089.00 1041.00 1046.00 1047.65 1067.80
810464   86541556460000
12 2007-12-12 MUNDRAPORT     EQ    1047.65 1032.00 1065.00 1016.00 1036.90 1036.80 1043.92
744799   77751369165000
13 2007-12-13 MUNDRAPORT     EQ    1036.80 1040.00 1150.00 1030.25 1131.15 1129.95 1109.09
3067687  340233907520000
14 2007-12-14 MUNDRAPORT     EQ    1129.95 1139.90 1140.00 1101.10 1107.00 1110.50 1119.55
1070737  119874627765000
15 2007-12-17 MUNDRAPORT     EQ    1110.50 1140.00 1168.00 1021.50 1052.00 1044.25 1102.42
1404955  154884767715000
16 2007-12-18 MUNDRAPORT     EQ    1044.25 1045.00 1109.90 1031.55 1085.00 1074.95 1077.84
1226984  132249513310000
17 2007-12-19 MUNDRAPORT     EQ    1074.95 1091.00 1116.00 1046.30 1078.00 1066.90 1082.93
845666   91579757645000
18 2007-12-20 MUNDRAPORT     EQ    1066.90 1083.50 1083.50 1051.00 1067.00 1060.20 1065.52
623288   66412706110000
19 2007-12-24 MUNDRAPORT     EQ    1060.20 1095.00 1192.00 1085.25 1160.00 1156.80 1160.77
2060892  239221361310000
20 2007-12-26 MUNDRAPORT     EQ    1156.80 1175.00 1214.00 1148.00 1212.00 1199.90 1183.30
1467031  173593856540000
```

21 2007-12-27 MUNDRAPORT EQ 1199.90 1215.00 1240.00 1204.00 1209.00 1211.65 1222.58
977495 119506465945000
22 2007-12-28 MUNDRAPORT EQ 1211.65 1189.40 1274.00 1175.00 1270.00 1249.10 1221.31
1164138 142177280540000
23 2007-12-31 MUNDRAPORT EQ 1249.10 1263.35 1295.00 1261.00 1268.00 1268.80 1277.64
737249 94194213815000
24 2008-01-01 MUNDRAPORT EQ 1268.80 1279.00 1319.00 1263.70 1308.00 1296.85 1285.72
491348 63173462100000
25 2008-01-02 MUNDRAPORT EQ 1296.85 1310.25 1324.00 1270.00 1300.15 1307.45 1302.15
703815 91647340425000
26 2008-01-03 MUNDRAPORT EQ 1307.45 1305.00 1314.70 1261.15 1267.15 1275.80 1289.24
505058 65114250075000
27 2008-01-04 MUNDRAPORT EQ 1275.80 1278.80 1294.80 1233.00 1239.90 1240.35 1256.03
550795 69181674340000
28 2008-01-07 MUNDRAPORT EQ 1240.35 1240.00 1278.90 1215.00 1233.00 1227.25 1244.76
630963 78539769975000
29 2008-01-08 MUNDRAPORT EQ 1227.25 1240.00 1255.00 1185.00 1202.00 1204.80 1217.08
530499 64565951270000
30 2008-01-09 MUNDRAPORT EQ 1204.80 1200.00 1210.00 1151.00 1181.00 1180.25 1176.37
627507 73818313330000
31 2008-01-10 MUNDRAPORT EQ 1180.25 1185.00 1199.80 1110.00 1118.00 1121.55 1156.44
438806 50745246590000
32 2008-01-11 MUNDRAPORT EQ 1121.55 1128.00 1130.00 1063.00 1096.00 1085.85 1087.78
616938 67109272025000
33 2008-01-14 MUNDRAPORT EQ 1085.85 1082.40 1082.40 1031.10 1035.00 1035.15 1042.40
835916 87135710755000
34 2008-01-15 MUNDRAPORT EQ 1035.15 1045.60 1078.70 1036.05 1057.00 1049.55 1050.69
830493 87259337110000
35 2008-01-16 MUNDRAPORT EQ 1049.55 1046.00 1064.00 1000.00 1038.30 1030.40 1032.86
816188 84300609685000
36 2008-01-17 MUNDRAPORT EQ 1030.40 1050.00 1053.50 1011.00 1014.95 1020.90 1033.73
336003 34733490900000
37 2008-01-18 MUNDRAPORT EQ 1020.90 1010.00 1072.00 974.90 995.00 994.60 1022.57
676854 69213280915000
38 2008-01-21 MUNDRAPORT EQ 994.60 995.00 1005.00 795.70 853.00 825.05 880.77 788623
69459899855000
39 2008-01-22 MUNDRAPORT EQ 825.05 700.00 810.00 660.05 739.00 735.55 703.20 546161
38406113705000
40 2008-01-23 MUNDRAPORT EQ 735.55 760.00 881.90 760.00 862.20 857.00 818.67 535462
43836526980000
41 2008-01-24 MUNDRAPORT EQ 857.00 875.00 935.00 812.00 814.70 814.15 854.83 511017
43683319425000
42 2008-01-25 MUNDRAPORT EQ 814.15 820.00 883.00 820.00 866.00 865.70 858.33 404045
34680333860000

43 2008-01-28 MUNDRAPORT EQ 865.70 835.00 835.00 783.20 822.00 820.80 804.38 467052
37568552380000
44 2008-01-29 MUNDRAPORT EQ 820.80 840.00 860.00 820.05 840.00 840.75 841.27 220070
18513823345000
45 2008-01-30 MUNDRAPORT EQ 840.75 849.80 864.00 822.25 834.00 830.45 833.82 286190
23863110660000
46 2008-01-31 MUNDRAPORT EQ 830.45 831.00 849.90 812.55 836.60 837.65 833.58 194300
16196555895000
47 2008-02-01 MUNDRAPORT EQ 837.65 831.65 852.30 820.00 826.00 825.35 828.09 204391
16925451805000
48 2008-02-04 MUNDRAPORT EQ 825.35 847.90 872.40 840.00 859.00 856.10 858.77 280230
24065208695000
49 2008-02-05 MUNDRAPORT EQ 856.10 856.00 857.00 830.00 834.65 834.30 842.06 162093
13649192020000
50 2008-02-06 MUNDRAPORT EQ 834.30 803.00 824.90 780.00 809.00 807.50 810.50 193260
15663794125000
51 2008-02-07 MUNDRAPORT EQ 807.50 825.00 830.00 792.00 795.90 796.25 809.53 212932
17237575975000
52 2008-02-08 MUNDRAPORT EQ 796.25 810.00 830.00 765.15 786.00 784.05 781.48 285025
22274252000000
53 2008-02-11 MUNDRAPORT EQ 784.05 785.00 785.00 695.00 699.00 711.20 736.23 223955
16488264325000
54 2008-02-12 MUNDRAPORT EQ 711.20 725.00 734.95 655.60 689.00 681.30 681.38 303409
20673577510000
55 2008-02-13 MUNDRAPORT EQ 681.30 815.90 815.90 664.00 678.00 670.95 681.68 214900
14649214640000
56 2008-02-14 MUNDRAPORT EQ 670.95 680.00 714.00 680.00 710.00 709.80 704.71 269032
18959036175000
57 2008-02-15 MUNDRAPORT EQ 709.80 700.00 763.70 681.25 729.00 728.75 734.23 353049
25921872820000
58 2008-02-18 MUNDRAPORT EQ 728.75 735.00 775.00 735.00 772.00 771.60 762.33 342580
26115882900000
59 2008-02-19 MUNDRAPORT EQ 771.60 779.00 786.90 760.20 767.00 763.90 772.24 137412
10611555840000
60 2008-02-20 MUNDRAPORT EQ 763.90 750.00 760.00 720.00 740.00 732.10 730.61 197489
14428706935000
61 2008-02-21 MUNDRAPORT EQ 732.10 762.00 762.00 730.10 738.90 737.60 741.53 125558
9310465240000
62 2008-02-22 MUNDRAPORT EQ 737.60 723.00 737.00 715.00 724.50 724.00 726.52 81070
5889922195000
63 2008-02-25 MUNDRAPORT EQ 724.00 725.05 758.90 702.30 707.00 707.65 711.70 152803
10875065635000
64 2008-02-26 MUNDRAPORT EQ 707.65 725.00 744.00 713.00 735.00 735.80 733.73 251269
18436350425000

65 2008-02-27 MUNDRAPORT EQ 735.80 749.70 783.40 741.00 744.00 746.40 762.47 305320
23279802440000
66 2008-02-28 MUNDRAPORT EQ 746.40 740.00 754.90 725.05 740.00 737.75 738.91 112491
8312092510000

	Trades	Deliverable.Volume	X.Deliverble
1	NA	9859619	0.3612
2	NA	1453278	0.3172
3	NA	1069678	0.2088
4	NA	1260913	0.2735
5	NA	816123	0.2741
6	NA	1537667	0.3171
7	NA	904260	0.3175
8	NA	825691	0.4720
9	NA	697763	0.3104
10	NA	417514	0.4124
11	NA	415191	0.5123
12	NA	363848	0.4885
13	NA	1040076	0.3390
14	NA	525239	0.4905
15	NA	670298	0.4771
16	NA	449420	0.3663
17	NA	344171	0.4070
18	NA	276356	0.4434
19	NA	807879	0.3920
20	NA	469389	0.3200
21	NA	355431	0.3636
22	NA	503564	0.4326
23	NA	316377	0.4291
24	NA	172911	0.3519
25	NA	221397	0.3146
26	NA	217437	0.4305
27	NA	230237	0.4180
28	NA	239404	0.3794
29	NA	228866	0.4314
30	NA	259280	0.4132
31	NA	200150	0.4561
32	NA	312121	0.5059
33	NA	570824	0.6829
34	NA	504259	0.6072
35	NA	478517	0.5863
36	NA	145194	0.4321
37	NA	278615	0.4116
38	NA	474223	0.6013
39	NA	376194	0.6888

```
40    NA          283881      0.5302
41    NA          258346      0.5056
42    NA          178177      0.4410
43    NA          241365      0.5168
44    NA           74141     0.3369
45    NA          165926      0.5798
46    NA          103890      0.5347
47    NA          115715      0.5661
48    NA          128195      0.4575
49    NA           96153     0.5932
50    NA          110565      0.5721
51    NA          106275      0.4991
52    NA          154857      0.5433
53    NA          118002      0.5269
54    NA          187180      0.6169
55    NA          108761      0.5061
56    NA          148611      0.5524
57    NA          110621      0.3133
58    NA          154099      0.4498
59    NA           47543     0.3460
60    NA           89397     0.4527
61    NA           37956     0.3023
62    NA           31808     0.3924
63    NA           71403     0.4673
64    NA           53136     0.2115
65    NA           84490     0.2767
66    NA           36730     0.3265
 [ reached 'max' / getOption("max.print") -- omitted 3256 rows ]
> names(df_AP)
 [1] "Date"          "Symbol"          "Series"          "Prev.Close"       "Open"
 [6] "High"          "Low"             "Last"            "Close"            "VWAP"
[11] "Volume"         "Turnover"         "Trades"           "Deliverable.Volume" "X.Deliverble"
> head(df_AP)    #get first 6 rows
      Date     Symbol Series Prev.Close  Open   High Low Last  Close   VWAP  Volume       Turnover
Trades
1 2007-11-27 MUNDRAPORT    EQ    440.00 770.00 1050.00 770  959  962.90  984.72 27294366
2687719053785000     NA
2 2007-11-28 MUNDRAPORT    EQ    962.90 984.00  990.00 874  885  893.90  941.38  4581338
431276530165000     NA
3 2007-11-29 MUNDRAPORT    EQ    893.90 909.00  914.75 841  887  884.20  888.09  5124121
455065846265000     NA
4 2007-11-30 MUNDRAPORT    EQ    884.20 890.00  958.00 890  929  921.55  929.17  4609762
428325662830000     NA
```

```
5 2007-12-03 MUNDRAPORT     EQ     921.55 939.75  995.00 922  980  969.30  965.65  2977470
287519974300000     NA
6 2007-12-04 MUNDRAPORT     EQ     969.30 985.00 1056.00 976 1049 1041.45 1015.39  4849250
492386736075000     NA
  Deliverable.Volume X.Deliverble
1          9859619      0.3612
2          1453278      0.3172
3          1069678      0.2088
4          1260913      0.2735
5           816123      0.2741
6          1537667      0.3171
> v <- c(8, 14, 26, 5, 43)
> plot(v, type="o")  #Line plot with points
> plot(v, type="p")  #Points plot
> plot(v, type="l")  #Line plot without points
> plot(v, type="o", col="red", xlab="Month", ylab="Rainfall", main="Rainfall Chart")
> v <- c(12, 14, 28, 5, 44)
> t <- c(15, 8, 8, 10, 13)
> plot(v, type="o", col="blue", xlab="Month", ylab="Rainfall", main="Rainfall Chart")
> lines(t, type="o", col="red")
> df_aapl <- read.csv("AAPL.csv")
> head(df_aapl)
        Date   Open   High    Low  Close Adj.Close    Volume
1 2021-02-17 131.25 132.22 129.47 130.84  130.0669  97918500
2 2021-02-18 129.20 130.00 127.41 129.71  128.9436  96856700
3 2021-02-19 130.24 130.71 128.80 129.87  129.1027  87668800
4 2021-02-22 128.01 129.72 125.60 126.00  125.2555 103916400
5 2021-02-23 123.76 126.71 118.39 125.86  125.1164 158273000
6 2021-02-24 124.94 125.56 122.23 125.35  124.6094 111039900
> df_waltdisney <- read.csv("DIS.csv")
> head(df_waltdisney)
        Date   Open   High    Low  Close Adj.Close   Volume
1 2021-02-17 185.36 187.63 182.16 186.44    186.44 11391800
2 2021-02-18 184.79 186.40 182.84 183.00    183.00 12380900
3 2021-02-19 184.27 184.78 182.79 183.65    183.65  8834500
4 2021-02-22 181.74 194.02 181.53 191.76    191.76 18799600
5 2021-02-23 193.59 198.94 188.66 197.09    197.09 23191400
6 2021-02-24 197.58 200.60 195.33 197.51    197.51 16205900
> df_nike <- read.csv("NKE.csv")
> head(df_nike)
        Date   Open   High    Low  Close Adj.Close   Volume
1 2021-02-17 141.30 144.56 140.21 143.99  142.9153  6437100
2 2021-02-18 142.98 145.39 141.21 145.09  144.0071  4486800
3 2021-02-19 145.43 145.50 141.50 142.02  140.9601  7486000
```

```
4 2021-02-22 141.54 142.46 136.26 136.67  135.6500  8985900
5 2021-02-23 136.03 136.83 131.58 136.13  135.1140 10364100
6 2021-02-24 135.06 135.96 133.95 135.65  134.6376  6360900
> df_aapl <- cbind(df_aapl, Stock="")
> df_waltdisney <- cbind(df_waltdisney, Stock="")
> df_nike <- cbind(df_nike, Stock="")
> head(df_aapl)
        Date   Open   High    Low  Close Adj.Close    Volume Stock
1 2021-02-17 131.25 132.22 129.47 130.84  130.0669  97918500
2 2021-02-18 129.20 130.00 127.41 129.71  128.9436  96856700
3 2021-02-19 130.24 130.71 128.80 129.87  129.1027  87668800
4 2021-02-22 128.01 129.72 125.60 126.00  125.2555 103916400
5 2021-02-23 123.76 126.71 118.39 125.86  125.1164 158273000
6 2021-02-24 124.94 125.56 122.23 125.35  124.6094 111039900
> head(df_waltdisney)
        Date   Open   High    Low  Close Adj.Close   Volume Stock
1 2021-02-17 185.36 187.63 182.16 186.44    186.44 11391800
2 2021-02-18 184.79 186.40 182.84 183.00    183.00 12380900
3 2021-02-19 184.27 184.78 182.79 183.65    183.65  8834500
4 2021-02-22 181.74 194.02 181.53 191.76    191.76 18799600
5 2021-02-23 193.59 198.94 188.66 197.09    197.09 23191400
6 2021-02-24 197.58 200.60 195.33 197.51    197.51 16205900
> head(df_nike)
        Date   Open   High    Low  Close Adj.Close   Volume Stock
1 2021-02-17 141.30 144.56 140.21 143.99  142.9153  6437100
2 2021-02-18 142.98 145.39 141.21 145.09  144.0071  4486800
3 2021-02-19 145.43 145.50 141.50 142.02  140.9601  7486000
4 2021-02-22 141.54 142.46 136.26 136.67  135.6500  8985900
5 2021-02-23 136.03 136.83 131.58 136.13  135.1140 10364100
6 2021-02-24 135.06 135.96 133.95 135.65  134.6376  6360900
> df_aapl$Stock <- paste(df_aapl$Stock, "Bertrandt", sep="")
> df_waltdisney$Stock <- paste(df_waltdisney$Stock, "Deutsche Bank", sep="")
> df_nike$Stock <- paste(df_nike$Stock, "Siemens", sep="")
> head(df_aapl)
        Date   Open   High    Low  Close Adj.Close    Volume     Stock
1 2021-02-17 131.25 132.22 129.47 130.84  130.0669  97918500 Bertrandt
2 2021-02-18 129.20 130.00 127.41 129.71  128.9436  96856700 Bertrandt
3 2021-02-19 130.24 130.71 128.80 129.87  129.1027  87668800 Bertrandt
4 2021-02-22 128.01 129.72 125.60 126.00  125.2555 103916400 Bertrandt
5 2021-02-23 123.76 126.71 118.39 125.86  125.1164 158273000 Bertrandt
6 2021-02-24 124.94 125.56 122.23 125.35  124.6094 111039900 Bertrandt
> head(df_waltdisney)
        Date   Open   High    Low  Close Adj.Close   Volume         Stock
1 2021-02-17 185.36 187.63 182.16 186.44    186.44 11391800 Deutsche Bank
```

```
2 2021-02-18 184.79 186.40 182.84 183.00    183.00 12380900 Deutsche Bank
3 2021-02-19 184.27 184.78 182.79 183.65    183.65  8834500 Deutsche Bank
4 2021-02-22 181.74 194.02 181.53 191.76    191.76 18799600 Deutsche Bank
5 2021-02-23 193.59 198.94 188.66 197.09    197.09 23191400 Deutsche Bank
6 2021-02-24 197.58 200.60 195.33 197.51    197.51 16205900 Deutsche Bank
> head(df_nike)
        Date   Open   High    Low  Close Adj.Close   Volume   Stock
1 2021-02-17 141.30 144.56 140.21 143.99  142.9153  6437100 Siemens
2 2021-02-18 142.98 145.39 141.21 145.09  144.0071  4486800 Siemens
3 2021-02-19 145.43 145.50 141.50 142.02  140.9601  7486000 Siemens
4 2021-02-22 141.54 142.46 136.26 136.67  135.6500  8985900 Siemens
5 2021-02-23 136.03 136.83 131.58 136.13  135.1140 10364100 Siemens
6 2021-02-24 135.06 135.96 133.95 135.65  134.6376  6360900 Siemens
> df_allStocks <- rbind(df_aapl, df_waltdisney, df_nike)
> df_allStocks
          Date   Open   High    Low  Close Adj.Close    Volume     Stock
1   2021-02-17 131.25 132.22 129.47 130.84  130.0669  97918500 Bertrandt
2   2021-02-18 129.20 130.00 127.41 129.71  128.9436  96856700 Bertrandt
3   2021-02-19 130.24 130.71 128.80 129.87  129.1027  87668800 Bertrandt
4   2021-02-22 128.01 129.72 125.60 126.00  125.2555 103916400 Bertrandt
5   2021-02-23 123.76 126.71 118.39 125.86  125.1164 158273000 Bertrandt
6   2021-02-24 124.94 125.56 122.23 125.35  124.6094 111039900 Bertrandt
7   2021-02-25 124.68 126.46 120.54 120.99  120.2751 148199500 Bertrandt
8   2021-02-26 122.59 124.85 121.20 121.26  120.5436 164560400 Bertrandt
9   2021-03-01 123.75 127.93 122.79 127.79  127.0350 116307900 Bertrandt
10  2021-03-02 128.41 128.72 125.01 125.12  124.3807 102260900 Bertrandt
11  2021-03-03 124.81 125.71 121.84 122.06  121.3388 112966300 Bertrandt
12  2021-03-04 121.75 123.60 118.62 120.13  119.4202 178155000 Bertrandt
13  2021-03-05 120.98 121.94 117.57 121.42  120.7026 153766600 Bertrandt
14  2021-03-08 120.93 121.00 116.21 116.36  115.6725 154376600 Bertrandt
15  2021-03-09 119.03 122.06 118.79 121.09  120.3745 129525800 Bertrandt
16  2021-03-10 121.69 122.17 119.45 119.98  119.2711 111943300 Bertrandt
17  2021-03-11 122.54 123.21 121.26 121.96  121.2394 103026500 Bertrandt
18  2021-03-12 120.40 121.17 119.16 121.03  120.3149  88105100 Bertrandt
19  2021-03-15 121.41 124.00 120.42 123.99  123.2574  92403800 Bertrandt
20  2021-03-16 125.70 127.22 124.72 125.57  124.8281 115227900 Bertrandt
21  2021-03-17 124.05 125.86 122.34 124.76  124.0229 111932600 Bertrandt
22  2021-03-18 122.88 123.18 120.32 120.53  119.8179 121229700 Bertrandt
23  2021-03-19 119.90 121.43 119.68 119.99  119.2811 185549500 Bertrandt
24  2021-03-22 120.33 123.87 120.26 123.39  122.6610 111912300 Bertrandt
25  2021-03-23 123.33 124.24 122.14 122.54  121.8160  95467100 Bertrandt
26  2021-03-24 122.82 122.90 120.07 120.09  119.3805  88530500 Bertrandt
27  2021-03-25 119.54 121.66 119.00 120.59  119.8775  98844700 Bertrandt
28  2021-03-26 120.35 121.48 118.92 121.21  120.4938  94071200 Bertrandt
```

29	2021-03-29	121.65	122.58	120.73	121.39	120.6728	80819200	Bertrandt
30	2021-03-30	120.11	120.40	118.86	119.90	119.1916	85671900	Bertrandt
31	2021-03-31	121.65	123.52	121.15	122.15	121.4283	118323800	Bertrandt
32	2021-04-01	123.66	124.18	122.49	123.00	122.2733	75089100	Bertrandt
33	2021-04-05	123.87	126.16	123.07	125.90	125.1561	88651200	Bertrandt
34	2021-04-06	126.50	127.13	125.65	126.21	125.4643	80171300	Bertrandt
35	2021-04-07	125.83	127.92	125.14	127.90	127.1443	83466700	Bertrandt
36	2021-04-08	128.95	130.39	128.52	130.36	129.5898	88844600	Bertrandt
37	2021-04-09	129.80	133.04	129.47	133.00	132.2142	106686700	Bertrandt
38	2021-04-12	132.52	132.85	130.63	131.24	130.4646	91420000	Bertrandt
39	2021-04-13	132.44	134.66	131.93	134.43	133.6357	91266500	Bertrandt
40	2021-04-14	134.94	135.00	131.66	132.03	131.2499	87222800	Bertrandt
41	2021-04-15	133.82	135.00	133.64	134.50	133.7053	89347100	Bertrandt
42	2021-04-16	134.30	134.67	133.28	134.16	133.3673	84922400	Bertrandt
43	2021-04-19	133.51	135.47	133.34	134.84	134.0433	94264200	Bertrandt
44	2021-04-20	135.02	135.53	131.81	133.11	132.3235	94812300	Bertrandt
45	2021-04-21	132.36	133.75	131.30	133.50	132.7112	68847100	Bertrandt
46	2021-04-22	133.04	134.15	131.41	131.94	131.1605	84566500	Bertrandt
47	2021-04-23	132.16	135.12	132.16	134.32	133.5264	78657500	Bertrandt
48	2021-04-26	134.83	135.06	133.56	134.72	133.9240	66905100	Bertrandt
49	2021-04-27	135.01	135.41	134.11	134.39	133.5960	66015800	Bertrandt
50	2021-04-28	134.31	135.02	133.08	133.58	132.7907	107760100	Bertrandt
51	2021-04-29	136.47	137.07	132.45	133.48	132.6913	151101000	Bertrandt
52	2021-04-30	131.78	133.56	131.07	131.46	130.6833	109839500	Bertrandt
53	2021-05-03	132.04	134.07	131.83	132.54	131.7569	75135100	Bertrandt
54	2021-05-04	131.19	131.49	126.70	127.85	127.0946	137564700	Bertrandt
55	2021-05-05	129.20	130.45	127.97	128.10	127.3431	84000900	Bertrandt
56	2021-05-06	127.89	129.75	127.13	129.74	128.9735	78128300	Bertrandt
57	2021-05-07	130.85	131.26	129.48	130.21	129.6606	78973300	Bertrandt
58	2021-05-10	129.41	129.54	126.81	126.85	126.3147	88071200	Bertrandt
59	2021-05-11	123.50	126.27	122.77	125.91	125.3787	126142800	Bertrandt
60	2021-05-12	123.40	124.64	122.25	122.77	122.2519	112172300	Bertrandt
61	2021-05-13	124.58	126.15	124.26	124.97	124.4426	105861300	Bertrandt
62	2021-05-14	126.25	127.89	125.85	127.45	126.9122	81918000	Bertrandt
63	2021-05-17	126.82	126.93	125.17	126.27	125.7372	74244600	Bertrandt
64	2021-05-18	126.56	126.99	124.78	124.85	124.3232	63342900	Bertrandt
65	2021-05-19	123.16	124.92	122.86	124.69	124.1638	92612000	Bertrandt
66	2021-05-20	125.23	127.72	125.10	127.31	126.7728	76857100	Bertrandt
67	2021-05-21	127.82	128.00	125.21	125.43	124.9007	79295400	Bertrandt
68	2021-05-24	126.01	127.94	125.94	127.10	126.5637	63092900	Bertrandt
69	2021-05-25	127.82	128.32	126.32	126.90	126.3645	72009500	Bertrandt
70	2021-05-26	126.96	127.39	126.42	126.85	126.3147	56575900	Bertrandt
71	2021-05-27	126.44	127.64	125.08	125.28	124.7513	94625600	Bertrandt
72	2021-05-28	125.57	125.80	124.55	124.61	124.0842	71311100	Bertrandt

73	2021-06-01	125.08	125.35	123.94	124.28	123.7556	67637100	Bertrandt
74	2021-06-02	124.28	125.24	124.05	125.06	124.5323	59278900	Bertrandt
75	2021-06-03	124.68	124.85	123.13	123.54	123.0187	76229200	Bertrandt
76	2021-06-04	124.07	126.16	123.85	125.89	125.3588	75169300	Bertrandt
77	2021-06-07	126.17	126.32	124.83	125.90	125.3687	71057600	Bertrandt
78	2021-06-08	126.60	128.46	126.21	126.74	126.2052	74403800	Bertrandt
79	2021-06-09	127.21	127.75	126.52	127.13	126.5935	56877900	Bertrandt
80	2021-06-10	127.02	128.19	125.94	126.11	125.5778	71186400	Bertrandt
81	2021-06-11	126.53	127.44	126.10	127.35	126.8126	53522400	Bertrandt
82	2021-06-14	127.82	130.54	127.07	130.48	129.9294	96906500	Bertrandt
83	2021-06-15	129.94	130.60	129.39	129.64	129.0929	62746300	Bertrandt
84	2021-06-16	130.37	130.89	128.46	130.15	129.6008	91815000	Bertrandt
85	2021-06-17	129.80	132.55	129.65	131.79	131.2339	96721700	Bertrandt
86	2021-06-18	130.71	131.51	130.24	130.46	129.9095	108953300	Bertrandt
87	2021-06-21	130.30	132.41	129.21	132.30	131.7417	79663300	Bertrandt
88	2021-06-22	132.13	134.08	131.62	133.98	133.4146	74783600	Bertrandt
89	2021-06-23	133.77	134.32	133.23	133.70	133.1358	60214200	Bertrandt
90	2021-06-24	134.45	134.64	132.93	133.41	132.8470	68711000	Bertrandt
91	2021-06-25	133.46	133.89	132.81	133.11	132.5483	70783700	Bertrandt
92	2021-06-28	133.41	135.25	133.35	134.78	134.2113	62111300	Bertrandt
93	2021-06-29	134.80	136.49	134.35	136.33	135.7547	64556100	Bertrandt
94	2021-06-30	136.17	137.41	135.87	136.96	136.3821	63261400	Bertrandt
95	2021-07-01	136.60	137.33	135.76	137.27	136.6908	52485800	Bertrandt
96	2021-07-02	137.90	140.00	137.75	139.96	139.3694	78852600	Bertrandt
97	2021-07-06	140.07	143.15	140.07	142.02	141.4207	108181800	Bertrandt
98	2021-07-07	143.54	144.89	142.66	144.57	143.9599	104911600	Bertrandt
99	2021-07-08	141.58	144.06	140.67	143.24	142.6355	105575500	Bertrandt
100	2021-07-09	142.75	145.65	142.65	145.11	144.4977	99890800	Bertrandt
101	2021-07-12	146.21	146.32	144.00	144.50	143.8902	76299700	Bertrandt
102	2021-07-13	144.03	147.46	143.63	145.64	145.0254	100827100	Bertrandt
103	2021-07-14	148.10	149.57	147.68	149.15	148.5206	127050800	Bertrandt
104	2021-07-15	149.24	150.00	147.09	148.48	147.8534	106820300	Bertrandt
105	2021-07-16	148.46	149.76	145.88	146.39	145.7722	93251400	Bertrandt
106	2021-07-19	143.75	144.07	141.67	142.45	141.8489	121434600	Bertrandt
107	2021-07-20	143.46	147.10	142.96	146.15	145.5332	96350000	Bertrandt
108	2021-07-21	145.53	146.13	144.63	145.40	144.7864	74993500	Bertrandt
109	2021-07-22	145.94	148.20	145.81	146.80	146.1805	77338200	Bertrandt
110	2021-07-23	147.55	148.72	146.92	148.56	147.9331	71447400	Bertrandt
111	2021-07-26	148.27	149.83	147.70	148.99	148.3613	72434100	Bertrandt
112	2021-07-27	149.12	149.21	145.55	146.77	146.1507	104818600	Bertrandt
113	2021-07-28	144.81	146.97	142.54	144.98	144.3682	118931200	Bertrandt
114	2021-07-29	144.69	146.55	144.58	145.64	145.0254	56699500	Bertrandt
115	2021-07-30	144.38	146.33	144.11	145.86	145.2445	70382000	Bertrandt
116	2021-08-02	146.36	146.95	145.25	145.52	144.9059	62880000	Bertrandt

```
117 2021-08-03 145.81 148.04 145.18 147.36  146.7382  64786600 Bertrandt
118 2021-08-04 147.27 147.79 146.28 146.95  146.3299  56368300 Bertrandt
119 2021-08-05 146.98 147.84 146.17 147.06  146.4394  46397700 Bertrandt
120 2021-08-06 146.35 147.11 145.63 146.14  145.7413  54067400 Bertrandt
121 2021-08-09 146.20 146.70 145.52 146.09  145.6915  48908700 Bertrandt
122 2021-08-10 146.44 147.71 145.30 145.60  145.2028  69023100 Bertrandt
123 2021-08-11 146.05 146.72 145.53 145.86  145.4621  48493500 Bertrandt
124 2021-08-12 146.19 149.05 145.84 148.89  148.4838  72282600 Bertrandt
125 2021-08-13 148.97 149.44 148.27 149.10  148.6933  59318800 Bertrandt
 [ reached 'max' / getOption("max.print") -- omitted 637 rows ]
> df_allStocks$Date <- as.character(df_allStocks$Date)
> datesplit_list <- strsplit(df_allStocks$Date, "-")
> df_dates <- ldply(datesplit_list)
> colnames(df_dates) <- c("Year", "Month", "Day")
> df_allStocks <- cbind(df_allStocks, df_dates)
> names(df_allStocks)
 [1] "Date"      "Open"      "High"      "Low"       "Close"     "Adj.Close" "Volume"    "Stock"     "Year"
[10] "Month"     "Day"
> head(df_allStocks)
        Date   Open   High    Low  Close Adj.Close    Volume     Stock Year Month Day
1 2021-02-17 131.25 132.22 129.47 130.84  130.0669  97918500 Bertrandt 2021    02  17
2 2021-02-18 129.20 130.00 127.41 129.71  128.9436  96856700 Bertrandt 2021    02  18
3 2021-02-19 130.24 130.71 128.80 129.87  129.1027  87668800 Bertrandt 2021    02  19
4 2021-02-22 128.01 129.72 125.60 126.00  125.2555 103916400 Bertrandt 2021    02  22
5 2021-02-23 123.76 126.71 118.39 125.86  125.1164 158273000 Bertrandt 2021    02  23
6 2021-02-24 124.94 125.56 122.23 125.35  124.6094 111039900 Bertrandt 2021    02  24
> g <- ggplot(data=df_aapl, aes(x=Date, y=Open, group=1))  # group 1st param
> g <- g + geom_line(linetype="dashed")
> g
> g <- ggplot(data=df_aapl, aes(x=Date, y=Open, group=1))  # group 1st param
> g <- g + geom_line(linetype="dashed", col="red")
> g
> g <- ggplot(data=df_aapl, aes(x=Date, y=Open, group=1))  # group 1st param
> g <- g + geom_line(linetype="solid", col="red", size=1.5)
> g <- g + labs(title="Apple Inc", subtitle="Open Prices", y="Open", x="Year", caption="Yearwise Apple
Stock")
> g
> options(scipen = 999)
> ggplot(data=df_allStocks, aes(x=Stock, y=Volume)) +
+   geom_bar(stat="identity")   #if we want heights of the bars to represent values in the data, map a
value to y aes
> #scipen - avoid scientific notations by giving largest limit eg. 999
> ggplot(data=df_allStocks, aes(x=Stock, y=Volume)) +
+   geom_bar(stat="identity") + coord_flip()   #coord_flip to create horizontal plot
```

```
> ggplot(data=df_allStocks, aes(x=Stock, y=Volume)) +
+   geom_bar(stat="identity", width=0.5)   #change width of bars
> ggplot(data=df_allStocks, aes(x=Stock, y=Volume)) +
+   geom_bar(stat="identity", width=0.5, col="blue")
> ggplot(data=df_allStocks, aes(x=Stock, y=Volume, fill=Stock)) +
+   geom_bar(stat="identity", width=0.5)
> #fill=Stock - fill colors automatically as per the levels of the bar
> ggplot(df_nike, aes(x=Open)) + geom_histogram()
`stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
> ggplot(df_waltdisney, aes(x=Open)) + geom_histogram()
`stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
> ggplot(df_nike, aes(x=Volume)) + geom_histogram(fill="lightblue", color="darkblue")
`stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
> ggplot(df_nike, aes(x=Close)) + geom_histogram(fill="lightblue", color="darkblue")
`stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
> ggplot(df_nike, aes(x=Close)) + geom_histogram(fill="lightblue", color="darkblue", binwidth=3)
> ggplot(df_nike, aes(x=Open)) +
+   geom_histogram(aes(y=..density..),fill="white", colour="black") +
+   geom_density(alpha=.2, fill="Turquoise")   #alpha controls the transparency
`stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
> ggplot(df_nike, aes(x=Open, col=Stock)) + geom_histogram(fill="light blue", binwidth=3)
> ggplot(df_allStocks, aes(x=Open, col=Stock)) + geom_histogram(fill="light blue", binwidth=3)
#Different outline color for different stock category
> ggplot(df_waltdisney, aes(x=Open, y=Close)) + geom_point()
> ggplot(df_nike, aes(x=Open, y=Close)) + geom_point(size=2, shape=23) + geom_smooth(method="lm")
`geom_smooth()` using formula 'y ~ x'
> #size - size of point, shape - shape of point (0-25), method="lm" - draw linear model (linear regression)
line
> ggplot(df_nike, aes(x=Open, y=Close)) +
+   geom_point(shape=18, color="dark grey") +
+   geom_smooth(method="lm", linetype="dashed", color="red")
> df_midwest = read.csv("http://goo.gl/G1K41K")
> dim(df_midwest)
[1] 437  28
> summary(df_midwest)
      PID         county            state               area            poptotal        popdensity
 Min.   : 561   Length:437        Length:437        Min.   :0.00500   Min.   :   1701   Min.   :   85.05
 1st Qu.: 670   Class :character   Class :character   1st Qu.:0.02400   1st Qu.:  18840   1st Qu.:  622.41
 Median :1221   Mode  :character   Mode  :character   Median :0.03000   Median :  35324   Median :
1156.21
 Mean   :1437                                      Mean   :0.03317   Mean   :  96130   Mean   : 3097.74
 3rd Qu.:2059                                      3rd Qu.:0.03800   3rd Qu.:  75651   3rd Qu.: 2330.00
 Max.   :3052                                     Max.   :0.11000   Max.   :5105067   Max.   :88018.40
    popwhite         popblack       popamerindian        popasian         popother         percwhite
```

```
 Min.   :    416   Min.   :      0   Min.   :    4.0   Min.   :      0   Min.   :      0   Min.   :10.69
 1st Qu.:  18630   1st Qu.:     29   1st Qu.:   44.0   1st Qu.:     35   1st Qu.:     20   1st Qu.:94.89
 Median :  34471   Median :    201   Median :   94.0   Median :    102   Median :     66   Median :98.03
 Mean   :  81840   Mean   :  11024   Mean   :  343.1   Mean   :   1310   Mean   :   1613   Mean   :95.56
 3rd Qu.:  72968   3rd Qu.:   1291   3rd Qu.:  288.0   3rd Qu.:    401   3rd Qu.:    345   3rd Qu.:99.07
 Max.   :3204947   Max.   :1317147   Max.   :10289.0   Max.   :188565   Max.   :384119   Max.   :99.82
   percblack       percamerindan       percasian        percother          popadults          perchsd
 Min.   : 0.0000   Min.   : 0.05623   Min.   :0.0000   Min.   :0.00000   Min.   :   1287   Min.   :46.91
 1st Qu.: 0.1157   1st Qu.: 0.15793   1st Qu.:0.1737   1st Qu.:0.09102   1st Qu.:  12271   1st Qu.:71.33
 Median : 0.5390   Median : 0.21502   Median :0.2972   Median :0.17844   Median :  22188   Median
:74.25
 Mean   : 2.6763   Mean   : 0.79894   Mean   :0.4872   Mean   :0.47906   Mean   :  60973   Mean   :73.97
 3rd Qu.: 2.6014   3rd Qu.: 0.38362   3rd Qu.:0.5212   3rd Qu.:0.48050   3rd Qu.:  47541   3rd Qu.:77.20
 Max.   :40.2100   Max.   :89.17738   Max.   :5.0705   Max.   :7.52427   Max.   :3291995   Max.   :88.90
   percollege        percprof       poppovertyknown   percpovertyknown percbelowpoverty
percchildbelowpovert
 Min.   : 7.336   Min.   : 0.5203   Min.   :   1696   Min.   :80.90    Min.   : 2.180   Min.   : 1.919
 1st Qu.:14.114   1st Qu.: 2.9980   1st Qu.:  18364   1st Qu.:96.89    1st Qu.: 9.199   1st Qu.:11.624
 Median :16.798   Median : 3.8142   Median :  33788   Median :98.17    Median :11.822   Median :15.270
 Mean   :18.273   Mean   : 4.4473   Mean   :  93642   Mean   :97.11    Mean   :12.511   Mean   :16.447
 3rd Qu.:20.550   3rd Qu.: 4.9493   3rd Qu.:  72840   3rd Qu.:98.60    3rd Qu.:15.133   3rd Qu.:20.352
 Max.   :48.079   Max.   :20.7913   Max.   :5023523   Max.   :99.86    Max.   :48.691   Max.   :64.308
 percadultpoverty percelderlypoverty    inmetro         category
 Min.   : 1.938   Min.   : 3.547     Min.   :0.0000   Length:437
 1st Qu.: 7.668   1st Qu.: 8.912     1st Qu.:0.0000   Class :character
 Median :10.008   Median :10.869     Median :0.0000   Mode  :character
 Mean   :10.919   Mean   :11.389     Mean   :0.3432
 3rd Qu.:13.182   3rd Qu.:13.412     3rd Qu.:1.0000
 Max.   :43.312   Max.   :31.162     Max.   :1.0000
> ggplot(df_midwest, aes(x=area, y=poptotal)) +
+   geom_point(shape=18, color="dark grey") +
+   geom_smooth(method="lm", linetype="dashed", color="red")
`geom_smooth()` using formula 'y ~ x'
> ggplot(df_midwest, aes(x=area, y=poptotal)) + geom_point(shape=18, color="dark
grey")+geom_smooth(method="lm", linetype="dashed", color="red") + coord_cartesian(xlim=c(0,0.1),
ylim=c(0,600000))
`geom_smooth()` using formula 'y ~ x'
> seq(1, 20, 3)
[1]  1  4  7 10 13 16 19
> g <- ggplot(df_midwest, aes(x=area, y=poptotal)) +
+   geom_point(size=2) +
+   geom_smooth(method="lm",col="black") +
+   coord_cartesian(xlim=c(0,0.1), ylim=c(0,1000000)) +
```

```
+   labs(title="Area Vs Population", subtitle = "Using midwest dataset", y="Population", x="area", caption
= "Midwest Demographics")
> g + scale_x_continuous(breaks=seq(0, 0.10, 0.01))
`geom_smooth()` using formula 'y ~ x'
> g + scale_y_continuous(breaks=seq(0, 1000000, 50000))
`geom_smooth()` using formula 'y ~ x'
> g <- ggplot(df_midwest, aes(x=area, y=poptotal)) +
+   geom_point(aes(color=state), size=2) +
+   geom_smooth(method="lm",col="black") +
+   coord_cartesian(xlim=c(0,0.1), ylim=c(0,1000000)) +
+   labs(title="Area Vs Population", subtitle = "Using midwest dataset", y="Population", x="area", caption
= "Midwest Demographics")
> g + scale_x_continuous(breaks=seq(0, 0.10, 0.01))
`geom_smooth()` using formula 'y ~ x'
> g + scale_y_continuous(breaks=seq(0, 1000000, 50000))
> ggplot(df_allStocks, aes(x=Month, y=Close)) + geom_boxplot()
> ggplot(df_allStocks, aes(x=Month, y=Close)) + geom_boxplot() + coord_flip()
> ggplot(df_allStocks, aes(x=Month, y=Close, color=Month)) + geom_boxplot() + coord_flip()
> ggplot(df_midwest, aes(x=state, y=poptotal)) + geom_boxplot(outlier.color = "red", outlier.shape = 1,
outlier.size = 2)
> ggplot(df_allStocks, aes(x=Year, y=Close)) + geom_boxplot() + facet_grid(~ Stock)
> ggplot(df_allStocks, aes(x=Month, y=Close)) + geom_boxplot() + facet_grid(Stock ~ Year)
> ggplot(df_allStocks, aes(x=Open)) +
+   geom_histogram(color="black", fill="white") +
+   facet_grid(Stock ~ .)
`stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
> ggplot(df_allStocks, aes(x=Open, color=Stock)) +
+   geom_histogram(fill="white") +
+   facet_grid(Stock ~ .)
`stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
> ggplot(df_allStocks, aes(x=Close, color=Stock)) +
+   geom_histogram(fill="white") +
+   facet_grid(Stock ~ ., scales="free_y")
```

Day 5 – R Programming

```
> dev.off()
null device
          1
> setwd("C:/zubeda/PGA02_Zubu/R Programming")
> library("plyr")
> library("ggplot2")
> g <- ggplot(df_midwest, aes(x=area, y=poptotal)) +
+   geom_point(shape=18, color="dark grey") +
+   geom_smooth(method="lm", linetype="dashed", col="red") +
```

```
+   coord_cartesian(xlim=c(0, 0.1), ylim=c(0, 600000))
> g <- g + theme_light()
> g
`geom_smooth()` using formula 'y ~ x'
> ggplot(df_waltdisney, aes(x=Open, y=Close)) +
+   geom_point() + theme(panel.grid.major = element_line(size=0.5, linetype="dashed", colour="red"),
panel.background=element_rect(fill="lightblue"))
> ggplot(df_allStocks, aes(x=Stock, y=Volume)) +
+   geom_bar(stat="identity") + theme(panel.grid.major = element_line(size=0.5, linetype="solid",
colour="blue"), panel.background=element_rect(fill="lightblue"))
> library(RColorBrewer)
> head(brewer.pal.info, 12)
         maxcolors category
BrBG            11      div
PiYG            11      div
PRGn            11      div
PuOr            11      div
RdBu            11      div
RdGy            11      div
RdYlBu          11      div
RdYlGn          11      div
Spectral        11      div
Accent           8     qual
Dark2            8     qual
Paired          12     qual
         colorblind
BrBG           TRUE
PiYG           TRUE
PRGn           TRUE
PuOr           TRUE
RdBu           TRUE
RdGy          FALSE
RdYlBu         TRUE
RdYlGn        FALSE
Spectral      FALSE
Accent        FALSE
Dark2          TRUE
Paired         TRUE
> display.brewer.all()
> g <- ggplot(df_midwest, aes(x=area, y=poptotal)) +
+   geom_point(aes(color=state), size=2) +
+   geom_smooth(method="lm",col="black") +
+   coord_cartesian(xlim=c(0, 0.1), ylim=c(0, 1000000)) +
```

```
+   labs(title="Area Vs Population", subtitle = "Using midwest dataset", y="Population", x="area", caption = "Midwest Demographics")
> g <- g + scale_colour_brewer(palette="Dark2")
> g
`geom_smooth()` using formula 'y ~ x'
> g <- ggplot(df_midwest, aes(x=area, y=poptotal)) +
+   geom_point(aes(color=state), size=2) +
+   geom_smooth(method="lm",col="black") +
+   coord_cartesian(xlim=c(0, 0.1), ylim=c(0, 1000000)) +
+   labs(title="Area Vs Population", subtitle = "Using midwest dataset", y="Population", x="area", caption = "Midwest Demographics")
> library(grid)
> annotate_text <- "Showing population by area with best fit regression line"
> g
`geom_smooth()` using formula 'y ~ x'
> annotatechart <- grid.text(annotate_text, x=0.5, y=0.9, gp=gpar(col="darkred", fontsize=9, fontface="plain"))
```

Class Assessment

```
> setwd("C:/zubeda/PGA02_Zubu/R Programming")
> #Q1)
> #II. Create a vector of length 4 using seq() function and showcase how to access the elements using numeric indexes, logical indexes and character indexes.
> v <- seq(11, 15, length.out=4)   #returns 4 numbers, including 1st, last and middle numbers averaged if numbers are more then limit
> v
[1] 11.00000 12.33333 13.66667 15.00000
> v[1]
[1] 11
> v[3]
[1] 13.66667
> v[c(2, 4)]
[1] 12.33333 15.00000
> v[c(TRUE, FALSE, TRUE, FALSE)]
[1] 11.00000 13.66667
> names(v) <- c("el1", "el2", "el3", "el4")
> v
     el1      el2      el3      el4
11.00000 12.33333 13.66667 15.00000
> v["el1"]
el1
 11
> y <- c("Mumbai"=400, "Delhi"=100, "Chennai"=300, "Kolkata"=200)
> y
```

```
 Mumbai   Delhi Chennai Kolkata
    400     100     300     200
> y["Chennai"]
Chennai
    300
> y["Mumbai"]
Mumbai
   400
>
> #I.	Load the in-built dataset called trees, that consists of measurements of the girth, height, and volume of 31 black cherry trees and display rows where height is greater than 82
> ?trees
> trees
   Girth Height Volume
1    8.3     70   10.3
2    8.6     65   10.3
3    8.8     63   10.2
4   10.5     72   16.4
5   10.7     81   18.8
6   10.8     83   19.7
7   11.0     66   15.6
8   11.0     75   18.2
9   11.1     80   22.6
10  11.2     75   19.9
11  11.3     79   24.2
12  11.4     76   21.0
13  11.4     76   21.4
14  11.7     69   21.3
15  12.0     75   19.1
16  12.9     74   22.2
17  12.9     85   33.8
18  13.3     86   27.4
19  13.7     71   25.7
20  13.8     64   24.9
21  14.0     78   34.5
22  14.2     80   31.7
23  14.5     74   36.3
24  16.0     72   38.3
25  16.3     77   42.6
26  17.3     81   55.4
27  17.5     82   55.7
28  17.9     80   58.3
29  18.0     80   51.5
30  18.0     80   51.0
```

```
31  20.6     87   77.0
> dim(trees)
[1] 31  3
> nrow(trees)
[1] 31
> ncol(trees)
[1] 3
> summary(trees)
     Girth           Height       Volume
 Min.   : 8.30   Min.   :63   Min.   :10.20
 1st Qu.:11.05   1st Qu.:72   1st Qu.:19.40
 Median :12.90   Median :76   Median :24.20
 Mean   :13.25   Mean   :76   Mean   :30.17
 3rd Qu.:15.25   3rd Qu.:80   3rd Qu.:37.30
 Max.   :20.60   Max.   :87   Max.   :77.00
> names(trees)
[1] "Girth"  "Height" "Volume"
> str(trees)
'data.frame':	31 obs. of  3 variables:
 $ Girth : num  8.3 8.6 8.8 10.5 10.7 10.8 11 11 11.1 11.2 ...
 $ Height: num  70 65 63 72 81 83 66 75 80 75 ...
 $ Volume: num  10.3 10.3 10.2 16.4 18.8 19.7 15.6 18.2 22.6 19.9 ...
> trees[trees$Height > 82,]
   Girth Height Volume
6   10.8     83   19.7
17  12.9     85   33.8
18  13.3     86   27.4
31  20.6     87   77.0
>
> #Q2) For the 'StudentsPerformance' dataset, perform the following tasks:
> #I.	Analyze the student's performance in exams and write your own observations about the students and plot the results
> #II.	Create a function to remove outliers using the IQR method
>
> #Function definition such that outliers of passed columns are removed
> students <- read.csv("StudentsPerformance.csv")
> #Get Dimensions
> nrow(students)
[1] 1000
> ncol(students)
[1] 8
> #Get data types
> str(students)
'data.frame':	1000 obs. of  8 variables:
```

```
 $ gender                  : chr  "female" "female" "female" "male" ...
 $ race.ethnicity          : chr  "group B" "group C" "group B" "group A" ...
 $ parental.level.of.education: chr  "bachelor's degree" "some college" "master's degree" "associate's
degree" ...
 $ lunch                   : chr  "standard" "standard" "standard" "free/reduced" ...
 $ test.preparation.course    : chr  "none" "completed" "none" "none" ...
 $ math.score              : int  72 69 90 47 76 71 88 40 64 38 ...
 $ reading.score           : int  72 90 95 57 78 83 95 43 64 60 ...
 $ writing.score           : int  74 88 93 44 75 78 92 39 67 50 ...
> #rename column names with new column names
> namesOfColumns <- c("Gender", "Race", "Parent_Education", "Lunch", "Test_Prep", "Math_Score",
"Reading_Score", "Writing_Score")
> colnames(students) <- namesOfColumns
> colnames(students)
[1] "Gender"         "Race"           "Parent_Education" "Lunch"          "Test_Prep"       "Math_Score"
"Reading_Score"
[8] "Writing_Score"
> summary(students)  #Summary statistics of numeric variable
   Gender              Race               Parent_Education      Lunch             Test_Prep           Math_Score
Reading_Score
 Length:1000        Length:1000        Length:1000        Length:1000        Length:1000        Min.   :  0.00   Min.
: 17.00
 Class :character   Class :character   Class :character   Class :character   Class :character   1st Qu.: 57.00
1st Qu.: 59.00
 Mode  :character   Mode  :character   Mode  :character   Mode  :character   Mode  :character   Median :
66.00   Median : 70.00
                                                                                                 Mean   : 66.09   Mean   : 69.17
                                                                                                 3rd Qu.: 77.00   3rd Qu.: 79.00
                                                                                                 Max.   :100.00   Max.   :100.00
 Writing_Score
 Min.   : 10.00
 1st Qu.: 57.75
 Median : 69.00
 Mean   : 68.05
 3rd Qu.: 79.00
 Max.   :100.00
>
> #Obervations
> #1. There are more females than males
> #2. Group C has the largest number of members
> #3. some college and associates degree are the most frequently occuring #parental levels of education
> #4. most students have a standard lunch
> #5. most students have not completed the test prep course
> #6. the scores for math, reading and writing are on the same scale 0-100
```

```
>
> remove_outliers <- function(x, na.rm=TRUE, ...) {
+   qnt <- quantile(x, probs=c(.25, .75), na.rm=na.rm, ...)
+   H <- 1.5 * IQR(x, na.rm = na.rm)
+   y <- x
+   y[x < (qnt[1] - H)] <- NA
+   y[x > (qnt[2] + H)] <- NA
+   y
+ }
> #Combine columns categorical cols as it is, and last 3 cols with outliers removed
> performance_data <- cbind(students[1:5], apply(students[6], 2, remove_outliers), apply(students[7], 2, remove_outliers), apply(students[8], 2, remove_outliers))
> performance_data
   Gender    Race   Parent_Education        Lunch Test_Prep Math_Score Reading_Score Writing_Score
1   female group B  bachelor's degree     standard      none         72           72           74
2   female group C       some college     standard completed         69           90           88
3   female group B    master's degree     standard      none         90           95           93
4     male group A associate's degree free/reduced      none         47           57           44
5     male group C       some college     standard      none         76           78           75
6   female group B associate's degree     standard      none         71           83           78
7   female group B       some college     standard completed         88           95           92
8     male group B       some college free/reduced      none         40           43           39
9     male group D        high school free/reduced completed         64           64           67
10  female group B        high school free/reduced      none         38           60           50
11    male group C associate's degree     standard      none         58           54           52
12    male group D associate's degree     standard      none         40           52           43
13  female group B        high school     standard      none         65           81           73
14    male group A       some college     standard completed         78           72           70
15  female group A    master's degree     standard      none         50           53           58
16  female group C   some high school     standard      none         69           75           78
17    male group C        high school     standard      none         88           89           86
18  female group B   some high school free/reduced      none         NA           32           28
19    male group C    master's degree free/reduced completed         46           42           46
20  female group C associate's degree free/reduced      none         54           58           61
21    male group D        high school     standard      none         66           69           63
22  female group B       some college free/reduced completed         65           75           70
23    male group D       some college     standard      none         44           54           53
24  female group C   some high school     standard      none         69           73           73
25    male group D  bachelor's degree free/reduced completed         74           71           80
26    male group A    master's degree free/reduced      none         73           74           72
27    male group B       some college     standard      none         69           54           55
28  female group C  bachelor's degree     standard      none         67           69           75
29    male group C        high school     standard      none         70           70           65
30  female group D    master's degree     standard      none         62           70           75
```

31	female	group D	some college	standard	none	69	74	74
32	female	group B	some college	standard	none	63	65	61
33	female	group E	master's degree	free/reduced	none	56	72	65
34	male	group D	some college	standard	none	40	42	38
35	male	group E	some college	standard	none	97	87	82
36	male	group E	associate's degree	standard	completed	81	81	79
37	female	group D	associate's degree	standard	none	74	81	83
38	female	group D	some high school	free/reduced	none	50	64	59
39	female	group D	associate's degree	free/reduced	completed	75	90	88
40	male	group B	associate's degree	free/reduced	none	57	56	57
41	male	group C	associate's degree	free/reduced	none	55	61	54
42	female	group C	associate's degree	standard	none	58	73	68
43	female	group B	associate's degree	standard	none	53	58	65
44	male	group B	some college	free/reduced	completed	59	65	66
45	female	group E	associate's degree	free/reduced	none	50	56	54
46	male	group B	associate's degree	standard	none	65	54	57
47	female	group A	associate's degree	standard	completed	55	65	62
48	female	group C	high school	standard	none	66	71	76
49	female	group D	associate's degree	free/reduced	completed	57	74	76
50	male	group C	high school	standard	completed	82	84	82
51	male	group E	some college	standard	none	53	55	48
52	male	group E	associate's degree	free/reduced	completed	77	69	68
53	male	group C	some college	standard	none	53	44	42
54	male	group D	high school	standard	none	88	78	75
55	female	group C	some high school	free/reduced	completed	71	84	87
56	female	group C	high school	free/reduced	none	33	41	43
57	female	group E	associate's degree	standard	completed	82	85	86
58	male	group D	associate's degree	standard	none	52	55	49
59	male	group D	some college	standard	completed	58	59	58
60	female	group C	some high school	free/reduced	none	NA	NA	NA
61	male	group E	bachelor's degree	free/reduced	completed	79	74	72
62	male	group A	some high school	free/reduced	none	39	39	34
63	male	group A	associate's degree	free/reduced	none	62	61	55
64	female	group C	associate's degree	standard	none	69	80	71
65	female	group D	some high school	standard	none	59	58	59
66	male	group B	some high school	standard	none	67	64	61
67	male	group D	some high school	free/reduced	none	45	37	37
68	female	group C	some college	standard	none	60	72	74
69	male	group B	associate's degree	free/reduced	none	61	58	56
70	female	group C	associate's degree	standard	none	39	64	57
71	female	group D	some college	free/reduced	completed	58	63	73
72	male	group D	some college	standard	completed	63	55	63
73	female	group A	associate's degree	free/reduced	none	41	51	48
74	male	group C	some high school	free/reduced	none	61	57	56

75	male	group C	some high school	standard	none	49	49	41
76	male	group B	associate's degree	free/reduced	none	44	41	38
77	male	group E	some high school	standard	none	30	NA	NA
78	male	group A	bachelor's degree	standard	completed	80	78	81
79	female	group D	some high school	standard	completed	61	74	72
80	female	group E	master's degree	standard	none	62	68	68
81	female	group B	associate's degree	standard	none	47	49	50
82	male	group B	high school	free/reduced	none	49	45	45
83	male	group A	some college	free/reduced	completed	50	47	54
84	male	group E	associate's degree	standard	none	72	64	63
85	male	group D	high school	free/reduced	none	42	39	34
86	female	group C	some college	standard	none	73	80	82
87	female	group C	some college	free/reduced	none	76	83	88
88	female	group D	associate's degree	standard	none	71	71	74
89	female	group A	some college	standard	none	58	70	67
90	female	group D	some high school	standard	none	73	86	82
91	female	group C	bachelor's degree	standard	none	65	72	74
92	male	group C	high school	free/reduced	none	27	34	36
93	male	group C	high school	standard	none	71	79	71
94	male	group C	associate's degree	free/reduced	completed	43	45	50
95	female	group B	some college	standard	none	79	86	92
96	male	group C	associate's degree	free/reduced	completed	78	81	82
97	male	group B	some high school	standard	completed	65	66	62
98	female	group E	some college	standard	completed	63	72	70
99	female	group D	some college	free/reduced	none	58	67	62
100	female	group D	bachelor's degree	standard	none	65	67	62
101	male	group B	some college	standard	none	79	67	67
102	male	group D	bachelor's degree	standard	completed	68	74	74
103	female	group D	associate's degree	standard	none	85	91	89
104	male	group B	high school	standard	completed	60	44	47
105	male	group C	some college	standard	completed	98	86	90
106	female	group C	some college	standard	none	58	67	72
107	female	group D	master's degree	standard	none	87	100	100
108	male	group E	associate's degree	standard	completed	66	63	64
109	female	group B	associate's degree	free/reduced	none	52	76	70
110	female	group B	some high school	standard	none	70	64	72
111	female	group D	associate's degree	free/reduced	completed	77	89	98
112	male	group C	high school	standard	none	62	55	49
113	male	group A	associate's degree	standard	none	54	53	47
114	female	group D	some college	standard	none	51	58	54
115	female	group E	bachelor's degree	standard	completed	99	100	100
116	male	group C	high school	standard	none	84	77	74
117	female	group B	bachelor's degree	free/reduced	none	75	85	82
118	female	group D	bachelor's degree	standard	none	78	82	79

```
119 female group D   some high school     standard      none        51          63          61
120 female group C      some college     standard      none       55          69          65
121 female group C  bachelor's degree     standard completed        79          92          89
122   male group B associate's degree     standard completed        91          89          92
123 female group C      some college     standard completed        88          93          93
124   male group D       high school free/reduced      none        63          57          56
125   male group E      some college     standard      none        83          80          73
 [ reached 'max' / getOption("max.print") -- omitted 875 rows ]
> dim(performance_data)
[1] 1000    8
> sum(is.na(performance_data))   # Sum of null values
[1] 19
> performance_1 <- na.omit(performance_data)
> performance_1
   Gender    Race   Parent_Education        Lunch Test_Prep Math_Score Reading_Score Writing_Score
1   female group B  bachelor's degree     standard      none        72          72          74
2   female group C      some college     standard completed        69          90          88
3   female group B    master's degree     standard      none        90          95          93
4    male group A associate's degree free/reduced      none        47          57          44
5    male group C      some college     standard      none        76          78          75
6   female group B associate's degree     standard      none        71          83          78
7   female group B      some college     standard completed        88          95          92
8    male group B      some college free/reduced      none        40          43          39
9    male group D       high school free/reduced completed        64          64          67
10  female group B       high school free/reduced      none        38          60          50
11   male group C associate's degree     standard      none        58          54          52
12   male group D associate's degree     standard      none        40          52          43
13  female group B       high school     standard      none        65          81          73
14   male group A      some college     standard completed        78          72          70
15  female group A    master's degree     standard      none        50          53          58
16  female group C   some high school     standard      none        69          75          78
17   male group C       high school     standard      none        88          89          86
19   male group C    master's degree free/reduced completed        46          42          46
20  female group C associate's degree free/reduced      none        54          58          61
21   male group D       high school     standard      none        66          69          63
22  female group B      some college free/reduced completed        65          75          70
23   male group D      some college     standard      none        44          54          53
24  female group C   some high school     standard      none        69          73          73
25   male group D  bachelor's degree free/reduced completed        74          71          80
26   male group A    master's degree free/reduced      none        73          74          72
27   male group B      some college     standard      none        69          54          55
28  female group C  bachelor's degree     standard      none        67          69          75
29   male group C       high school     standard      none        70          70          65
30  female group D    master's degree     standard      none        62          70          75
```

31	female	group D	some college	standard	none	69	74	74
32	female	group B	some college	standard	none	63	65	61
33	female	group E	master's degree	free/reduced	none	56	72	65
34	male	group D	some college	standard	none	40	42	38
35	male	group E	some college	standard	none	97	87	82
36	male	group E	associate's degree	standard	completed	81	81	79
37	female	group D	associate's degree	standard	none	74	81	83
38	female	group D	some high school	free/reduced	none	50	64	59
39	female	group D	associate's degree	free/reduced	completed	75	90	88
40	male	group B	associate's degree	free/reduced	none	57	56	57
41	male	group C	associate's degree	free/reduced	none	55	61	54
42	female	group C	associate's degree	standard	none	58	73	68
43	female	group B	associate's degree	standard	none	53	58	65
44	male	group B	some college	free/reduced	completed	59	65	66
45	female	group E	associate's degree	free/reduced	none	50	56	54
46	male	group B	associate's degree	standard	none	65	54	57
47	female	group A	associate's degree	standard	completed	55	65	62
48	female	group C	high school	standard	none	66	71	76
49	female	group D	associate's degree	free/reduced	completed	57	74	76
50	male	group C	high school	standard	completed	82	84	82
51	male	group E	some college	standard	none	53	55	48
52	male	group E	associate's degree	free/reduced	completed	77	69	68
53	male	group C	some college	standard	none	53	44	42
54	male	group D	high school	standard	none	88	78	75
55	female	group C	some high school	free/reduced	completed	71	84	87
56	female	group C	high school	free/reduced	none	33	41	43
57	female	group E	associate's degree	standard	completed	82	85	86
58	male	group D	associate's degree	standard	none	52	55	49
59	male	group D	some college	standard	completed	58	59	58
61	male	group E	bachelor's degree	free/reduced	completed	79	74	72
62	male	group A	some high school	free/reduced	none	39	39	34
63	male	group A	associate's degree	free/reduced	none	62	61	55
64	female	group C	associate's degree	standard	none	69	80	71
65	female	group D	some high school	standard	none	59	58	59
66	male	group B	some high school	standard	none	67	64	61
67	male	group D	some high school	free/reduced	none	45	37	37
68	female	group C	some college	standard	none	60	72	74
69	male	group B	associate's degree	free/reduced	none	61	58	56
70	female	group C	associate's degree	standard	none	39	64	57
71	female	group D	some college	free/reduced	completed	58	63	73
72	male	group D	some college	standard	completed	63	55	63
73	female	group A	associate's degree	free/reduced	none	41	51	48
74	male	group C	some high school	free/reduced	none	61	57	56
75	male	group C	some high school	standard	none	49	49	41

76	male	group B	associate's degree	free/reduced	none	44	41	38
78	male	group A	bachelor's degree	standard	completed	80	78	81
79	female	group D	some high school	standard	completed	61	74	72
80	female	group E	master's degree	standard	none	62	68	68
81	female	group B	associate's degree	standard	none	47	49	50
82	male	group B	high school	free/reduced	none	49	45	45
83	male	group A	some college	free/reduced	completed	50	47	54
84	male	group E	associate's degree	standard	none	72	64	63
85	male	group D	high school	free/reduced	none	42	39	34
86	female	group C	some college	standard	none	73	80	82
87	female	group C	some college	free/reduced	none	76	83	88
88	female	group D	associate's degree	standard	none	71	71	74
89	female	group A	some college	standard	none	58	70	67
90	female	group D	some high school	standard	none	73	86	82
91	female	group C	bachelor's degree	standard	none	65	72	74
92	male	group C	high school	free/reduced	none	27	34	36
93	male	group C	high school	standard	none	71	79	71
94	male	group C	associate's degree	free/reduced	completed	43	45	50
95	female	group B	some college	standard	none	79	86	92
96	male	group C	associate's degree	free/reduced	completed	78	81	82
97	male	group B	some high school	standard	completed	65	66	62
98	female	group E	some college	standard	completed	63	72	70
99	female	group D	some college	free/reduced	none	58	67	62
100	female	group D	bachelor's degree	standard	none	65	67	62
101	male	group B	some college	standard	none	79	67	67
102	male	group D	bachelor's degree	standard	completed	68	74	74
103	female	group D	associate's degree	standard	none	85	91	89
104	male	group B	high school	standard	completed	60	44	47
105	male	group C	some college	standard	completed	98	86	90
106	female	group C	some college	standard	none	58	67	72
107	female	group D	master's degree	standard	none	87	100	100
108	male	group E	associate's degree	standard	completed	66	63	64
109	female	group B	associate's degree	free/reduced	none	52	76	70
110	female	group B	some high school	standard	none	70	64	72
111	female	group D	associate's degree	free/reduced	completed	77	89	98
112	male	group C	high school	standard	none	62	55	49
113	male	group A	associate's degree	standard	none	54	53	47
114	female	group D	some college	standard	none	51	58	54
115	female	group E	bachelor's degree	standard	completed	99	100	100
116	male	group C	high school	standard	none	84	77	74
117	female	group B	bachelor's degree	free/reduced	none	75	85	82
118	female	group D	bachelor's degree	standard	none	78	82	79
119	female	group D	some high school	standard	none	51	63	61
120	female	group C	some college	standard	none	55	69	65

```
121 female group C  bachelor's degree     standard completed        79          92          89
122   male group B associate's degree     standard completed        91          89          92
123 female group C       some college     standard completed        88          93          93
124   male group D        high school free/reduced      none        63          57          56
125   male group E       some college     standard      none        83          80          73
126 female group B        high school     standard      none        87          95          86
127   male group B   some high school     standard      none        72          68          67
128   male group D       some college     standard completed        65          77          74
 [ reached 'max' / getOption("max.print") -- omitted 863 rows ]
> nrow(performance_1)
[1] 988
> library(ggplot2)
> Data <- performance_1
>
> plot1 <-
+   ggplot() +
+   geom_bar(data = Data, aes(x = Gender), width = 0.2, fill = "green") +
+   geom_text(stat='count', data = Data, aes(x = Gender, label=..count..), vjust=-0.2) +
+   theme_bw() +
+   xlab("Gender") +
+   ylab("Number of Students") +
+   theme_classic() +
+   ggtitle("Number of Students by Gender") +
+   scale_fill_brewer(type = "qual", palette = 1, direction = 1,
+                     aesthetics = "fill") +
+   ylim(0, 600)
>
> plot1
>
> #There are more 510 female students and 478 male students.
>
> #Students By race:
> plot2 <- ggplot() +
+   geom_bar(data = Data, aes(x = Race), width = 0.6, fill = "green") +
+   geom_text(data = Data, aes(x = Race, label = ..count..), stat = "count", vjust = -0.2) +
+   theme_bw() +
+   xlab("Race/Ethnicity") +
+   ylab("Number of Students") +
+   theme(
+     text = element_text(family = "Tahoma")
+   ) +
+   theme_classic()+
+   scale_fill_brewer(type = "qual", palette = 1, direction = 1,
+                     aesthetics = "fill") +
```

```
+   ggtitle("Number of Students by Race/Ethnicity")
> plot2
> #There are 316 students in group C, 261 students in group D while there are only 88 students in group
A.
>
> #Plot scores by Gender to determine if there is a different score tendency for each gender
> # Math scores by Gender plot
> p <- ggplot(students, aes(Math_Score)) + geom_histogram(binwidth=5, color="gray", aes(fill=Gender))
> p <- p + xlab("Math Scores") + ylab("Gender") + ggtitle("Math Scores by Gender")
> p
>
> # Boxplot of scores and Test Prep by Gender
> b <- ggplot(students, aes(Gender, Writing_Score, color = Test_Prep))
> b <- b + geom_boxplot()
> b <- b + ggtitle("Writing scores by Gender Boxplot")
> b <- b + xlab("Gender") + ylab("Writing Scores")
> b
>
> # Reading scores by Gender plot
> p1 <- ggplot(students, aes(Reading_Score)) + geom_histogram(binwidth=5, color="gray",
aes(fill=Gender))
> p1 <- p1 + xlab("Reading Scores") + ylab("Gender") + ggtitle("Reading Scores by Gender")
> p1
>
> b1 <- ggplot(students, aes(Gender, Math_Score, color = Test_Prep))
> b1 <- b1 + geom_boxplot()
> b1 <- b1 + ggtitle("Math scores by Gender Boxplot")
> b1 <- b1 + xlab("Gender") + ylab("Math Scores")
> b1
>
> # Writing scores by Gender plot
> p2 <- ggplot(students, aes(Writing_Score)) +  geom_histogram(binwidth=5, color="gray",
aes(fill=Gender))
> p2 <- p2 + xlab("Writing Scores") + ylab("Gender") + ggtitle("Writing Scores by Gender")
> p2
>
> b2 <- ggplot(students, aes(Gender, Reading_Score, color = Test_Prep))
> b2 <- b2 + geom_boxplot()
> b2 <- b2 + ggtitle("Reading scores by Gender Boxplot")
> b2 <- b2 + xlab("Gender") + ylab("Reading Scores")
> b2
>
> #Conclusion :
>
```

```
> #1. students who completed the prep class had better scores in all three tests.
> #2. male students have received better scores in Math while female students in reading and writing.
>
> #Which gender does better in tests
> # To find out the result, we need to create a columns that stores average of score
> performance_2 <- performance_1
> performance_2$Total_score = performance_2$Math_Score + performance_2$Reading_Score
+performance_2$Writing_Score
> performance_2$Avg_score = round((performance_2$Total_score)/3,0)
> performance_2
   Gender    Race   Parent_Education       Lunch Test_Prep Math_Score Reading_Score Writing_Score
Total_score Avg_score
1   female group B  bachelor's degree     standard      none        72          72          74        218       73
2   female group C       some college     standard completed        69          90          88        247       82
3   female group B    master's degree     standard      none        90          95          93        278       93
4     male group A associate's degree free/reduced      none        47          57          44        148       49
5     male group C       some college     standard      none        76          78          75        229       76
6   female group B associate's degree     standard      none        71          83          78        232       77
7   female group B       some college     standard completed        88          95          92        275       92
8     male group B       some college free/reduced      none        40          43          39        122       41
9     male group D        high school free/reduced completed        64          64          67        195       65
10  female group B        high school free/reduced      none        38          60          50        148       49
11    male group C associate's degree     standard      none        58          54          52        164       55
12    male group D associate's degree     standard      none        40          52          43        135       45
13  female group B        high school     standard      none        65          81          73        219       73
14    male group A       some college     standard completed        78          72          70        220       73
15  female group A    master's degree     standard      none        50          53          58        161       54
16  female group C   some high school     standard      none        69          75          78        222       74
17    male group C        high school     standard      none        88          89          86        263       88
19    male group C    master's degree free/reduced completed        46          42          46        134       45
20  female group C associate's degree free/reduced      none        54          58          61        173       58
21    male group D        high school     standard      none        66          69          63        198       66
22  female group B       some college free/reduced completed        65          75          70        210       70
23    male group D       some college     standard      none        44          54          53        151       50
24  female group C   some high school     standard      none        69          73          73        215       72
25    male group D  bachelor's degree free/reduced completed        74          71          80        225       75
26    male group A    master's degree free/reduced      none        73          74          72        219       73
27    male group B       some college     standard      none        69          54          55        178       59
28  female group C  bachelor's degree     standard      none        67          69          75        211       70
29    male group C        high school     standard      none        70          70          65        205       68
30  female group D    master's degree     standard      none        62          70          75        207       69
31  female group D       some college     standard      none        69          74          74        217       72
32  female group B       some college     standard      none        63          65          61        189       63
33  female group E    master's degree free/reduced      none        56          72          65        193       64
```

34	male	group D	some college	standard	none	40	42	38	120	40
35	male	group E	some college	standard	none	97	87	82	266	89
36	male	group E	associate's degree	standard	completed	81	81	79	241	80
37	female	group D	associate's degree	standard	none	74	81	83	238	79
38	female	group D	some high school	free/reduced	none	50	64	59	173	58
39	female	group D	associate's degree	free/reduced	completed	75	90	88	253	84
40	male	group B	associate's degree	free/reduced	none	57	56	57	170	57
41	male	group C	associate's degree	free/reduced	none	55	61	54	170	57
42	female	group C	associate's degree	standard	none	58	73	68	199	66
43	female	group B	associate's degree	standard	none	53	58	65	176	59
44	male	group B	some college	free/reduced	completed	59	65	66	190	63
45	female	group E	associate's degree	free/reduced	none	50	56	54	160	53
46	male	group B	associate's degree	standard	none	65	54	57	176	59
47	female	group A	associate's degree	standard	completed	55	65	62	182	61
48	female	group C	high school	standard	none	66	71	76	213	71
49	female	group D	associate's degree	free/reduced	completed	57	74	76	207	69
50	male	group C	high school	standard	completed	82	84	82	248	83
51	male	group E	some college	standard	none	53	55	48	156	52
52	male	group E	associate's degree	free/reduced	completed	77	69	68	214	71
53	male	group C	some college	standard	none	53	44	42	139	46
54	male	group D	high school	standard	none	88	78	75	241	80
55	female	group C	some high school	free/reduced	completed	71	84	87	242	81
56	female	group C	high school	free/reduced	none	33	41	43	117	39
57	female	group E	associate's degree	standard	completed	82	85	86	253	84
58	male	group D	associate's degree	standard	none	52	55	49	156	52
59	male	group D	some college	standard	completed	58	59	58	175	58
61	male	group E	bachelor's degree	free/reduced	completed	79	74	72	225	75
62	male	group A	some high school	free/reduced	none	39	39	34	112	37
63	male	group A	associate's degree	free/reduced	none	62	61	55	178	59
64	female	group C	associate's degree	standard	none	69	80	71	220	73
65	female	group D	some high school	standard	none	59	58	59	176	59
66	male	group B	some high school	standard	none	67	64	61	192	64
67	male	group D	some high school	free/reduced	none	45	37	37	119	40
68	female	group C	some college	standard	none	60	72	74	206	69
69	male	group B	associate's degree	free/reduced	none	61	58	56	175	58
70	female	group C	associate's degree	standard	none	39	64	57	160	53
71	female	group D	some college	free/reduced	completed	58	63	73	194	65
72	male	group D	some college	standard	completed	63	55	63	181	60
73	female	group A	associate's degree	free/reduced	none	41	51	48	140	47
74	male	group C	some high school	free/reduced	none	61	57	56	174	58
75	male	group C	some high school	standard	none	49	49	41	139	46
76	male	group B	associate's degree	free/reduced	none	44	41	38	123	41
78	male	group A	bachelor's degree	standard	completed	80	78	81	239	80
79	female	group D	some high school	standard	completed	61	74	72	207	69

```
80  female group E    master's degree     standard      none        62          68          68        198       66
81  female group B associate's degree     standard      none        47          49          50        146       49
82    male group B       high school free/reduced       none        49          45          45        139       46
83    male group A      some college free/reduced completed         50          47          54        151       50
84    male group E associate's degree     standard      none        72          64          63        199       66
85    male group D       high school free/reduced       none        42          39          34        115       38
86  female group C      some college     standard      none        73          80          82        235       78
87  female group C      some college free/reduced       none        76          83          88        247       82
88  female group D associate's degree     standard      none        71          71          74        216       72
89  female group A      some college     standard      none        58          70          67        195       65
90  female group D   some high school     standard      none        73          86          82        241       80
91  female group C  bachelor's degree     standard      none        65          72          74        211       70
92    male group C       high school free/reduced       none        27          34          36         97       32
93    male group C       high school     standard      none        71          79          71        221       74
94    male group C associate's degree free/reduced completed        43          45          50        138       46
95  female group B      some college     standard      none        79          86          92        257       86
96    male group C associate's degree free/reduced completed        78          81          82        241       80
97    male group B   some high school     standard completed        65          66          62        193       64
98  female group E      some college     standard completed        63          72          70        205       68
99  female group D      some college free/reduced       none        58          67          62        187       62
100 female group D  bachelor's degree     standard      none        65          67          62        194       65
101   male group B      some college     standard      none        79          67          67        213       71
102   male group D  bachelor's degree     standard completed        68          74          74        216       72
103 female group D associate's degree     standard      none        85          91          89        265       88
 [ reached 'max' / getOption("max.print") -- omitted 888 rows ]
>
> #comparison of avg scores - male vs female
> ggplot(performance_2, aes( x= Avg_score, color = Gender))+
+   geom_density() +
+   geom_vline( color = "red",linetype = "dashed", lwd =0.5  ,xintercept =
mean(performance_2[performance_2$Gender == "female",]$Avg_score))+
+   geom_vline( color = "cyan",linetype = "dashed", lwd=0.5 ,  xintercept =
mean(performance_2[performance_2$Gender == "male",]$Avg_score)) +
+   labs(title ="Distribution of scores by Gender", x ="Score", y = " Density")
>
> #From the above density plot, we see that scores of female students have a higher mean than male
students.
>
> #Q3) For the given ‘chinook’ database, perform the following tasks:
> #install.packages("DBI")
> library(DBI)
> #install.packages("readr")
> library(readr)
> #install.packages("RSQLite")
```

```
> library(RSQLite)
>
> #I.    Connect to the above database and convert all the tables into data frame
> con <- dbConnect(RSQLite::SQLite(),"chinook.db")
> db <- dbConnect(dbDriver("SQLite"), dbname="chinook.db")
> dbListTables(db)
 [1] "albums"        "artists"        "customers"      "employees"      "genres"         "invoice_items"
"invoices"
 [8] "media_types"     "playlist_track"  "playlists"       "sqlite_sequence" "sqlite_stat1"    "tracks"
>
> albums <- dbReadTable(db, "albums")
> head(albums)
  AlbumId                                 Title ArtistId
1       1 For Those About To Rock We Salute You        1
2       2                     Balls to the Wall        2
3       3                     Restless and Wild        2
4       4                     Let There Be Rock        1
5       5                              Big Ones        3
6       6                    Jagged Little Pill        4
> artists <- dbReadTable(db, "artists")
> head(artists)
  ArtistId                 Name
1        1                AC/DC
2        2               Accept
3        3            Aerosmith
4        4    Alanis Morissette
5        5      Alice In Chains
6        6 Antônio Carlos Jobim
> customers <- dbReadTable(db, "customers")
> head(customers)
  CustomerId FirstName    LastName                                          Company                         Address                City
1          1      Luís   Gonçalves Embraer - Empresa Brasileira de Aeronáutica S.A. Av. Brigadeiro Faria Lima,
2170 São José dos Campos
2          2    Leonie      Köhler                                             <NA>         Theodor-Heuss-Straße 34           Stuttgart
3          3  François    Tremblay                                             <NA>               1498 rue Bélanger            Montréal
4          4     Bjørn      Hansen                                             <NA>                Ullevålsveien 14                Oslo
5          5 František Wichterlová                                 JetBrains s.r.o.                  Klanova 9/506              Prague
6          6    Helena        Holý                                             <NA>                   Rilská 3174/6              Prague
  State        Country PostalCode              Phone                 Fax                    Email SupportRepId
1    SP         Brazil  12227-000 +55 (12) 3923-5555 +55 (12) 3923-5566    luisg@embraer.com.br            3
2  <NA>        Germany      70174   +49 0711 2842222                <NA>    leonekohler@surfeu.de            5
3    QC         Canada    H2G 1A7  +1 (514) 721-4711                <NA>      ftremblay@gmail.com            3
4  <NA>         Norway       0171    +47 22 44 22 22                <NA>    bjorn.hansen@yahoo.no            4
```

```
5  <NA> Czech Republic      14700   +420 2 4172 5555   +420 2 4172 5555 frantisekw@jetbrains.com
4
6  <NA> Czech Republic      14300   +420 2 4177 0449            <NA>        hholy@gmail.com          5
> employees <- dbReadTable(db, "employees")
> head(employees)
  EmployeeId LastName FirstName             Title ReportsTo          BirthDate            HireDate
Address     City
1          1    Adams    Andrew    General Manager        NA 1962-02-18 00:00:00 2002-08-14 00:00:00
11120 Jasper Ave NW Edmonton
2          2  Edwards     Nancy      Sales Manager         1 1958-12-08 00:00:00 2002-05-01 00:00:00        825
8 Ave SW  Calgary
3          3  Peacock      Jane Sales Support Agent        2 1973-08-29 00:00:00 2002-04-01 00:00:00       1111
6 Ave SW  Calgary
4          4     Park  Margaret Sales Support Agent        2 1947-09-19 00:00:00 2003-05-03 00:00:00     683
10 Street SW  Calgary
5          5  Johnson     Steve Sales Support Agent        2 1965-03-03 00:00:00 2003-10-17 00:00:00
7727B 41 Ave  Calgary
6          6 Mitchell   Michael         IT Manager         1 1973-07-01 00:00:00 2003-10-17 00:00:00 5827
Bowness Road NW  Calgary
  State Country PostalCode             Phone               Fax                    Email
1    AB  Canada    T5K 2N1 +1 (780) 428-9482 +1 (780) 428-3457   andrew@chinookcorp.com
2    AB  Canada    T2P 2T3 +1 (403) 262-3443 +1 (403) 262-3322    nancy@chinookcorp.com
3    AB  Canada    T2P 5M5 +1 (403) 262-3443 +1 (403) 262-6712     jane@chinookcorp.com
4    AB  Canada    T2P 5G3 +1 (403) 263-4423 +1 (403) 263-4289 margaret@chinookcorp.com
5    AB  Canada    T3B 1Y7  1 (780) 836-9987  1 (780) 836-9543    steve@chinookcorp.com
6    AB  Canada    T3B 0C5 +1 (403) 246-9887 +1 (403) 246-9899  michael@chinookcorp.com
> genres <- dbReadTable(db, "genres")
> head(genres)
  GenreId               Name
1       1               Rock
2       2               Jazz
3       3              Metal
4       4 Alternative & Punk
5       5      Rock And Roll
6       6              Blues
> invoice_items <- dbReadTable(db, "invoice_items")
> head(invoice_items)
  InvoiceLineId InvoiceId TrackId UnitPrice Quantity
1             1         1       2      0.99        1
2             2         1       4      0.99        1
3             3         2       6      0.99        1
4             4         2       8      0.99        1
5             5         2      10      0.99        1
6             6         2      12      0.99        1
```

```
> invoices <- dbReadTable(db, "invoices")
> head(invoices)
  InvoiceId CustomerId         InvoiceDate          BillingAddress BillingCity BillingState BillingCountry
BillingPostalCode Total
1         1          2 2009-01-01 00:00:00 Theodor-Heuss-Straße 34   Stuttgart         <NA>        Germany
70174  1.98
2         2          4 2009-01-02 00:00:00         Ullevålsveien 14        Oslo         <NA>         Norway            0171
3.96
3         3          8 2009-01-03 00:00:00          Grétrystraat 63    Brussels         <NA>        Belgium            1000
5.94
4         4         14 2009-01-06 00:00:00           8210 111 ST NW    Edmonton           AB         Canada          T6G
2C7  8.91
5         5         23 2009-01-11 00:00:00          69 Salem Street      Boston           MA            USA            2113
13.86
6         6         37 2009-01-19 00:00:00          Berger Straße 10   Frankfurt         <NA>        Germany
60316  0.99
> media_types <- dbReadTable(db, "media_types")
> head(media_types)
  MediaTypeId                        Name
1           1             MPEG audio file
2           2    Protected AAC audio file
3           3 Protected MPEG-4 video file
4           4    Purchased AAC audio file
5           5              AAC audio file
> playlist_track <- dbReadTable(db, "playlist_track")
> head(playlist_track)
  PlaylistId TrackId
1          1    3402
2          1    3389
3          1    3390
4          1    3391
5          1    3392
6          1    3393
> playlists <- dbReadTable(db, "playlists")
> head(playlists)
  PlaylistId       Name
1          1      Music
2          2     Movies
3          3   TV Shows
4          4 Audiobooks
5          5 90’s Music
6          6 Audiobooks
> tracks <- dbReadTable(db, "tracks")
> head(tracks)
```

```
  TrackId                               Name AlbumId MediaTypeId GenreId
1       1 For Those About To Rock (We Salute You)       1           1       1
2       2                       Balls to the Wall       2           2       1
3       3                         Fast As a Shark       3           2       1
4       4                       Restless and Wild       3           2       1
5       5                    Princess of the Dawn       3           2       1
6       6                   Put The Finger On You       1           1       1
                                                    Composer Milliseconds    Bytes UnitPrice
1                          Angus Young, Malcolm Young, Brian Johnson       343719 11170334      0.99
2                                                               <NA>       342562  5510424      0.99
3                 F. Baltes, S. Kaufman, U. Dirkscneider & W. Hoffman       230619  3990994      0.99
4 F. Baltes, R.A. Smith-Diesel, S. Kaufman, U. Dirkscneider & W. Hoffman       252051  4331779      0.99
5                                      Deaffy & R.A. Smith-Diesel       375418  6290521      0.99
6                          Angus Young, Malcolm Young, Brian Johnson       205662  6713451      0.99
>
> #II.     Print the different types of music available
> genres$Name
 [1] "Rock"               "Jazz"               "Metal"              "Alternative & Punk" "Rock And Roll"      "Blues"
 [7] "Latin"              "Reggae"             "Pop"                "Soundtrack"         "Bossa Nova"         "Easy Listening"
[13] "Heavy Metal"        "R&B/Soul"           "Electronica/Dance"  "World"              "Hip Hop/Rap"
"Science Fiction"
[19] "TV Shows"           "Sci Fi & Fantasy"   "Drama"              "Comedy"             "Alternative"        "Classical"
[25] "Opera"
>
> #III.     List out all the artists from the entire database
> artists$Name
  [1] "AC/DC"
  [2] "Accept"
  [3] "Aerosmith"
  [4] "Alanis Morissette"
  [5] "Alice In Chains"
  [6] "Antônio Carlos Jobim"
  [7] "Apocalyptica"
  [8] "Audioslave"
  [9] "BackBeat"
 [10] "Billy Cobham"
 [11] "Black Label Society"
 [12] "Black Sabbath"
 [13] "Body Count"
 [14] "Bruce Dickinson"
 [15] "Buddy Guy"
 [16] "Caetano Veloso"
 [17] "Chico Buarque"
 [18] "Chico Science & Nação Zumbi"
```

[19] "Cidade Negra"
[20] "Cláudio Zoli"
[21] "Various Artists"
[22] "Led Zeppelin"
[23] "Frank Zappa & Captain Beefheart"
[24] "Marcos Valle"
[25] "Milton Nascimento & Bebeto"
[26] "Azymuth"
[27] "Gilberto Gil"
[28] "João Gilberto"
[29] "Bebel Gilberto"
[30] "Jorge Vercilo"
[31] "Baby Consuelo"
[32] "Ney Matogrosso"
[33] "Luiz Melodia"
[34] "Nando Reis"
[35] "Pedro Luís & A Parede"
[36] "O Rappa"
[37] "Ed Motta"
[38] "Banda Black Rio"
[39] "Fernanda Porto"
[40] "Os Cariocas"
[41] "Elis Regina"
[42] "Milton Nascimento"
[43] "A Cor Do Som"
[44] "Kid Abelha"
[45] "Sandra De Sá"
[46] "Jorge Ben"
[47] "Hermeto Pascoal"
[48] "Barão Vermelho"
[49] "Edson, DJ Marky & DJ Patife Featuring Fernanda Porto"
[50] "Metallica"
[51] "Queen"
[52] "Kiss"
[53] "Spyro Gyra"
[54] "Green Day"
[55] "David Coverdale"
[56] "Gonzaguinha"
[57] "Os Mutantes"
[58] "Deep Purple"
[59] "Santana"
[60] "Santana Feat. Dave Matthews"
[61] "Santana Feat. Everlast"
[62] "Santana Feat. Rob Thomas"

[63] "Santana Feat. Lauryn Hill & Cee-Lo"
[64] "Santana Feat. The Project G&B"
[65] "Santana Feat. Maná"
[66] "Santana Feat. Eagle-Eye Cherry"
[67] "Santana Feat. Eric Clapton"
[68] "Miles Davis"
[69] "Gene Krupa"
[70] "Toquinho & Vinícius"
[71] "Vinícius De Moraes & Baden Powell"
[72] "Vinícius De Moraes"
[73] "Vinícius E Qurteto Em Cy"
[74] "Vinícius E Odette Lara"
[75] "Vinicius, Toquinho & Quarteto Em Cy"
[76] "Creedence Clearwater Revival"
[77] "Cássia Eller"
[78] "Def Leppard"
[79] "Dennis Chambers"
[80] "Djavan"
[81] "Eric Clapton"
[82] "Faith No More"
[83] "Falamansa"
[84] "Foo Fighters"
[85] "Frank Sinatra"
[86] "Funk Como Le Gusta"
[87] "Godsmack"
[88] "Guns N' Roses"
[89] "Incognito"
[90] "Iron Maiden"
[91] "James Brown"
[92] "Jamiroquai"
[93] "JET"
[94] "Jimi Hendrix"
[95] "Joe Satriani"
[96] "Jota Quest"
[97] "João Suplicy"
[98] "Judas Priest"
[99] "Legião Urbana"
[100] "Lenny Kravitz"
[101] "Lulu Santos"
[102] "Marillion"
[103] "Marisa Monte"
[104] "Marvin Gaye"
[105] "Men At Work"
[106] "Motörhead"

[107] "Motörhead & Girlschool"
[108] "Mônica Marianno"
[109] "Mötley Crüe"
[110] "Nirvana"
[111] "O Terço"
[112] "Olodum"
[113] "Os Paralamas Do Sucesso"
[114] "Ozzy Osbourne"
[115] "Page & Plant"
[116] "Passengers"
[117] "Paul D'Ianno"
[118] "Pearl Jam"
[119] "Peter Tosh"
[120] "Pink Floyd"
[121] "Planet Hemp"
[122] "R.E.M. Feat. Kate Pearson"
[123] "R.E.M. Feat. KRS-One"
[124] "R.E.M."
[125] "Raimundos"
[126] "Raul Seixas"
[127] "Red Hot Chili Peppers"
[128] "Rush"
[129] "Simply Red"
[130] "Skank"
[131] "Smashing Pumpkins"
[132] "Soundgarden"
[133] "Stevie Ray Vaughan & Double Trouble"
[134] "Stone Temple Pilots"
[135] "System Of A Down"
[136] "Terry Bozzio, Tony Levin & Steve Stevens"
[137] "The Black Crowes"
[138] "The Clash"
[139] "The Cult"
[140] "The Doors"
[141] "The Police"
[142] "The Rolling Stones"
[143] "The Tea Party"
[144] "The Who"
[145] "Tim Maia"
[146] "Titãs"
[147] "Battlestar Galactica"
[148] "Heroes"
[149] "Lost"
[150] "U2"

[151] "UB40"
[152] "Van Halen"
[153] "Velvet Revolver"
[154] "Whitesnake"
[155] "Zeca Pagodinho"
[156] "The Office"
[157] "Dread Zeppelin"
[158] "Battlestar Galactica (Classic)"
[159] "Aquaman"
[160] "Christina Aguilera featuring BigElf"
[161] "Aerosmith & Sierra Leone's Refugee Allstars"
[162] "Los Lonely Boys"
[163] "Corinne Bailey Rae"
[164] "Dhani Harrison & Jakob Dylan"
[165] "Jackson Browne"
[166] "Avril Lavigne"
[167] "Big & Rich"
[168] "Youssou N'Dour"
[169] "Black Eyed Peas"
[170] "Jack Johnson"
[171] "Ben Harper"
[172] "Snow Patrol"
[173] "Matisyahu"
[174] "The Postal Service"
[175] "Jaguares"
[176] "The Flaming Lips"
[177] "Jack's Mannequin & Mick Fleetwood"
[178] "Regina Spektor"
[179] "Scorpions"
[180] "House Of Pain"
[181] "Xis"
[182] "Nega Gizza"
[183] "Gustavo & Andres Veiga & Salazar"
[184] "Rodox"
[185] "Charlie Brown Jr."
[186] "Pedro Luís E A Parede"
[187] "Los Hermanos"
[188] "Mundo Livre S/A"
[189] "Otto"
[190] "Instituto"
[191] "Nação Zumbi"
[192] "DJ Dolores & Orchestra Santa Massa"
[193] "Seu Jorge"
[194] "Sabotage E Instituto"

[195] "Stereo Maracana"
[196] "Cake"
[197] "Aisha Duo"
[198] "Habib Koité and Bamada"
[199] "Karsh Kale"
[200] "The Posies"
[201] "Luciana Souza/Romero Lubambo"
[202] "Aaron Goldberg"
[203] "Nicolaus Esterhazy Sinfonia"
[204] "Temple of the Dog"
[205] "Chris Cornell"
[206] "Alberto Turco & Nova Schola Gregoriana"
[207] "Richard Marlow & The Choir of Trinity College, Cambridge"
[208] "English Concert & Trevor Pinnock"
[209] "Anne-Sophie Mutter, Herbert Von Karajan & Wiener Philharmoniker"
[210] "Hilary Hahn, Jeffrey Kahane, Los Angeles Chamber Orchestra & Margaret Batjer"
[211] "Wilhelm Kempff"
[212] "Yo-Yo Ma"
[213] "Scholars Baroque Ensemble"
[214] "Academy of St. Martin in the Fields & Sir Neville Marriner"
[215] "Academy of St. Martin in the Fields Chamber Ensemble & Sir Neville Marriner"
[216] "Berliner Philharmoniker, Claudio Abbado & Sabine Meyer"
[217] "Royal Philharmonic Orchestra & Sir Thomas Beecham"
[218] "Orchestre Révolutionnaire et Romantique & John Eliot Gardiner"
[219] "Britten Sinfonia, Ivor Bolton & Lesley Garrett"
[220] "Chicago Symphony Chorus, Chicago Symphony Orchestra & Sir Georg Solti"
[221] "Sir Georg Solti & Wiener Philharmoniker"
[222] "Academy of St. Martin in the Fields, John Birch, Sir Neville Marriner & Sylvia McNair"
[223] "London Symphony Orchestra & Sir Charles Mackerras"
[224] "Barry Wordsworth & BBC Concert Orchestra"
[225] "Herbert Von Karajan, Mirella Freni & Wiener Philharmoniker"
[226] "Eugene Ormandy"
[227] "Luciano Pavarotti"
[228] "Leonard Bernstein & New York Philharmonic"
[229] "Boston Symphony Orchestra & Seiji Ozawa"
[230] "Aaron Copland & London Symphony Orchestra"
[231] "Ton Koopman"
[232] "Sergei Prokofiev & Yuri Temirkanov"
[233] "Chicago Symphony Orchestra & Fritz Reiner"
[234] "Orchestra of The Age of Enlightenment"
[235] "Emanuel Ax, Eugene Ormandy & Philadelphia Orchestra"
[236] "James Levine"
[237] "Berliner Philharmoniker & Hans Rosbaud"
[238] "Maurizio Pollini"

```
[239] "Academy of St. Martin in the Fields, Sir Neville Marriner & William Bennett"
[240] "Gustav Mahler"
[241] "Felix Schmidt, London Symphony Orchestra & Rafael Frühbeck de Burgos"
[242] "Edo de Waart & San Francisco Symphony"
[243] "Antal Doráti & London Symphony Orchestra"
[244] "Choir Of Westminster Abbey & Simon Preston"
[245] "Michael Tilson Thomas & San Francisco Symphony"
[246] "Chor der Wiener Staatsoper, Herbert Von Karajan & Wiener Philharmoniker"
[247] "The King's Singers"
[248] "Berliner Philharmoniker & Herbert Von Karajan"
[249] "Sir Georg Solti, Sumi Jo & Wiener Philharmoniker"
[250] "Christopher O'Riley"
[251] "Fretwork"
[252] "Amy Winehouse"
[253] "Calexico"
[254] "Otto Klemperer & Philharmonia Orchestra"
[255] "Yehudi Menuhin"
[256] "Philharmonia Orchestra & Sir Neville Marriner"
[257] "Academy of St. Martin in the Fields, Sir Neville Marriner & Thurston Dart"
[258] "Les Arts Florissants & William Christie"
[259] "The 12 Cellists of The Berlin Philharmonic"
[260] "Adrian Leaper & Doreen de Feis"
[261] "Roger Norrington, London Classical Players"
[262] "Charles Dutoit & L'Orchestre Symphonique de Montréal"
[263] "Equale Brass Ensemble, John Eliot Gardiner & Munich Monteverdi Orchestra and Choir"
[264] "Kent Nagano and Orchestre de l'Opéra de Lyon"
[265] "Julian Bream"
[266] "Martin Roscoe"
[267] "Göteborgs Symfoniker & Neeme Järvi"
[268] "Itzhak Perlman"
[269] "Michele Campanella"
[270] "Gerald Moore"
[271] "Mela Tenenbaum, Pro Musica Prague & Richard Kapp"
[272] "Emerson String Quartet"
[273] "C. Monteverdi, Nigel Rogers - Chiaroscuro; London Baroque; London Cornett & Sackbu"
[274] "Nash Ensemble"
[275] "Philip Glass Ensemble"
>
> #IV.    List out all the countries where the customer resides and plot a bar graph showing the number of customers from the respective country
> unique(customers$Country)
 [1] "Brazil"         "Germany"        "Canada"         "Norway"         "Czech Republic" "Austria"        "Belgium"
 [8] "Denmark"        "USA"            "Portugal"       "France"         "Finland"        "Hungary"        "Ireland"
[15] "Italy"          "Netherlands"    "Poland"         "Spain"          "Sweden"         "United Kingdom" "Australia"
```

```
[22] "Argentina"      "Chile"         "India"
> plot2 <-
+   ggplot() +
+   geom_bar(data = customers, aes(x = Country), width = 0.3, fill = "turquoise") +
+   geom_text(stat='count', data = customers, aes(x = Country, label=..count..), vjust=-0.2) +
+   theme_bw() +
+   xlab("Country") +
+   ylab("Number of Customers") +
+   theme_classic() +
+   theme(axis.text.x=element_text(angle=90, hjust=1)) +
+   ggtitle("Number of Customers by Country") +
+   scale_fill_brewer(type = "qual", palette = 1, direction = 1,
+                     aesthetics = "fill")
> plot2
```

Day 6 – R Programming

Happiness ~ Income (Simple Linear Regression)

```
> #install.packages("broom")
> #install.packages("ggpubr")
> library(ggplot2)
> library(dplyr)
> library(broom)
> library(ggpubr)
> setwd("C:/zubeda/PGA02_Zubu/R Programming/Models")
> dev.off()
null device
          1
>
> #Importing Data
> income.data <- read.csv("income.data.csv")
> income.data
      X   income happiness
1     1 3.862647 2.3144890
2     2 4.979381 3.4334898
3     3 4.923957 4.5993734
4     4 3.214372 2.7911138
5     5 7.196409 5.5963983
6     6 3.729643 2.4585559
7     7 4.674517 3.1929918
8     8 4.498104 1.9071368
9     9 3.121631 2.9424499
10   10 4.639914 3.7379416
11   11 4.632840 3.1754061
12   12 2.773179 2.0090465
```

13	13	7.119479	5.9518141
14	14	7.466653	5.9605473
15	15	2.117742	1.4457989
16	16	2.559166	2.8985831
17	17	2.354793	1.2311675
18	18	2.388157	2.3129881
19	19	4.755680	2.6661160
20	20	1.994275	2.5847290
21	21	7.310916	5.7474441
22	22	3.528319	2.5465246
23	23	2.428752	1.2007855
24	24	3.542748	3.0782934
25	25	5.227201	4.3177609
26	26	6.691993	5.3814787
27	27	3.900410	3.5652243
28	28	2.291055	0.9534130
29	29	2.380513	2.1691613
30	30	2.549609	2.0607943
31	31	6.933296	6.2991013
32	32	1.855645	1.5903559
33	33	3.589023	2.2509294
34	34	6.826478	5.9142477
35	35	2.070602	2.1918337
36	36	5.224205	5.7678144
37	37	2.243114	0.9728829
38	38	7.076166	5.0105774
39	39	4.190672	2.2396650
40	40	1.956486	1.9275788
41	41	5.061758	3.3580716
42	42	3.982190	2.4000873
43	43	3.065059	3.4079800
44	44	3.682877	2.5761763
45	45	3.789429	2.4730794
46	46	5.358716	3.7526595
47	47	5.196120	4.0876312
48	48	5.241190	3.5432037
49	49	7.101620	5.3483529
50	50	3.424021	3.0563767
51	51	2.253399	1.5584226
52	52	5.370337	3.2251328
53	53	6.225606	5.0342310
54	54	5.482862	3.8574243
55	55	4.034172	3.6190555
56	56	6.510219	4.0045377

57	57	6.029214	4.8020918
58	58	6.949113	4.6588904
59	59	7.195037	5.2317030
60	60	2.757338	2.4806065
61	61	6.956079	5.4981472
62	62	4.670193	4.5506370
63	63	6.368293	3.5700136
64	64	6.166681	4.7196653
65	65	6.074158	4.5031082
66	66	5.484719	5.0460818
67	67	1.589575	0.6697159
68	68	1.680474	1.6060724
69	69	5.499948	4.8266027
70	70	4.043891	2.2082405
71	71	5.005093	4.0564931
72	72	4.863582	3.5679052
73	73	1.506275	1.3084873
74	74	2.864664	4.1596093
75	75	5.877906	4.6339151
76	76	6.483984	5.0687479
77	77	4.938037	3.0407973
78	78	5.625434	3.8042989
79	79	7.228265	5.0340038
80	80	5.337460	3.7034379
81	81	2.825827	2.1889381
82	82	5.931367	5.5380475
83	83	3.520255	3.5838752
84	84	3.239941	3.0968856
85	85	3.498386	2.2009822
86	86	7.186112	5.1515983
87	87	4.719166	5.9509863
88	88	3.594802	2.9681871
89	89	3.233942	2.3995613
90	90	1.514153	0.8594991
91	91	4.002537	1.7759326
92	92	6.198104	4.6612612
93	93	2.280651	0.7272212
94	94	2.189866	0.7712866
95	95	3.434151	3.3487882
96	96	5.932270	3.9662154
97	97	5.307839	2.8904474
98	98	5.664345	3.7732607
99	99	7.439248	6.3596000
100	100	2.134702	0.2687221

101	101	6.501275	4.3748323
102	102	3.651183	2.1558433
103	103	2.286495	1.8935569
104	104	4.748859	4.9029916
105	105	5.459161	4.8335064
106	106	3.433065	3.1722995
107	107	7.176400	5.0299517
108	108	5.506395	4.2610130
109	109	3.097616	1.6723906
110	110	4.647556	1.4970241
111	111	1.828306	1.2654889
112	112	3.534566	2.6674654
113	113	4.606176	1.9993255
114	114	5.361503	5.2318633
115	115	6.879333	5.2114013
116	116	4.317032	3.6616565
117	117	3.383164	1.4150347
118	118	4.932207	4.9330441
119	119	4.935597	4.1307783
120	120	2.601553	2.2822669
121	121	5.711264	3.9011703
122	122	6.117531	4.6919989
123	123	3.771415	3.5778007
124	124	7.117220	5.5625455
125	125	2.194882	2.3932281
126	126	5.952002	3.5647237
127	127	3.922303	2.2537215
128	128	7.081589	4.1216477
129	129	6.950745	4.1691008
130	130	3.660877	3.8238987
131	131	1.789092	0.4583776
132	132	3.540341	2.5769400
133	133	4.533395	2.9475315
134	134	4.867339	3.7399958
135	135	4.056005	3.5714465
136	136	5.634643	4.8081504
137	137	5.461636	4.0176112
138	138	3.186176	1.8398020
139	139	4.417666	3.4685738
140	140	5.760289	4.7587855
141	141	3.716700	2.3916775
142	142	2.182562	0.9929174
143	143	4.291984	3.1693802
144	144	3.410030	2.0890424

145	145	3.581097	1.8436758
146	146	3.509663	1.6166074
147	147	6.660216	5.9493475
148	148	6.271786	4.9402278
149	149	3.735018	2.8412387
150	150	4.393208	2.9443913
151	151	3.512217	3.0269182
152	152	6.239740	5.0978025
153	153	3.681486	3.0368199
154	154	7.241313	4.6828170
155	155	6.345370	4.0008192
156	156	5.939742	4.5708011
157	157	2.459321	2.0427393
158	158	2.539089	1.7486511
159	159	6.708604	6.0640130
160	160	6.831322	5.1324015
161	161	5.082658	3.1133421
162	162	6.030607	4.8218103
163	163	6.574595	4.1795210
164	164	3.574297	1.6312611
165	165	5.529908	3.8221796
166	166	2.409382	1.8575419
167	167	4.264790	3.7510893
168	168	3.530345	3.1586186
169	169	6.143150	4.9271326
170	170	5.157697	4.6001148
171	171	4.710847	2.4083799
172	172	6.847515	4.4867037
173	173	5.464640	2.7277157
174	174	4.176532	3.0207244
175	175	3.748093	3.7491807
176	176	2.274523	2.3115542
177	177	1.576366	0.9876032
178	178	1.924134	1.4611444
179	179	5.904246	4.5768565
180	180	5.189031	4.7675956
181	181	1.879868	0.4381716
182	182	2.544348	2.3512025
183	183	3.221394	3.4531027
184	184	7.260374	5.3813985
185	185	6.481617	4.8200912
186	186	5.688488	4.6587427
187	187	6.633619	5.3800702
188	188	5.972741	3.3159895

189	189	3.897738	2.7997475
190	190	6.461243	4.2067549
191	191	6.628036	4.4026632
192	192	3.118959	2.7691181
193	193	4.695964	2.7842445
194	194	1.573694	0.6880906
195	195	3.670377	3.4764991
196	196	7.194407	5.8361967
197	197	1.780479	2.0039261
198	198	2.142360	0.9713325
199	199	3.656486	2.8576144
200	200	2.090354	1.8521177
201	201	3.363097	3.5151792
202	202	2.423144	2.1005525
203	203	7.111584	6.0864783
204	204	3.039942	4.0838212
205	205	2.373232	1.5247823
206	206	1.984564	2.5795165
207	207	2.628483	1.6192261
208	208	7.136760	5.5066215
209	209	3.104918	1.0959993
210	210	1.558657	0.9685273
211	211	7.478447	4.8777255
212	212	2.813900	1.7596987
213	213	5.744540	5.0322110
214	214	6.540988	5.7138023
215	215	6.562794	4.7958429
216	216	5.470125	4.6602900
217	217	2.085131	2.7403954
218	218	4.589572	4.2505681
219	219	5.074502	3.9410193
220	220	7.463510	4.5034454
221	221	5.853906	4.6332287
222	222	3.764540	4.0308649
223	223	7.062792	6.8633880
224	224	6.377376	5.0794759
225	225	1.920863	1.6252684
226	226	7.364213	6.6182798
227	227	6.535799	4.6569986
228	228	7.300903	6.0049875
229	229	3.037232	2.4106356
230	230	6.703267	4.2612200
231	231	1.927997	1.0656238
232	232	4.223554	2.2957001

233	233	2.922706	1.9195111
234	234	4.427109	3.5813810
235	235	2.070562	0.6289421
236	236	5.224070	3.3441084
237	237	7.161873	6.1604341
238	238	2.210696	2.8612744
239	239	7.207060	4.2094706
240	240	2.184085	1.2262981
241	241	4.414998	4.7933252
242	242	5.014810	3.3137965
243	243	2.602037	2.1526984
244	244	2.917049	2.9236839
245	245	6.244342	3.4520672
246	246	6.859654	4.2583115
247	247	2.371230	3.9610328
248	248	5.964058	3.4277232
249	249	7.153674	4.7209873
250	250	2.289897	1.5719498
251	251	6.376228	6.1869456
252	252	3.540504	3.5527370
253	253	3.139826	1.2457859
254	254	6.460742	3.4269117
255	255	2.641348	1.8771470
256	256	2.002214	1.8178891
257	257	6.391428	5.2399095
258	258	2.763720	1.3840441
259	259	6.831258	3.8379778
260	260	3.827255	2.3216977
261	261	3.770671	3.3115782
262	262	3.159855	3.0573878
263	263	5.099417	4.7370460
264	264	5.610391	4.5909705
265	265	1.856372	0.6542421
266	266	5.363730	3.9811277
267	267	2.336134	2.3362760
268	268	4.975851	4.5406361
269	269	2.629547	2.1927449
270	270	2.646458	1.7881763
271	271	3.859892	2.7194322
272	272	4.121531	4.2126973
273	273	6.385941	4.4772714
274	274	3.842710	2.4688600
275	275	4.990549	3.4517725
276	276	3.400597	0.6869208

277	277	3.820115	2.2467457
278	278	1.909499	1.6276951
279	279	2.858464	2.4532091
280	280	7.451501	4.1099447
281	281	3.354252	2.5725982
282	282	6.707825	4.8104241
283	283	6.325906	4.8274163
284	284	1.931181	2.4657611
285	285	5.128545	3.7573790
286	286	4.278210	3.6913482
287	287	1.579470	1.9818099
288	288	2.907682	1.4102296
289	289	5.644714	3.7543014
290	290	3.571175	4.2452156
291	291	2.345108	1.4378767
292	292	5.845197	4.1598871
293	293	5.298480	3.5148918
294	294	3.434700	2.7144980
295	295	1.865995	1.3573731
296	296	5.095615	5.6384604
297	297	1.530808	2.4214647
298	298	1.618311	1.5112051
299	299	6.687677	5.5228917
300	300	7.347246	4.3966223
301	301	5.983846	3.9884528
302	302	5.196082	3.3176068
303	303	4.193007	2.7796826
304	304	2.347688	0.8465055
305	305	4.709489	2.4579217
306	306	2.307198	3.5096047
307	307	2.730769	2.6469045
308	308	3.882514	2.9779167
309	309	3.574394	1.4244505
310	310	4.159576	1.7903423
311	311	1.544634	1.6446366
312	312	3.383897	0.7316363
313	313	3.614745	2.9017886
314	314	6.503882	4.3171332
315	315	1.848413	2.0973691
316	316	4.420312	4.3915930
317	317	6.477153	5.3139569
318	318	6.561420	6.2813702
319	319	7.180907	5.6079666
320	320	2.809091	2.9468704

```
321 321 5.686205 3.8727888
322 322 4.800344 2.8428684
323 323 2.412912 1.4606163
324 324 2.925704 3.7528248
325 325 3.174176 3.1266032
326 326 2.685530 2.8211926
327 327 2.124429 2.7349601
328 328 2.694022 2.1975921
329 329 4.230889 4.1555409
330 330 5.350516 4.0782088
331 331 5.091580 4.4569636
332 332 6.250302 4.9392590
333 333 5.324633 3.7305700
 [ reached 'max' / getOption("max.print") -- omitted 165 rows ]
> dim(income.data)
[1] 498   3
> summary(income.data)
       X             income        happiness    
 Min.   :  1.0   Min.   :1.506   Min.   :0.266  
 1st Qu.:125.2   1st Qu.:3.006   1st Qu.:2.266  
 Median :249.5   Median :4.424   Median :3.473  
 Mean   :249.5   Mean   :4.467   Mean   :3.393  
 3rd Qu.:373.8   3rd Qu.:5.992   3rd Qu.:4.503  
 Max.   :498.0   Max.   :7.482   Max.   :6.863  
> 
> #Assumptions
> hist(income.data$happiness)   #normally distributed
> plot(happiness ~ income, data=income.data)   #linearity x ~ y
> #Homoscedasticity or homogeneity of variance will be checked after model building
> 
> #Linear Regression Analysis
> income.happiness.lm <- lm(happiness ~ income, data=income.data)
> summary(income.happiness.lm)

Call:
lm(formula = happiness ~ income, data = income.data)

Residuals:
     Min       1Q   Median       3Q      Max 
-2.02479 -0.48526  0.04078  0.45898  2.37805 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept)  0.20427    0.08884   2.299   0.0219 *  
```

```
income       0.71383    0.01854  38.505   <2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.7181 on 496 degrees of freedom
Multiple R-squared:  0.7493,	Adjusted R-squared:  0.7488
F-statistic:  1483 on 1 and 496 DF,  p-value: < 2.2e-16

> par(mfrow=c(2, 2))
> plot(income.happiness.lm)   #Homoscedasticity, Residuals normally distributed
> par(mfrow=c(1, 1))
>
> #Visualize results
> income.graph <- ggplot(income.data, aes(x=income, y=happiness)) + geom_point()
> income.graph
> income.graph <- income.graph + geom_smooth(method = "lm", col="black")
> income.graph
`geom_smooth()` using formula 'y ~ x'
> income.graph <- income.graph + stat_regline_equation(label.x=3, label.y=7)  #regression line eq. y = mx + c
> income.graph
`geom_smooth()` using formula 'y ~ x'
> income.graph + theme_bw() +
+   labs(title="Reported Happiness as a function of Income", x="Income(x$10,000)", y="Happiness(1 to 10)")
`geom_smooth()` using formula 'y ~ x'
```

Cars Distance ~ Speed (Simple Linear Regression)

```
> cars
   speed dist
1      4    2
2      4   10
3      7    4
4      7   22
5      8   16
6      9   10
7     10   18
8     10   26
9     10   34
10    11   17
11    11   28
12    12   14
13    12   20
14    12   24
```

```
15   12   28
16   13   26
17   13   34
18   13   34
19   13   46
20   14   26
21   14   36
22   14   60
23   14   80
24   15   20
25   15   26
26   15   54
27   16   32
28   16   40
29   17   32
30   17   40
31   17   50
32   18   42
33   18   56
34   18   76
35   18   84
36   19   36
37   19   46
38   19   68
39   20   32
40   20   48
41   20   52
42   20   56
43   20   64
44   22   66
45   23   54
46   24   70
47   24   92
48   24   93
49   24  120
50   25   85
> ?cars
> summary(cars)
     speed           dist       
 Min.   : 4.0   Min.   :  2.00  
 1st Qu.:12.0   1st Qu.: 26.00  
 Median :15.0   Median : 36.00  
 Mean   :15.4   Mean   : 42.98  
 3rd Qu.:19.0   3rd Qu.: 56.00  
```

```
 Max.   :25.0   Max.   :120.00
> plot(cars, col="blue", pch=20, cex=2, main="Relationship between Speed and Stopping Distance for 10 Cars", xlab="Speed in mph", ylab="Stopping Distance in feet")
> set.seed(1)   #generates random numbers, gives same set of numbers (Set seed every time if we need same number)
> sample(3)
[1] 1 2 3
>
> mt <- matrix(1:10, ncol = 5)
> mt
     [,1] [,2] [,3] [,4] [,5]
[1,]    1    3    5    7    9
[2,]    2    4    6    8   10
> scale(mt, center=TRUE, scale=FALSE)
     [,1] [,2] [,3] [,4] [,5]
[1,] -0.5 -0.5 -0.5 -0.5 -0.5
[2,]  0.5  0.5  0.5  0.5  0.5
attr(,"scaled:center")
[1] 1.5 3.5 5.5 7.5 9.5
>
> set.seed(2)   #Works like random_state from python
> speed.c <- scale(cars$speed, center=TRUE, scale=FALSE)
> mod1 <- lm(formula=dist ~ speed.c, data=cars)
> mod1

Call:
lm(formula = dist ~ speed.c, data = cars)

Coefficients:
(Intercept)      speed.c
     42.980        3.932

> summary(mod1)

Call:
lm(formula = dist ~ speed.c, data = cars)

Residuals:
    Min      1Q  Median      3Q     Max
-29.069  -9.525  -2.272   9.215  43.201

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  42.9800     2.1750  19.761  < 2e-16 ***
```

```
speed.c      3.9324     0.4155   9.464 1.49e-12 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 15.38 on 48 degrees of freedom
Multiple R-squared:  0.6511,      Adjusted R-squared:  0.6438
F-statistic: 89.57 on 1 and 48 DF,  p-value: 1.49e-12
```

Heart Disease Prediction (Simple Linear Regression)

```
> library(ggplot2)
> library(dplyr)
> library(broom)
> library(ggpubr)
> setwd("C:/zubeda/PGA02_Zubu/R Programming/Models")
> dev.off()
null device
          1
>
> #Importing Data
> heart.data <- read.csv("heart.data.csv")
> heart.data
     X    biking    smoking heart.disease
1    1 30.801246 10.8966080    11.7694228
2    2 65.129215  2.2195632     2.8540815
3    3  1.959665 17.5883305    17.1778035
4    4 44.800196  2.8025589     6.8166469
5    5 69.428454 15.9745046     4.0622235
6    6 54.403626 29.3331755     9.5500460
7    7 49.056162  9.0608458     7.6245070
8    8  4.784604 12.8350208    15.8546544
9    9 65.730788 11.9912973     3.0674617
10  10 35.257449 23.2776834    12.0984844
11  11 51.825567 14.4351184     6.4302482
12  12 52.936197 25.0748686     8.6082721
13  13 48.767478 11.0232710     6.7225238
14  14 26.166801  6.6457495    10.5978071
15  15 10.553075  5.9905063    14.0794783
16  16 47.163716 14.0978372     8.7448453
17  17 61.685256 16.8408167     5.4433420
18  18 33.944394  5.7585952     9.1623064
19  19 39.697624 12.6628694     9.7471858
20  20 63.124698 22.9174800     5.8582779
21  21 28.510129 14.8551064    11.7247416
22  22 18.525973 26.4049774    16.0281877
```

23	23	24.479470	26.9249607	15.0007154
24	24	18.358646	23.4319568	16.4882059
25	25	30.388184	16.9860864	12.3566075
26	26	52.985220	27.6890270	9.0884449
27	27	60.509448	3.9819621	3.2172143
28	28	45.247110	2.1374753	6.5937191
29	29	48.597044	10.3884264	6.6594202
30	30	25.139771	5.8363728	11.4829371
31	31	44.173095	3.9676057	7.9822275
32	32	61.146946	27.7834060	7.7224625
33	33	27.267898	16.8532932	12.1648166
34	34	49.527100	15.2497308	8.0274043
35	35	20.197206	10.3314895	13.4165474
36	36	18.811228	16.7534420	15.0873602
37	37	67.350765	23.8737268	5.2798634
38	38	29.904475	24.5845499	13.5076192
39	39	14.011760	21.0121418	15.9929864
40	40	45.815488	4.7161269	7.0406211
41	41	31.477251	22.1658255	13.2385262
42	42	17.108204	1.3528783	11.5357354
43	43	9.665082	3.5042963	12.8868168
44	44	23.933005	4.1858692	10.8944571
45	45	22.636301	13.6789828	13.4660327
46	46	27.247477	13.3797768	12.3288989
47	47	20.789602	19.4554286	14.4157831
48	48	46.613715	9.2493326	6.8093546
49	49	28.622632	12.4827339	11.3683650
50	50	21.127498	18.9413483	14.8253623
51	51	68.574349	1.8047036	0.6839264
52	52	41.684367	13.0672050	9.0888493
53	53	69.879593	17.3516599	4.2564834
54	54	9.817277	23.8189949	17.8341328
55	55	4.379280	20.6629714	17.4118109
56	56	28.610378	1.1479621	10.0558055
57	57	21.460016	22.9760018	15.0086341
58	58	27.601656	4.3883804	10.8936852
59	59	57.230504	12.8050000	5.3068180
60	60	26.397282	7.7639305	11.0243252
61	61	39.010480	0.7676801	8.7639297
62	62	11.527487	6.7396220	14.6003016
63	63	17.684287	8.2780091	13.8021255
64	64	19.935253	6.0764383	11.6320723
65	65	42.310040	8.4123966	7.4630017
66	66	1.119154	19.5503583	17.7101910

67	67	23.276821	14.3066349	13.2663850
68	68	14.965816	12.8691532	15.3003824
69	69	30.663350	16.6632038	11.7531562
70	70	22.925183	24.5987873	16.1991180
71	71	59.770308	8.7007692	5.6856819
72	72	70.456061	12.7407109	2.9059872
73	73	21.750385	18.8575107	14.1798442
74	74	49.360686	23.2094802	9.4660260
75	75	4.487242	23.4190000	18.7669334
76	76	14.693269	17.8743936	16.3479313
77	77	40.611628	25.9937106	12.0466432
78	78	8.764197	3.8990404	13.3730055
79	79	56.725412	16.1809774	6.7357709
80	80	60.551149	18.0652019	6.5167036
81	81	64.384893	10.5295309	4.5668945
82	82	20.262798	11.1787830	12.3790734
83	83	30.520086	12.4334774	11.1407355
84	84	30.461542	29.8608106	14.3299759
85	85	3.705894	21.4941886	17.8776920
86	86	15.082469	16.6152001	14.9359167
87	87	30.997842	29.0164264	13.7579867
88	88	14.625411	6.7983462	14.4789601
89	89	28.103061	14.7313505	12.7945548
90	90	34.680241	8.6381050	8.3317181
91	91	6.947463	26.1056583	18.6897979
92	92	26.860662	16.7194132	13.4876889
93	93	41.019323	12.9991873	10.2648899
94	94	31.932738	28.1828730	13.1737410
95	95	69.877147	26.3311967	6.0724855
96	96	63.029854	22.2471319	6.3567192
97	97	21.349999	12.8558535	13.7996830
98	98	3.811338	17.7237882	18.4623513
99	99	67.514106	26.9553999	6.5826851
100	100	12.580893	16.9897825	14.8556848
101	101	61.232197	4.6135505	4.3616062
102	102	64.332304	29.4237378	7.7073976
103	103	31.872439	16.2418978	10.5260293
104	104	11.936562	3.4865672	14.3914484
105	105	52.360268	19.8410769	7.8163886
106	106	22.516638	7.6951729	11.6647648
107	107	49.764822	3.5729441	6.2645744
108	108	22.792636	9.8168286	10.9117655
109	109	49.748748	20.4918307	8.7001979
110	110	68.204122	2.1929445	1.9664482

111	111	15.185101	14.5203621	14.5878253
112	112	58.046901	15.7135850	6.5593358
113	113	69.499688	23.1678748	5.7536536
114	114	2.616135	4.3190804	14.8517766
115	115	2.136343	25.8401303	19.2426878
116	116	25.771571	28.5403473	14.4293484
117	117	11.615646	6.3455289	14.4114830
118	118	17.197456	20.7334559	15.1981843
119	119	27.681662	18.6370085	11.5075951
120	120	9.648096	8.3700383	15.5763683
121	121	65.621956	14.1066235	4.0442469
122	122	46.556228	26.8937498	10.5427902
123	123	48.300472	12.0117870	7.2992774
124	124	15.141292	7.8937497	13.6191300
125	125	71.579351	18.9378855	4.0907648
126	126	4.681250	10.9882819	15.8685828
127	127	22.476723	25.9723233	14.8303440
128	128	49.296201	5.0887881	6.0737923
129	129	17.648135	27.1925636	15.1389680
130	130	9.413778	8.6085378	13.5250712
131	131	64.915668	10.8893906	3.7743880
132	132	3.950367	6.3595796	14.6357693
133	133	67.342457	16.6226193	3.7444984
134	134	71.238955	25.1941518	6.0015227
135	135	70.323878	26.2334743	5.2357312
136	136	28.424901	20.2084486	13.0038423
137	137	73.713732	14.1016522	2.9890677
138	138	56.058032	8.2061159	5.5239310
139	139	21.588199	2.5734949	11.3742898
140	140	16.276161	3.4634491	13.5987548
141	141	26.988690	15.8833260	12.9216726
142	142	11.326814	9.2306549	14.5243489
143	143	55.580584	7.6713074	5.7268270
144	144	50.603802	28.9184712	10.6355905
145	145	60.401739	24.8321410	7.5726142
146	146	71.486751	21.6373684	3.6684273
147	147	37.978507	13.9453443	10.1067938
148	148	48.692115	23.5640881	9.5990941
149	149	40.016400	7.8116349	7.6581870
150	150	32.148553	0.9690843	8.8739847
151	151	12.318283	26.8908059	17.7585948
152	152	31.659667	21.3995087	12.0931244
153	153	55.841893	2.2487162	4.5002273
154	154	28.826953	11.4098666	12.8126371

155	155	55.472287	5.2674378	4.9163834
156	156	54.354034	16.3415202	7.5917973
157	157	70.331699	15.0588637	3.0374318
158	158	54.062609	20.3888136	8.0511951
159	159	59.575645	24.7926381	8.5199279
160	160	2.818204	23.2461405	18.4829598
161	161	30.460335	2.5256544	8.9365829
162	162	22.343450	23.2046425	15.9085493
163	163	14.696886	9.8316822	13.2582035
164	164	70.902815	29.9140032	6.3350215
165	165	35.335113	9.1475020	9.7518942
166	166	72.173766	15.2736628	2.8283567
167	167	44.698217	10.0025889	7.4494126
168	168	70.361366	20.3399150	3.8971099
169	169	7.619084	26.6615229	18.6881304
170	170	29.673634	2.3967956	9.3879078
171	171	28.485683	12.6628036	11.2860570
172	172	67.423291	28.6574311	5.8094945
173	173	10.145069	11.3175197	14.8048065
174	174	59.989904	25.4558391	7.4743720
175	175	38.155015	20.1042221	11.5622808
176	176	15.466010	11.5711484	13.3774300
177	177	73.767713	16.1513316	2.3548085
178	178	31.179629	5.5684413	9.8617288
179	179	5.201611	4.3599032	15.6431142
180	180	50.249614	4.1290591	5.5419066
181	181	60.940141	21.8644959	5.4159174
182	182	20.068674	11.9294173	12.6884954
183	183	41.211215	4.1514402	7.5901660
184	184	72.394856	7.5198372	1.8701100
185	185	10.610969	19.3015155	16.7460356
186	186	45.579836	20.6168515	9.7984834
187	187	29.658506	12.1518990	12.8178071
188	188	40.056854	16.5064944	9.3926393
189	189	5.510300	17.8842193	16.0139208
190	190	32.056529	12.4794809	11.5360652
191	191	46.842870	27.3216486	10.2505847
192	192	42.425007	10.9547393	8.8282361
193	193	31.212374	7.7973828	9.7753859
194	194	13.176628	9.9874669	14.5477545
195	195	33.779739	0.9653903	6.9442975
196	196	70.690083	22.7107707	2.7084606
197	197	60.284951	15.1081402	4.9479908
198	198	16.003605	19.8941489	15.3662877

199	199	39.677219	10.2721672	9.5436557
200	200	12.885185	25.2101825	16.2725863
201	201	35.023450	22.6640373	12.5158362
202	202	10.343753	27.6468493	17.4485160
203	203	20.640893	15.3841384	14.5572879
204	204	63.238037	20.5047412	5.5609216
205	205	23.984565	7.6121169	11.5562573
206	206	44.014897	6.5796621	8.5037463
207	207	67.127924	5.8203635	2.5511506
208	208	36.538050	2.0528831	8.4950339
209	209	7.831481	26.8269709	17.5604514
210	210	40.395401	7.0274602	6.7390807
211	211	16.249914	28.3617369	17.3545771
212	212	47.584661	29.4683524	11.5999032
213	213	15.481362	18.8152012	14.9407560
214	214	70.085196	10.3826455	2.2392169
215	215	1.330485	28.7937440	20.4534962
216	216	61.542692	12.8374854	4.9734613
217	217	23.097771	0.9073611	10.6947889
218	218	65.069917	1.4876797	2.1856508
219	219	71.014542	15.2819055	4.0768235
220	220	64.065982	28.4724464	7.2862524
221	221	61.152760	2.1900536	3.5229763
222	222	22.672368	28.7353995	16.7786346
223	223	49.728740	3.4945424	5.0730746
224	224	35.480794	15.8501562	11.3729972
225	225	59.461885	18.8075341	5.6906489
226	226	31.697593	23.1906637	12.5378281
227	227	62.772108	15.2398175	4.9345783
228	228	58.668542	19.5857407	6.6620753
229	229	25.254140	7.1689519	12.0465147
230	230	22.722701	8.2882507	11.2963162
231	231	1.616922	1.4584368	16.3351186
232	232	10.353648	16.0131718	14.0886540
233	233	44.721586	7.2730788	7.6450248
234	234	29.224098	27.3570273	13.6443726
235	235	66.111593	18.6293941	4.5900481
236	236	46.728488	8.4995461	7.8255359
237	237	39.172427	17.5563764	10.7150869
238	238	9.937617	22.3426031	16.3846780
239	239	61.393845	22.9195492	6.8381847
240	240	21.608228	9.3488357	13.9959895
241	241	6.283610	20.1753293	17.5981661
242	242	61.436136	24.4659138	7.6206629

```
243 243  1.257841 15.1309519    16.7368633
244 244  4.699863  9.8599366    15.5694794
245 245 54.938223 13.0462165     6.2944988
246 246  7.749030  8.3870192    13.2815287
247 247 49.563424 16.2590213     8.1434851
248 248 29.731403 23.3339016    12.9964758
249 249 45.716267 23.8302604     9.2300743
250 250 47.537321 20.0742953     8.9378932
 [ reached 'max' / getOption("max.print") -- omitted 248 rows ]
> dim(heart.data)
[1] 498   4
> summary(heart.data)
       X             biking          smoking        heart.disease
 Min.   :  1.0   Min.   : 1.119   Min.   : 0.5259   Min.   : 0.5519
 1st Qu.:125.2   1st Qu.:20.205   1st Qu.: 8.2798   1st Qu.: 6.5137
 Median :249.5   Median :35.824   Median :15.8146   Median :10.3853
 Mean   :249.5   Mean   :37.788   Mean   :15.4350   Mean   :10.1745
 3rd Qu.:373.8   3rd Qu.:57.853   3rd Qu.:22.5689   3rd Qu.:13.7240
 Max.   :498.0   Max.   :74.907   Max.   :29.9467   Max.   :20.4535
>
> #Assumptions
> cor(heart.data$biking, heart.data$smoking)   #independent predictors
[1] 0.01513618
> hist(heart.data$heart.disease)    #normally distributed
> plot(heart.disease ~ biking, data=heart.data)     #linearity x ~ y
> plot(heart.disease ~ smoking, data=heart.data)    #linearity x ~ y
> #Homoscedasticity or homogeneity of variance will be checked after model building
>
> #Linear Regression Analysis
> heart.disease.lm <- lm(heart.disease ~ biking+smoking, data=heart.data)
> summary(heart.disease.lm)

Call:
lm(formula = heart.disease ~ biking + smoking, data = heart.data)

Residuals:
    Min      1Q  Median      3Q     Max
-2.1789 -0.4463  0.0362  0.4422  1.9331

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) 14.984658   0.080137  186.99   <2e-16 ***
biking      -0.200133   0.001366 -146.53   <2e-16 ***
smoking      0.178334   0.003539   50.39   <2e-16 ***
```

```
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.654 on 495 degrees of freedom
Multiple R-squared:  0.9796,	Adjusted R-squared:  0.9795 
F-statistic: 1.19e+04 on 2 and 495 DF,  p-value: < 2.2e-16

> par(mfrow=c(2, 2))
> plot(heart.disease.lm)   #Homoscedasticity, Residuals normally distributed
> par(mfrow=c(1, 1))
> 
> #Visualize results
> #1.	Create a new dataframe with the information needed to plot the model
> plotting.data <- expand.grid(biking=seq(min(heart.data$biking), max(heart.data$biking), 
length.out=30),
+                              smoking=c(min(heart.data$smoking), mean(heart.data$smoking), 
max(heart.data$smoking)))
> #2.	Predict the values of heart disease based on your linear model
> plotting.data$predicted.y <- predict.lm(heart.disease.lm, newdata = plotting.data)
> #3.	Round the smoking numbers to two decimals
> plotting.data$smoking <- round(plotting.data$smoking, digits = 2)
> #4.	Change the 'smoking' variable into a factor
> plotting.data$smoking <- as.factor(plotting.data$smoking)
> #5.	Plot the original data
> heart.plot <- ggplot(heart.data, aes(x=biking, y=heart.disease)) + geom_point()
> heart.plot
> #6.	Add the regression lines
> heart.plot <- heart.plot +
+   geom_line(data=plotting.data, aes(x=biking, y=predicted.y, color=smoking), size=1.25)
> heart.plot
> #7.	Make the graph ready for publication
> heart.plot <-
+   heart.plot +
+   theme_bw() +
+   labs(title = "Rates of heart disease (% of population) \n as a function of biking to work and smoking",
+        x = "Biking to work (% of population)",
+        y = "Heart disease (% of population)",
+        color = "Smoking \n (% of population)")
> heart.plot
> 
> heart.plot + annotate(geom="text", x=30, y=1.75, label=" = 15 + (-0.2*biking) + (0.178*smoking)")
```

Day 7 – R Programming

Class Assessment – Property Price Prediction

```
> # In this case study we build a linear regression model
> # We use the model to predict our test data
> # We check the model performance using
> # RMSE metric
> # We demonstrate tests for autocorrelation & heteroskedasticity
> # We demonstrate VIF to detect multicollinearity
>
> library(ggplot2)
> library(dplyr)
> library(broom)
> library(ggpubr)
> # Set your working directory.
> setwd("C:/zubeda/PGA02_Zubu/R Programming/Models")
>
> #Importing Data
> propertytrainData <- read.csv("PropertyTrainData.csv")
> head(propertytrainData)
  Price Sea   Area Elevation Sewer Days Flood Distance
1   4.5   1  138.4        10  3000 -103     0      0.3
2  10.6   1   52.0         4     0 -103     0      2.5
3   1.7   0   16.1         0  2640  -98     1     10.3
4   5.0   0 1695.2         1  3500  -93     0     14.0
5   5.0   0  845.0         1  1000  -92     1     14.0
6   3.3   1    6.9         2 10000  -86     0      0.0
> dim(propertytrainData)
[1] 31  8
> summary(propertytrainData)
     Price          Sea              Area           Elevation         Sewer           Days             Flood           Distance
 Min.   : 1.70   Min.   :0.0000   Min.   :   6.90   Min.   : 0.000   Min.   :    0   Min.   :-103.00   Min.   :0.0000
Min.   : 0.000
 1st Qu.: 5.35   1st Qu.:0.0000   1st Qu.:  20.35   1st Qu.: 2.000   1st Qu.:    0   1st Qu.: -63.50   1st
Qu.:0.0000   1st Qu.: 0.850
 Median :11.70   Median :1.0000   Median :  51.40   Median : 4.000   Median :  900   Median : -59.00
Median :0.0000   Median : 4.900
 Mean   :11.95   Mean   :0.6129   Mean   : 139.97   Mean   : 4.645   Mean   : 1981   Mean   : -58.65   Mean
:0.1613   Mean   : 5.132
 3rd Qu.:16.05   3rd Qu.:1.0000   3rd Qu.: 104.10   3rd Qu.: 7.000   3rd Qu.: 3450   3rd Qu.: -51.00   3rd
Qu.:0.0000   3rd Qu.: 5.500
 Max.   :37.20   Max.   :1.0000   Max.   :1695.20   Max.   :20.000   Max.   :10000   Max.   :  -4.00   Max.
:1.0000   Max.   :16.500
> propertytestData <- read.csv("PropertyTestData.csv")
> head(propertytestData)
```

```
  Price Sea Area Elevation Sewer Days Flood Distance
1    12   1 1472       20  4811  -36    1       8
2     5   0 1301        1  4070  -79    0       1
3    12   1   39       17  1200  -40    1       4
4    36   0    7       18  3240  -46    0      12
5     2   0  357        7  5619  -88    1       1
6     9   1  686       12  5056  -39    1      12
> dim(propertytestData)
[1] 31  8
> summary(propertytestData)
    Price           Sea             Area          Elevation         Sewer           Days            Flood           Distance
 Min.   : 2.00   Min.   :0.0000   Min.   :   7.0   Min.   : 1.00   Min.   :   0   Min.   :-96.00   Min.   :0.0000   Min.
: 0.000
 1st Qu.: 6.50   1st Qu.:0.0000   1st Qu.: 325.5   1st Qu.: 7.00   1st Qu.:1142   1st Qu.:-77.00   1st
Qu.:0.0000   1st Qu.: 2.500
 Median : 9.00   Median :0.0000   Median : 657.0   Median :12.00   Median :2814   Median :-49.00
Median :1.0000   Median : 6.000
 Mean   :11.76   Mean   :0.4839   Mean   : 733.7   Mean   :11.32   Mean   :2819   Mean   :-55.39   Mean
:0.6129   Mean   : 6.145
 3rd Qu.:16.90   3rd Qu.:1.0000   3rd Qu.:1166.5   3rd Qu.:16.50   3rd Qu.:4273   3rd Qu.:-39.50   3rd
Qu.:1.0000   3rd Qu.: 9.500
 Max.   :36.00   Max.   :1.0000   Max.   :1556.0   Max.   :20.00   Max.   :5775   Max.   : -7.00   Max.   :1.0000
Max.   :12.000
>
> #EDA ~ Assumptions
> #Check for normality of dependent variable
> hist(propertytrainData$Price)
> shapiro.test(propertytrainData$Price)   #Price is not normality distributed

	Shapiro-Wilk normality test

data:  propertytrainData$Price
W = 0.90607, p-value = 0.01025

> logPrice <- log(propertytrainData$Price)
> hist(logPrice)
> shapiro.test(logPrice)   #Price is now normality distributed (p-value > 0.05)

	Shapiro-Wilk normality test

data:  logPrice
W = 0.95854, p-value = 0.2668

> propertytrainData$Logprice <- logPrice
```

```
> dim(propertytrainData)
[1] 31  9
> names(propertytrainData)
[1] "Price"     "Sea"      "Area"      "Elevation" "Sewer"     "Days"      "Flood"     "Distance"  "Logprice"
> summary(propertytrainData)
    Price           Sea             Area           Elevation         Sewer           Days            Flood           Distance
Logprice
 Min.   : 1.70   Min.   :0.0000   Min.   :    6.90   Min.   : 0.000   Min.   :     0   Min.   :-103.00   Min.   :0.0000
Min.   : 0.000   Min.   :0.5306
 1st Qu.: 5.35   1st Qu.:0.0000   1st Qu.:  20.35   1st Qu.: 2.000   1st Qu.:     0   1st Qu.: -63.50   1st
Qu.:0.0000   1st Qu.: 0.850   1st Qu.:1.6750
 Median :11.70   Median :1.0000   Median :  51.40   Median : 4.000   Median :  900   Median : -59.00
Median :0.0000   Median : 4.900   Median :2.4596
 Mean   :11.95   Mean   :0.6129   Mean   : 139.97   Mean   : 4.645   Mean   : 1981   Mean   : -58.65   Mean
:0.1613   Mean   : 5.132   Mean   :2.2594
 3rd Qu.:16.05   3rd Qu.:1.0000   3rd Qu.: 104.10   3rd Qu.: 7.000   3rd Qu.: 3450   3rd Qu.: -51.00   3rd
Qu.:0.0000   3rd Qu.: 5.500   3rd Qu.:2.7746
 Max.   :37.20   Max.   :1.0000   Max.   :1695.20   Max.   :20.000   Max.   :10000   Max.   :  -4.00   Max.
:1.0000   Max.   :16.500   Max.   :3.6163
> #Check Correlation
> cor(propertytrainData[, -1])  #Read all rows, skip 1st column. Use a negative index to skip the column
from the left
                  Sea        Area   Elevation       Sewer        Days       Flood    Distance    Logprice
Sea        1.00000000 -0.33944108  0.47517280 -0.05004423 -0.36983885 -0.55180357 -0.74220440 -
0.04416109
Area      -0.33944108  1.00000000 -0.20945610  0.05338087 -0.34946290  0.10890203  0.55694587 -
0.22024015
Elevation  0.47517280 -0.20945610  1.00000000 -0.35940756 -0.05650853 -0.37308077 -0.36246039
0.43335591
Sewer     -0.05004423  0.05338087 -0.35940756  1.00000000 -0.15149473 -0.11305464 -0.15865389 -
0.46759131
Days      -0.36983885 -0.34946290 -0.05650853 -0.15149473  1.00000000  0.01536084  0.04438251
0.62016026
Flood     -0.55180357  0.10890203 -0.37308077 -0.11305464  0.01536084  1.00000000  0.42330840 -
0.40729809
Distance  -0.74220440  0.55694587 -0.36246039 -0.15865389  0.04438251  0.42330840  1.00000000
0.06587072
Logprice  -0.04416109 -0.22024015  0.43335591 -0.46759131  0.62016026 -0.40729809  0.06587072
1.00000000
> cormat <- round(cor(propertytrainData[, -1]), 3)
> cormat
             Sea   Area Elevation  Sewer   Days  Flood Distance Logprice
Sea        1.000 -0.339     0.475 -0.050 -0.370 -0.552   -0.742   -0.044
Area      -0.339  1.000    -0.209  0.053 -0.349  0.109    0.557   -0.220
```

```
Elevation  0.475 -0.209    1.000 -0.359 -0.057 -0.373   -0.362    0.433
Sewer     -0.050  0.053    -0.359  1.000 -0.151 -0.113   -0.159   -0.468
Days      -0.370 -0.349    -0.057 -0.151  1.000  0.015    0.044    0.620
Flood     -0.552  0.109    -0.373 -0.113  0.015  1.000    0.423   -0.407
Distance  -0.742  0.557    -0.362 -0.159  0.044  0.423    1.000    0.066
Logprice  -0.044 -0.220    0.433 -0.468  0.620 -0.407    0.066    1.000
> write.csv(cormat, "corrmatrix.csv")
>
> #install.packages("GGally")
> library(GGally)
> GGally::ggpairs(propertytrainData[, -1])   #Pairplot - histogram & lineplot
> # Note we see variables which are faily correlated with log(price)
> # are Elevation, Sewer, Date & Flood
> # Note we also see some of the IVs are also correlated with each other
> # like Elevation & Sea, Distance & Sea, Area & Distance and so on
>
> # Another way to better appreciate the relationship
> # between variables is to look at scatter plot
> # Lets plot log(price) and Date
> plot(propertytrainData$Logprice, propertytrainData$Days)
> plot(propertytrainData$Logprice, propertytrainData$Area)
> #Linear Regression Analysis
> reg_model <- lm(Logprice ~ ., data = propertytrainData[, -1])   # . refers to rest all variables
> summary(reg_model)

Call:
lm(formula = Logprice ~ ., data = propertytrainData[, -1])

Residuals:
     Min       1Q   Median       3Q      Max
-0.41605 -0.22833  0.01037  0.22662  0.63418

Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept)  3.099e+00  2.815e-01  11.006 1.22e-10 ***
Sea         -1.596e-01  2.685e-01  -0.594 0.558013
Area        -2.578e-04  2.574e-04  -1.002 0.327001
Elevation    5.053e-02  1.754e-02   2.880 0.008448 **
Sewer       -8.338e-05  3.066e-05  -2.720 0.012214 *
Days         1.479e-02  3.577e-03   4.135 0.000403 ***
Flood       -9.819e-01  2.198e-01  -4.468 0.000175 ***
Distance     4.889e-02  2.496e-02   1.958 0.062407 .
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
Residual standard error: 0.3258 on 23 degrees of freedom
Multiple R-squared:  0.8416,	Adjusted R-squared:  0.7934
F-statistic: 17.46 on 7 and 23 DF,  p-value: 8.112e-08
> #White Spaces - Not significant, . - Poorly Significant(0.05-0.1), * - Average Significance(0.01-0.05), ** - Significant(0.001-0.01), *** - Highly Significant(0-0.001) (in Coefficients)
>
> #New models without Sea, Area
> reg_model1 <- lm(Logprice ~ Area+Days+Distance+Flood+Elevation+Sewer, data = propertytrainData[, -1])
> summary(reg_model1)

Call:
lm(formula = Logprice ~ Area + Days + Distance + Flood + Elevation +
    Sewer, data = propertytrainData[, -1])

Residuals:
     Min       1Q   Median       3Q      Max
-0.37796 -0.22920 -0.01371  0.20334  0.68359

Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept)  3.006e+00  2.310e-01  13.009 2.31e-12 ***
Area        -2.256e-04  2.482e-04  -0.909  0.37240
Days         1.614e-02  2.729e-03   5.914 4.22e-06 ***
Distance     5.826e-02  1.908e-02   3.053  0.00547 **
Flood       -9.154e-01  1.866e-01  -4.906 5.27e-05 ***
Elevation    4.992e-02  1.727e-02   2.890  0.00806 **
Sewer       -7.653e-05  2.802e-05  -2.731  0.01164 *
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.3214 on 24 degrees of freedom
Multiple R-squared:  0.8392,	Adjusted R-squared:  0.799
F-statistic: 20.87 on 6 and 24 DF,  p-value: 1.978e-08

> reg_model2 <- lm(Logprice ~ Days+Distance+Flood+Elevation+Sewer, data = propertytrainData[, -1])
> summary(reg_model2)

Call:
lm(formula = Logprice ~ Days + Distance + Flood + Elevation +
    Sewer, data = propertytrainData[, -1])

Residuals:
```

```
    Min       1Q   Median       3Q      Max 
-0.38511 -0.25256 -0.01794  0.20994  0.72640 

Coefficients:
             Estimate Std. Error t value Pr(>|t|)    
(Intercept)  3.089e+00  2.115e-01  14.603 9.60e-14 ***
Days         1.724e-02  2.435e-03   7.080 2.02e-07 ***
Distance     4.784e-02  1.521e-02   3.147  0.00424 ** 
Flood       -8.835e-01  1.826e-01  -4.838 5.66e-05 ***
Elevation    5.048e-02  1.720e-02   2.934  0.00707 ** 
Sewer       -7.859e-05  2.783e-05  -2.824  0.00919 ** 
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.3203 on 25 degrees of freedom
Multiple R-squared:  0.8336,	Adjusted R-squared:  0.8003 
F-statistic: 25.05 on 5 and 25 DF,  p-value: 5.474e-09

> 
> library(car)
> car::vif(reg_model)   #Lower the value, relevant is the variable
      Sea      Area Elevation     Sewer      Days     Flood  Distance 
 4.995597  2.003925  1.649759  1.635122  2.174889  1.907942  3.623612 
> car::vif(reg_model1)
     Area      Days  Distance     Flood Elevation     Sewer 
 1.915504  1.300847  2.176858  1.412978  1.644029  1.403977 
> car::vif(reg_model2)
     Days  Distance     Flood Elevation     Sewer 
 1.043122  1.391449  1.363245  1.641901  1.394802 
> #property.train1 <- subset(property.train, select=-c(Sea, Distance))
> #head(property.train1)
> 
> par(mfrow=c(2, 2))
> plot(reg_model2)   #Homoscedasticity, Residuals normally distributed
> par(mfrow=c(1, 1))
> # Check residual vs fitted plot to check Heteroscedasticity
> # If there is absolutely no heteroscedasticity, you should
> # see a completely random, equal distribution of points
> # throughout the range of X axis and a flat red line.
> # In our case,
> # the red line is slightly curved and the residuals seem to
> # increase as the fitted Y values increase.
> # So, the inference here is, heteroscedasticity exists.
> # Check the Residuals Vs Fitted Curve
```

```
>
> # Alternate Check for Breusch-Pagan Test for Heteroscedasticity;
> # Ho: Homoscedasticity (Variance of residuals is constant)
> # Ha: Heteroscedasticity
> #install.packages("lmtest")
> library(lmtest)
> lmtest::bptest(reg_model)

	studentized Breusch-Pagan test

data:  reg_model
BP = 15.953, df = 7, p-value = 0.02555

> lmtest::bptest(reg_model2)

	studentized Breusch-Pagan test

data:  reg_model2
BP = 7.7561, df = 5, p-value = 0.1702
> library(e1071)
> library(caret)
Loading required package: lattice
> # How to rectify?
> # Re-build the model with new predictors.
> # Variable transformation such as Box-Cox transformation can also be tried instead of log price
> # (Normal Distribution).
> boxcoxprice <- caret::BoxCoxTrans(propertytrainData$Price)   #Normalizing Price using BoxCox
Transformation
> print(boxcoxprice)
Box-Cox Transformation

31 data points used to estimate Lambda

Input data summary:
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
   1.70    5.35   11.70   11.95   16.05   37.20 

Largest/Smallest: 21.9 
Sample Skewness: 1.03 

Estimated Lambda: 0.3 
> propertytrainData <- cbind(propertytrainData, Newprice=predict(boxcoxprice,
propertytrainData$Price))   #add predicted normalized price column
> head(propertytrainData)
```

```
  Price Sea   Area Elevation Sewer Days Flood Distance  Logprice  Newprice
1   4.5   1  138.4        10  3000 -103     0      0.3 1.5040774 1.9007725
2  10.6   1   52.0         4     0 -103     0      2.5 2.3608540 3.4348249
3   1.7   0   16.1         0  2640  -98     1     10.3 0.5306283 0.5751964
4   5.0   0 1695.2         1  3500  -93     0     14.0 1.6094379 2.0688553
5   5.0   0  845.0         1  1000  -92     1     14.0 1.6094379 2.0688553
6   3.3   1    6.9         2 10000  -86     0      0.0 1.1939225 1.4357282
> reg_model22 <- lm(Newprice ~ Days+Distance+Flood+Elevation+Sewer, data = propertytrainData)
> summary(reg_model22)

Call:
lm(formula = Newprice ~ Days + Distance + Flood + Elevation +
    Sewer, data = propertytrainData)

Residuals:
    Min      1Q  Median      3Q     Max
-0.7648 -0.4730 -0.0573  0.4420  1.7239

Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept)  4.944e+00  4.399e-01  11.238 2.89e-11 ***
Days         3.365e-02  5.064e-03   6.644 5.81e-07 ***
Distance     9.468e-02  3.162e-02   2.994 0.006128 **
Flood       -1.619e+00  3.798e-01  -4.262 0.000252 ***
Elevation    9.871e-02  3.578e-02   2.759 0.010694 *
Sewer       -1.401e-04  5.789e-05  -2.421 0.023095 *
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.6661 on 25 degrees of freedom
Multiple R-squared:  0.8072,	Adjusted R-squared:  0.7686
F-statistic: 20.93 on 5 and 25 DF,  p-value: 3.31e-08

> lmtest::bptest(reg_model22)

	studentized Breusch-Pagan test

data:  reg_model22
BP = 4.3918, df = 5, p-value = 0.4945

> plot(reg_model22)
Hit <Return> to see next plot:
Hit <Return> to see next plot:
Hit <Return> to see next plot:
```

```
Hit <Return> to see next plot:
>
> #Autocorrelation: Durbin watson Test
> #H0: No Autocorrelation, Ha: Autocorrelation present
> lmtest::dwtest(reg_model)     #p-vale: 0.74

	Durbin-Watson test

data:  reg_model
DW = 2.4103, p-value = 0.7465
alternative hypothesis: true autocorrelation is greater than 0

> lmtest::dwtest(reg_model1)    #p-vale: 0.73

	Durbin-Watson test

data:  reg_model1
DW = 2.3327, p-value = 0.7328
alternative hypothesis: true autocorrelation is greater than 0

> lmtest::dwtest(reg_model2)    #p-vale: 0.67

	Durbin-Watson test

data:  reg_model2
DW = 2.265, p-value = 0.6728
alternative hypothesis: true autocorrelation is greater than 0

>
> #Fitting the model
> predicted_salesPrice <- predict(reg_model2, newdata = propertytrainData)
> predicted_salesPrice
        1         2         3         4         5         6         7         8         9        10        11        12        13
1.5963200 1.6344573 0.8009036 1.9305666 1.2607307 0.9211005 2.1182612 2.1872227 2.9673940 1.5482586 1.5654990 1.8520608 1.9548896
       14        15        16        17        18        19        20        21        22        23        24        25        26
2.6858571 2.7772486 2.7411216 2.7002102 2.6088186 2.8468662 2.7145633 2.6327763 2.6588934 2.5390465 1.8259669 1.7226961 3.0131863
       27        28        29        30        31
2.5844959 2.8667676 3.0611836 2.4123110 3.3130961
> propertytestData$PredictedPrice <- exp(predicted_salesPrice)
> write.csv(propertytestData, "predictedresult.csv")
>
> cor(propertytestData$Price, propertytestData$PredictedPrice)
```

```
[1] 0.2907895
> plot(propertytestData$Price, propertytestData$PredictedPrice)
>
> #install.packages("Metrics")
> library(Metrics)
> Metrics::rmse(propertytestData$Price, propertytestData$PredictedPrice)
[1] 8.158264
```

Day 8 – R Programming

Admission Classification – Logistic Regression

```
> df <- read.csv("https://stats.idre.ucla.edu/stat/data/binary.csv")
> head(df)
  admit gre  gpa rank
1     0 380 3.61    3
2     1 660 3.67    3
3     1 800 4.00    1
4     1 640 3.19    4
5     0 520 2.93    4
6     1 760 3.00    2
> str(df)
'data.frame':	400 obs. of  4 variables:
 $ admit: int  0 1 1 1 0 1 1 0 1 0 ...
 $ gre  : int  380 660 800 640 520 760 560 400 540 700 ...
 $ gpa  : num  3.61 3.67 4 3.19 2.93 3 2.98 3.08 3.39 3.92 ...
 $ rank : int  3 3 1 4 4 2 1 2 3 2 ...
> dim(df)
[1] 400   4
> edit(df)
    admit gre  gpa rank
1       0 380 3.61    3
2       1 660 3.67    3
3       1 800 4.00    1
4       1 640 3.19    4
5       0 520 2.93    4
6       1 760 3.00    2
7       1 560 2.98    1
8       0 400 3.08    2
9       1 540 3.39    3
10      0 700 3.92    2
11      0 800 4.00    4
12      0 440 3.22    1
13      1 760 4.00    1
14      0 700 3.08    2
15      1 700 4.00    1
```

16	0	480	3.44	3
17	0	780	3.87	4
18	0	360	2.56	3
19	0	800	3.75	2
20	1	540	3.81	1
21	0	500	3.17	3
22	1	660	3.63	2
23	0	600	2.82	4
24	0	680	3.19	4
25	1	760	3.35	2
26	1	800	3.66	1
27	1	620	3.61	1
28	1	520	3.74	4
29	1	780	3.22	2
30	0	520	3.29	1
31	0	540	3.78	4
32	0	760	3.35	3
33	0	600	3.40	3
34	1	800	4.00	3
35	0	360	3.14	1
36	0	400	3.05	2
37	0	580	3.25	1
38	0	520	2.90	3
39	1	500	3.13	2
40	1	520	2.68	3
41	0	560	2.42	2
42	1	580	3.32	2
43	1	600	3.15	2
44	0	500	3.31	3
45	0	700	2.94	2
46	1	460	3.45	3
47	1	580	3.46	2
48	0	500	2.97	4
49	0	440	2.48	4
50	0	400	3.35	3
51	0	640	3.86	3
52	0	440	3.13	4
53	0	740	3.37	4
54	1	680	3.27	2
55	0	660	3.34	3
56	1	740	4.00	3
57	0	560	3.19	3
58	0	380	2.94	3
59	0	400	3.65	2

60	0	600	2.82	4
61	1	620	3.18	2
62	0	560	3.32	4
63	0	640	3.67	3
64	1	680	3.85	3
65	0	580	4.00	3
66	0	600	3.59	2
67	0	740	3.62	4
68	0	620	3.30	1
69	0	580	3.69	1
70	0	800	3.73	1
71	0	640	4.00	3
72	0	300	2.92	4
73	0	480	3.39	4
74	0	580	4.00	2
75	0	720	3.45	4
76	0	720	4.00	3
77	0	560	3.36	3
78	1	800	4.00	3
79	0	540	3.12	1
80	1	620	4.00	1
81	0	700	2.90	4
82	0	620	3.07	2
83	0	500	2.71	2
84	0	380	2.91	4
85	1	500	3.60	3
86	0	520	2.98	2
87	0	600	3.32	2
88	0	600	3.48	2
89	0	700	3.28	1
90	1	660	4.00	2
91	0	700	3.83	2
92	1	720	3.64	1
93	0	800	3.90	2
94	0	580	2.93	2
95	1	660	3.44	2
96	0	660	3.33	2
97	0	640	3.52	4
98	0	480	3.57	2
99	0	700	2.88	2
100	0	400	3.31	3
101	0	340	3.15	3
102	0	580	3.57	3
103	0	380	3.33	4

104	0	540	3.94	3
105	1	660	3.95	2
106	1	740	2.97	2
107	1	700	3.56	1
108	0	480	3.13	2
109	0	400	2.93	3
110	0	480	3.45	2
111	0	680	3.08	4
112	0	420	3.41	4
113	0	360	3.00	3
114	0	600	3.22	1
115	0	720	3.84	3
116	0	620	3.99	3
117	1	440	3.45	2
118	0	700	3.72	2
119	1	800	3.70	1
120	0	340	2.92	3
121	1	520	3.74	2
122	1	480	2.67	2
123	0	520	2.85	3
124	0	500	2.98	3
125	0	720	3.88	3
126	0	540	3.38	4
127	1	600	3.54	1
128	0	740	3.74	4
129	0	540	3.19	2
130	0	460	3.15	4
131	1	620	3.17	2
132	0	640	2.79	2
133	0	580	3.40	2
134	0	500	3.08	3
135	0	560	2.95	2
136	0	500	3.57	3
137	0	560	3.33	4
138	0	700	4.00	3
139	0	620	3.40	2
140	1	600	3.58	1
141	0	640	3.93	2
142	1	700	3.52	4
143	0	620	3.94	4
144	0	580	3.40	3
145	0	580	3.40	4
146	0	380	3.43	3
147	0	480	3.40	2

148	0	560	2.71	3
149	1	480	2.91	1
150	0	740	3.31	1
151	1	800	3.74	1
152	0	400	3.38	2
153	1	640	3.94	2
154	0	580	3.46	3
155	0	620	3.69	3
156	1	580	2.86	4
157	0	560	2.52	2
158	1	480	3.58	1
159	0	660	3.49	2
160	0	700	3.82	3
161	0	600	3.13	2
162	0	640	3.50	2
163	1	700	3.56	2
164	0	520	2.73	2
165	0	580	3.30	2
166	0	700	4.00	1
167	0	440	3.24	4
168	0	720	3.77	3
169	0	500	4.00	3
170	0	600	3.62	3
171	0	400	3.51	3
172	0	540	2.81	3
173	0	680	3.48	3
174	1	800	3.43	2
175	0	500	3.53	4
176	1	620	3.37	2
177	0	520	2.62	2
178	1	620	3.23	3
179	0	620	3.33	3
180	0	300	3.01	3
181	0	620	3.78	3
182	0	500	3.88	4
183	0	700	4.00	2
184	1	540	3.84	2
185	0	500	2.79	4
186	0	800	3.60	2
187	0	560	3.61	3
188	0	580	2.88	2
189	0	560	3.07	2
190	0	500	3.35	2
191	1	640	2.94	2

192	0	800	3.54	3
193	0	640	3.76	3
194	0	380	3.59	4
195	1	600	3.47	2
196	0	560	3.59	2
197	0	660	3.07	3
198	1	400	3.23	4
199	0	600	3.63	3
200	0	580	3.77	4
201	0	800	3.31	3
202	1	580	3.20	2
203	1	700	4.00	1
204	0	420	3.92	4
205	1	600	3.89	1
206	1	780	3.80	3
207	0	740	3.54	1
208	1	640	3.63	1
209	0	540	3.16	3
210	0	580	3.50	2
211	0	740	3.34	4
212	0	580	3.02	2
213	0	460	2.87	2
214	0	640	3.38	3
215	1	600	3.56	2
216	1	660	2.91	3
217	0	340	2.90	1
218	1	460	3.64	1
219	0	460	2.98	1
220	1	560	3.59	2
221	0	540	3.28	3
222	0	680	3.99	3
223	1	480	3.02	1
224	0	800	3.47	3
225	0	800	2.90	2
226	1	720	3.50	3
227	0	620	3.58	2
228	0	540	3.02	4
229	0	480	3.43	2
230	1	720	3.42	2
231	0	580	3.29	4
232	0	600	3.28	3
233	0	380	3.38	2
234	0	420	2.67	3
235	1	800	3.53	1

```
236    0 620 3.05    2
237    1 660 3.49    2
238    0 480 4.00    2
239    0 500 2.86    4
240    0 700 3.45    3
241    0 440 2.76    2
242    1 520 3.81    1
243    1 680 2.96    3
244    0 620 3.22    2
245    0 540 3.04    1
246    0 800 3.91    3
247    0 680 3.34    2
248    0 440 3.17    2
249    0 680 3.64    3
250    0 640 3.73    3
 [ reached 'max' / getOption("max.print") -- omitted 150 rows ]
> sum(is.na(df))
[1] 0
> summary(df)
     admit            gre             gpa             rank      
 Min.   :0.0000   Min.   :220.0   Min.   :2.260   Min.   :1.000  
 1st Qu.:0.0000   1st Qu.:520.0   1st Qu.:3.130   1st Qu.:2.000  
 Median :0.0000   Median :580.0   Median :3.395   Median :2.000  
 Mean   :0.3175   Mean   :587.7   Mean   :3.390   Mean   :2.485  
 3rd Qu.:1.0000   3rd Qu.:660.0   3rd Qu.:3.670   3rd Qu.:3.000  
 Max.   :1.0000   Max.   :800.0   Max.   :4.000   Max.   :4.000  
> #Mean < 0.5 means more rejection of students for admission then acceptance
> xtabs(~ admit + rank, data = df)   #Frequency table
     rank
admit  1  2  3  4
    0 28 97 93 55
    1 33 54 28 12
> df$rank <- as.factor(df$rank)
> logit <- glm(admit ~ gre+gpa+rank, data=df, family="binomial")
> summary(logit)

Call:
glm(formula = admit ~ gre + gpa + rank, family = "binomial", 
    data = df)

Deviance Residuals: 
    Min       1Q   Median       3Q      Max  
-1.6268  -0.8662  -0.6388   1.1490   2.0790  
```

```
Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept) -3.989979   1.139951  -3.500 0.000465 ***
gre          0.002264   0.001094   2.070 0.038465 *
gpa          0.804038   0.331819   2.423 0.015388 *
rank2       -0.675443   0.316490  -2.134 0.032829 *
rank3       -1.340204   0.345306  -3.881 0.000104 ***
rank4       -1.551464   0.417832  -3.713 0.000205 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 499.98  on 399  degrees of freedom
Residual deviance: 458.52  on 394  degrees of freedom
AIC: 470.52

Number of Fisher Scoring iterations: 4
> #Predicting raw data
> x <- data.frame(gre=790, gpa=3.8, rank=as.factor(1))
> p <- predict(logit, x)
> p
      1
0.85426
> x <- data.frame(gre=600, gpa=3.0, rank=as.factor(3))
> p <- predict(logit, x)
> p
        1
-1.559415
```

Automatic/Manual Car Classification – Logistic Regression

> #The in-built data set "mtcars" describes different models of a car with their various engine specifications. In "mtcars" data set, the transmission mode (automatic or manual) is described by the column am which is a binary value (0 or 1). We can create a logistic regression model between the columns "am" and 3 other columns - hp, wt and cyl.

```
> mtcars
                   mpg cyl  disp  hp drat    wt  qsec vs am gear carb
Mazda RX4         21.0   6 160.0 110 3.90 2.620 16.46  0  1    4    4
Mazda RX4 Wag     21.0   6 160.0 110 3.90 2.875 17.02  0  1    4    4
Datsun 710        22.8   4 108.0  93 3.85 2.320 18.61  1  1    4    1
Hornet 4 Drive    21.4   6 258.0 110 3.08 3.215 19.44  1  0    3    1
Hornet Sportabout 18.7   8 360.0 175 3.15 3.440 17.02  0  0    3    2
Valiant           18.1   6 225.0 105 2.76 3.460 20.22  1  0    3    1
Duster 360        14.3   8 360.0 245 3.21 3.570 15.84  0  0    3    4
```

```
Merc 240D          24.4   4 146.7  62 3.69 3.190 20.00  1  0    4    2
Merc 230           22.8   4 140.8  95 3.92 3.150 22.90  1  0    4    2
Merc 280           19.2   6 167.6 123 3.92 3.440 18.30  1  0    4    4
Merc 280C          17.8   6 167.6 123 3.92 3.440 18.90  1  0    4    4
Merc 450SE         16.4   8 275.8 180 3.07 4.070 17.40  0  0    3    3
Merc 450SL         17.3   8 275.8 180 3.07 3.730 17.60  0  0    3    3
Merc 450SLC        15.2   8 275.8 180 3.07 3.780 18.00  0  0    3    3
Cadillac Fleetwood  10.4   8 472.0 205 2.93 5.250 17.98  0  0    3    4
Lincoln Continental 10.4   8 460.0 215 3.00 5.424 17.82  0  0    3    4
Chrysler Imperial   14.7   8 440.0 230 3.23 5.345 17.42  0  0    3    4
Fiat 128           32.4   4  78.7  66 4.08 2.200 19.47  1  1    4    1
Honda Civic        30.4   4  75.7  52 4.93 1.615 18.52  1  1    4    2
Toyota Corolla     33.9   4  71.1  65 4.22 1.835 19.90  1  1    4    1
Toyota Corona      21.5   4 120.1  97 3.70 2.465 20.01  1  0    3    1
Dodge Challenger   15.5   8 318.0 150 2.76 3.520 16.87  0  0    3    2
AMC Javelin        15.2   8 304.0 150 3.15 3.435 17.30  0  0    3    2
Camaro Z28         13.3   8 350.0 245 3.73 3.840 15.41  0  0    3    4
Pontiac Firebird   19.2   8 400.0 175 3.08 3.845 17.05  0  0    3    2
Fiat X1-9          27.3   4  79.0  66 4.08 1.935 18.90  1  1    4    1
Porsche 914-2      26.0   4 120.3  91 4.43 2.140 16.70  0  1    5    2
Lotus Europa       30.4   4  95.1 113 3.77 1.513 16.90  1  1    5    2
Ford Pantera L     15.8   8 351.0 264 4.22 3.170 14.50  0  1    5    4
Ferrari Dino       19.7   6 145.0 175 3.62 2.770 15.50  0  1    5    6
Maserati Bora      15.0   8 301.0 335 3.54 3.570 14.60  0  1    5    8
Volvo 142E         21.4   4 121.0 109 4.11 2.780 18.60  1  1    4    2
> str(mtcars)
'data.frame':	32 obs. of  11 variables:
 $ mpg : num  21 21 22.8 21.4 18.7 18.1 14.3 24.4 22.8 19.2 ...
 $ cyl : num  6 6 4 6 8 6 8 4 4 6 ...
 $ disp: num  160 160 108 258 360 ...
 $ hp  : num  110 110 93 110 175 105 245 62 95 123 ...
 $ drat: num  3.9 3.9 3.85 3.08 3.15 2.76 3.21 3.69 3.92 3.92 ...
 $ wt  : num  2.62 2.88 2.32 3.21 3.44 ...
 $ qsec: num  16.5 17 18.6 19.4 17 ...
 $ vs  : num  0 0 1 1 0 1 0 1 1 1 ...
 $ am  : num  1 1 1 0 0 0 0 0 0 0 ...
 $ gear: num  4 4 4 3 3 3 3 4 4 4 ...
 $ carb: num  4 4 1 1 2 1 4 2 2 4 ...
> dim(mtcars)
[1] 32 11
> sum(is.na(mtcars))
[1] 0
> summary(mtcars)
      mpg             cyl             disp             hp             drat             wt
```

```
 Min.   :10.40   Min.   :4.000   Min.   : 71.1   Min.   : 52.0   Min.   :2.760   Min.   :1.513  
 1st Qu.:15.43   1st Qu.:4.000   1st Qu.:120.8   1st Qu.: 96.5   1st Qu.:3.080   1st Qu.:2.581  
 Median :19.20   Median :6.000   Median :196.3   Median :123.0   Median :3.695   Median :3.325  
 Mean   :20.09   Mean   :6.188   Mean   :230.7   Mean   :146.7   Mean   :3.597   Mean   :3.217  
 3rd Qu.:22.80   3rd Qu.:8.000   3rd Qu.:326.0   3rd Qu.:180.0   3rd Qu.:3.920   3rd Qu.:3.610  
 Max.   :33.90   Max.   :8.000   Max.   :472.0   Max.   :335.0   Max.   :4.930   Max.   :5.424  
      qsec             vs               am              gear            carb      
 Min.   :14.50   Min.   :0.0000   Min.   :0.0000   Min.   :3.000   Min.   :1.000  
 1st Qu.:16.89   1st Qu.:0.0000   1st Qu.:0.0000   1st Qu.:3.000   1st Qu.:2.000  
 Median :17.71   Median :0.0000   Median :0.0000   Median :4.000   Median :2.000  
 Mean   :17.85   Mean   :0.4375   Mean   :0.4062   Mean   :3.688   Mean   :2.812  
 3rd Qu.:18.90   3rd Qu.:1.0000   3rd Qu.:1.0000   3rd Qu.:4.000   3rd Qu.:4.000  
 Max.   :22.90   Max.   :1.0000   Max.   :1.0000   Max.   :5.000   Max.   :8.000  
> xtabs(~ am + cyl, data=mtcars)
   cyl
am   4  6  8
  0  3  4 12
  1  8  3  2
> table(mtcars$am, mtcars$cyl)
   
     4  6  8
  0  3  4 12
  1  8  3  2
> 
> cars1 <- mtcars[, c("cyl", "hp", "wt", "am")]
> head(cars1)
                  cyl  hp    wt am
Mazda RX4           6 110 2.620  1
Mazda RX4 Wag       6 110 2.875  1
Datsun 710          4  93 2.320  1
Hornet 4 Drive      6 110 3.215  0
Hornet Sportabout   8 175 3.440  0
Valiant             6 105 3.460  0
> 
> logit <- glm(formula=am ~ cyl+hp+wt, data=cars1, family="binomial")
> summary(logit)

Call:
glm(formula = am ~ cyl + hp + wt, family = "binomial", data = cars1)

Deviance Residuals: 
     Min        1Q    Median        3Q       Max  
-2.17272  -0.14907  -0.01464   0.14116   1.27641  
```

```
Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept) 19.70288    8.11637   2.428   0.0152 *
cyl          0.48760    1.07162   0.455   0.6491
hp           0.03259    0.01886   1.728   0.0840 .
wt          -9.14947    4.15332  -2.203   0.0276 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 43.2297  on 31  degrees of freedom
Residual deviance:  9.8415  on 28  degrees of freedom
AIC: 17.841

Number of Fisher Scoring iterations: 8

> x <- data.frame(cyl=6, hp=110, wt=3.200)
> p <- predict(logit, x)
> p
        1
-3.064753
```

Day 9 – R Programming

German Credit – Decision Tree

```
> setwd("C:/zubeda/PGA02_Zubu/R Programming/Models")
>
> #Read the data file
> data <- read.csv("german_credit.csv")
> #Check attributes of data
> str(data)
'data.frame':	1000 obs. of  21 variables:
 $ Creditability                    : int  1 1 1 1 1 1 1 1 1 1 ...
 $ Account.Balance                  : int  1 1 2 1 1 1 1 1 4 2 ...
 $ Duration.of.Credit..month.       : int  18 9 12 12 12 10 8 6 18 24 ...
 $ Payment.Status.of.Previous.Credit: int  4 4 2 4 4 4 4 4 4 2 ...
 $ Purpose                          : int  2 0 9 0 0 0 0 0 3 3 ...
 $ Credit.Amount                    : int  1049 2799 841 2122 2171 2241 3398 1361 1098 3758 ...
 $ Value.Savings.Stocks             : int  1 1 2 1 1 1 1 1 1 3 ...
 $ Length.of.current.employment     : int  2 3 4 3 3 2 4 2 1 1 ...
 $ Instalment.per.cent              : int  4 2 2 3 4 1 1 2 4 1 ...
 $ Sex...Marital.Status             : int  2 3 2 3 3 3 3 3 2 2 ...
 $ Guarantors                       : int  1 1 1 1 1 1 1 1 1 1 ...
 $ Duration.in.Current.address      : int  4 2 4 2 4 3 4 4 4 4 ...
```

```
 $ Most.valuable.available.asset    : int  2 1 1 1 2 1 1 1 3 4 ...
 $ Age..years.                  : int  21 36 23 39 38 48 39 40 65 23 ...
 $ Concurrent.Credits             : int  3 3 3 3 1 3 3 3 3 3 ...
 $ Type.of.apartment              : int  1 1 1 1 2 1 2 2 2 1 ...
 $ No.of.Credits.at.this.Bank      : int  1 2 1 2 2 2 2 1 2 1 ...
 $ Occupation                   : int  3 3 2 2 2 2 2 2 1 1 ...
 $ No.of.dependents              : int  1 2 1 2 1 2 1 2 1 1 ...
 $ Telephone                    : int  1 1 1 1 1 1 1 1 1 1 ...
 $ Foreign.Worker                : int  1 1 1 2 2 2 2 2 1 1 ...
> #Columns of data
> names(data)
 [1] "Creditability"                   "Account.Balance"
 [3] "Duration.of.Credit..month."       "Payment.Status.of.Previous.Credit"
 [5] "Purpose"                         "Credit.Amount"
 [7] "Value.Savings.Stocks"            "Length.of.current.employment"
 [9] "Instalment.per.cent"             "Sex...Marital.Status"
[11] "Guarantors"                      "Duration.in.Current.address"
[13] "Most.valuable.available.asset"     "Age..years."
[15] "Concurrent.Credits"              "Type.of.apartment"
[17] "No.of.Credits.at.this.Bank"       "Occupation"
[19] "No.of.dependents"                "Telephone"
[21] "Foreign.Worker"
> #Check no. of rows & columns
> dim(data)
[1] 1000   21
> head(data)  #First 6 rows
  Creditability Account.Balance Duration.of.Credit..month. Payment.Status.of.Previous.Credit Purpose
1             1               1                         18                                 4       2
2             1               1                          9                                 4       0
3             1               2                         12                                 2       9
4             1               1                         12                                 4       0
5             1               1                         12                                 4       0
6             1               1                         10                                 4       0
  Credit.Amount Value.Savings.Stocks Length.of.current.employment Instalment.per.cent
Sex...Marital.Status
1          1049                    1                            2                   4                    2
2          2799                    1                            3                   2                    3
3           841                    2                            4                   2                    2
4          2122                    1                            3                   3                    3
5          2171                    1                            3                   4                    3
6          2241                    1                            2                   1                    3
  Guarantors Duration.in.Current.address Most.valuable.available.asset Age..years. Concurrent.Credits
1          1                           4                             2          21                  3
2          1                           2                             1          36                  3
```

```
3        1                 4                 1      23             3
4        1                 2                 1      39             3
5        1                 4                 2      38             1
6        1                 3                 1      48             3
  Type.of.apartment No.of.Credits.at.this.Bank Occupation No.of.dependents Telephone Foreign.Worker
1            1                    1       3          1      1          1
2            1                    2       3          2      1          1
3            1                    1       2          1      1          1
4            1                    2       2          2      1          2
5            2                    2       2          1      1          2
6            1                    2       2          2      1          2
>
> #Make dependent variable Credibility into factor (categorical)
> class(data$Creditability)
[1] "integer"
> data$Creditability <- as.factor(data$Creditability)
> class(data$Creditability)
[1] "factor"
> class(data)
[1] "data.frame"
>
> set.seed(123)   #Maintains the state
> #Splitting the data into training 70% and validation 30%
> dt <- sort(sample(nrow(data), nrow(data) * .7))  #Select 70% random row indices
> train <- data[dt, ]   #Selected 70% rows & all the columns
> val <- data[-dt, ]    #Not selected rows 30% & all the columns
> #Check no.of rows in training data set
> nrow(train)
[1] 700
> #Check no.of rows in validation data set
> nrow(val)
[1] 300
> #View datasets
> edit(train)
   Creditability Account.Balance Duration.of.Credit..month. Payment.Status.of.Previous.Credit Purpose
2          1           1                 9                        4      0
5          1           1                12                         4      0
6          1           1                10                         4      0
8          1           1                 6                        4      0
10          1           2                 24                         2      3
11          1           1                 11                         4      0
13          1           1                  6                        4      3
14          1           2                 48                         3     10
16          1           1                  6                        2      3
```

19	1	2	36	4	3
20	1	4	11	4	0
23	0	2	36	2	5
24	1	2	12	4	4
26	1	2	11	3	3
29	1	4	15	2	0
30	1	3	42	4	1
31	1	3	30	4	3
33	1	4	36	4	0
34	1	4	24	2	3
36	1	1	6	4	0
37	1	4	12	4	0
38	1	4	12	4	3
39	1	4	18	2	1
40	1	4	24	4	1
41	1	4	12	4	5
45	1	2	18	2	6
46	0	1	18	2	0
48	0	4	18	4	6
49	1	4	24	2	0
51	1	4	12	2	0
52	1	3	36	2	3
53	1	4	9	4	0
54	1	4	12	4	3
55	1	4	24	2	1
56	1	1	12	4	3
57	1	4	12	4	3
59	1	4	21	2	3
61	1	4	12	4	0
64	1	4	36	3	0
65	1	1	12	3	0
67	1	4	12	2	3
68	1	4	24	2	3
69	1	2	12	2	3
71	1	2	21	4	2
72	1	4	30	2	3
74	1	4	24	2	2
76	1	2	9	2	2

	Credit.Amount	Value.Savings.Stocks	Length.of.current.employment	Instalment.per.cent	Sex...Marital.Status
2	2799	1	3	2	3
5	2171	1	3	4	3
6	2241	1	2	1	3
8	1361	1	2	2	3

10	3758	3	1	1	2
11	3905	1	3	2	3
13	1957	1	4	1	2
14	7582	2	1	2	3
16	2647	3	3	2	3
19	2337	1	5	4	3
20	7228	1	3	1	3
23	2384	1	2	4	3
24	1424	1	4	4	3
26	4771	1	4	2	3
29	3556	5	3	3	3
30	4796	1	5	4	3
31	3017	1	5	4	3
33	6614	1	5	4	3
34	1376	3	4	4	2
36	860	1	5	1	2
37	1495	1	5	4	3
38	1934	1	5	2	3
39	3378	5	3	2	3
40	3868	1	5	4	2
41	996	5	4	4	2
45	1239	5	3	4	3
46	1216	1	2	4	2
48	1864	2	3	4	2
49	1474	2	2	4	4
51	640	1	3	4	1
52	3919	1	3	2	3
53	1224	1	3	3	3
54	2331	5	5	1	3
55	6313	5	5	3	3
56	385	1	4	4	2
57	1655	1	5	2	3
59	3160	5	5	4	3
61	1163	3	3	4	3
64	10875	1	5	2	3
65	1344	1	3	4	3
67	3077	1	3	2	3
68	2284	1	4	4	3
69	1567	1	3	1	2
71	2745	4	4	3	3
72	1867	5	5	4	3
74	929	5	4	4	3
76	2030	5	4	2	3

Guarantors Duration.in.Current.address Most.valuable.available.asset Age..years. Concurrent.Credits

2	1	2	1	36	3
5	1	4	2	38	1
6	1	3	1	48	3
8	1	4	1	40	3
10	1	4	4	23	3
11	1	2	1	36	3
13	1	4	3	31	3
14	1	4	4	31	3
16	1	3	1	44	3
19	1	4	1	36	3
20	1	4	2	39	3
23	1	1	4	33	3
24	1	3	2	26	3
26	1	4	2	51	3
29	1	2	4	29	3
30	1	4	4	56	3
31	1	4	2	47	3
33	1	4	3	34	3
34	1	1	3	28	3
36	1	4	4	39	3
37	1	1	1	38	3
38	1	2	4	26	3
39	1	1	2	31	3
40	1	2	3	41	3
41	1	4	1	23	3
45	1	4	4	61	3
46	1	3	3	23	3
48	1	2	1	30	3
49	1	3	1	33	3
51	1	2	1	49	3
52	1	2	1	23	3
53	1	1	1	30	3
54	2	4	1	49	3
55	1	4	3	41	3
56	1	3	1	58	3
57	1	4	1	63	3
59	1	3	2	41	3
61	1	4	1	44	3
64	1	2	3	45	3
65	1	2	1	43	3
67	1	4	3	52	3
68	1	2	3	28	3
69	1	1	3	22	3
71	1	2	3	32	3

72	1	4	3	58	3
74	1	2	3	31	2
76	1	1	3	24	3

	Type.of.apartment	No.of.Credits.at.this.Bank	Occupation	No.of.dependents	Telephone	Foreign.Worker
2	1	2	3	2	1	1
5	2	2	2	1	1	2
6	1	2	2	2	1	2
8	2	1	2	2	1	2
10	1	1	1	1	1	1
11	1	2	3	2	1	1
13	2	1	3	1	1	1
14	2	1	4	1	2	1
16	1	1	3	2	1	1
19	2	1	3	1	1	1
20	2	2	2	1	1	1
23	1	1	2	1	1	1
24	2	1	3	1	1	1
26	2	1	3	1	1	1
29	2	1	3	1	1	1
30	3	1	3	1	1	1
31	2	1	3	1	1	1
33	2	2	4	1	2	1
34	2	1	3	1	1	1
36	2	2	3	1	2	1
37	2	2	2	2	1	1
38	2	2	3	1	1	1
39	2	1	3	1	2	1
40	1	2	4	1	2	1
41	2	2	3	1	1	1
45	3	1	3	1	1	1
46	1	1	3	1	2	1
48	2	2	3	1	1	1
49	2	1	3	1	2	1
51	2	1	2	1	1	1
52	2	1	3	1	2	1
53	2	2	3	1	1	1
54	2	1	3	1	2	1
55	2	1	4	2	2	1
56	2	4	2	1	2	1
57	2	2	2	1	2	1
59	2	1	3	1	2	1
61	2	1	3	1	2	1
64	2	2	3	2	2	1

```
65         2           2     2          2     1        1
67         2           1     3          1     2        1
68         2           1     3          1     2        1
69         2           1     3          1     2        1
71         2           2     3          1     2        1
72         2           1     3          1     2        1
74         2           1     3          1     2        1
76         2           1     3          1     2        1
 [ reached 'max' / getOption("max.print") -- omitted 653 rows ]
> edit(val)
   Creditability Account.Balance Duration.of.Credit..month. Payment.Status.of.Previous.Credit Purpose
1          1          1                18                        4    2
3          1          2                12                        2    9
4          1          1                12                        4    0
7          1          1                 8                        4    0
9          1          4                18                        4    3
12         1          1                30                        4    1
15         1          1                18                        2    3
17         1          1                11                        4    0
18         1          2                18                        2    3
21         1          1                 6                        4    0
22         1          2                12                        4    0
25         1          1                 6                        4    0
27         1          1                12                        2    2
28         1          2                 9                        4    3
32         1          4                36                        4    0
35         1          1                15                        2    0
42         1          1                24                        2   10
43         1          4                18                        4    0
44         1          2                24                        4    9
47         1          4                24                        2    9
50         1          1                24                        4    9
58         1          1                15                        2    3
60         1          4                36                        2    0
62         1          4                24                        2    1
63         1          4                48                        4    3
66         1          4                 6                        4    3
70         1          4                24                        3    0
73         1          4                36                        2    3
75         1          3                12                        2    3
77         1          4                21                        4    1
82         1          4                36                        4    3
86         1          1                12                        2    0
92         1          2                30                        2    3
```

93	1	2	30	0	9
97	1	2	12	4	3
99	1	1	9	2	2
101	1	4	24	4	2
102	1	1	15	2	9
103	1	2	24	3	9
107	1	4	12	4	9
109	1	4	24	2	3
112	1	1	6	2	2
114	1	4	12	2	2
123	1	2	6	2	3
126	1	4	24	4	3
133	1	3	6	2	2
140	1	3	24	4	3

	Credit.Amount	Value.Savings.Stocks	Length.of.current.employment	Instalment.per.cent	Sex...Marital.Status
1	1049	1	2	4	2
3	841	2	4	2	2
4	2122	1	3	3	3
7	3398	1	4	1	3
9	1098	1	1	4	2
12	6187	2	4	1	4
15	1936	5	4	2	4
17	3939	1	3	1	3
18	3213	3	2	1	4
21	3676	1	3	1	3
22	3124	1	2	1	3
25	4716	5	2	1	3
27	652	1	5	4	2
28	1154	1	5	2	3
32	3535	1	4	4	3
35	1721	1	2	2	3
42	1755	1	5	4	2
43	1028	1	3	4	2
44	2825	5	4	4	3
47	1258	1	4	4	3
50	1382	2	4	4	3
58	1053	1	2	4	4
60	3079	5	3	4	3
62	2679	1	2	4	2
63	3578	5	5	4	3
66	1237	2	3	1	2
70	2032	1	5	4	3
73	2299	3	5	4	3

75	3399	5	5	2	3
77	3275	1	5	1	3
82	3342	5	5	4	3
86	3651	4	3	1	3
92	2991	5	5	2	2
93	4221	1	3	2	2
97	3573	1	3	1	2
99	2136	1	3	3	3
101	3777	4	3	4	3
102	806	1	3	4	2
103	4712	5	3	4	3
107	1412	1	3	4	2
109	1533	1	2	4	2
112	428	1	5	2	2
114	763	1	3	4	2
123	2063	1	2	4	4
126	5103	1	2	3	4
133	2116	1	3	2	3
140	3148	5	3	3	3

	Guarantors	Duration.in.Current.address	Most.valuable.available.asset	Age..years.	Concurrent.Credits
1	1	4	2	21	3
3	1	4	1	23	3
4	1	2	1	39	3
7	1	4	1	39	3
9	1	4	3	65	3
12	1	4	3	24	3
15	1	4	3	23	3
17	1	2	1	40	3
18	1	3	1	25	3
21	1	3	1	37	3
22	1	3	1	49	1
25	1	3	1	44	3
27	1	4	2	24	3
28	1	4	1	37	3
32	1	4	3	37	3
35	1	3	1	36	3
42	3	4	1	58	3
43	1	3	1	36	3
44	1	3	4	34	3
47	1	1	1	25	3
50	1	1	1	26	3
58	1	2	1	27	3
60	1	4	1	36	3
62	1	1	4	29	3

63	1	1	1	47	3
66	1	1	2	27	3
70	1	4	4	60	3
73	1	4	3	39	3
75	1	3	3	37	3
77	1	4	3	36	3
82	1	2	3	51	3
86	1	3	2	31	3
92	1	4	3	25	3
93	1	1	3	28	3
97	1	1	1	23	3
99	1	2	1	25	3
101	1	4	1	40	3
102	1	4	2	22	3
103	1	2	2	34	1
107	3	2	1	29	3
109	1	3	3	38	2
112	1	1	2	49	1
114	1	1	1	26	3
123	1	3	3	30	3
126	1	3	4	47	3
133	1	2	1	41	3
140	1	2	3	31	3

	Type.of.apartment	No.of.Credits.at.this.Bank	Occupation	No.of.dependents	Telephone	Foreign.Worker
1	1	1	3	1	1	1
3	1	1	2	1	1	1
4	1	2	2	2	1	2
7	2	2	2	1	1	2
9	2	2	1	1	1	1
12	1	2	3	1	1	1
15	1	2	2	1	1	1
17	2	2	2	2	1	1
18	1	1	3	1	1	1
21	1	3	3	2	1	1
22	2	2	2	2	1	1
25	2	2	2	2	1	1
27	1	1	3	1	1	1
28	2	3	2	1	1	1
32	2	2	3	1	2	1
35	2	1	3	1	1	1
42	2	1	2	1	2	1
43	2	2	3	1	1	1
44	2	2	3	2	2	1

```
47             2                1      3          1      2          1
50             2                2      3          1      2          1
58             2                1      3          1      1          2
60             2                1      3          1      1          1
62             2                1      4          1      2          1
63             2                1      3          1      2          1
66             2                2      3          1      1          1
70             3                2      3          1      2          1
73             2                1      3          1      1          1
75             2                1      4          1      1          1
77             2                1      4          1      2          1
82             2                1      3          1      2          1
86             2                1      3          2      1          1
92             2                1      3          1      1          1
93             2                2      3          1      1          1
97             2                1      2          1      1          1
99             2                1      3          1      1          1
101            2                1      3          1      2          1
102            2                1      2          1      1          1
103            2                2      4          1      2          1
107            2                2      4          1      2          1
109            2                1      3          1      2          1
112            2                1      3          1      2          1
114            2                1      3          1      2          1
123            1                1      4          1      2          1
126            3                3      3          1      2          1
133            2                1      3          1      2          1
140            2                2      3          1      2          1
 [ reached 'max' / getOption("max.print") -- omitted 253 rows ]
>
> #Decision Tree model
> library(rpart)
> mtree <- rpart(Creditability ~ ., data=train, method="class",
+             control=rpart.control(minsplit=20, minbucket=7, maxdepth=10, usesurrogate=2, xval=10))
> #xval = no. of cross validation
> #rpart.control to group multiple parameters
> #method="class" for classification
> #usesurrogate dealing with missing values
> mtree
n= 700

node), split, n, loss, yval, (yprob)
      * denotes terminal node
```

```
 1) root 700 205 1 (0.29285714 0.70714286)
   2) Account.Balance< 2.5 375 168 1 (0.44800000 0.55200000)
     4) Duration.of.Credit..month.>=22.5 160  69 0 (0.56875000 0.43125000)
       8) Value.Savings.Stocks< 3.5 134  50 0 (0.62686567 0.37313433)
        16) Age..years.< 26.5 37   8 0 (0.78378378 0.21621622) *
        17) Age..years.>=26.5 97  42 0 (0.56701031 0.43298969)
          34) Instalment.per.cent>=2.5 66  22 0 (0.66666667 0.33333333) *
          35) Instalment.per.cent< 2.5 31  11 1 (0.35483871 0.64516129) *
       9) Value.Savings.Stocks>=3.5 26   7 1 (0.26923077 0.73076923) *
     5) Duration.of.Credit..month.< 22.5 215  77 1 (0.35813953 0.64186047)
      10) Payment.Status.of.Previous.Credit< 1.5 15   3 0 (0.80000000 0.20000000) *
      11) Payment.Status.of.Previous.Credit>=1.5 200  65 1 (0.32500000 0.67500000)
        22) Guarantors< 1.5 169  62 1 (0.36686391 0.63313609)
          44) Payment.Status.of.Previous.Credit< 2.5 109  50 1 (0.45871560 0.54128440)
            88) Credit.Amount< 971 23   7 0 (0.69565217 0.30434783)
             176) Most.valuable.available.asset>=1.5 13   0 0 (1.00000000 0.00000000) *
             177) Most.valuable.available.asset< 1.5 10   3 1 (0.30000000 0.70000000) *
            89) Credit.Amount>=971 86  34 1 (0.39534884 0.60465116)
             178) Value.Savings.Stocks< 1.5 50  25 0 (0.50000000 0.50000000)
               356) Credit.Amount< 1354.5 15   4 0 (0.73333333 0.26666667) *
               357) Credit.Amount>=1354.5 35  14 1 (0.40000000 0.60000000) *
             179) Value.Savings.Stocks>=1.5 36   9 1 (0.25000000 0.75000000) *
          45) Payment.Status.of.Previous.Credit>=2.5 60  12 1 (0.20000000 0.80000000) *
        23) Guarantors>=1.5 31   3 1 (0.09677419 0.90322581) *
   3) Account.Balance>=2.5 325  37 1 (0.11384615 0.88615385) *
> #Plot tree
> plot(mtree)
> text(mtree)  #Add text to the plot
>
> #Beautify tree
> #install.packages("rattle")
> library(RColorBrewer)
> library(rattle)
> library(rpart.plot)
> #view1
> prp(mtree, faclen=0, cex=0.8, extra=1)
> #faclen = Length of factor level  names in splits
> #cex = text size
> #extra = Number of obs. that fall in the node
> #view2 - total count of each node
> tot_count <- function(x, labs, digits, varlen) {
+   paste(labs, "\n\nn=", x$frame$n)
+ }
> prp(mtree, faclen=0, cex=0.8, node.fun=tot_count)
```

```
> #node.fun - function generates the text at the node labels
> #Pruning
> printcp(mtree)  #Provides optimal pruning based on cp value. Select one with small cross validated
error(xerror).

Classification tree:
rpart(formula = Creditability ~ ., data = train, method = "class",
    control = rpart.control(minsplit = 20, minbucket = 7, maxdepth = 10,
        usesurrogate = 2, xval = 10))

Variables actually used in tree construction:
[1] Account.Balance                Age..years.                    Credit.Amount
Duration.of.Credit..month.
[5] Guarantors                     Instalment.per.cent            Most.valuable.available.asset
Payment.Status.of.Previous.Credit
[9] Value.Savings.Stocks

Root node error: 205/700 = 0.29286

n= 700

        CP nsplit rel error  xerror     xstd
1 0.053659      0   1.00000 1.00000 0.058732
2 0.043902      3   0.83415 1.00000 0.058732
3 0.021951      4   0.79024 1.00000 0.058732
4 0.014634      6   0.74634 0.98537 0.058477
5 0.010000     12   0.64878 0.92683 0.057393
> bestcp <- mtree$cptable[which.min(mtree$cptable[,"xerror"]), "CP"]
> bestcp
[1] 0.01
> #Prune the tree using best cp
> pruned <- prune(mtree, cp=bestcp)
> #Plot pruned tree
> prp(pruned, faclen=0, cex=0.8, extra=1)
>
> #Confusion matrix (training data)
> conf.matrix <- table(train$Creditability, predict(pruned, type="class"))
> rownames(conf.matrix) <- paste("Actual", rownames(conf.matrix), sep = ":")
> colnames(conf.matrix) <- paste("Pred", colnames(conf.matrix), sep = ":")
> print(conf.matrix)

          Pred:0 Pred:1
  Actual:0    109     96
  Actual:1     37    458
```

```
> accuracy_test <- sum(diag(conf.matrix)) /sum(conf.matrix)
> accuracy_test
[1] 0.81
```

Day 10 – R Programming

Iris – Random Forest

```
> #Load dataset
> data("iris")
> #Structure
> 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 ...
> head(iris)
  Sepal.Length Sepal.Width Petal.Length Petal.Width Species
1          5.1         3.5          1.4         0.2  setosa
2          4.9         3.0          1.4         0.2  setosa
3          4.7         3.2          1.3         0.2  setosa
4          4.6         3.1          1.5         0.2  setosa
5          5.0         3.6          1.4         0.2  setosa
6          5.4         3.9          1.7         0.4  setosa
> #Installing packages
> #install.packages("caTools")       #For sampling dataset
> #install.packages("randomForest")  #For implementing Random Forest Algorithm
> #Loading packages
> library(caTools)
> library(randomForest)
>
> #Splitting data in train and test data
> dim(iris)
[1] 150   5
> split <- sample.split(iris, SplitRatio = 0.7)
> split
[1] FALSE FALSE  TRUE  TRUE  TRUE
> train <- subset(iris, split == "TRUE")
> dim(train)
[1] 90  5
> test <- subset(iris, split == "FALSE")
> dim(test)
[1] 60  5
> #Fitting Random Forest to train dataset
```

```
> set.seed(120)
> classifier_RF = randomForest(x=train[-5], y=train$Species, ntree=500)  #First 4 columns as features, Species as dependent variable
> classifier_RF

Call:
 randomForest(x = train[-5], y = train$Species, ntree = 500)
               Type of random forest: classification
                     Number of trees: 500
No. of variables tried at each split: 2

        OOB estimate of  error rate: 5.56%
Confusion matrix:
           setosa versicolor virginica class.error
setosa         30          0         0  0.00000000
versicolor      0         27         3  0.10000000
virginica       0          2        28  0.06666667
>
> #Predicting the Test set results
> y_pred = predict(classifier_RF, newdata=test[-5])
> #Confusion Matrix
> confusion_mtx <- table(test[, 5], y_pred)
> confusion_mtx
            y_pred
             setosa versicolor virginica
  setosa         20          0         0
  versicolor      0         19         1
  virginica       0          1        19
> #Plotting the model
> plot(classifier_RF)
> #Important features
> importance(classifier_RF)
             MeanDecreaseGini
Sepal.Length         4.778206
Sepal.Width          2.705124
Petal.Length        27.798137
Petal.Width         24.011866
> #Variable importance plot
> varImpPlot(classifier_RF)
```

R Programming Exercise:

```
> #1. Execute the following lines which create two vectors of random integers which are chosen with
> #replacement from the integers 0, 1, : : : , 999. Both vectors have length 250.
> set.seed(100)
```

```
> x <- sample(0:999, 250, replace=T)
> y <- sample(0:999, 250, replace=T)
> x
  [1] 713 502 357 623 984 717 918 469 965 515 822 837  97 902   6 182 298 503 465 956 907 994 306
455 145 792 257 434 323  67 509 947 559 287 340 346 166 376 783 970 627 449 965 604 300 669 157
732  86 606 864 222 924 731 250
 [56] 542 693 424 488 296 501 919 170 518 702 448 392 997 659 909 362 845 599 386 877 419 370 882
922 429 954 941 253 964  46 438 942 707  11 946 120  15 951 405 977 948 642 132 555 852 155 947
756 280 553 654 184 297 843 420
[111] 665 489 870 791 395 136 249 362 566 842 702 290 313 537 232  47 254 847 117  36 221 730 657
327  90 583 193 146 862 662 843 793 260 758 333 295 878 848 843 221 999 447 222 693 169 386 741
335 729 421 727 426 722 899 713
[166] 493 817 779 894 315 812 363 804 489 963  99 200 921 282 926 941  70 660 550 987 704 271  81
647 393 708 470 669 209 480 710 457 176 227 129 650  46 113   0 713 799 456 124 632 268 317 394  99
441 909 249 169 915 575 718
[221]  14 275 689 127 748 558 182 401 893 613 472  52 851 522 204 917 598 307 639 934 926 333 228
971 813 984 228 447 491 506
> y
  [1] 658 650 971 659 841 301 337 667 527 324 134 976 695 164 371 246 839 998 969 260 851 875 466
45 115 531 721 823 448 803 505 805 183 554 394 572 402 629 425 525 382 825 877 514 554 475 391
953 331 659 566 208 114 792 490
 [56] 559 681 303 550 219 895 702 518 552 509 762 350  18 176 353 944 129 636 465 693  55 139 382
291 904 128  86 110 396 694 810 296 326 618  75 324 964 980 572 719 953 460 223  37 373 903 684
236  97 162 716 220 902 897 614
[111] 694 232 638 823 584 174 903 922  27  11 106 678 643 118 810 295 751 145 624 358 217 323 400
119 754 908 354 975  22 736 792  84   1 405 326 567 267 128  81 844 945 388 423 983 702  82 288 339
488 129 734 958 295 893 804
[166] 655 639 365 946  20 626 633 401 684 246 886  13 608 556 132 370 399 369 696  25 164 640 186
321 996 173 231 916 348 196 307 591 592 595 955  30 752 885 487 680 515 261 508  78 975 897 287
871 276 697 212 628  12 272 280
[221]  75 730 302 329 617 466 848 536 282 815 986 811 137 697 116 325 247 669 646  97 575 595 759
193 213 470 422 900 473 782
> #(a) Identify out the values in y which are > 500.
> y[y > 500]
  [1] 658 650 971 659 841 667 527 976 695 839 998 969 851 875 531 721 823 803 505 805 554 572 629
525 825 877 514 554 953 659 566 792 559 681 550 895 702 518 552 509 762 944 636 693 904 694 810
618 964 980 572 719 953 903 684
 [56] 716 902 897 614 694 638 823 584 903 922 678 643 810 751 624 754 908 975 736 792 567 844 945
983 702 734 958 893 804 655 639 946 626 633 684 886 608 556 696 640 996 916 591 592 595 955 752
885 680 515 508 975 897 871 697
[111] 628 730 617 848 536 815 986 811 697 669 646 575 595 759 900 782
> #(b) Identify the index positions in y of the values which are > 700?
> which(y > 700)
```

```
 [1]   3   5  12  17  18  19  21  22  27  28  30  32  42  43  48  54  61  62  66  71  80  86  92  93  95  96 101 106
108 109 114 117 118 125 127 135 136 138 140 141 150 151 154 155 161 162 164 165 169 176 190 193
200 202 203
[56] 210 211 213 222 227 230 231 232 243 248 250
> #(c) What are the values in x which are in same index position to the values in y which are > 400?
> y1 <- which(y > 400)
> y1
  [1]   1   2   3   4   5   8   9  12  13  17  18  19  21  22  23  26  27  28  29  30  31  32  34  36  37  38  39  40  42
43  44  45  46  48  50  51  54  55  56  57  59  61  62  63  64  65  66  71  73  74  75  80  85  86  89
 [56]  92  93  94  95  96  97 101 102 106 108 109 110 111 113 114 115 117 118 122 123 125 127 129 135
136 138 140 141 144 146 150 151 153 154 155 159 161 162 164 165 166 167 169 171 172 173 174 176
178 179 184 187 190 193 197
[111] 198 199 200 202 203 204 205 206 208 210 211 213 215 217 222 225 226 227 228 230 231 232 234
238 239 241 242 243 246 247 248 249 250
> x[y1]
  [1] 713 502 357 623 984 469 965 837  97 298 503 465 907 994 306 792 257 434 323  67 509 947 287
346 166 376 783 970 449 965 604 300 669 732 606 864 731 250 542 693 488 501 919 170 518 702 448
362 599 386 877 429  46 438  11
 [56]  15 951 405 977 948 642 155 947 654 297 843 420 665 870 791 395 249 362 290 313 232 254 117
90 583 146 662 843 758 295 221 999 222 693 169 729 727 426 899 713 493 817 894 812 363 804 489
99 921 282 550 271 393 669 457
[111] 176 227 129  46 113   0 713 799 124 268 317  99 909 169 275 748 558 182 401 613 472  52 522
307 639 926 333 228 984 228 447 491 506
> #(d) How many values in y are within 200 of the maximum value of the terms in y?
> count <-length(which(y <= 200))
> count
[1] 48
> #(e) How many numbers in x are divisible by 2?
> n <- length(x[x%%2 == 0])
> n
[1] 119
> #(f) Sort the numbers in the vector x in the order of increasing values in y.
> y2 <- order(y)
> y2
  [1] 143 120 218 177  68 170 139 185 119 201  99  24  76  90 221 209 149 156 142  82 104 240 121  83
53  25 235 124 134  81 148  72 160 180  11 233  77 128 105  14 186 191 116  69  33 188 244 195  52 216
245 131  60 107  98
 [56] 192 112 103  16 175 237  20 207 147 219 214 220 229 212 157  79 126 163  87   6 223  58 196 189
132  10  91 236  88 145 224  49   7 158 194  67  70 137 130 168 183 181  15 100  41  78 152  47  35  84
182 133 173  37 144
[111] 247 153  39  29  97  74  23 226 246 249  46 204 159  55  31 208  65  44 206  63  40   9  26 228  59
64  34  45 179  56  51 146  36  94 241 115 197 198 199 242 178 110 225  89 129 171 217  38 172  73 113
167 187 123 239
```

```
[166]   2 166   1   4  50   8 238 122 205  57 102 174  75  85 111  13 184 215 234  62 155 106  95  27 222
161 140 127 202 135 243  66 250  54 141  30 165  32  86 125 232 230  28 114  42  17   5 150 227  21 213
22  43 203 176
[221] 164  61 109 211 248 108 101 117  80 136 193 118  71 151 169  48  96 200 162  92  19   3 138 210
12  93 154 231 190  18
> x[y2]
  [1] 260 842 915 200 997 315 862 987 566 650 555 455 419 946  14 632 843 386 793 941 280 934 702
253 924 145 204 537 327 954 848 845 421 926 822 851 370 847 553 902 704 708 136 659 559  81 971
480 222 249 813 221 296 184 132
 [56] 470 489 756 182 963 598 956 456 878 575 441 718 893 394 741 922  47 722 942 717 689 424 710
647 730 515 120 917 707 333 127  86 918 335 209 392 909 193  36 779 660 941   6 852 627 882 447 157
340 964  70 657 804 166 758
[111] 228 222 783 323 642 386 306 558 984 491 669   0 729 250 509 124 702 604 799 170 970 965 792
401 488 518 287 300 282 542 864 295 346 405 926 395 457 176 227 333 921 420 748  11 117 812 169
376 363 599 870 817 271 313 639
[166] 502 493 713 623 606 469 307 290 713 693 947 489 877  46 665  97 550 909 522 919 169 654 977
257 275 727 662 254  46  90 228 448 506 731 843  67 713 947 438 232  52 613 434 791 449 298 984 221
182 907  99 994 965 113  99
[221] 899 501 843 317 447 297 155 249 429 583 669 362 362 999 894 732 948 129 426  15 465 357 146
268 837 951 693 472 393 503
>
> #2. Use the function paste to create the following character vectors of length 30:
> #(a) ("Label 1", "Label 2", ....., "Label 30").
> #*Note that there is a single space between label and the number following.
> v <- c(1:30)
> v
 [1]  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30
> paste("Label", v)
 [1] "Label 1"  "Label 2"  "Label 3"  "Label 4"  "Label 5"  "Label 6"  "Label 7"  "Label 8"  "Label 9"  "Label
10" "Label 11" "Label 12" "Label 13" "Label 14" "Label 15" "Label 16" "Label 17" "Label 18" "Label 19"
"Label 20"
[21] "Label 21" "Label 22" "Label 23" "Label 24" "Label 25" "Label 26" "Label 27" "Label 28" "Label 29"
"Label 30"
> #(b) ("FN1", "FN2", ..., "FN30").
> #**In this case, there is no space between fn and the number following.
> paste("FN", v, sep="")
 [1] "FN1"  "FN2"  "FN3"  "FN4"  "FN5"  "FN6"  "FN7"  "FN8"  "FN9"  "FN10" "FN11" "FN12" "FN13"
"FN14" "FN15" "FN16" "FN17" "FN18" "FN19" "FN20" "FN21" "FN22" "FN23" "FN24" "FN25" "FN26"
"FN27" "FN28" "FN29" "FN30"
>
> #3. Compound interest can be computed using the formula
> #A = P × (1 + R/100)n, where P is the original money lent, A is what it amounts to in n years at R
> #percent per year interest.
```

```
> #Write R code to calculate the amount of money owed after n years, where n changes from 1 to 15 in yearly increments, if the money lent originally is 10000 Rupees and the interest rate remains constant throughout the period at 11.5%.
> P <- 10000
> R <- 11.5
> n <- 1
> for(i in 1:15) {
+   A <- P * (1 + (R / 100)) * n
+   P <- A
+   cat("For ", n, " year(s), A = ", A, "\n")
+ }
For  1  year(s), A =  11150
For  1  year(s), A =  12432.25
For  1  year(s), A =  13861.96
For  1  year(s), A =  15456.08
For  1  year(s), A =  17233.53
For  1  year(s), A =  19215.39
For  1  year(s), A =  21425.16
For  1  year(s), A =  23889.05
For  1  year(s), A =  26636.29
For  1  year(s), A =  29699.47
For  1  year(s), A =  33114.91
For  1  year(s), A =  36923.12
For  1  year(s), A =  41169.28
For  1  year(s), A =  45903.75
For  1  year(s), A =  51182.68
>
> #4. Generate the following matrices.
> #[,1] [,2] [,3] [,4]
> #[1,]       1 101 201 301
> #[2,]       2 102 202 302
> #[3,]       3 103 203 303
> #[4,]       4 104 204 304
> #[5,]       5 105 205 305
> v <- c(1:5, 101:105, 201:205, 301:305)
> v
 [1]   1   2   3   4   5 101 102 103 104 105 201 202 203 204 205 301 302 303 304 305
> matrix(v, nrow = 5)
     [,1] [,2] [,3] [,4]
[1,]    1  101  201  301
[2,]    2  102  202  302
[3,]    3  103  203  303
[4,]    4  104  204  304
[5,]    5  105  205  305
```

```
>
> #5. Create a 6 by 10 matrix of random integers chosen from 1 to 10 by executing the following two
lines of code:
> set.seed(100)
> GMAT <- matrix(sample(10, size=60, replace=T), nr=6)
> GMAT
     [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10]
[1,]   10    7    2    3    3    9    6    7    3     7
[2,]    7    6    7    3    4    4    9    1    4     4
[3,]    6    6    7    8    4    2    9    9    3     3
[4,]    3    4    7    2    4    6    9    6    3     9
[5,]    9    7    8    9    5    7    6    4    4     8
[6,]   10    6    2    2    7    1    8    8    5     6
> #(a) Find the number of entries in each row which are greater than 4.
> apply(GMAT, 1, function(x) { sum(x > 4) })
[1] 6 4 6 5 8 7
> #(b) Which rows contain exactly two occurrences of the number seven?
> which(apply(GMAT, 1, function(x) { sum(x == 7) == 2 }))
[1] 2 5
> #(c) Find those pairs of columns whose total (over both columns) is >= 50. The answer should be a
matrix with two columns.
> n <- ncol(GMAT) - 1
> n
[1] 9
> m <- matrix(ncol=2)
> s <- sapply(1:n, function(x) {
+   if(sum(GMAT[,x]) + sum(GMAT[,x + 1]) >= 50) {
+     c(x, x + 1)
+
+   }
+
+ })
> t(s)
      [,1] [,2]
 [1,]    1    2
 [2,]    2    3
 [3,]    3    4
 [4,]    4    5
 [5,]    5    6
 [6,]    6    7
 [7,]    7    8
 [8,]    8    9
 [9,]    9   10
```

Day 11 – R Programming

Iris – KNearestNeighbors

```
> #Load data
> df <- iris
> head(df)
  Sepal.Length Sepal.Width Petal.Length Petal.Width Species
1          5.1         3.5          1.4         0.2  setosa
2          4.9         3.0          1.4         0.2  setosa
3          4.7         3.2          1.3         0.2  setosa
4          4.6         3.1          1.5         0.2  setosa
5          5.0         3.6          1.4         0.2  setosa
6          5.4         3.9          1.7         0.4  setosa
> str(df)
'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 ...
> dim(df)
[1] 150   5
>
> #Generate a random number that is 90% of the total no. of rows in dataset
> ran <- sample(1:nrow(df), 0.9 * nrow(df))
> ran
  [1]  75  46  51  60  62 117  55 114  22  12 115  44 100  61 123  66  19  77 103  94  56 108 107  83 102 130
 [27]  26 129  45  38  10  40 120 121  91 148  29 131  87  54  85 135 112  64 134 106 133  33 145 128  37
71
 [53] 124  81  47 138  32 122 119  36 125  50  68  92  28  90 111 144 116 149  86 141  11  30  78  17  23
84
 [79]   4  70  41  48   7 140  24 101 147  16  65  96  18 143  49  25  63   5  74   3   6  57 105  76  43  79
[105] 132   8  88  15  95  35 139  13  20   9 137  14  80  42 113  67  73  72  89  97   2 109 150  82  99 118
[131] 136 110  98  21  53
> #Normalizing data
> nor <- function(x) { (x - min(x)) / (max(x) - min(x)) }
> iris_norm <- as.data.frame(lapply(df[, c(1, 2, 3, 4)], nor))   #Appling normalization function on
predictors i.e. first 4 columns
> summary(iris_norm)
  Sepal.Length     Sepal.Width      Petal.Length     Petal.Width
 Min.   :0.0000   Min.   :0.0000   Min.   :0.0000   Min.   :0.00000
 1st Qu.:0.2222   1st Qu.:0.3333   1st Qu.:0.1017   1st Qu.:0.08333
 Median :0.4167   Median :0.4167   Median :0.5678   Median :0.50000
 Mean   :0.4287   Mean   :0.4406   Mean   :0.4675   Mean   :0.45806
```

```
 3rd Qu.:0.5833   3rd Qu.:0.5417   3rd Qu.:0.6949   3rd Qu.:0.70833
 Max.   :1.0000   Max.   :1.0000   Max.   :1.0000   Max.   :1.00000
> head(iris_norm)
  Sepal.Length Sepal.Width Petal.Length Petal.Width
1   0.22222222   0.6250000   0.06779661  0.04166667
2   0.16666667   0.4166667   0.06779661  0.04166667
3   0.11111111   0.5000000   0.05084746  0.04166667
4   0.08333333   0.4583333   0.08474576  0.04166667
5   0.19444444   0.6666667   0.06779661  0.04166667
6   0.30555556   0.7916667   0.11864407  0.12500000
>
> #Extract Training data
> iris_train = iris_norm[ran,]
> dim(iris_train)
[1] 135   4
> head(iris_train)
    Sepal.Length Sepal.Width Petal.Length Petal.Width
75     0.5833333   0.3750000   0.55932203  0.50000000
46     0.1388889   0.4166667   0.06779661  0.08333333
51     0.7500000   0.5000000   0.62711864  0.54166667
60     0.2500000   0.2916667   0.49152542  0.54166667
62     0.4444444   0.4166667   0.54237288  0.58333333
117    0.6111111   0.4166667   0.76271186  0.70833333
> #Extract Test data
> iris_test = iris_norm[-ran,]
> dim(iris_test)
[1] 15  4
> head(iris_test)
   Sepal.Length Sepal.Width Petal.Length Petal.Width
1    0.22222222   0.6250000   0.06779661  0.04166667
27   0.19444444   0.5833333   0.10169492  0.12500000
31   0.13888889   0.4583333   0.10169492  0.04166667
34   0.33333333   0.9166667   0.06779661  0.04166667
39   0.02777778   0.4166667   0.05084746  0.04166667
52   0.58333333   0.5000000   0.59322034  0.58333333
> #Extract dependent variable of train dataset
> iris_target_category <- df[ran, 5]
> head(iris_target_category)
[1] versicolor setosa     versicolor versicolor versicolor virginica
Levels: setosa versicolor virginica
> #Extract dependent variable of test dataset
> iris_test_category <- df[-ran, 5]
> head(iris_test_category)
[1] setosa     setosa     setosa     setosa     setosa     versicolor
```

```
Levels: setosa versicolor virginica
> library(class)
> #Run KNN function
> pr <- knn(iris_train, iris_test, cl=iris_target_category, k=13)
> pr
 [1] setosa     setosa     setosa     setosa     setosa     versicolor versicolor versicolor versicolor
[10] versicolor virginica  virginica  virginica  virginica  virginica
Levels: setosa versicolor virginica
> #Create confusion matrix
> tab <- table(pr, iris_test_category)
> tab
            iris_test_category
pr           setosa versicolor virginica
  setosa          5          0         0
  versicolor      0          5         0
  virginica       0          0         5
> #Accuracy score
> accuracy <- function(x) { sum(diag(x) / sum(rowSums(x))) * 100 }
> accuracy(tab)
[1] 100
```

Day 12 – R Programming

Social Ads Marketing – SVM

```
setwd("C:/zubeda/PGA02_Zubu/R Programming/Models")
> #Importing Dataset
> dataset <- read.csv("Social_Network_Ads.csv")
> head(dataset)
   User.ID Gender Age EstimatedSalary Purchased
1 15624510   Male  19           19000         0
2 15810944   Male  35           20000         0
3 15668575 Female  26           43000         0
4 15603246 Female  27           57000         0
5 15804002   Male  19           76000         0
6 15728773   Male  27           58000         0
> str(dataset)
'data.frame':	400 obs. of  5 variables:
 $ User.ID        : int  15624510 15810944 15668575 15603246 15804002 15728773 15598044 15694829
15600575 15727311 ...
 $ Gender         : chr  "Male" "Male" "Female" "Female" ...
 $ Age            : int  19 35 26 27 19 27 27 32 25 35 ...
 $ EstimatedSalary: int  19000 20000 43000 57000 76000 58000 84000 150000 33000 65000 ...
 $ Purchased      : int  0 0 0 0 0 0 0 1 0 0 ...
> dim(dataset)
[1] 400   5
```

```
> dataset <- dataset[3:5]     #User.ID and Gender are not considered
> head(dataset)
  Age EstimatedSalary Purchased
1  19           19000         0
2  35           20000         0
3  26           43000         0
4  27           57000         0
5  19           76000         0
6  27           58000         0
> #Encoding target feature by factorizing
> dataset$Purchased <- factor(dataset$Purchased, labels=c(0, 1))
> class(dataset$Purchased)
[1] "factor"
>
> #Splitting the dataset
> library(caTools)
> set.seed(123)
> split <- sample.split(dataset$Purchased, SplitRatio=0.75)
> split
  [1]  TRUE FALSE  TRUE FALSE FALSE  TRUE  TRUE  TRUE FALSE  TRUE  TRUE FALSE  TRUE  TRUE  TRUE
TRUE  TRUE FALSE FALSE FALSE
 [21]  TRUE FALSE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE FALSE  TRUE  TRUE FALSE  TRUE FALSE FALSE
TRUE  TRUE FALSE  TRUE  TRUE
 [41]  TRUE  TRUE  TRUE  TRUE FALSE FALSE  TRUE FALSE  TRUE  TRUE  TRUE FALSE  TRUE  TRUE  TRUE
TRUE  TRUE  TRUE  TRUE  TRUE
 [61]  TRUE  TRUE  TRUE  TRUE  TRUE FALSE  TRUE  TRUE FALSE  TRUE  TRUE  TRUE  TRUE FALSE FALSE
TRUE  TRUE  TRUE  TRUE  TRUE
 [81]  TRUE FALSE  TRUE FALSE FALSE FALSE FALSE  TRUE FALSE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
TRUE  TRUE  TRUE  TRUE  TRUE
[101]  TRUE  TRUE FALSE FALSE  TRUE  TRUE FALSE FALSE FALSE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
TRUE FALSE  TRUE  TRUE  TRUE
[121]  TRUE  TRUE  TRUE FALSE  TRUE FALSE FALSE  TRUE  TRUE  TRUE FALSE  TRUE  TRUE FALSE  TRUE
TRUE  TRUE  TRUE FALSE  TRUE
[141]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE FALSE  TRUE  TRUE  TRUE  TRUE  TRUE FALSE  TRUE
FALSE  TRUE  TRUE FALSE  TRUE
[161]  TRUE FALSE FALSE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE FALSE  TRUE  TRUE  TRUE  TRUE FALSE
FALSE  TRUE  TRUE  TRUE  TRUE
[181]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE FALSE  TRUE  TRUE
TRUE  TRUE  TRUE FALSE FALSE
[201]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE FALSE  TRUE  TRUE  TRUE  TRUE FALSE  TRUE  TRUE
TRUE  TRUE  TRUE  TRUE  TRUE
[221]  TRUE  TRUE  TRUE FALSE  TRUE FALSE  TRUE FALSE FALSE FALSE  TRUE  TRUE  TRUE FALSE  TRUE
FALSE FALSE  TRUE FALSE  TRUE
```

```
[241] FALSE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE FALSE
TRUE  TRUE  TRUE  TRUE  TRUE
[261]  TRUE  TRUE  TRUE FALSE FALSE FALSE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE FALSE FALSE  TRUE
TRUE  TRUE  TRUE  TRUE  TRUE
[281] FALSE  TRUE  TRUE  TRUE  TRUE FALSE  TRUE  TRUE  TRUE  TRUE  TRUE FALSE  TRUE  TRUE  TRUE
TRUE  TRUE  TRUE FALSE  TRUE
[301]  TRUE FALSE  TRUE  TRUE FALSE  TRUE FALSE  TRUE  TRUE FALSE  TRUE  TRUE  TRUE  TRUE  TRUE
FALSE  TRUE  TRUE  TRUE  TRUE
[321]  TRUE  TRUE  TRUE FALSE  TRUE FALSE  TRUE  TRUE  TRUE  TRUE  TRUE FALSE  TRUE  TRUE  TRUE
TRUE  TRUE  TRUE FALSE  TRUE
[341] FALSE  TRUE FALSE  TRUE  TRUE  TRUE FALSE  TRUE  TRUE  TRUE  TRUE  TRUE FALSE  TRUE  TRUE
TRUE  TRUE  TRUE  TRUE  TRUE
[361]  TRUE  TRUE FALSE FALSE  TRUE  TRUE FALSE FALSE FALSE  TRUE  TRUE FALSE FALSE  TRUE  TRUE
TRUE  TRUE  TRUE  TRUE FALSE
[381]  TRUE  TRUE FALSE  TRUE  TRUE  TRUE  TRUE  TRUE FALSE  TRUE  TRUE FALSE  TRUE  TRUE FALSE
TRUE  TRUE  TRUE  TRUE FALSE
> training_set <- subset(dataset, split == TRUE)
> test_set <- subset(dataset, split == FALSE)
> dim(training_set)
[1] 300   3
> dim(test_set)
[1] 100   3
> #Feature scaling
> training_set[-3] <- scale(training_set[-3])  #Except target feature, scale all the features
> test_set[-3] <- scale(test_set[-3])
> head(training_set)
          Age EstimatedSalary Purchased
1  -1.7655475      -1.4733414         0
3  -1.0962966      -0.7883761         0
6  -1.0006894      -0.3602727         0
7  -1.0006894       0.3817730         0
8  -0.5226531       2.2654277         1
10 -0.2358313      -0.1604912         0
> head(test_set)
          Age EstimatedSalary Purchased
2  -0.3041906      -1.5135434         0
4  -1.0599437      -0.3245603         0
5  -1.8156969       0.2859986         0
9  -1.2488820      -1.0957926         0
12 -1.1544129      -0.4852337         0
18  0.6405008      -1.3207353         1
> #Fitting SVM to training set
> library(e1071)
> classifier <- svm(formula=Purchased ~ ., data=training_set, type="C-classification", kernel="linear")
```

```
> #Predicting the test set result
> y_pred <- predict(classifier, newdata=test_set[-3])
> y_pred
  2   4   5   9  12  18  19  20  22  29  32  34  35  38  45  46  48  52  66  69  74  75  82  84  85  86  87  89 103
104 107
  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   1   0
108 109 117 124 126 127 131 134 139 148 154 156 159 162 163 170 175 176 193 199 200 208 213 224
226 228 229 230 234 236 237
  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   1   1   1   0   1   0   1   1   1   0
239 241 255 264 265 266 273 274 281 286 292 299 302 305 307 310 316 324 326 332 339 341 343 347
353 363 364 367 368 369 372
  1   1   1   0   1   1   1   1   1   0   1   1   1   0   1   0   0   0   0   1   0   1   0   1   1   0   1   1   1   0   1
373 380 383 389 392 395 400
  0   1   1   0   0   0   0
Levels: 0 1
> #Making confusion matrix
> cm <- table(test_set[, 3], y_pred)
> cm
   y_pred
     0  1
  0 57  7
  1 13 23
> #Visualizing results
> # Download package tarball from CRAN archive
> # Download package tarball from CRAN archive
> url <- "https://cran.r-
project.org/src/contrib/Archive/ElemStatLearn/ElemStatLearn_2015.6.26.2.tar.gz"
> pkgFile <- "ElemStatLearn_2015.6.26.2.tar.gz"
> download.file(url = url, destfile = pkgFile)
trying URL 'https://cran.r-
project.org/src/contrib/Archive/ElemStatLearn/ElemStatLearn_2015.6.26.2.tar.gz'
Content type 'application/x-gzip' length 12169918 bytes (11.6 MB)
downloaded 11.6 MB

> # Install package
> install.packages(pkgs=pkgFile, type="source", repos=NULL)
WARNING: Rtools is required to build R packages but is not currently installed. Please download and
install the appropriate version of Rtools before proceeding:

https://cran.rstudio.com/bin/windows/Rtools/
Installing package into ‘C:/Users/ashraf/Documents/R/win-library/4.1’
(as ‘lib’ is unspecified)
* installing *source* package 'ElemStatLearn' ...
** package 'ElemStatLearn' successfully unpacked and MD5 sums checked
```

```
** using staged installation
** R
** data
*** moving datasets to lazyload DB
** byte-compile and prepare package for lazy loading
** help
*** installing help indices
  converting help for package 'ElemStatLearn'
    finding HTML links ... done
    SAheart                                 html
    bone                                    html
    countries                               html
    galaxy                                  html
    marketing                               html
    mixture.example                         html
    nci                                     html
    orange10.test                           html
    orange10.train                          html
    orange4.test                            html
    orange4.train                           html
    ozone                                   html
    phoneme                                 html
    prostate                                html
    simple.ridge                            html
    spam                                    html
    vowel.test                              html
    vowel.train                             html
    waveform                                html
    waveform.test                           html
    waveform.train                          html
    zip.test                                html
    zip.train                               html
    zip2image                               html
** building package indices
** testing if installed package can be loaded from temporary location
*** arch - i386
*** arch - x64
** testing if installed package can be loaded from final location
*** arch - i386
*** arch - x64
** testing if installed package keeps a record of temporary installation path
* DONE (ElemStatLearn)
> # Delete package tarball
> unlink(pkgFile)
```

```
> library(ElemStatLearn)
> # Plotting the training data set results
> set = training_set
> X1 = seq(min(set[, 1]) - 1, max(set[, 1]) + 1, by = 0.01)
> X2 = seq(min(set[, 2]) - 1, max(set[, 2]) + 1, by = 0.01)
>
> grid_set = expand.grid(X1, X2)
> #expand.grid() - Create a data frame from all combinations of the supplied vectors or factors.
> colnames(grid_set) = c('Age', 'EstimatedSalary')
> y_grid = predict(classifier, newdata = grid_set)
>
> plot(set[, -3],
+     main = 'SVM (Training set)',
+     xlab = 'Age', ylab = 'Estimated Salary',
+     xlim = range(X1), ylim = range(X2))
>
> contour(X1, X2, matrix(as.numeric(y_grid), length(X1), length(X2)), add = TRUE)
>
> points(grid_set, pch = '.', col = ifelse(y_grid == 1, 'coral1', 'aquamarine'))
>
> points(set, pch = 21, bg = ifelse(set[, 3] == 1, 'green4', 'red3'))
> #Plotting the test set results
> set = test_set
> X1 = seq(min(set[, 1]) - 1, max(set[, 1]) + 1, by = 0.01)
> X2 = seq(min(set[, 2]) - 1, max(set[, 2]) + 1, by = 0.01)
>
> grid_set = expand.grid(X1, X2)
> colnames(grid_set) = c('Age', 'EstimatedSalary')
> y_grid = predict(classifier, newdata = grid_set)
>
> plot(set[, -3], main = 'SVM (Test set)',
+     xlab = 'Age', ylab = 'Estimated Salary',
+     xlim = range(X1), ylim = range(X2))
>
> contour(X1, X2, matrix(as.numeric(y_grid), length(X1), length(X2)), add = TRUE)
>
> points(grid_set, pch = '.', col = ifelse(y_grid == 1, 'coral1', 'aquamarine'))
>
> points(set, pch = 21, bg = ifelse(set[, 3] == 1, 'green4', 'red3'))
```

Chemical Classification – SVM

```
> setwd("C:/zubeda/PGA02_Zubu/R Programming/Models")
> #Import dataset
> biodeg <- read.csv("biodeg.csv", sep=";")
```

```
> head(biodeg)
  X3.919 X2.6909 X0 X0.1 X0.2 X0.3 X0.4 X31.4 X2 X0.5 X0.6   X0.7 X3.106 X2.55 X9.002 X0.8 X0.96 X1.142
X0.9 X0.10 X0.11 X1.201 X0.12 X0.13 X0.14 X0.15 X1.932
1  4.170  2.1144  0    0    0    0    0  30.8  1    1    0  0.000  2.461 1.393  8.723    1 0.989  1.144    0    0    0
1.104    1    0    0    0  2.214
2  3.932  3.2512  0    0    0    0    0  26.7  2    4    0  0.000  3.279 2.585  9.110    0 1.009  1.152    0    0    0
1.092    0    0    0    0  1.942
3  3.000  2.7098  0    0    0    0    0  20.0  0    2    0  0.000  2.100 0.918  6.594    0 1.108  1.167    0    0    0
1.024    0    0    0    0  1.414
4  4.236  3.3944  0    0    0    0    0  29.4  2    4    0 -0.271  3.449 2.753  9.528    2 1.004  1.147    0    0    0
1.137    0    0    0    0  1.985
5  4.236  3.4286  0    0    0    0    0  28.6  2    4    0 -0.275  3.313 2.522  9.383    1 1.014  1.149    0    0    0
1.119    0    0    0    0  1.980
6  5.000  5.0476  1    0    0    0    0  11.1  0    3    0  0.000  2.872 0.722  9.657    0 1.092  1.153    0    0    0
1.125    0    0    0    0  2.000
  X0.011 X0.16  X0.17 X4.489 X0.18 X0.19 X0.20 X0.21 X2.949 X1.591 X0.22 X7.253 X0.23 X0.24 RB
1 -0.204     0  0.000  1.542     0     0     0     0  3.315  1.967     0  7.257     0     0 RB
2 -0.008     0  0.000  4.891     0     0     0     1  3.076  2.417     0  7.601     0     0 RB
3  1.073     0  8.361  1.333     0     0     0     1  3.046  5.000     0  6.690     0     0 RB
4 -0.002     0 10.348  5.588     0     0     0     0  3.351  2.405     0  8.003     0     0 RB
5 -0.008     0 10.276  4.746     0     0     0     0  3.351  2.556     0  7.904     0     0 RB
6  0.446     0 18.375  0.800     0     0     0     1  4.712  4.583     0  9.303     0     0 RB
> str(biodeg)
'data.frame':	1054 obs. of  42 variables:
 $ X3.919 : num  4.17 3.93 3 4.24 4.24 ...
 $ X2.6909: num  2.11 3.25 2.71 3.39 3.43 ...
 $ X0     : int  0 0 0 0 0 1 0 0 0 0 ...
 $ X0.1   : int  0 0 0 0 0 0 0 0 0 0 ...
 $ X0.2   : int  0 0 0 0 0 0 0 0 1 0 ...
 $ X0.3   : int  0 0 0 0 0 0 0 0 0 0 ...
 $ X0.4   : int  0 0 0 0 0 0 0 2 2 2 ...
 $ X31.4  : num  30.8 26.7 20 29.4 28.6 11.1 31.6 44.4 41.2 52.9 ...
 $ X2     : int  1 2 0 2 2 0 3 2 0 0 ...
 $ X0.5   : int  1 4 2 4 4 3 2 0 4 2 ...
 $ X0.6   : int  0 0 0 0 0 0 0 0 3 0 ...
 $ X0.7   : num  0 0 0 -0.271 -0.275 0 -0.039 0 -1.29 -0.302 ...
 $ X3.106 : num  2.46 3.28 2.1 3.45 3.31 ...
 $ X2.55  : num  1.393 2.585 0.918 2.753 2.522 ...
 $ X9.002 : num  8.72 9.11 6.59 9.53 9.38 ...
 $ X0.8   : int  1 0 0 2 1 0 5 0 8 5 ...
 $ X0.96  : num  0.989 1.009 1.108 1.004 1.014 ...
 $ X1.142 : num  1.14 1.15 1.17 1.15 1.15 ...
 $ X0.9   : int  0 0 0 0 0 0 0 0 0 0 ...
 $ X0.10  : int  0 0 0 0 0 0 0 0 1 0 ...
```

```
 $ X0.11  : int  0 0 0 0 0 0 0 0 0 0 ...
 $ X1.201 : num  1.1 1.09 1.02 1.14 1.12 ...
 $ X0.12  : int  1 0 0 0 0 0 0 1 1 3 ...
 $ X0.13  : int  0 0 0 0 0 0 0 0 0 0 ...
 $ X0.14  : int  0 0 0 0 0 0 0 0 0 0 ...
 $ X0.15  : int  0 0 0 0 0 0 0 0 0 0 ...
 $ X1.932 : num  2.21 1.94 1.41 1.99 1.98 ...
 $ X0.011 : num  -0.204 -0.008 1.073 -0.002 -0.008 ...
 $ X0.16  : int  0 0 0 0 0 0 0 0 0 0 ...
 $ X0.17  : num  0 0 8.36 10.35 10.28 ...
 $ X4.489 : num  1.54 4.89 1.33 5.59 4.75 ...
 $ X0.18  : int  0 0 0 0 0 0 0 0 0 0 ...
 $ X0.19  : int  0 0 0 0 0 0 0 0 1 1 ...
 $ X0.20  : int  0 0 0 0 0 0 0 0 2 0 ...
 $ X0.21  : int  0 1 1 0 0 1 0 0 1 0 ...
 $ X2.949 : num  3.31 3.08 3.05 3.35 3.35 ...
 $ X1.591 : num  1.97 2.42 5 2.4 2.56 ...
 $ X0.22  : int  0 0 0 0 0 0 0 0 1 0 ...
 $ X7.253 : num  7.26 7.6 6.69 8 7.9 ...
 $ X0.23  : int  0 0 0 0 0 0 0 0 0 0 ...
 $ X0.24  : int  0 0 0 0 0 0 0 0 0 0 ...
 $ RB     : chr  "RB" "RB" "RB" "RB" ...
> dim(biodeg)
[1] 1054   42
> sum(is.na(biodeg))
[1] 0
> #Data Pre-processing
> biodeg$RB <- ifelse(biodeg$RB == "RB", 1, 0)
> class(biodeg$RB)
[1] "numeric"
> biodeg$RB <- factor(biodeg$RB, labels=c(1, 0))
> class(biodeg$RB)
[1] "factor"
> #Feature Selection
> biodeg[-42] <- scale(biodeg[-42])
> head(biodeg)
     X3.919    X2.6909        X0       X0.1       X0.2       X0.3       X0.4      X31.4         X2       X0.5       X0.6       X0.7
X3.106      X2.55
1 -1.1224746 -1.1489213 -0.4902786 -0.1666917 -0.4203431 -0.2702577 -0.7405779 -0.6844765 -
0.1912642 -0.4537005 -0.4613371  0.25625322 -1.7391211  0.05526774
2 -1.5579514  0.2180477 -0.4902786 -0.1666917 -0.4203431 -0.2702577 -0.7405779 -1.1327038
0.3178077  1.2360533 -0.4613371  0.25625322 -0.3391925  1.57244420
3 -3.2632642 -0.4329701 -0.4902786 -0.1666917 -0.4203431 -0.2702577 -0.7405779 -1.8651727 -
0.7003361  0.1095508 -0.4613371  0.25625322 -2.3569379 -0.54931181
```

```
4 -1.0017121  0.3902415 -0.4902786 -0.1666917 -0.4203431 -0.2702577 -0.7405779 -0.8375297
0.3178077  1.2360533 -0.4613371 -0.09569312 -0.0482538  1.78627444
5 -1.0017121  0.4313660 -0.4902786 -0.1666917 -0.4203431 -0.2702577 -0.7405779 -0.9249887
0.3178077  1.2360533 -0.4613371 -0.10088790 -0.2810048  1.49225786
6  0.3962053  2.3781666  0.1932580 -0.1666917 -0.4203431 -0.2702577 -0.7405779 -2.8381539 -
0.7003361  0.6728020 -0.4613371  0.25625322 -1.0357340 -0.79878043
    X9.002     X0.8     X0.96   X1.142     X0.9     X0.10     X0.11   X1.201     X0.12     X0.13     X0.14
X0.15     X1.932     X0.011
1 -1.3086107 -0.5908038 -0.52386167 0.4279226 -0.08383824 -0.2329964 -0.1348551 -1.396441 -
0.08496205 -0.2036237 -0.4165988 -0.156832 -0.008448142 -1.275335545
2 -0.8918860 -0.8151220 -0.09363711 0.6932151 -0.08383824 -0.2329964 -0.1348551 -1.520787 -
0.29370332 -0.2036237 -0.4165988 -0.156832 -1.211617959 -0.042657773
3 -3.6011346 -0.8151220  2.03597445 1.1906385 -0.08383824 -0.2329964 -0.1348551 -2.225418 -
0.29370332 -0.2036237 -0.4165988 -0.156832 -3.547182898  6.755937489
4 -0.4417803 -0.3664857 -0.20119325 0.5274073 -0.08383824 -0.2329964 -0.1348551 -1.054488 -
0.29370332 -0.2036237 -0.4165988 -0.156832 -1.021410966 -0.004922739
5 -0.5979175 -0.5908038  0.01391903 0.5937304 -0.08383824 -0.2329964 -0.1348551 -1.241008 -
0.29370332 -0.2036237 -0.4165988 -0.156832 -1.043528058 -0.042657773
6 -0.3028721 -0.8151220  1.69179481 0.7263766 -0.08383824 -0.2329964 -0.1348551 -1.178834 -
0.29370332 -0.2036237 -0.4165988 -0.156832 -0.955059689  2.812626453
     X0.16     X0.17    X4.489     X0.18     X0.19     X0.20     X0.21    X2.949     X1.591     X0.22    X7.253
X0.23     X0.24 RB
1 -0.1651198 -0.73865380 -0.5363359 -0.2017241 -0.5813821 -0.5609848 -0.7651948 -0.6043829 -
0.922097187 -0.6297848 -1.1062502 -0.160547 -0.3231392  0
2 -0.1651198 -0.73865380  1.0608207 -0.2017241 -0.5813821 -0.5609848  0.0301852 -0.8434713 -
0.221575026 -0.6297848 -0.8292443 -0.160547 -0.3231392  0
3 -0.1651198 -0.03595765 -0.6360092 -0.2017241 -0.5813821 -0.5609848  0.0301852 -0.8734824
3.799422176 -0.6297848 -1.5628267 -0.160547 -0.3231392  0
4 -0.1651198  0.13103879  1.3932238 -0.2017241 -0.5813821 -0.5609848 -0.7651948 -0.5683696 -
0.240255617 -0.6297848 -0.5055340 -0.160547 -0.3231392  0
5 -0.1651198  0.12498758  0.9916694 -0.2017241 -0.5813821 -0.5609848 -0.7651948 -0.5683696 -
0.005191515 -0.6297848 -0.5852537 -0.160547 -0.3231392  0
6 -0.1651198  0.80566407 -0.8901998 -0.2017241 -0.5813821 -0.5609848  0.0301852  0.7931339
3.150271640 -0.6297848  0.5412904 -0.160547 -0.3231392  0
> #install.packages("mlbench")
> library(mlbench)
> library(caret)
> control <- trainControl(method="repeatedcv", number=15, repeats=3)
> # train the model
> set.seed(111)
> model <- train(RB~., data=biodeg, method="lvq", trControl=control)
> # estimate variable importance
> importance <- varImp(model, scale=FALSE)
> # summarize importance
```

```
> print(importance)
ROC curve variable importance

  only 20 most important variables shown (out of 41)

       Importance
X2.949     0.7981
X7.253     0.7737
X3.919     0.7658
X1.932     0.7632
X1.201     0.7604
X9.002     0.7283
X3.106     0.7128
X0.4       0.7034
X0.12      0.7004
X0.19      0.6911
X0         0.6778
X2.55      0.6662
X1.142     0.6590
X0.22      0.6531
X0.6       0.6519
X0.20      0.6496
X0.2       0.6491
X0.24      0.6397
X0.5       0.6308
X31.4      0.6283
> ImpMeasure <- data.frame(importance$importance)
> ImpMeasure<- ImpMeasure[order(-ImpMeasure$X1, -ImpMeasure$X0),]
> ImpMeasure
              X1        X0
X2.949  0.7981382 0.7981382
X7.253  0.7736706 0.7736706
X3.919  0.7658184 0.7658184
X1.932  0.7631969 0.7631969
X1.201  0.7604324 0.7604324
X9.002  0.7283463 0.7283463
X3.106  0.7128292 0.7128292
X0.4    0.7033912 0.7033912
X0.12   0.7003627 0.7003627
X0.19   0.6911181 0.6911181
X0      0.6777852 0.6777852
X2.55   0.6662254 0.6662254
X1.142  0.6589957 0.6589957
X0.22   0.6530597 0.6530597
```

```
X0.6    0.6519213 0.6519213
X0.20   0.6496444 0.6496444
X0.2    0.6490782 0.6490782
X0.24   0.6396845 0.6396845
X0.5    0.6308046 0.6308046
X31.4   0.6283262 0.6283262
X0.7    0.6192085 0.6192085
X0.14   0.5945979 0.5945979
X0.3    0.5860767 0.5860767
X1.591  0.5830865 0.5830865
X0.17   0.5751677 0.5751677
X4.489  0.5736364 0.5736364
X0.8    0.5432731 0.5432731
X0.10   0.5408692 0.5408692
X2.6909 0.5275182 0.5275182
X2      0.5264926 0.5264926
X0.23   0.5253763 0.5253763
X0.96   0.5253360 0.5253360
X0.13   0.5236716 0.5236716
X0.18   0.5229563 0.5229563
X0.1    0.5214391 0.5214391
X0.15   0.5165085 0.5165085
X0.16   0.5157811 0.5157811
X0.11   0.5157368 0.5157368
X0.21   0.5094501 0.5094501
X0.9    0.5057225 0.5057225
X0.011  0.5031695 0.5031695
> imp_vars <- row.names(ImpMeasure)[1:15]
> imp_vars <- append(imp_vars, "RB")
> imp_vars
 [1] "X2.949" "X7.253" "X3.919" "X1.932" "X1.201" "X9.002" "X3.106" "X0.4"   "X0.12"  "X0.19"  "X0"
"X2.55"  "X1.142" "X0.22"  "X0.6"   "RB"
> # plot importance
> plot(importance)
>
> biodeg1 <- biodeg[imp_vars]
> head(biodeg1)
     X2.949     X7.253     X3.919      X1.932    X1.201     X9.002     X3.106       X0.4       X0.12      X0.19        X0
X2.55    X1.142      X0.22
1 -0.6043829 -1.1062502 -1.1224746 -0.008448142 -1.396441 -1.3086107 -1.7391211 -0.7405779 -
0.08496205 -0.5813821 -0.4902786  0.05526774 0.4279226 -0.6297848
2 -0.8434713 -0.8292443 -1.5579514 -1.211617959 -1.520787 -0.8918860 -0.3391925 -0.7405779 -
0.29370332 -0.5813821 -0.4902786  1.57244420 0.6932151 -0.6297848
```

```
3 -0.8734824 -1.5628267 -3.2632642 -3.547182898 -2.225418 -3.6011346 -2.3569379 -0.7405779 -
0.29370332 -0.5813821 -0.4902786 -0.54931181 1.1906385 -0.6297848
4 -0.5683696 -0.5055340 -1.0017121 -1.021410966 -1.054488 -0.4417803 -0.0482538 -0.7405779 -
0.29370332 -0.5813821 -0.4902786  1.78627444 0.5274073 -0.6297848
5 -0.5683696 -0.5852537 -1.0017121 -1.043528058 -1.241008 -0.5979175 -0.2810048 -0.7405779 -
0.29370332 -0.5813821 -0.4902786  1.49225786 0.5937304 -0.6297848
6  0.7931339  0.5412904  0.3962053 -0.955059689 -1.178834 -0.3028721 -1.0357340 -0.7405779 -
0.29370332 -0.5813821  0.1932580 -0.79878043 0.7263766 -0.6297848
       X0.6 RB
1 -0.4613371  0
2 -0.4613371  0
3 -0.4613371  0
4 -0.4613371  0
5 -0.4613371  0
6 -0.4613371  0
> str(biodeg1)
'data.frame':	1054 obs. of  16 variables:
 $ X2.949: num  -0.604 -0.843 -0.873 -0.568 -0.568 ...
 $ X7.253: num  -1.106 -0.829 -1.563 -0.506 -0.585 ...
 $ X3.919: num  -1.12 -1.56 -3.26 -1 -1 ...
 $ X1.932: num  -0.00845 -1.21162 -3.54718 -1.02141 -1.04353 ...
 $ X1.201: num  -1.4 -1.52 -2.23 -1.05 -1.24 ...
 $ X9.002: num  -1.309 -0.892 -3.601 -0.442 -0.598 ...
 $ X3.106: num  -1.7391 -0.3392 -2.3569 -0.0483 -0.281 ...
 $ X0.4  : num  -0.741 -0.741 -0.741 -0.741 -0.741 ...
 $ X0.12 : num  -0.085 -0.294 -0.294 -0.294 -0.294 ...
 $ X0.19 : num  -0.581 -0.581 -0.581 -0.581 -0.581 ...
 $ X0    : num  -0.49 -0.49 -0.49 -0.49 -0.49 ...
 $ X2.55 : num  0.0553 1.5724 -0.5493 1.7863 1.4923 ...
 $ X1.142: num  0.428 0.693 1.191 0.527 0.594 ...
 $ X0.22 : num  -0.63 -0.63 -0.63 -0.63 -0.63 ...
 $ X0.6  : num  -0.461 -0.461 -0.461 -0.461 -0.461 ...
 $ RB    : Factor w/ 2 levels "1","0": 2 2 2 2 2 2 2 2 2 2 ...
> dim(biodeg1)
[1] 1054   16
> #Splitting dataset
> library(caTools)
> set.seed(122)
> split <- sample.split(biodeg1$RB, SplitRatio=0.80)
> training_set <- subset(biodeg1, split == TRUE)
> test_set <- subset(biodeg1, split == FALSE)
> dim(training_set)
[1] 843  16
> dim(test_set)
```

```
[1] 211  16
> head(training_set)
    X2.949    X7.253    X3.919    X1.932    X1.201    X9.002    X3.106      X0.4      X0.12     X0.19        X0
X2.55    X1.142     X0.22
3 -0.8734824 -1.5628267 -3.2632642 -3.5471829 -2.2254177 -3.6011346 -2.3569379 -0.7405779 -
0.29370332 -0.5813821 -0.4902786 -0.5493118  1.1906385 -0.6297848
4 -0.5683696 -0.5055340 -1.0017121 -1.0214110 -1.0544878 -0.4417803 -0.0482538 -0.7405779 -
0.29370332 -0.5813821 -0.4902786  1.7862744  0.5274073 -0.6297848
5 -0.5683696 -0.5852537 -1.0017121 -1.0435281 -1.2410076 -0.5979175 -0.2810048 -0.7405779 -
0.29370332 -0.5813821 -0.4902786  1.4922579  0.5937304 -0.6297848
6  0.7931339  0.5412904  0.3962053 -0.9550597 -1.1788343 -0.3028721 -1.0357340 -0.7405779 -
0.29370332 -0.5813821  0.1932580 -0.7987804  0.7263766 -0.6297848
7 -0.5403592 -0.5482122 -0.4729187 -0.4286729 -0.6192749 -0.1639639 -0.1013073 -0.7405779 -
0.29370332 -0.5813821 -0.4902786  1.4235267  0.3615995 -0.6297848
8 -0.2932678 -0.5570700 -0.3430076 -0.1809615  0.9246947 -0.4288587 -0.8680164  0.1581776 -
0.08496205 -0.5813821 -0.4902786 -0.6040422 -0.5337627 -0.6297848
       X0.6 RB
3 -0.4613371  0
4 -0.4613371  0
5 -0.4613371  0
6 -0.4613371  0
7 -0.4613371  0
8 -0.4613371  0
> head(test_set)
     X2.949    X7.253    X3.919      X1.932     X1.201    X9.002    X3.106      X0.4      X0.12     X0.19        X0
X2.55    X1.142     X0.22
1  -0.6043829 -1.1062502 -1.122475 -0.008448142 -1.3964407 -1.308611 -1.7391211 -0.7405779 -
0.08496205 -0.5813821 -0.4902786  0.05526774 0.4279226 -0.6297848
2  -0.8434713 -0.8292443 -1.557951 -1.211617959 -1.5207873 -0.891886 -0.3391925 -0.7405779 -
0.29370332 -0.5813821 -0.4902786  1.57244420 0.6932151 -0.6297848
11 -1.1255756 -1.8503008 -2.132488 -2.140535832 -0.6296371 -2.245434 -1.9581808 -0.7405779 -
0.29370332 -0.5813821 -0.4902786  0.21945881 0.5274073 -0.6297848
31 -0.6584029 -1.2350901 -1.433529 -2.140535832 -0.4534795 -1.744718 -2.0505966 -0.7405779 -
0.29370332 -0.5813821 -0.4902786 -0.68550131 0.4942457 -0.6297848
34 -0.5823748 -1.0764560 -1.122475 -1.627419293 -1.3135431 -1.444288 -1.6518395 -0.7405779 -
0.29370332 -0.5813821 -0.4902786 -0.48185346 0.6600535 -0.6297848
36 -0.5393588 -0.8759488 -1.041966 -1.388554697 -0.5467394 -1.274153 -1.3078472 -0.7405779 -
0.29370332 -0.5813821 -0.4902786  0.13927247 0.2621148 -0.6297848
       X0.6 RB
1  -0.4613371  0
2  -0.4613371  0
11 -0.4613371  0
31 -0.4613371  0
34 -0.4613371  0
```

```
36 -0.4613371  0
>
> #Logistic Regression
> logit <- glm(formula=RB ~ ., data=training_set, family="binomial")
> summary(logit)

Call:
glm(formula = RB ~ ., family = "binomial", data = training_set)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.3812  -0.5517  -0.1152   0.4973   4.2061

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept) -2.03660    0.20515  -9.927  < 2e-16 ***
X2.949      -1.56867    0.83012  -1.890 0.058800 .
X7.253       3.45085    1.16249   2.968 0.002993 **
X3.919      -1.75952    0.48936  -3.596 0.000324 ***
X1.932      -0.47167    0.50247  -0.939 0.347888
X1.201      -2.08451    0.30669  -6.797 1.07e-11 ***
X9.002       1.96000    0.75519   2.595 0.009449 **
X3.106      -2.97724    0.73541  -4.048 5.16e-05 ***
X0.4        -0.76567    0.34152  -2.242 0.024964 *
X0.12       -0.24686    0.81100  -0.304 0.760836
X0.19       -0.47989    0.31700  -1.514 0.130058
X0          -1.82153    0.43487  -4.189 2.81e-05 ***
X2.55        0.76591    0.19984   3.833 0.000127 ***
X1.142      -2.96774    0.51363  -5.778 7.56e-09 ***
X0.22       -0.38575    0.29209  -1.321 0.186616
X0.6        -0.04537    0.38768  -0.117 0.906845
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 1077.3  on 842  degrees of freedom
Residual deviance:  601.6  on 827  degrees of freedom
AIC: 633.6

Number of Fisher Scoring iterations: 7

> #Predicting the test set result
> y_pred <- predict(logit, newdata=test_set[-16], type = "response")
```

```
> y_pred
       1            2          11           31           34           36           38           49           50           52           59           61
9.674875e-01 9.674555e-01 8.213433e-01 8.787283e-01 9.225592e-01 9.016214e-01 8.958062e-01
9.136167e-01 9.248425e-01 2.985116e-01 5.135801e-01 2.709491e-01
       67           71           72           74           82           89           90           96           99          101          109
112
5.859405e-01 8.057486e-01 5.424879e-01 2.707282e-01 5.677640e-02 6.755034e-02 5.015474e-01
9.255828e-01 7.130585e-01 8.779557e-01 8.622326e-01 7.634211e-01
      129          131          136          148          150          155          158          163          167          168          170
193
6.981148e-01 7.369943e-01 3.671884e-01 7.793587e-01 6.381805e-01 7.612499e-01 5.383913e-01
2.726270e-01 6.799265e-01 6.259890e-01 4.471750e-01 6.297829e-01
      199          202          209          218          221          237          240          256          258          265          267
269
8.811116e-01 8.789495e-01 5.331796e-01 9.358349e-01 6.525607e-01 9.137068e-01 8.577808e-01
3.285798e-01 3.837453e-01 5.668171e-01 2.534421e-03 8.796237e-01
      273          276          280          284          285          287          293          296          301          308          325
331
7.877231e-01 9.788238e-01 1.563304e-01 5.452727e-03 1.056816e-01 1.996479e-08 6.433515e-04
3.470356e-02 6.111223e-07 6.617056e-05 2.675553e-01 1.603295e-01
      345          346          351          356          357          358          364          365          367          380          382
385
4.576989e-04 3.203877e-02 7.358303e-02 4.582022e-02 4.388011e-02 1.105999e-01 2.106732e-01
3.267120e-04 8.066657e-08 6.087532e-01 6.779853e-02 1.333534e-02
      390          391          395          397          399          402          410          417          420          425          431
435
3.321955e-03 2.877990e-02 5.295935e-02 4.141410e-02 3.322551e-02 7.925728e-03 7.357424e-02
3.594833e-01 7.589268e-02 2.007533e-02 2.406825e-02 1.722865e-02
      439          446          449          457          460          462          468          470          471          477          479
487
2.815296e-03 1.901912e-01 5.130638e-01 3.631151e-05 2.716762e-01 4.151631e-02 7.096890e-02
1.048132e-01 3.007797e-03 5.764408e-03 1.437819e-01 7.199900e-02
      495          499          509          512          518          530          532          533          538          539          541
544
8.157142e-03 5.296943e-03 2.426239e-01 4.662292e-02 4.144843e-03 4.301046e-03 1.972991e-02
2.875917e-01 4.566659e-02 2.589967e-02 2.383228e-01 3.060365e-01
      547          548          566          573          581          588          599          610          611          618          625
628
7.264459e-01 9.746340e-03 2.569285e-09 1.749316e-01 8.118469e-01 9.715693e-05 2.492411e-02
6.956192e-01 6.188371e-01 1.638564e-03 5.471111e-01 2.017040e-01
      635          638          649          651          652          653          656          661          662          666          669
673
2.074937e-01 2.691046e-02 6.769847e-04 9.795639e-04 1.808056e-01 6.334276e-05 1.041701e-02
5.483338e-01 2.414291e-02 5.093503e-01 2.342886e-01 4.074473e-01
```

```
      679        685        692        693        696        698        701        708        713        716        721
723
7.278373e-02 4.944221e-04 2.839162e-03 4.239003e-01 3.892071e-01 1.256617e-01 1.833482e-01
2.164956e-01 2.013613e-01 2.776185e-01 5.707713e-02 6.353242e-01
      735        736        737        742        743        753        759        761        771        775        776
778
8.516332e-01 6.972746e-01 8.454286e-01 5.547208e-01 8.295279e-01 6.409358e-01 1.207089e-02
9.207194e-04 1.997617e-01 2.085584e-01 3.011911e-01 1.878334e-01
      789        794        795        811        813        816        817        821        825        827        829
835
4.982716e-03 2.532091e-01 8.489912e-01 9.112727e-05 9.886689e-02 1.434788e-02 1.526234e-02
3.056702e-01 3.192758e-08 6.251275e-02 5.554658e-03 1.813295e-01
      836        837        840        843        848        850        853        858        862        869        872
876
1.687740e-09 8.181908e-01 7.239384e-01 8.916309e-01 8.743828e-01 4.873044e-01 7.059348e-01
6.402549e-01 7.540568e-01 9.696596e-01 9.414780e-01 8.451545e-01
      877        885        888        895        900        902        903        905        907        917        919
941
6.888675e-01 4.181915e-01 9.378296e-01 8.801193e-01 7.665428e-01 9.617948e-01 7.989257e-01
1.275893e-01 5.299961e-01 5.895782e-01 2.998635e-03 4.592207e-02
      943        950        957        964        973        975        981        989        995       1000       1009
1010
4.476849e-02 5.737395e-01 8.496155e-02 1.016873e-01 1.058375e-02 7.763831e-01 1.227196e-01
1.245936e-01 3.209016e-05 1.584910e-02 1.299333e-01 1.481877e-01
     1013       1014       1017       1018       1019       1025       1048
2.959929e-01 4.728785e-02 1.216112e-01 1.104870e-01 1.009405e-01 2.158973e-04 1.312417e-02
> y_pred_final <- factor(ifelse(y_pred > 0.5, 1, 0))
> y_pred_final
  1   2  11  31  34  36  38  49  50  52  59  61  67  71  72  74  82  89  90  96  99 101 109 112
129 131 136 148 150 155 158 163
  1   1   1   1   1   1   1   1   1   0   1   0   1   1   1   0   0   0   1   1   1   1   1   1   1   1   0   1   1   1
1   0
 167 168 170 193 199 202 209 218 221 237 240 256 258 265 267 269 273 276 280 284 285
287 293 296 301 308 325 331 345 346 351 356
  1   1   0   1   1   1   1   1   1   1   1   0   0   1   0   1   1   1   0   0   0   0   0   0   0   0   0   0   0   0
0   0
 357 358 364 365 367 380 382 385 390 391 395 397 399 402 410 417 420 425 431 435 439
446 449 457 460 462 468 470 471 477 479 487
  0   0   0   0   0   1   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   1   0   0   0   0   0   0   0
0   0
 495 499 509 512 518 530 532 533 538 539 541 544 547 548 566 573 581 588 599 610 611
618 625 628 635 638 649 651 652 653 656 661
  0   0   0   0   0   0   0   0   0   0   0   0   1   0   0   0   1   0   0   1   1   0   1   0   0   0   0   0   0   0
0   1
```

```
 662  666  669  673  679  685  692  693  696  698  701  708  713  716  721  723  735  736  737  742  743
753  759  761  771  775  776  778  789  794  795  811
   0    1    0    0    0    0    0    0    0    0    0    0    0    0    0    1    1    1    1    1    1    1    0    0    0    0    0    0    0    0
1    0
 813  816  817  821  825  827  829  835  836  837  840  843  848  850  853  858  862  869  872  876  877
885  888  895  900  902  903  905  907  917  919  941
   0    0    0    0    0    0    0    0    0    1    1    1    1    0    1    1    1    1    1    1    1    0    1    1    1    1    1    0    1    1
0    0
 943  950  957  964  973  975  981  989  995 1000 1009 1010 1013 1014 1017 1018 1019 1025 1048
   0    1    0    0    0    1    0    0    0    0    0    0    0    0    0    0    0    0    0
Levels: 0 1
> #Making confusion matrix
> cm <- table(test_set$RB, y_pred_final)
> cm
   y_pred_final
     0   1
  1 120  20
  0  15  56
> accuracy <- (sum(diag(cm)) / sum(cm)) * 100
> accuracy
[1] 83.41232
> precision <- (cm[1, "0"] / (cm[1, "0"] + cm[1, "1"])) * 100
> precision
[1] 85.71429
> sensitivity <- (cm[1, "0"] / (cm[1, "0"] + cm[2, "0"])) * 100
> sensitivity
[1] 88.88889
> specificity <- (cm[1, "1"] / (cm[1, "1"] + cm[1, "1"])) * 100
> sensitivity
[1] 88.88889
> #install.packages("ROCR")
> library(ROCR)
> ROCPred <- prediction(as.numeric(y_pred_final), as.numeric(test_set$RB))
> ROCPer <- performance(ROCPred, measure = "tpr", x.measure = "fpr")
> auc <- performance(ROCPred, measure = "auc")
> auc <- auc@y.values[[1]]
> auc
[1] 0.8229376
> # Plotting curve
> plot(ROCPer)
> plot(ROCPer, colorize = TRUE,
+      print.cutoffs.at = seq(0.1, by = 0.1),
+      main = "ROC CURVE")
> abline(a = 0, b = 1)
```

```
>
> auc <- round(auc, 4)
> legend(.6, .4, auc, title = "AUC", cex = 1)
>
> #SVM
> library(e1071)
> #Linear
> svc <- svm(formula=RB ~ ., data=training_set, type="C-classification", kernel="linear")
> #Predicting the test set result
> y_pred_final <- predict(svc, newdata=test_set[-16], type = "class")
> y_pred_final
  1   2  11  31  34  36  38  49  50  52  59  61  67  71  72  74  82  89  90  96  99 101 109 112 
129 131 136 148 150 155 158 163 
  0   0   0   0   0   0   0   0   0   1   0   1   0   0   1   1   1   1   1   0   0   0   0   0   0   0   1   0   0   0 
0   1 
 167 168 170 193 199 202 209 218 221 237 240 256 258 265 267 269 273 276 280 284 285 
287 293 296 301 308 325 331 345 346 351 356 
  0   0   1   0   0   0   0   0   0   0   0   1   1   0   1   0   0   0   1   1   1   1   1   1   1   1   1   1   1   1 
1   1 
 357 358 364 365 367 380 382 385 390 391 395 397 399 402 410 417 420 425 431 435 439 
446 449 457 460 462 468 470 471 477 479 487 
  1   1   1   1   1   0   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   0   1   1   1   1   1   1   1 
1   1 
 495 499 509 512 518 530 532 533 538 539 541 544 547 548 566 573 581 588 599 610 611 
618 625 628 635 638 649 651 652 653 656 661 
  1   1   1   1   1   1   1   1   1   1   1   1   0   1   1   1   0   1   1   0   0   1   0   1   1   1   1   1   1   1 
1   1 
 662 666 669 673 679 685 692 693 696 698 701 708 713 716 721 723 735 736 737 742 743 
753 759 761 771 775 776 778 789 794 795 811 
  1   0   1   1   1   1   1   1   1   1   1   1   1   1   1   1   0   0   0   1   0   1   1   1   1   1   1   1   1   1 
0   1 
 813 816 817 821 825 827 829 835 836 837 840 843 848 850 853 858 862 869 872 876 877 
885 888 895 900 902 903 905 907 917 919 941 
  1   1   1   1   1   1   1   1   1   0   0   0   0   0   0   0   0   0   0   0   0   1   0   0   0   0   0   1   1   1 
1   1 
 943  950  957  964  973  975  981  989  995 1000 1009 1010 1013 1014 1017 1018 1019 1025 1048 
  1   0   1   1   1   0   1   1   1   1   1   1   1   1   1   1   1   1   1 
Levels: 1 0
> #Making confusion matrix
> cm <- table(test_set$RB, y_pred_final)
> cm
   y_pred_final
     1   0
  1 125  15
```

```
  0  17  54
> accuracy <- (sum(diag(cm)) / sum(cm)) * 100
> accuracy
[1] 84.83412
> precision <- (cm[1, "1"] / (cm[1, "1"] + cm[1, "0"])) * 100
> precision
[1] 89.28571
> sensitivity <- (cm[1, "1"] / (cm[1, "1"] + cm[2, "1"])) * 100
> sensitivity
[1] 88.02817
> specificity <- (cm[1, "0"] / (cm[1, "0"] + cm[1, "0"])) * 100
> sensitivity
[1] 88.02817
> ROCPred <- prediction(as.numeric(y_pred_final), as.numeric(test_set$RB))
> ROCPer <- performance(ROCPred, measure = "tpr", x.measure = "fpr")
> auc <- performance(ROCPred, measure = "auc")
> auc <- auc@y.values[[1]]
> auc
[1] 0.8267103
> # Plotting curve
> plot(ROCPer)
> plot(ROCPer, colorize = TRUE,
+      print.cutoffs.at = seq(0.1, by = 0.1),
+      main = "ROC CURVE")
> abline(a = 0, b = 1)
>
> auc <- round(auc, 4)
> legend(.6, .4, auc, title = "AUC", cex = 1)
>
> #Radial
> svc <- svm(formula=RB ~ ., data=training_set, type="C-classification", kernel="radial")
> #Predicting the test set result
> y_pred_final <- predict(svc, newdata=test_set[-16])
> y_pred_final
  1   2  11  31  34  36  38  49  50  52  59  61  67  71  72  74  82  89  90  96  99 101 109 112
129 131 136 148 150 155 158 163
  0   0   0   0   0   0   0   0   0   1   0   1   0   0   1   1   1   1   1   0   0   0   0   0   0   0   1   0   0   1
0   1
 167 168 170 193 199 202 209 218 221 237 240 256 258 265 267 269 273 276 280 284 285
287 293 296 301 308 325 331 345 346 351 356
  0   0   1   0   0   0   0   0   0   0   0   0   1   0   1   0   0   0   1   1   1   1   1   1   1   1   0   1   1   1
1   1
 357 358 364 365 367 380 382 385 390 391 395 397 399 402 410 417 420 425 431 435 439
446 449 457 460 462 468 470 471 477 479 487
```

```
  1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   0   1   1   1   1   1   1   1
1   1
 495  499  509  512  518  530  532  533  538  539  541  544  547  548  566  573  581  588  599  610  611
618  625  628  635  638  649  651  652  653  656  661
  1   1   1   1   1   1   1   1   1   1   1   1   0   1   1   1   0   1   1   1   1   1   1   1   1   1   1   1   1   1
1   1
 662  666  669  673  679  685  692  693  696  698  701  708  713  716  721  723  735  736  737  742  743
753  759  761  771  775  776  778  789  794  795  811
  1   0   1   1   1   1   1   1   1   1   0   1   1   1   1   1   0   0   0   1   0   1   1   1   1   1   1   1   1   1
0   1
 813  816  817  821  825  827  829  835  836  837  840  843  848  850  853  858  862  869  872  876  877
885  888  895  900  902  903  905  907  917  919  941
  1   1   1   1   1   1   1   1   1   0   0   0   0   0   0   0   0   0   0   0   0   1   0   0   0   0   0   1   1   1
1   1
 943  950  957  964  973  975  981  989  995 1000 1009 1010 1013 1014 1017 1018 1019 1025 1048
  1   0   1   1   1   0   1   1   1   1   1   1   1   1   1   1   1   1   1
Levels: 1 0
> #Making confusion matrix
> cm <- table(test_set$RB, y_pred_final)
> cm
   y_pred_final
      1   0
  1 127  13
  0  17  54
> accuracy <- (sum(diag(cm)) / sum(cm)) * 100
> accuracy
[1] 85.78199
> precision <- (cm[1, "1"] / (cm[1, "1"] + cm[1, "0"])) * 100
> precision
[1] 90.71429
> sensitivity <- (cm[1, "1"] / (cm[1, "1"] + cm[2, "1"])) * 100
> sensitivity
[1] 88.19444
> specificity <- (cm[1, "0"] / (cm[1, "0"] + cm[1, "0"])) * 100
> sensitivity
[1] 88.19444
> ROCPred <- prediction(as.numeric(y_pred_final), as.numeric(test_set$RB))
> ROCPer <- performance(ROCPred, measure = "tpr", x.measure = "fpr")
> auc <- performance(ROCPred, measure = "auc")
> auc <- auc@y.values[[1]]
> auc
[1] 0.8338531
> # Plotting curve
> plot(ROCPer)
```

```
> plot(ROCPer, colorize = TRUE,
+      print.cutoffs.at = seq(0.1, by = 0.1),
+      main = "ROC CURVE")
> abline(a = 0, b = 1)
>
> auc <- round(auc, 4)
> legend(.6, .4, auc, title = "AUC", cex = 1)
>
> #Decision Tree
> library(rpart)
> #Max-Depth - 8
> mtree <- rpart(RB ~ ., data=training_set, method="class", control=rpart.control(minsplit=20,
minbucket=7, maxdepth=8, usesurrogate=2, xval=10))
> #Predicting the test set result
> y_pred_final <- predict(mtree, newdata=test_set[-16], type="class")
> y_pred_final
   1    2   11   31   34   36   38   49   50   52   59   61   67   71   72   74   82   89   90   96   99  101  109  112
129  131  136  148  150  155  158  163
   0    0    0    0    0    0    0    0    0    1    0    1    0    1    1    0    1    1    1    0    0    0    0    0    0    0    0    0    0    1
0    1
 167  168  170  193  199  202  209  218  221  237  240  256  258  265  267  269  273  276  280  284  285
287  293  296  301  308  325  331  345  346  351  356
   0    0    1    0    0    0    1    0    0    0    0    1    0    0    1    0    0    0    1    1    1    1    1    1    1    1    1    0    1    1
1    1
 357  358  364  365  367  380  382  385  390  391  395  397  399  402  410  417  420  425  431  435  439
446  449  457  460  462  468  470  471  477  479  487
   1    1    1    1    1    1    1    1    1    1    1    1    1    1    1    0    1    1    1    1    1    1    0    1    1    1    1    1    1    1
1    1
 495  499  509  512  518  530  532  533  538  539  541  544  547  548  566  573  581  588  599  610  611
618  625  628  635  638  649  651  652  653  656  661
   1    1    1    0    1    1    1    1    1    1    1    1    0    1    1    1    1    1    1    1    1    1    1    1    0    1    1    1    1    1
1    1
 662  666  669  673  679  685  692  693  696  698  701  708  713  716  721  723  735  736  737  742  743
753  759  761  771  775  776  778  789  794  795  811
   1    0    0    1    1    1    1    0    0    1    1    1    1    1    1    1    0    0    1    1    0    1    1    1    1    0    1    1    1    1
0    1
 813  816  817  821  825  827  829  835  836  837  840  843  848  850  853  858  862  869  872  876  877
885  888  895  900  902  903  905  907  917  919  941
   1    1    1    1    1    1    1    1    1    0    0    0    0    0    0    0    0    0    0    0    0    1    0    0    0    0    0    1    0    1
1    1
 943  950  957  964  973  975  981  989  995 1000 1009 1010 1013 1014 1017 1018 1019 1025 1048
   1    0    1    1    1    0    1    1    1    1    1    1    1    1    1    1    1    1    1
Levels: 1 0
> #Making confusion matrix
```

```
> cm <- table(test_set$RB, y_pred_final)
> cm
   y_pred_final
     1   0
  1 123  17
  0  16  55
> accuracy <- (sum(diag(cm)) / sum(cm)) * 100
> accuracy
[1] 84.36019
> precision <- (cm[1, "1"] / (cm[1, "1"] + cm[1, "0"])) * 100
> precision
[1] 87.85714
> sensitivity <- (cm[1, "1"] / (cm[1, "1"] + cm[2, "1"])) * 100
> sensitivity
[1] 88.48921
> specificity <- (cm[1, "0"] / (cm[1, "0"] + cm[1, "0"])) * 100
> sensitivity
[1] 88.48921
> ROCPred <- prediction(as.numeric(y_pred_final), as.numeric(test_set$RB))
> ROCPer <- performance(ROCPred, measure = "tpr", x.measure = "fpr")
> auc <- performance(ROCPred, measure = "auc")
> auc <- auc@y.values[[1]]
> auc
[1] 0.8266097
> # Plotting curve
> plot(ROCPer)
> plot(ROCPer, colorize = TRUE,
+      print.cutoffs.at = seq(0.1, by = 0.1),
+      main = "ROC CURVE")
> abline(a = 0, b = 1)
>
> auc <- round(auc, 4)
> legend(.6, .4, auc, title = "AUC", cex = 1)
>
> #Max-Depth - 10
> mtree <- rpart(RB ~ ., data=training_set, method="class", control=rpart.control(minsplit=20,
minbucket=7, maxdepth=10, usesurrogate=2, xval=10))
> #Predicting the test set result
> y_pred_final <- predict(mtree, newdata=test_set[-16], type="class")
> y_pred_final
  1   2  11  31  34  36  38  49  50  52  59  61  67  71  72  74  82  89  90  96  99 101 109 112
129 131 136 148 150 155 158 163
  0   0   0   0   0   0   0   0   0   1   0   1   0   1   1   0   1   1   1   0   0   0   0   0   0   0   0   0   0   1
0   1
```

```
 167  168  170  193  199  202  209  218  221  237  240  256  258  265  267  269  273  276  280  284  285 
287  293  296  301  308  325  331  345  346  351  356 
  0    0    1    0    0    0    1    0    0    0    0    1    0    0    1    0    0    0    1    1    1    1    1    1    1    1    1    0    1    1 
1    1 
 357  358  364  365  367  380  382  385  390  391  395  397  399  402  410  417  420  425  431  435  439 
446  449  457  460  462  468  470  471  477  479  487 
  1    1    1    1    1    1    1    1    1    1    1    1    1    1    1    0    1    1    1    1    1    1    0    1    1    1    1    1    1    1 
1    1 
 495  499  509  512  518  530  532  533  538  539  541  544  547  548  566  573  581  588  599  610  611 
618  625  628  635  638  649  651  652  653  656  661 
  1    1    1    0    1    1    1    1    1    1    1    1    0    1    1    1    1    1    1    1    1    1    1    1    0    1    1    1    1    1 
1    1 
 662  666  669  673  679  685  692  693  696  698  701  708  713  716  721  723  735  736  737  742  743 
753  759  761  771  775  776  778  789  794  795  811 
  1    0    0    1    1    1    1    0    0    1    1    1    1    1    1    1    0    0    1    1    0    1    1    1    1    0    1    1    1    1 
0    1 
 813  816  817  821  825  827  829  835  836  837  840  843  848  850  853  858  862  869  872  876  877 
885  888  895  900  902  903  905  907  917  919  941 
  1    1    1    1    1    1    1    1    1    0    0    0    0    0    0    0    0    0    0    0    0    1    0    0    0    0    0    1    0    1 
1    1 
 943  950  957  964  973  975  981  989  995 1000 1009 1010 1013 1014 1017 1018 1019 1025 1048 
  1    0    1    1    1    0    1    1    1    1    1    1    1    1    1    1    1    1    1 
Levels: 1 0
> #Making confusion matrix
> cm <- table(test_set$RB, y_pred_final)
> cm
   y_pred_final
     1   0
  1 123  17
  0  16  55
> accuracy <- (sum(diag(cm)) / sum(cm)) * 100
> accuracy
[1] 84.36019
> precision <- (cm[1, "1"] / (cm[1, "1"] + cm[1, "0"])) * 100
> precision
[1] 87.85714
> sensitivity <- (cm[1, "1"] / (cm[1, "1"] + cm[2, "1"])) * 100
> sensitivity
[1] 88.48921
> specificity <- (cm[1, "0"] / (cm[1, "0"] + cm[1, "0"])) * 100
> sensitivity
[1] 88.48921
> ROCPred <- prediction(as.numeric(y_pred_final), as.numeric(test_set$RB))
> ROCPer <- performance(ROCPred, measure = "tpr", x.measure = "fpr")
```

```
> auc <- performance(ROCPred, measure = "auc")
> auc <- auc@y.values[[1]]
> auc
[1] 0.8266097
> # Plotting curve
> plot(ROCPer)
> plot(ROCPer, colorize = TRUE,
+      print.cutoffs.at = seq(0.1, by = 0.1),
+      main = "ROC CURVE")
> abline(a = 0, b = 1)
>
> auc <- round(auc, 4)
> legend(.6, .4, auc, title = "AUC", cex = 1)
>
> #Random Forest
> #No. of trees - 500
> library(randomForest)
> classifier_RF = randomForest(x=training_set[-16], y=training_set$RB, ntree=500)  #First 4 columns as features, Species as dependent variable
> classifier_RF

Call:
 randomForest(x = training_set[-16], y = training_set$RB, ntree = 500)
               Type of random forest: classification
                     Number of trees: 500
No. of variables tried at each split: 3

        OOB estimate of  error rate: 16.01%
Confusion matrix:
    1   0 class.error
1 503  56   0.1001789
0  79 205   0.2781690
> y_pred_final <- predict(classifier_RF, newdata=test_set[-16], type="class")
> y_pred_final
  1   2  11  31  34  36  38  49  50  52  59  61  67  71  72  74  82  89  90  96  99 101 109 112
129 131 136 148 150 155 158 163
  0   0   0   0   0   0   0   0   0   0   0   1   0   1   1   1   0   1   1   0   0   0   0   0   0   0   1   0   0   1
0   1
 167 168 170 193 199 202 209 218 221 237 240 256 258 265 267 269 273 276 280 284 285
287 293 296 301 308 325 331 345 346 351 356
  0   0   1   0   0   0   1   0   1   0   0   0   1   0   1   0   0   0   1   1   1   1   1   1   1   1   1   1   1   1
1   1
 357 358 364 365 367 380 382 385 390 391 395 397 399 402 410 417 420 425 431 435 439
446 449 457 460 462 468 470 471 477 479 487
```

```
  1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   0   1   1   1   1   1   1   1
1   1
 495  499  509  512  518  530  532  533  538  539  541  544  547  548  566  573  581  588  599  610  611
618  625  628  635  638  649  651  652  653  656  661
  1   1   1   1   1   1   1   1   1   1   1   1   0   1   1   1   0   1   1   1   1   1   1   1   1   1   1   1   1   1
1   1
 662  666  669  673  679  685  692  693  696  698  701  708  713  716  721  723  735  736  737  742  743
753  759  761  771  775  776  778  789  794  795  811
  1   0   1   1   1   1   1   1   1   1   1   1   1   1   1   1   0   0   0   1   0   1   1   1   1   1   1   1   1   1
0   1
 813  816  817  821  825  827  829  835  836  837  840  843  848  850  853  858  862  869  872  876  877
885  888  895  900  902  903  905  907  917  919  941
  1   1   1   1   1   1   1   1   1   0   0   0   0   0   0   0   0   0   0   0   0   1   0   0   0   0   0   1   0   1
1   1
 943  950  957  964  973  975  981  989  995 1000 1009 1010 1013 1014 1017 1018 1019 1025 1048
  1   0   1   1   1   0   1   1   1   1   1   1   1   1   1   1   1   1   1
Levels: 1 0
> #Making confusion matrix
> cm <- table(test_set$RB, y_pred_final)
> cm
   y_pred_final
     1   0
  1 129  11
  0  17  54
> accuracy <- (sum(diag(cm)) / sum(cm)) * 100
> accuracy
[1] 86.72986
> precision <- (cm[1, "1"] / (cm[1, "1"] + cm[1, "0"])) * 100
> precision
[1] 92.14286
> sensitivity <- (cm[1, "1"] / (cm[1, "1"] + cm[2, "1"])) * 100
> sensitivity
[1] 88.35616
> specificity <- (cm[1, "0"] / (cm[1, "0"] + cm[1, "0"])) * 100
> sensitivity
[1] 88.35616
> ROCPred <- prediction(as.numeric(y_pred_final), as.numeric(test_set$RB))
> ROCPer <- performance(ROCPred, measure = "tpr", x.measure = "fpr")
> auc <- performance(ROCPred, measure = "auc")
> auc <- auc@y.values[[1]]
> auc
[1] 0.840996
> # Plotting curve
> plot(ROCPer)
```

```
> plot(ROCPer, colorize = TRUE,
+      print.cutoffs.at = seq(0.1, by = 0.1),
+      main = "ROC CURVE")
> abline(a = 0, b = 1)
>
> auc <- round(auc, 4)
> legend(.6, .4, auc, title = "AUC", cex = 1)
>
> #No. of trees - 800
> library(randomForest)
> classifier_RF = randomForest(x=training_set[-16], y=training_set$RB, ntree=800)  #First 4 columns as features, Species as dependent variable
> classifier_RF

Call:
 randomForest(x = training_set[-16], y = training_set$RB, ntree = 800)
               Type of random forest: classification
                     Number of trees: 800
No. of variables tried at each split: 3

        OOB estimate of  error rate: 15.54%
Confusion matrix:
    1   0 class.error
1 507  52  0.09302326
0  79 205  0.27816901
> y_pred_final <- predict(classifier_RF, newdata=test_set[-16], type="class")
> y_pred_final
  1   2  11  31  34  36  38  49  50  52  59  61  67  71  72  74  82  89  90  96  99 101 109 112
129 131 136 148 150 155 158 163
  0   0   0   0   0   0   0   0   0   0   0   1   0   1   1   1   0   1   1   0   0   0   0   0   0   0   1   0   0   1
0   1
 167 168 170 193 199 202 209 218 221 237 240 256 258 265 267 269 273 276 280 284 285
287 293 296 301 308 325 331 345 346 351 356
  0   0   1   0   0   0   1   0   0   0   0   0   1   0   1   0   0   0   1   1   1   1   1   1   1   1   1   1   1   1
1   1
 357 358 364 365 367 380 382 385 390 391 395 397 399 402 410 417 420 425 431 435 439
446 449 457 460 462 468 470 471 477 479 487
  1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   0   1   1   1   1   1   1   1
1   1
 495 499 509 512 518 530 532 533 538 539 541 544 547 548 566 573 581 588 599 610 611
618 625 628 635 638 649 651 652 653 656 661
  1   1   1   1   1   1   1   0   1   1   1   1   0   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1
1   1
```

```
 662  666  669  673  679  685  692  693  696  698  701  708  713  716  721  723  735  736  737  742  743
753  759  761  771  775  776  778  789  794  795  811
  1    0    1    1    1    1    1    1    1    1    1    1    1    1    1    1    0    0    0    1    0    1    1    1    1    1    1    1    1    1
0    1
 813  816  817  821  825  827  829  835  836  837  840  843  848  850  853  858  862  869  872  876  877
885  888  895  900  902  903  905  907  917  919  941
  1    1    1    1    1    1    1    1    1    0    0    0    0    0    0    0    0    0    0    0    0    1    0    0    0    0    0    1    0    1
1    1
 943  950  957  964  973  975  981  989  995 1000 1009 1010 1013 1014 1017 1018 1019 1025 1048
  1    0    1    1    1    0    1    1    1    1    1    1    1    1    1    1    1    1    1
Levels: 1 0
> #Making confusion matrix
> cm <- table(test_set$RB, y_pred_final)
> cm
   y_pred_final
     1   0
  1 129  11
  0  16  55
> accuracy <- (sum(diag(cm)) / sum(cm)) * 100
> accuracy
[1] 87.20379
> precision <- (cm[1, "1"] / (cm[1, "1"] + cm[1, "0"])) * 100
> precision
[1] 92.14286
> sensitivity <- (cm[1, "1"] / (cm[1, "1"] + cm[2, "1"])) * 100
> sensitivity
[1] 88.96552
> specificity <- (cm[1, "0"] / (cm[1, "0"] + cm[1, "0"])) * 100
> sensitivity
[1] 88.96552
> ROCPred <- prediction(as.numeric(y_pred_final), as.numeric(test_set$RB))
> ROCPer <- performance(ROCPred, measure = "tpr", x.measure = "fpr")
> auc <- performance(ROCPred, measure = "auc")
> auc <- auc@y.values[[1]]
> auc
[1] 0.8480382
> # Plotting curve
> plot(ROCPer)
> plot(ROCPer, colorize = TRUE,
+      print.cutoffs.at = seq(0.1, by = 0.1),
+      main = "ROC CURVE")
> abline(a = 0, b = 1)
>
> auc <- round(auc, 4)
```

```
> legend(.6, .4, auc, title = "AUC", cex = 1)
```

Day 13 – R Programming

Football Segmentation – KMeans

```
> setwd("C:/zubeda/PGA02_Zubu/R Programming/Models/KMeans/Dataset")
> library(caret)
> library(broom)
> library(dplyr)
> #install.packages("dummy")
> library(dummy)
> library(ggplot2)
> #install.packages("ROCit")
> library(ROCit)
> library(purrr)
> library(tidyverse)
> library(magrittr)
> #install.packages("maps")
> library(maps)
> #install.packages("plotly")
> library(plotly)
> #install.packages("DT")
> library(DT)
> #install.packages("tidytext")
> library(tidytext)
> library(gridExtra)
> #install.packages("factoextra")
> library(factoextra)
>
> #Read the Data
> raw_data <- read.csv("data.csv")
> head(raw_data)
  ï..    ID            Name Age                                 Photo Nationality                          Flag Overall
Potential
1   0 158023         L. Messi  31 https://cdn.sofifa.org/players/4/19/158023.png   Argentina
https://cdn.sofifa.org/flags/52.png      94       94
2   1  20801 Cristiano Ronaldo  33  https://cdn.sofifa.org/players/4/19/20801.png    Portugal
https://cdn.sofifa.org/flags/38.png      94       94
3   2 190871        Neymar Jr  26 https://cdn.sofifa.org/players/4/19/190871.png      Brazil
https://cdn.sofifa.org/flags/54.png      92       93
4   3 193080          De Gea  27 https://cdn.sofifa.org/players/4/19/193080.png       Spain
https://cdn.sofifa.org/flags/45.png      91       93
5   4 192985      K. De Bruyne  27 https://cdn.sofifa.org/players/4/19/192985.png     Belgium
https://cdn.sofifa.org/flags/7.png      91       92
```

6   5 183277   E. Hazard 27 https://cdn.sofifa.org/players/4/19/183277.png   Belgium
https://cdn.sofifa.org/flags/7.png   91   91
Club   Club.Logo   Value   Wage Special Preferred.Foot
International.Reputation Weak.Foot
1   FC Barcelona https://cdn.sofifa.org/teams/2/light/241.png â‚¬110.5M â‚¬565K   2202   Left
5   4
2   Juventus https://cdn.sofifa.org/teams/2/light/45.png   â‚¬77M â‚¬405K   2228   Right
5   4
3 Paris Saint-Germain https://cdn.sofifa.org/teams/2/light/73.png â‚¬118.5M â‚¬290K   2143   Right
5   5
4   Manchester United https://cdn.sofifa.org/teams/2/light/11.png   â‚¬72M â‚¬260K   1471   Right
4   3
5   Manchester City https://cdn.sofifa.org/teams/2/light/10.png   â‚¬102M â‚¬355K   2281   Right
4   5
6   Chelsea   https://cdn.sofifa.org/teams/2/light/5.png   â‚¬93M â‚¬340K   2142   Right
4   4
Skill.Moves   Work.Rate Body.Type Real.Face Position Jersey.Number   Joined Loaned.From
Contract.Valid.Until Height Weight   LS   ST   RS   LW
1   4 Medium/ Medium   Messi   Yes   RF   10 Jul 1, 2004   2021   5'7
159lbs 88+2 88+2 88+2 92+2
2   5   High/ Low C. Ronaldo   Yes   ST   7 Jul 10, 2018   2022   6'2 183lbs
91+3 91+3 91+3 89+3
3   5   High/ Medium   Neymar   Yes   LW   10 Aug 3, 2017   2022   5'9
150lbs 84+3 84+3 84+3 89+3
4   1 Medium/ Medium   Lean   Yes   GK   1 Jul 1, 2011   2020   6'4 168lbs
5   4   High/ High   Normal   Yes   RCM   7 Aug 30, 2015   2023   5'11 154lbs
82+3 82+3 82+3 87+3
6   4   High/ Medium   Normal   Yes   LF   10 Jul 1, 2012   2020   5'8 163lbs
83+3 83+3 83+3 89+3
LF   CF   RF   RW LAM CAM RAM   LM LCM   CM RCM   RM LWB LDM CDM RDM RWB   LB LCB   CB
RCB   RB Crossing Finishing HeadingAccuracy
1 93+2 93+2 93+2 92+2 93+2 93+2 93+2 91+2 84+2 84+2 84+2 91+2 64+2 61+2 61+2 61+2 64+2 59+2
47+2 47+2 47+2 59+2   84   95   70
2 90+3 90+3 90+3 89+3 88+3 88+3 88+3 88+3 81+3 81+3 81+3 88+3 65+3 61+3 61+3 61+3 65+3 61+3
53+3 53+3 53+3 61+3   84   94   89
3 89+3 89+3 89+3 89+3 89+3 89+3 89+3 88+3 81+3 81+3 81+3 88+3 65+3 60+3 60+3 60+3 65+3 60+3
47+3 47+3 47+3 60+3   79   87   62
4   17   13   21
5 87+3 87+3 87+3 87+3 88+3 88+3 88+3 88+3 87+3 87+3 87+3 88+3 77+3 77+3 77+3 77+3 77+3 73+3
66+3 66+3 66+3 73+3   93   82   55
6 88+3 88+3 88+3 89+3 89+3 89+3 89+3 89+3 82+3 82+3 82+3 89+3 66+3 63+3 63+3 63+3 66+3 60+3
49+3 49+3 49+3 60+3   81   84   61
ShortPassing Volleys Dribbling Curve FKAccuracy LongPassing BallControl Acceleration SprintSpeed
Agility Reactions Balance ShotPower Jumping Stamina

```
1       90    86     97   93       94       87       96       91       86    91      95    95      85    68
72
2       81    87     88   81       76       77       94       89       91    87      96    70      95    95
88
3       84    84     96   88       87       78       95       94       90    96      94    84      80    61
81
4       50    13     18   21       19       51       42       57       58    60      90    43      31    67
43
5       92    82     86   85       83       91       91       78       76    79      91    77      91    63
90
6       89    80     95   83       79       83       94       94       88    95      90    94      82    56
83
  Strength LongShots Aggression Interceptions Positioning Vision Penalties Composure Marking
StandingTackle SlidingTackle GKDiving GKHandling GKKicking
1      59      94      48         22       94   94      75      96     33         28        26      6       11      15
2      79      93      63         29       95   82      85      95     28         31        23      7       11      15
3      49      82      56         36       89   87      81      94     27         24        33      9        9      15
4      64      12      38         30       12   68      40      68     15         21        13     90       85      87
5      75      91      76         61       87   94      79      88     68         58        51     15       13       5
6      66      80      54         41       87   89      86      91     34         27        22     11       12       6
  GKPositioning GKReflexes Release.Clause
1          14        8     â‚¬226.5M
2          14       11     â‚¬127.1M
3          15       11     â‚¬228.1M
4          88       94     â‚¬138.6M
5          10       13     â‚¬196.4M
6           8        8    â‚¬172.1M
> names(raw_data)
 [1] "ï.."                    "ID"                     "Name"                   "Age"                    "Photo"
 [6] "Nationality"            "Flag"                   "Overall"                "Potential"              "Club"
[11] "Club.Logo"              "Value"                  "Wage"                   "Special"                "Preferred.Foot"
[16] "International.Reputation" "Weak.Foot"              "Skill.Moves"            "Work.Rate"
"Body.Type"
[21] "Real.Face"              "Position"               "Jersey.Number"          "Joined"                 "Loaned.From"
[26] "Contract.Valid.Until"     "Height"                 "Weight"                 "LS"                     "ST"
[31] "RS"                     "LW"                     "LF"                     "CF"                     "RF"
[36] "RW"                     "LAM"                    "CAM"                    "RAM"                    "LM"
[41] "LCM"                    "CM"                     "RCM"                    "RM"                     "LWB"
[46] "LDM"                    "CDM"                    "RDM"                    "RWB"                    "LB"
[51] "LCB"                    "CB"                     "RCB"                    "RB"                     "Crossing"
[56] "Finishing"              "HeadingAccuracy"        "ShortPassing"           "Volleys"                "Dribbling"
[61] "Curve"                  "FKAccuracy"             "LongPassing"            "BallControl"            "Acceleration"
[66] "SprintSpeed"            "Agility"                "Reactions"              "Balance"                "ShotPower"
[71] "Jumping"                "Stamina"                "Strength"               "LongShots"              "Aggression"
```

```
[76] "Interceptions"          "Positioning"           "Vision"               "Penalties"             "Composure"
[81] "Marking"                "StandingTackle"        "SlidingTackle"        "GKDiving"              "GKHandling"
[86] "GKKicking"              "GKPositioning"         "GKReflexes"           "Release.Clause"
>
> #Data types & Dimensions
> str(raw_data)
'data.frame':	18207 obs. of  89 variables:
 $ ï..                     : int  0 1 2 3 4 5 6 7 8 9 ...
 $ ID                      : int  158023 20801 190871 193080 192985 183277 177003 176580 155862 200389 ...
 $ Name                    : chr  "L. Messi" "Cristiano Ronaldo" "Neymar Jr" "De Gea" ...
 $ Age                     : int  31 33 26 27 27 27 32 31 32 25 ...
 $ Photo                   : chr  "https://cdn.sofifa.org/players/4/19/158023.png"
"https://cdn.sofifa.org/players/4/19/20801.png" "https://cdn.sofifa.org/players/4/19/190871.png"
"https://cdn.sofifa.org/players/4/19/193080.png" ...
 $ Nationality             : chr  "Argentina" "Portugal" "Brazil" "Spain" ...
 $ Flag                    : chr  "https://cdn.sofifa.org/flags/52.png" "https://cdn.sofifa.org/flags/38.png"
"https://cdn.sofifa.org/flags/54.png" "https://cdn.sofifa.org/flags/45.png" ...
 $ Overall                 : int  94 94 92 91 91 91 91 91 91 90 ...
 $ Potential               : int  94 94 93 93 92 91 91 91 91 93 ...
 $ Club                    : chr  "FC Barcelona" "Juventus" "Paris Saint-Germain" "Manchester United" ...
 $ Club.Logo               : chr  "https://cdn.sofifa.org/teams/2/light/241.png"
"https://cdn.sofifa.org/teams/2/light/45.png" "https://cdn.sofifa.org/teams/2/light/73.png"
"https://cdn.sofifa.org/teams/2/light/11.png" ...
 $ Value                   : chr  "â‚¬110.5M" "â‚¬77M" "â‚¬118.5M" "â‚¬72M" ...
 $ Wage                    : chr  "â‚¬565K" "â‚¬405K" "â‚¬290K" "â‚¬260K" ...
 $ Special                 : int  2202 2228 2143 1471 2281 2142 2280 2346 2201 1331 ...
 $ Preferred.Foot          : chr  "Left" "Right" "Right" "Right" ...
 $ International.Reputation: int  5 5 5 4 4 4 4 5 4 3 ...
 $ Weak.Foot               : int  4 4 5 3 5 4 4 4 3 3 ...
 $ Skill.Moves             : int  4 5 5 1 4 4 4 3 3 1 ...
 $ Work.Rate               : chr  "Medium/ Medium" "High/ Low" "High/ Medium" "Medium/ Medium" ...
 $ Body.Type               : chr  "Messi" "C. Ronaldo" "Neymar" "Lean" ...
 $ Real.Face               : chr  "Yes" "Yes" "Yes" "Yes" ...
 $ Position                : chr  "RF" "ST" "LW" "GK" ...
 $ Jersey.Number           : int  10 7 10 1 7 10 10 9 15 1 ...
 $ Joined                  : chr  "Jul 1, 2004" "Jul 10, 2018" "Aug 3, 2017" "Jul 1, 2011" ...
 $ Loaned.From             : chr  "" "" "" "" ...
 $ Contract.Valid.Until    : chr  "2021" "2022" "2022" "2020" ...
 $ Height                  : chr  "5'7" "6'2" "5'9" "6'4" ...
 $ Weight                  : chr  "159lbs" "183lbs" "150lbs" "168lbs" ...
 $ LS                      : chr  "88+2" "91+3" "84+3" "" ...
 $ ST                      : chr  "88+2" "91+3" "84+3" "" ...
 $ RS                      : chr  "88+2" "91+3" "84+3" "" ...
 $ LW                      : chr  "92+2" "89+3" "89+3" "" ...
```

```
$ LF                 : chr  "93+2" "90+3" "89+3" "" ...
$ CF                 : chr  "93+2" "90+3" "89+3" "" ...
$ RF                 : chr  "93+2" "90+3" "89+3" "" ...
$ RW                 : chr  "92+2" "89+3" "89+3" "" ...
$ LAM                : chr  "93+2" "88+3" "89+3" "" ...
$ CAM                : chr  "93+2" "88+3" "89+3" "" ...
$ RAM                : chr  "93+2" "88+3" "89+3" "" ...
$ LM                 : chr  "91+2" "88+3" "88+3" "" ...
$ LCM                : chr  "84+2" "81+3" "81+3" "" ...
$ CM                 : chr  "84+2" "81+3" "81+3" "" ...
$ RCM                : chr  "84+2" "81+3" "81+3" "" ...
$ RM                 : chr  "91+2" "88+3" "88+3" "" ...
$ LWB                : chr  "64+2" "65+3" "65+3" "" ...
$ LDM                : chr  "61+2" "61+3" "60+3" "" ...
$ CDM                : chr  "61+2" "61+3" "60+3" "" ...
$ RDM                : chr  "61+2" "61+3" "60+3" "" ...
$ RWB                : chr  "64+2" "65+3" "65+3" "" ...
$ LB                 : chr  "59+2" "61+3" "60+3" "" ...
$ LCB                : chr  "47+2" "53+3" "47+3" "" ...
$ CB                 : chr  "47+2" "53+3" "47+3" "" ...
$ RCB                : chr  "47+2" "53+3" "47+3" "" ...
$ RB                 : chr  "59+2" "61+3" "60+3" "" ...
$ Crossing           : int  84 84 79 17 93 81 86 77 66 13 ...
$ Finishing          : int  95 94 87 13 82 84 72 93 60 11 ...
$ HeadingAccuracy    : int  70 89 62 21 55 61 55 77 91 15 ...
$ ShortPassing       : int  90 81 84 50 92 89 93 82 78 29 ...
$ Volleys            : int  86 87 84 13 82 80 76 88 66 13 ...
$ Dribbling          : int  97 88 96 18 86 95 90 87 63 12 ...
$ Curve              : int  93 81 88 21 85 83 85 86 74 13 ...
$ FKAccuracy         : int  94 76 87 19 83 79 78 84 72 14 ...
$ LongPassing        : int  87 77 78 51 91 83 88 64 77 26 ...
$ BallControl        : int  96 94 95 42 91 94 93 90 84 16 ...
$ Acceleration       : int  91 89 94 57 78 94 80 86 76 43 ...
$ SprintSpeed        : int  86 91 90 58 76 88 72 75 75 60 ...
$ Agility            : int  91 87 96 60 79 95 93 82 78 67 ...
$ Reactions          : int  95 96 94 90 91 90 90 92 85 86 ...
$ Balance            : int  95 70 84 43 77 94 94 83 66 49 ...
$ ShotPower          : int  85 95 80 31 91 82 79 86 79 22 ...
$ Jumping            : int  68 95 61 67 63 56 68 69 93 76 ...
$ Stamina            : int  72 88 81 43 90 83 89 90 84 41 ...
$ Strength           : int  59 79 49 64 75 66 58 83 83 78 ...
$ LongShots          : int  94 93 82 12 91 80 82 85 59 12 ...
$ Aggression         : int  48 63 56 38 76 54 62 87 88 34 ...
$ Interceptions      : int  22 29 36 30 61 41 83 41 90 19 ...
```

```
 $ Positioning           : int  94 95 89 12 87 87 79 92 60 11 ...
 $ Vision                : int  94 82 87 68 94 89 92 84 63 70 ...
 $ Penalties             : int  75 85 81 40 79 86 82 85 75 11 ...
 $ Composure             : int  96 95 94 68 88 91 84 85 82 70 ...
 $ Marking               : int  33 28 27 15 68 34 60 62 87 27 ...
 $ StandingTackle        : int  28 31 24 21 58 27 76 45 92 12 ...
 $ SlidingTackle         : int  26 23 33 13 51 22 73 38 91 18 ...
 $ GKDiving              : int  6 7 9 90 15 11 13 27 11 86 ...
 $ GKHandling            : int  11 11 9 85 13 12 9 25 8 92 ...
 $ GKKicking             : int  15 15 15 87 5 6 7 31 9 78 ...
 $ GKPositioning         : int  14 14 15 88 10 8 14 33 7 88 ...
 $ GKReflexes            : int  8 11 11 94 13 8 9 37 11 89 ...
 $ Release.Clause        : chr  "â‚¬226.5M" "â‚¬127.1M" "â‚¬228.1M" "â‚¬138.6M" ...
> dim(raw_data)
[1] 18207    89
> nrow(raw_data)
[1] 18207
> ncol(raw_data)
[1] 89
>
> #Data Manipulation
> df <- raw_data[, c(2:4, 6, 8, 9, 10, 12, 13, 15:23, 27, 28, 55:60, 66:74, 89)]
> head(df)
      ID              Name Age Nationality Overall Potential                Club    Value    Wage Preferred.Foot
International.Reputation Weak.Foot
1 158023          L. Messi  31   Argentina      94        94        FC Barcelona â‚¬110.5M â‚¬565K           Left
5         4
2  20801 Cristiano Ronaldo  33    Portugal      94        94            Juventus    â‚¬77M â‚¬405K          Right
5         4
3 190871         Neymar Jr  26      Brazil      92        93 Paris Saint-Germain â‚¬118.5M â‚¬290K          Right
5         5
4 193080            De Gea  27       Spain      91        93   Manchester United    â‚¬72M â‚¬260K          Right
4         3
5 192985      K. De Bruyne  27     Belgium      91        92     Manchester City   â‚¬102M â‚¬355K          Right
4         5
6 183277         E. Hazard  27     Belgium      91        91             Chelsea    â‚¬93M â‚¬340K          Right
4         4
  Skill.Moves      Work.Rate  Body.Type Real.Face Position Jersey.Number Height Weight Crossing
Finishing HeadingAccuracy ShortPassing Volleys Dribbling
1           4 Medium/ Medium      Messi       Yes       RF            10    5'7 159lbs       84       95              70           90
86        97
2           5      High/ Low C. Ronaldo       Yes       ST             7    6'2 183lbs       84       94              89           81
87        88
```

```
3          5   High/ Medium     Neymar       Yes       LW          10    5'9 150lbs       79       87            62          84
84       96
4          1 Medium/ Medium       Lean       Yes       GK           1    6'4 168lbs       17       13            21          50
13       18
5          4    High/ High     Normal       Yes      RCM           7   5'11 154lbs      93       82            55          92
82       86
6          4   High/ Medium     Normal       Yes       LF          10    5'8 163lbs       81       84            61          89
80       95
  SprintSpeed Agility Reactions Balance ShotPower Jumping Stamina Strength LongShots Release.Clause
1        86     91       95      95       85      68      72      59       94      â‚¬226.5M
2        91     87       96      70       95      95      88      79       93      â‚¬127.1M
3        90     96       94      84       80      61      81      49       82      â‚¬228.1M
4        58     60       90      43       31      67      43      64       12      â‚¬138.6M
5        76     79       91      77       91      63      90      75       91      â‚¬196.4M
6        88     95       90      94       82      56      83      66       80      â‚¬172.1M
> #Create League variable & Sampling
> df <- raw_data
> bundesliga <- c("1. FC Nürnberg", "1. FSV Mainz 05", "Bayer 04 Leverkusen", "FC Bayern München",
+               "Borussia Dortmund", "Borussia Mönchengladbach", "Eintracht Frankfurt",
+               "FC Augsburg", "FC Schalke 04", "Fortuna Düsseldorf", "Hannover 96",
+               "Hertha BSC", "RB Leipzig", "SC Freiburg", "TSG 1899 Hoffenheim",
+               "VfB Stuttgart", "VfL Wolfsburg", "SV Werder Bremen")
> premierLeague <- c("Arsenal", "Bournemouth", "Brighton & Hove Albion", "Burnley",
+                  "Cardiff City", "Chelsea", "Crystal Palace", "Everton", "Fulham",
+                  "Huddersfield Town", "Leicester City", "Liverpool", "Manchester City",
+                  "Manchester United", "Newcastle United", "Southampton",
+                  "Tottenham Hotspur", "Watford", "West Ham United", "Wolverhampton Wanderers")
> laliga <- c("Athletic Club de Bilbao", "Atlético Madrid", "CD Leganés",
+           "Deportivo Alavés", "FC Barcelona", "Getafe CF", "Girona FC",
+           "Levante UD", "Rayo Vallecano", "RC Celta", "RCD Espanyol",
+           "Real Betis", "Real Madrid", "Real Sociedad", "Real Valladolid CF",
+           "SD Eibar", "SD Huesca", "Sevilla FC", "Valencia CF", "Villarreal CF")
> seriea <- c("Atalanta","Bologna","Cagliari","Chievo Verona","Empoli",
"Fiorentina","Frosinone","Genoa",
+           "Inter","Juventus","Lazio","Milan","Napoli","Parma","Roma","Sampdoria","Sassuolo","SPAL",
+           "Torino","Udinese")
> superlig <- c("Akhisar Belediyespor","Alanyaspor", "Antalyaspor","Medipol Başakşehir FK","BB
Erzurumspor","Beşiktaş JK",
+             "Bursaspor","Çaykur Rizespor","Fenerbahçe SK", "Galatasaray SK","Göztepe SK","Kasimpaşa
SK",
+             "Kayserispor","Atiker Konyaspor","MKE Ankaragücü", "Sivasspor","Trabzonspor","Yeni
Malatyaspor")
> ligue1 <- c("Amiens SC", "Angers SCO", "AS Monaco", "AS Saint-Étienne", "Dijon FCO", "En Avant de
Guingamp",
```

```
+           "FC Nantes", "FC Girondins de Bordeaux", "LOSC Lille", "Montpellier HSC", "Nîmes Olympique",
+           "OGC Nice", "Olympique Lyonnais","Olympique de Marseille", "Paris Saint-Germain",
+           "RC Strasbourg Alsace", "Stade Malherbe Caen", "Stade de Reims", "Stade Rennais FC",
"Toulouse Football Club")
> eredivisie <- c("ADO Den Haag","Ajax", "AZ Alkmaar", "De Graafschap","Excelsior","FC Emmen","FC
Groningen",
+              "FC Utrecht", "Feyenoord","Fortuna Sittard", "Heracles Almelo","NAC Breda",
+              "PEC Zwolle", "PSV","SC Heerenveen","Vitesse","VVV-Venlo","Willem II")
> liganos <- c("Os Belenenses", "Boavista FC", "CD Feirense", "CD Tondela", "CD Aves", "FC Porto",
+            "CD Nacional", "GD Chaves", "Clube Sport Marítimo", "Moreirense FC", "Portimonense SC",
"Rio Ave FC",
+            "Santa Clara", "SC Braga", "SL Benfica", "Sporting CP", "Vitória Guimarães", "Vitória de
Setúbal")
> df%<>%mutate(
+   League = case_when(
+     Club %in% bundesliga ~ "Bundesliga",
+     Club %in% premierLeague ~ "Premier League",
+     Club %in% laliga ~ "La Liga",
+     Club %in% seriea ~ "Serie A",
+     Club %in% superlig ~ "Süper Lig",
+     Club %in% ligue1 ~ "Ligue 1",
+     Club %in% liganos ~ "Liga Nos",
+     Club %in% eredivisie ~ "Eredivisie"
+   ),
+   Country = case_when(
+     League == "Bundesliga" ~ "Germany",
+     League == "Premier League" ~ "UK",
+     League == "La Liga" ~ "Spain",
+     League == "Serie A" ~ "Italy",
+     League == "Süper Lig" ~ "Turkey",
+     League == "Ligue 1" ~ "France",
+     League == "Liga Nos" ~ "Portugal",
+     League == "Eredivisie" ~ "Netherlands"
+   )
+ ) %>% filter(!is.na(League)) %>% mutate_if(is.factor, as.character)
> rm(bundesliga, premierLeague, laliga, seriea, superlig, ligue1, eredivisie, liganos)
> #String Manipulation
> head(df$Value)
[1] "â,¬110.5M" "â,¬77M"    "â,¬118.5M" "â,¬72M"    "â,¬102M"   "â,¬93M"
> df$Values <- str_remove_all(df$Value, "€")   #Player values
> df$Values <- str_replace_all(df$Values, "K", "000")
> df$Values <- str_remove_all(df$Values, "M")
> df$Values <- as.numeric(df$Values)
> head(df$Values)
```

```
[1] 110.5  77.0 118.5  72.0 102.0  93.0
> df$Wages <- str_remove_all(df$Wage, "€")   #Player wages
> df$Wages <- str_replace_all(df$Wages, "K", "000")
> df$Wages <- as.numeric(df$Wages)
> head(df$Wages)
[1] 565000 405000 290000 260000 355000 340000
> data_1 <- df%>%mutate(Values = if_else(df$Values < 1000, Values * 1000000, Values))  #Million
Tranformation
> #Create Position class
> unique(data_1$Position)
 [1] "RF"  "ST"  "LW"  "GK"  "RCM" "LF"  "RS"  "RCB" "LCM" "LDM" "CDM" "LS"  "LCB" "RM"  "CAM" "LM"
"LB"  "CB"  "RDM" "RW"  "RB"  "CM"  "RAM" "CF"
[25] "LAM" "RWB" "LWB" ""
> defence <- c("CB", "RB", "LB", "LWB", "RWB", "LCB", "RCB")
> midfielder <- c("CM", "CDM","CAM","LM","RM", "LAM", "RAM", "LCM", "RCM", "LDM", "RDM")
> data_2 <- data_1
> data_2 %<>% mutate(Class = if_else(Position%in%"GK", "Goal Keeper",
+                          if_else(Position%in%defence, "Defender",
+                              if_else(Position%in%midfielder, "Midfielder", "Forward"))))
> rm(defence, midfielder)
> head(data_2$Class)
[1] "Forward"     "Forward"     "Forward"     "Goal Keeper" "Midfielder"  "Forward"
> #Height & Weight
> data_3 <- data_2
> data_3 %<>% mutate(Height = round((as.numeric(str_sub(Height, start=1, end=1))*30.48) +
(as.numeric(str_sub(Height, start=3, end=5))*2.54)),
+                  Weight = round(as.numeric(str_sub(Weight, start=1, end=3)) / 2.204623))
> head(data_3)
  ï..     ID            Name Age                                Photo Nationality                          Flag Overall
Potential
1   0 158023         L. Messi  31 https://cdn.sofifa.org/players/4/19/158023.png   Argentina
https://cdn.sofifa.org/flags/52.png      94       94
2   1  20801 Cristiano Ronaldo  33  https://cdn.sofifa.org/players/4/19/20801.png    Portugal
https://cdn.sofifa.org/flags/38.png      94       94
3   2 190871         Neymar Jr  26 https://cdn.sofifa.org/players/4/19/190871.png      Brazil
https://cdn.sofifa.org/flags/54.png      92       93
4   3 193080           De Gea  27 https://cdn.sofifa.org/players/4/19/193080.png       Spain
https://cdn.sofifa.org/flags/45.png      91       93
5   4 192985      K. De Bruyne  27 https://cdn.sofifa.org/players/4/19/192985.png     Belgium
https://cdn.sofifa.org/flags/7.png      91       92
6   5 183277         E. Hazard  27 https://cdn.sofifa.org/players/4/19/183277.png     Belgium
https://cdn.sofifa.org/flags/7.png      91       91
               Club                             Club.Logo     Value    Wage Special Preferred.Foot
International.Reputation Weak.Foot
```

1 FC Barcelona https://cdn.sofifa.org/teams/2/light/241.png â‚¬110.5M â‚¬565K 2202 Left
5 4
2 Juventus https://cdn.sofifa.org/teams/2/light/45.png â‚¬77M â‚¬405K 2228 Right
5 4
3 Paris Saint-Germain https://cdn.sofifa.org/teams/2/light/73.png â‚¬118.5M â‚¬290K 2143 Right
5 5
4 Manchester United https://cdn.sofifa.org/teams/2/light/11.png â‚¬72M â‚¬260K 1471 Right
4 3
5 Manchester City https://cdn.sofifa.org/teams/2/light/10.png â‚¬102M â‚¬355K 2281 Right
4 5
6 Chelsea https://cdn.sofifa.org/teams/2/light/5.png â‚¬93M â‚¬340K 2142 Right
4 4

Skill.Moves Work.Rate Body.Type Real.Face Position Jersey.Number Joined Loaned.From
Contract.Valid.Until Height Weight LS ST RS LW
1 4 Medium/ Medium Messi Yes RF 10 Jul 1, 2004 2021 170 72
88+2 88+2 88+2 92+2
2 5 High/ Low C. Ronaldo Yes ST 7 Jul 10, 2018 2022 188 83
91+3 91+3 91+3 89+3
3 5 High/ Medium Neymar Yes LW 10 Aug 3, 2017 2022 175 68
84+3 84+3 84+3 89+3
4 1 Medium/ Medium Lean Yes GK 1 Jul 1, 2011 2020 193 76
5 4 High/ High Normal Yes RCM 7 Aug 30, 2015 2023 180 70
82+3 82+3 82+3 87+3
6 4 High/ Medium Normal Yes LF 10 Jul 1, 2012 2020 173 74
83+3 83+3 83+3 89+3

LF CF RF RW LAM CAM RAM LM LCM CM RCM RM LWB LDM CDM RDM RWB LB LCB CB
RCB RB Crossing Finishing HeadingAccuracy
1 93+2 93+2 93+2 92+2 93+2 93+2 93+2 91+2 84+2 84+2 84+2 91+2 64+2 61+2 61+2 61+2 64+2 59+2
47+2 47+2 47+2 59+2 84 95 70
2 90+3 90+3 90+3 89+3 88+3 88+3 88+3 88+3 81+3 81+3 81+3 88+3 65+3 61+3 61+3 61+3 65+3 61+3
53+3 53+3 53+3 61+3 84 94 89
3 89+3 89+3 89+3 89+3 89+3 89+3 89+3 88+3 81+3 81+3 81+3 88+3 65+3 60+3 60+3 60+3 65+3 60+3
47+3 47+3 47+3 60+3 79 87 62
4 17 13 21
5 87+3 87+3 87+3 87+3 88+3 88+3 88+3 88+3 87+3 87+3 87+3 88+3 77+3 77+3 77+3 77+3 77+3 73+3
66+3 66+3 66+3 73+3 93 82 55
6 88+3 88+3 88+3 89+3 89+3 89+3 89+3 89+3 82+3 82+3 82+3 89+3 66+3 63+3 63+3 63+3 66+3 60+3
49+3 49+3 49+3 60+3 81 84 61

ShortPassing Volleys Dribbling Curve FKAccuracy LongPassing BallControl Acceleration SprintSpeed
Agility Reactions Balance ShotPower Jumping Stamina
1 90 86 97 93 94 87 96 91 86 91 95 95 85 68
72
2 81 87 88 81 76 77 94 89 91 87 96 70 95 95
88

```
3       84     84      96   88       87        78        95         94        90     96       94     84       80     61
81
4       50     13      18   21       19        51        42         57        58     60       90     43       31     67
43
5       92     82      86   85       83        91        91         78        76     79       91     77       91     63
90
6       89     80      95   83       79        83        94         94        88     95       90     94       82     56
83
  Strength LongShots Aggression Interceptions Positioning Vision Penalties Composure Marking
StandingTackle SlidingTackle GKDiving GKHandling GKKicking
1      59       94       48          22        94    94      75      96     33          28          26      6       11       15
2      79       93       63          29        95    82      85      95     28          31          23      7       11       15
3      49       82       56          36        89    87      81      94     27          24          33      9        9       15
4      64       12       38          30        12    68      40      68     15          21          13     90       85       87
5      75       91       76          61        87    94      79      88     68          58          51     15       13        5
6      66       80       54          41        87    89      86      91     34          27          22     11       12        6
  GKPositioning GKReflexes Release.Clause        League Country    Values  Wages      Class
1          14         8     â‚¬226.5M       La Liga   Spain 110500000 565000     Forward
2          14        11      â‚¬127.1M       Serie A   Italy  77000000 405000     Forward
3          15        11      â‚¬228.1M       Ligue 1  France 118500000 290000     Forward
4          88        94      â‚¬138.6M Premier League      UK  72000000 260000 Goal Keeper
5          10        13      â‚¬196.4M Premier League      UK 102000000 355000  Midfielder
6           8         8     â‚¬172.1M Premier League      UK  93000000 340000     Forward
> #Correction of Preferred Foot variable
> data_4 <- data_3
> data_4 %<>% filter(Preferred.Foot%in%c("Left", "Right"))
> data_4$Preferred.Foot <- as.factor(as.character(data_4$Preferred.Foot))
> unique(data_4$Preferred.Foot)
[1] Left  Right
Levels: Left Right
> #Rename some variables
> data_5 <- data_4
> data_5 %<>% rename(
+   "Heading.Accuracy" = HeadingAccuracy,
+   "Short.Passing" = ShortPassing,
+   "FK.Accuracy" = FKAccuracy,
+   "Long.Passing" = LongPassing,
+   "Ball.Control" = BallControl,
+   "Sprint.Speed" = SprintSpeed,
+   "Shot.Power" = ShotPower,
+   "Long.Shots"= LongShots,
+   "Standing.Tackle"= StandingTackle,
+   "Sliding.Tackle"= SlidingTackle,
+   "GK.Diving"= GKDiving,
```

```
+   "GK.Handling"= GKHandling,
+   "GK.Kicking"= GKKicking,
+   "GK.Positioning"= GKPositioning,
+   "GK.Reflexes"= GKReflexes
+ )
> #Remove unnecessary values
> data_6 <- data_5
> data_6 %<>% select(-ID, -Body.Type, -Real.Face, -Joined, -Loaned.From, -Release.Clause, -Photo, -Flag,
-Special, -Work.Rate)
> data_manipulated <- data_6
>
> #Dealing with Missing Values
> colSums(is.na(data_manipulated))
              ï..                  Name                   Age           Nationality               Overall             Potential
                0                     0                     0                     0                     0                     0
             Club             Club.Logo                 Value                  Wage        Preferred.Foot
International.Reputation
                0                     0                     0                     0                     0                     0
        Weak.Foot           Skill.Moves              Position         Jersey.Number   Contract.Valid.Until
Height
                0                     0                     0                     0                     0                     0
           Weight                    LS                    ST                    RS                    LW                    LF
                0                     0                     0                     0                     0                     0
               CF                    RF                    RW                   LAM                   CAM                   RAM
                0                     0                     0                     0                     0                     0
               LM                   LCM                    CM                   RCM                    RM                   LWB
                0                     0                     0                     0                     0                     0
              LDM                   CDM                   RDM                   RWB                    LB                   LCB
                0                     0                     0                     0                     0                     0
               CB                   RCB                    RB              Crossing             Finishing
Heading.Accuracy
                0                     0                     0                     0                     0                     0
    Short.Passing               Volleys             Dribbling                 Curve           FK.Accuracy
Long.Passing
                0                     0                     0                     0                     0                     0
     Ball.Control          Acceleration          Sprint.Speed               Agility             Reactions
Balance
                0                     0                     0                     0                     0                     0
       Shot.Power               Jumping               Stamina              Strength            Long.Shots
Aggression
                0                     0                     0                     0                     0                     0
    Interceptions           Positioning                Vision             Penalties             Composure
Marking
                0                     0                     0                     0                     0                     0
```

```
    Standing.Tackle     Sliding.Tackle         GK.Diving       GK.Handling        GK.Kicking
GK.Positioning
              0              0              0              0              0              0
      GK.Reflexes           League         Country           Values            Wages
Class
              0              0              0              0              0              0
>
> #Visualization
> #Distribution & Avg. of Players in each League
> summ <- data_manipulated %>% group_by(League) %>% summarise(age=mean(Age), .groups='drop')
> summ
# A tibble: 8 x 2
  League         age
  <chr>         <dbl>
1 Bundesliga      24.2
2 Eredivisie      23.3
3 La Liga         24.8
4 Liga Nos        25.5
5 Ligue 1         24.3
6 Premier League  24.6
7 Serie A         25.7
8 Süper Lig       26.5
> options(repr.plot.width = 12, repr.plot.height = 8)
> ggplot()+
+   geom_histogram(data_manipulated, mapping = aes(Age, fill = League),bins=10)+
+   geom_vline(summ, mapping = aes(xintercept = age), color = "red", size = 1.5)+
+   geom_text(summ, mapping = aes(x = age+3, y = 65, label = round(age,digits = 2)))+
+   facet_wrap(League~.)+
+   theme_minimal()+
+   theme(legend.position = "bottom")+
+   labs(y = "Frequency", title = "Distribution & The Average Age of The Players in each League", caption
= "@EA Sports - FIFA 19")
> #Distribution of Total Market value in each League
> data_manipulated %>%
+   group_by(League) %>%
+   summarise(Total.Value = sum(as.numeric(Values), na.rm = TRUE),.groups = 'drop') %>%
+   ggplot(aes(reorder(League, Total.Value), Total.Value, fill = Total.Value))+
+   geom_col(show.legend = FALSE)+
+   coord_flip()+
+   theme_minimal()+
+   labs(x = NULL, y = "Market Values of rhe Leagues")+
+   scale_fill_gradient(low = "khaki", high = "seagreen")+
+   theme(axis.line.y = element_line(colour = "darkslategray"),
+         axis.ticks.x = element_line(colour = "darkslategray"))+
```

```
+   scale_y_continuous(labels = c("0 €", "2 Billion €", "4 Billion €", "6 Billion €"))
> #Interactive World Map & No. of Players
> options(repr.plot.width = 12, repr.plot.height = 8)
> world_map <- map_data("world")
> numofplayers <- world_map %>%
+   mutate(region = as.character(region)) %>%
+   left_join((data_manipulated %>% mutate(Nationality = as.character(Nationality),
+                                  Nationality = if_else(Nationality %in% "England",
+                                                        "UK", Nationality)) %>%
+             #filter(League == "Bundesliga") %>%
+             count(Nationality, name = "Number of Player") %>%
+             rename(region = Nationality) %>%
+             mutate(region = as.character(region))), by = "region")
> ggplot(numofplayers, aes(long, lat, group = group))+
+   geom_polygon(aes(fill = `Number of Player` ), color = "white", show.legend = FALSE)+
+   scale_fill_viridis_c(option = "C")+
+   theme_void()+
+   labs(fill = "Number of Player",
+        title = "Number of Player in FIFA 19")
> #Comparison of 2 players
> # Selection of the players
> players <- data_manipulated %>%
+   filter(Name %in% c("Cristiano Ronaldo", "L. Messi")) %>%
+   # Unite Name & Club variables
+   mutate(Name = paste0(Name, ", ", Club)) %>%
+   # Selection abilities of the players
+   select(Name,Crossing:Sliding.Tackle) %>%
+   # Correction of the punctuation
+   rename_all(funs(gsub("[[:punct:]]", " ", .))) %>%
+   # Tranform from Variable to Observation
+   gather(Skill, Exp, Crossing:`Sliding Tackle`, -Name)
> head(players)
                      Name            Skill Exp
1     L. Messi, FC Barcelona         Crossing  84
2 Cristiano Ronaldo, Juventus        Crossing  84
3     L. Messi, FC Barcelona        Finishing  95
4 Cristiano Ronaldo, Juventus       Finishing  94
5     L. Messi, FC Barcelona Heading Accuracy  70
6 Cristiano Ronaldo, Juventus Heading Accuracy  89
> options(repr.plot.width = 15, repr.plot.height = 8)
> ggplot(players, aes(Skill, Exp, fill = Name))+
+   geom_col(show.legend = FALSE)+
+   coord_flip()+
+   facet_wrap(Name~.)+
```

```
+   scale_fill_manual(values = c("black", "navy"))+
+   theme_minimal()
> options(repr.plot.width = 15, repr.plot.height = 8)
> ggplot(players, aes(Skill, Exp, fill = Name))+
+   geom_col(position = "fill")+
+   coord_flip()+
+   scale_fill_manual(values = c("black", "red"))+
+   theme_minimal()+
+   geom_hline(yintercept = 0.5, color = "white", size = 1, linetype = 2)+
+   theme(legend.position = "top", axis.text.x=element_blank())+
+   labs(title = "Messi VS Ronaldo")
> #BMI - Body Mass Index
> #1. Below the Ideal Weight: < 18.49
> #2. Ideal Weight: 18.5 - 24.99
> #3. Over the Ideal Weight: 25 - 29.99
> #4. Much Over The Ideal Weight: > 30
> unique(data_manipulated$Club)
  [1] "FC Barcelona"            "Juventus"                "Paris Saint-Germain"     "Manchester United"
"Manchester City"
  [6] "Chelsea"                 "Real Madrid"             "Tottenham Hotspur"       "Liverpool"               "Napoli"
 [11] "Arsenal"                 "Milan"                   "Inter"                   "Lazio"                   "Borussia Dortmund"
 [16] "Olympique Lyonnais"      "Roma"                    "Valencia CF"             "FC Porto"                "FC Schalke
04"
 [21] "Sporting CP"             "Real Betis"              "Olympique de Marseille"  "RC Celta"                "Bayer 04
Leverkusen"
 [26] "Real Sociedad"           "Villarreal CF"           "Sevilla FC"              "SL Benfica"              "AS Monaco"
 [31] "Leicester City"          "Atalanta"                "RB Leipzig"              "Ajax"                    "Everton"
 [36] "West Ham United"         "TSG 1899 Hoffenheim"     "OGC Nice"                "Wolverhampton
Wanderers"  "Hertha BSC"
 [41] "SV Werder Bremen"        "Athletic Club de Bilbao"  "Torino"                 "Crystal Palace"          "VfL
Wolfsburg"
 [46] "Sassuolo"                "PSV"                     "Levante UD"              "Fulham"                  "Watford"
 [51] "Montpellier HSC"         "Galatasaray SK"          "SD Eibar"                "Sampdoria"               "VfB
Stuttgart"
 [56] "SC Braga"                "Eintracht Frankfurt"     "Girona FC"               "Burnley"
"Southampton"
 [61] "Getafe CF"               "Chievo Verona"           "Genoa"                   "RCD Espanyol"            "Cagliari"
 [66] "1. FSV Mainz 05"         "Bournemouth"             "FC Augsburg"             "Fiorentina"              "FC
Nantes"
 [71] "Feyenoord"               "Brighton & Hove Albion"  "SC Freiburg"             "Stade Rennais FC"
"Trabzonspor"
 [76] "SPAL"                    "Portimonense SC"         "Newcastle United"        "Frosinone"               "Hannover
96"
```

```
 [81] "Stade Malherbe Caen"      "Toulouse Football Club"   "Huddersfield Town"       "CD Tondela"
"Rio Ave FC"
 [86] "FC Girondins de Bordeaux" "Parma"                  "RC Strasbourg Alsace"     "Bologna"
"Amiens SC"
 [91] "Udinese"                "Real Valladolid CF"      "Rayo Vallecano"         "En Avant de Guingamp"
"Akhisar Belediyespor"
 [96] "LOSC Lille"             "BB Erzurumspor"         "FC Groningen"           "Angers SCO"
"Antalyaspor"
[101] "Empoli"                 "VVV-Venlo"              "Alanyaspor"             "Cardiff City"           "Dijon FCO"
[106] "AZ Alkmaar"             "Willem II"              "Boavista FC"            "Atiker Konyaspor"       "GD
Chaves"
[111] "Stade de Reims"         "ADO Den Haag"           "SD Huesca"              "Vitesse"
"Kayserispor"
[116] "Yeni Malatyaspor"       "Bursaspor"              "Heracles Almelo"        "NAC Breda"
"Moreirense FC"
[121] "FC Utrecht"             "SC Heerenveen"          "Sivasspor"              "CD Feirense"            "CD Aves"
[126] "CD Nacional"            "Santa Clara"            "Fortuna Sittard"        "PEC Zwolle"             "Excelsior"
[131] "Os Belenenses"          "FC Emmen"               "De Graafschap"
> # Calculate BMI
> bmi <- data_manipulated %>%
+   filter(Club == "FC Barcelona") %>%
+   mutate(BMI = round(Weight/(Height/100)^2, digits = 4)) %>%
+   arrange(-BMI)%>%
+   select(Name, Age, Position, Class, Height, Weight, BMI)
> options(repr.plot.width = 12, repr.plot.height = 8)
> # Head & Tail Observations
> bmi2  <- rbind(
+   bmi %>% head(5) %>% mutate(BMI = BMI * -1),
+   bmi %>% tail(5)) %>% mutate(Type = if_else(BMI < 0, "Head", "Tail"))
> # BMI Visual
> bmi2 %>%
+   ggplot(aes(fct_reorder(paste(Name,",", Position), desc(BMI)), BMI))+
+   geom_col(aes(fill = Type))+
+   geom_text(aes(y = c(rep(-2,5), rep(2,5)),label = round(abs(BMI),digits = 2)),
+             color = "white", fontface = "bold", size = 4)+
+   coord_flip()+
+   theme_minimal()+
+   theme(axis.text.x = element_blank(),
+         legend.position = "top",
+         panel.background = element_rect(fill = "lightgray"),
+         panel.grid.minor = element_blank(),
+         axis.text = element_text(color = "slategray", face = "bold.italic",size = 12),
+         title = element_text(color = "slategray", face = "bold.italic",size = 20),
+         legend.box.background = element_rect(linetype = 2))+
```

```
+   labs(x = NULL, y = NULL, fill = NULL, title = "BMI Index")+
+   scale_fill_manual(values = c("steelblue", "khaki"))
>
> #Scale the Data
> numeric_feature <- Filter(is.numeric, data_manipulated)
> data_standardized <- data.frame(scale(numeric_feature))
> data_standardized
        ï..        Age  Overall Potential International.Reputation   Weak.Foot Skill.Moves Jersey.Number
Height      Weight   Crossing  Finishing
1  -1.087785  1.3717005 3.191279  3.050070              5.712935  1.39966159   1.6103701   -0.63100398 -
1.8365790 -0.553752087  1.4919060  2.1948857
2  -1.087585  1.8120458 3.191279  3.050070              5.712935  1.39966159   2.7646376   -0.79521543
0.9083646  0.988584356  1.4919060  2.1471341
3  -1.087386  0.2708374 2.909515  2.875068              5.712935  2.81949121   2.7646376   -0.63100398 -
1.0740947 -1.114601703  1.2407621  1.8128726
4  -1.087186  0.4910100 2.768633  2.875068              4.152847 -0.02016803  -1.8524324   -1.12363833
1.6708490  0.007097529 -1.8734215 -1.7207485
5  -1.086986  0.4910100 2.768633  2.700065              4.152847  2.81949121   1.6103701   -0.79521543 -
0.3116103 -0.834176895  1.9439649  1.5741144
6  -1.086787  0.4910100 2.768633  2.525063              4.152847  1.39966159   1.6103701   -0.63100398 -
1.3790884 -0.273327279  1.3412197  1.6696177
7  -1.086587  1.5918731 2.768633  2.525063              4.152847  1.39966159   1.6103701   -0.63100398 -
1.3790884 -1.395026511  1.5923635  1.0965981
8  -1.086388  1.3717005 2.768633  2.525063              5.712935  1.39966159   0.4561026   -0.68574113
0.1458803  1.409221568  1.1403046  2.0993824
9  -1.086188  1.5918731 2.768633  2.525063              4.152847 -0.02016803   0.4561026   -0.35731823
0.1458803  0.848371952  0.5877882  0.5235784
10 -1.085590  0.7111826 2.627750  2.350060              4.152847  2.81949121   0.4561026   -0.74047828
0.1458803  0.007097529  1.6928211  1.2876046
11 -1.085190  1.5918731 2.627750  2.350060              4.152847 -1.43999766   1.6103701   -0.02889533
-1.3790884 -1.254814107  1.4919060  1.2876046
12 -1.084991  0.4910100 2.486868  2.350060              2.592759 -0.02016803  -0.6981649   -0.46679253
-2.1415728 -0.553752087  0.6882457  0.7623366
13 -1.084791 -0.1695079 2.486868  3.050070              2.592759 -0.02016803   1.6103701   -0.02889533
-0.6166041 -0.133114875  1.3914484  1.6696177
14 -1.084592 -0.1695079 2.486868  2.525063              2.592759  1.39966159   0.4561026   -0.68574113
0.9083646  1.829858780  1.0398471  2.1471341
15 -1.084193  0.2708374 2.486868  2.700065              2.592759  1.39966159  -1.8524324    0.02584182
0.9083646  1.269009164 -1.9738790 -1.6729968
16 -1.083993  0.2708374 2.486868  2.350060              4.152847 -1.43999766  -1.8524324   -1.12363833
2.4333333  2.811345608 -2.0241078 -1.6729968
17 -1.083794  0.9313553 2.486868  2.175057              4.152847 -0.02016803   0.4561026   -0.90468973
0.9083646  0.007097529  0.3868731  0.8578399
```

18 -1.083594 1.3717005 2.486868 2.175057 4.152847 1.39966159 0.4561026 -0.02889533
0.4508740 0.147309933 0.7887032 1.9083759
19 -1.083195 1.1515279 2.486868 2.175057 4.152847 1.39966159 1.6103701 -0.63100398
-1.3790884 -0.834176895 0.7887032 2.0993824
20 -1.082995 1.8120458 2.486868 2.175057 4.152847 -0.02016803 -0.6981649 -1.01416403
0.9083646 1.269009164 0.1859580 -0.7657157
21 -1.082796 -1.2703711 2.345986 3.225073 2.592759 1.39966159 2.7646376 -0.63100398
-0.6166041 -0.413539683 1.1403046 1.8606242
Heading.Accuracy Short.Passing Volleys Dribbling Curve FK.Accuracy Long.Passing Ball.Control
Acceleration Sprint.Speed Agility Reactions
1 0.75369099 1.6848656 1.9863077 1.7997488 2.0295349 2.5052292 1.83445120 1.8047771
1.7115729 1.3810617 1.79984320 2.985502
2 1.76105880 1.1055559 2.0378096 1.3536297 1.4302396 1.5539810 1.21248899 1.6924545
1.5763900 1.7225829 1.52352339 3.093794
3 0.32953612 1.2986591 1.8833039 1.7501800 1.7798285 2.1352993 1.27468521 1.7486158
1.9143474 1.6542787 2.14524296 2.877210
4 -1.84425756 -0.8898441 -1.7733304 -2.1161850 -1.5662367 -1.4583048 -0.40461276 -
1.2279342 -0.5865377 -0.5314571 -0.34163533 2.444041
5 -0.04159938 1.8136010 1.7803002 1.2544922 1.6300047 1.9239108 2.08323609
1.5239705 0.8328836 0.6980193 0.97088377 2.552333
6 0.27651677 1.6204978 1.6772964 1.7006112 1.5301222 1.7125224 1.58566632 1.6924545
1.9143474 1.5176702 2.07616301 2.444041
7 -0.04159938 1.8779688 1.4712888 1.4527673 1.6300047 1.6596753 1.89664743
1.6362932 0.9680665 0.4248023 1.93800310 2.444041
8 1.12482650 1.1699236 2.0893115 1.3040610 1.6799460 1.9767580 0.40393811 1.4678092
1.3736155 0.6297150 1.17812362 2.660625
9 1.86709751 0.9124526 0.9562699 0.1144102 1.0806507 1.3425925 1.21248899 1.1308413
0.6977006 0.6297150 0.90180381 1.902580
10 -0.09461874 1.8136010 1.7803002 1.0066483 1.6799460 1.9767580 2.20762853
1.4678092 -0.1133973 -0.2582401 0.34916419 2.335749
11 -0.09461874 1.8779688 1.7803002 1.4031985 1.4801809 1.6068281 1.83445120
1.6924545 0.2921516 -0.1216317 1.86892315 2.444041
12 -0.09461874 1.4273946 0.4412510 0.9075107 -0.1678811 0.1271088 1.46127388
0.9061960 1.1032495 0.8346277 1.17812362 2.768917
13 0.64765227 1.4917623 2.0893115 1.5519049 1.7798285 2.1881464 1.08809655
1.5801318 1.4412070 1.1761490 1.79984320 2.010872
14 1.54898136 1.0411881 1.8833039 0.9570795 1.2804158 1.1312041 1.52347010
1.1308413 0.1569687 0.4248023 0.41824414 2.552333
15 -2.37445114 -1.7909925 -1.7218285 -2.1657538 -1.7160605 -1.8282346 -0.96437876 -
2.5758059 -1.8707760 -1.0778910 -1.93047424 1.902580
16 -2.26841243 -1.9840958 -1.8248323 -2.3640289 -1.6661193 -1.4054577 -1.39975230 -
2.2949993 -1.3300441 -0.9412826 -0.27255538 1.794288
17 0.64765227 1.6204978 -0.1767717 0.9570795 0.6811205 1.1312041 1.52347010
1.3554865 -1.0596782 -0.9412826 0.07284438 2.119164

18 1.76105880 0.9124526 2.1923153 0.9570795 1.2304745 1.5539810 -0.34241654
1.0185186 0.6301091 0.6980193 0.83272386 2.552333
19 1.12482650 1.1055559 1.9348058 1.4031985 1.4801809 1.3954397 0.40393811
1.4116479 1.5087985 0.9712362 1.45444343 2.444041
20 1.44294265 -0.3105345 -0.1252698 -0.1334337 0.3814729 -0.8241394 0.09295701 -
0.3855144 -0.1809888 0.6297150 -0.75611505 1.577704
21 1.12482650 1.1699236 1.5742926 1.4527673 1.2304745 0.8669685 0.96370411
1.5239705 2.0495304 2.0641041 1.86892315 2.119164
Balance Shot.Power Jumping Stamina Strength Long.Shots Aggression Interceptions
Positioning Vision Penalties Composure
1 2.1378487 1.34352854 0.15177264 0.48906217 -0.60256102 2.0376206 -0.64395656 -1.26756793
1.8930605 2.4020311 1.41774311 2.7751992
2 0.3973190 1.89288042 2.42074430 1.48199590 1.00233410 1.9893135 0.18980770 -0.95952933
1.9401677 1.6035537 2.02215398 2.6874373
3 1.3720156 1.06885260 -0.43647927 1.04758739 -1.40500859 1.4579349 -0.19928229 -
0.65149073 1.6575244 1.9362526 1.78038963 2.5996753
4 -1.4824530 -1.62297158 0.06773665 -1.31063020 -0.20133724 -1.9235649 -1.19979940 -
0.91552382 -1.9697314 0.6719968 -0.69769495 0.3178650
5 0.8846673 1.67313967 -0.26840730 1.60611261 0.68135508 1.8926992 0.91240339 0.44864713
1.5633100 2.4020311 1.65950746 2.0731037
6 2.0682275 1.17872298 -0.85665921 1.17170411 -0.04084773 1.3613207 -0.31045086 -
0.43146316 1.5633100 2.0693322 2.08259507 2.3363895
7 2.0682275 1.01391742 0.15177264 1.54405425 -0.68280578 1.4579349 0.13422341 1.41676844
1.1864522 2.2689515 1.84083072 1.7220560
8 1.3023945 1.39846373 0.23580863 1.60611261 1.32331312 1.6028563 1.52383051 -0.43146316
1.7988461 1.7366333 2.02215398 1.8098179
9 0.1188343 1.01391742 2.25267233 1.23376247 1.32331312 0.3468707 1.57941480 1.72480704
0.2914151 0.3392979 1.41774311 1.5465321
10 0.4669402 1.45339892 -3.04159488 0.67523725 0.52086556 1.9410063 0.02305485
1.37276293 1.1864522 1.8697129 1.29686093 1.8098179
11 1.7897428 0.62937110 -0.18437131 0.86141232 -1.16427432 1.1197850 -0.14369801 -
0.03541353 1.6575244 2.2689515 1.41774311 2.5119134
12 1.9289851 0.57443592 0.90809653 1.97846276 0.76159983 0.8299421 1.69058337
1.81281807 0.8095945 1.4039344 0.14848027 1.8098179
13 1.4416368 1.17872298 0.74002455 0.98552903 -0.12109249 1.7477778 -0.64395656 -
0.82751279 1.4219883 1.9362526 2.08259507 1.7220560
14 0.4669402 1.50833410 0.99213251 1.54405425 1.40355788 1.6028563 0.91240339 -
0.69549624 1.8459533 1.4704742 2.32435941 2.1608657
15 -1.4824530 -2.11738827 1.07616850 -1.80709707 1.00233410 -2.0201791 -0.92187798 -
1.26756793 -2.0168386 0.7385366 -1.60431126 0.4056270
16 -1.3432106 -1.34829564 0.15177264 -1.62092199 0.28013129 -1.6820292 -2.03356367 -
1.57560653 -1.9226242 -0.9249579 -1.48342908 0.1423412
17 -0.8558623 0.02508404 -0.01629934 1.35787918 0.84184459 0.1053350 1.41266195
1.59279050 1.0922378 1.9362526 0.51112680 2.2486276

18 -0.3685140 1.45339892 1.83249239 1.73022933 0.92208934 1.3130135 1.35707766 -0.12342456 1.8459533 1.2708548 2.02215398 1.5465321
19 1.8593639 1.50833410 1.24424048 0.73729560 0.52086556 1.5062421 0.30097627 -1.17955690 1.7988461 1.6700935 1.90127180 2.2486276
20 -0.6469987 0.95898223 1.91652838 0.05465367 1.80478166 -0.1362007 1.80175193 1.63679602 -1.2160159 -0.5257193 -0.09328407 1.7220560
21 1.3023945 1.01391742 0.74002455 1.17170411 0.36037605 1.2647064 0.13422341 -0.56347970 1.6104172 1.6035537 1.11553767 1.8975799
Marking Standing.Tackle Sliding.Tackle GK.Diving GK.Handling GK.Kicking GK.Positioning GK.Reflexes Values Wages
1 -0.81382432 -0.99693438 -0.98817606 -0.5749064 -0.3216783 -0.08855293 -0.1573438 -0.4700254 10.587303 13.398194
2 -1.04145761 -0.87043634 -1.11591110 -0.5236341 -0.3216783 -0.08855293 -0.1573438 -0.3181741 7.179220 9.419694
3 -1.08698427 -1.16559844 -0.69012766 -0.4210895 -0.4295405 -0.08855293 -0.1039817 -0.3181741 11.401173 6.560146
4 -1.63330419 -1.29209648 -1.54169454 3.7319654 3.6692221 3.96449911 3.7914527 3.8830465 6.670551 5.814177
5 0.77960877 0.26804604 0.07628253 -0.1134558 -0.2138161 -0.65147682 -0.3707923 -0.2169398 9.722565 8.176412
6 -0.76829766 -1.03910039 -1.15848944 -0.3185450 -0.2677472 -0.59518444 -0.4775165 -0.4700254 8.806961 7.803428
7 0.41539550 1.02703429 1.01300610 -0.2160004 -0.4295405 -0.53889205 -0.1573438 -0.4194083 6.161882 9.792678
8 0.50644882 -0.28011214 -0.47723594 0.5018116 0.4333569 0.81212530 0.8565364 0.9978709 7.484422 10.662975
9 1.64461531 1.70169052 1.77941629 -0.3185450 -0.4834716 -0.42630727 -0.5308786 -0.3181741 4.534142 8.798053
10 0.96171541 1.15353234 0.84269272 -0.3698172 -0.3216783 -0.20113771 -0.5308786 -0.3687912 7.128354 8.176412
11 0.36986884 0.05721597 -0.86044103 -0.5749064 -0.1059540 -0.53889205 -0.5842407 -0.2675569 5.449746 6.435818
12 1.78119529 1.65952450 1.52394623 -0.1134558 -0.2677472 -0.37001488 -0.5308786 -0.3687912 5.754947 4.943880
13 -1.26909091 -1.33426249 -1.24364613 -0.6261787 -0.6991959 -0.70776921 -0.6376029 -0.4700254 8.400026 4.446568
14 0.23328886 -0.65960627 -0.47723594 -0.4723618 -0.3756094 -0.31372249 -0.1573438 -0.3181741 7.840490 4.446568
15 -1.17803759 -1.62942459 -1.66942957 3.5781486 3.6692221 4.02079150 3.6313664 3.6805780 5.246278 5.316865
16 -1.40567089 -1.41859452 -1.41395950 3.4756040 3.9928086 3.12011327 3.6847285 3.5793438 4.788476 5.316865
17 1.78119529 1.44869443 1.31105451 -0.6261787 -0.4834716 -0.20113771 -0.4241544 -0.2169398 4.585009 7.181787

```
18  0.05118222     -0.28011214    -0.43465759 -0.2672727  -0.6452648 -0.20113771     -0.2107059  -
0.3687912  5.449746  4.322240
19 -0.95040430     -1.33426249    -1.58427288 -0.2160004  -0.1059540 -0.59518444     -0.3174302  -
0.1663227  5.907548  6.808803
20  1.91777527      1.74385653     1.73683795 -0.7287232  -0.7531270 -0.82035399     -0.6909650  -
0.7231110  2.092531  4.695224
21 -0.76829766     -0.74393830    -0.73270600 -0.2160004  -0.6452648 -0.53889205     -0.3174302  -
0.5712596  7.586156  1.835677
 [ reached 'max' / getOption("max.print") -- omitted 3851 rows ]
>
> #PCA
> data_pca <- prcomp(data_standardized,
+                  center = TRUE,
+                  scale. = TRUE)
> data_with_pca <- data.frame(as.matrix(data_standardized) %*% as.matrix(data_pca$rotation[,0:5]))
> data_with_pca
           PC1         PC2          PC3          PC4          PC5
1   -10.37151575  6.147904405 -11.26033150  2.694502132 -4.928103442
2    -9.32280116  7.023755019  -8.71100439 -0.823766672 -4.905720323
3    -9.50109278  4.311339068 -10.07240029  2.610463230 -4.461507817
4     6.83389697  8.455722811  -8.84177747  3.953419923 -3.060405336
5    -9.55848687  6.260124969  -7.12074247  2.023594038 -3.297437015
6    -9.02375298  4.004977409  -8.71030720  1.937465119 -3.905128241
7    -9.32657398  5.366431585  -6.18869269  3.920881421 -1.844468465
8    -7.86225118  7.820741740  -8.37111550  0.501872271 -3.762076060
9    -6.83254899  8.064057028  -2.12749246  1.704756898 -3.321272987
10   -7.98123670  6.164900996  -5.48511996  1.436842158 -1.359437987
11   -7.90941206  3.767611688  -6.48132487  2.449858047 -0.955387126
12   -6.69552177  5.679469373  -2.22525960  4.389718501 -2.395145571
13   -8.14738458  2.658595397  -7.52079879  0.461325534 -4.070270075
14   -7.23863956  5.741238435  -5.46658281 -1.655646889 -3.835411715
15    8.61543543  8.194210071  -7.64228019  2.708232248 -2.782813134
16    8.83262967  8.216708563  -7.62863384  2.019892341 -3.186838472
17   -5.74761186  7.674318988  -2.17072314  1.438785031 -1.559918304
18   -7.00138687  5.290122902  -4.95551272 -1.254892507 -2.847438842
19   -7.90106917  3.632098909  -7.97139007  0.327308944 -3.154246652
20   -2.88050721  7.812684734   0.91877550  0.800366037 -2.510303612
21   -7.87903905  2.341672381  -6.17997792  1.083637791 -4.698997893
22   -7.59797444  3.447927522  -6.32085082  2.110154503 -3.267704431
23   -5.63760709  7.704304261  -1.54281815  0.983022425 -2.757910194
24   -7.30352684  0.165506837  -7.04281905  3.229873194 -1.344711208
25   -7.54452125  3.917724495  -5.62500256  2.314954454 -2.531499224
26   -7.91301086  3.351778981  -6.30750679  1.267511721 -1.667931676
27   -8.67043820  3.048490511  -6.79359229  2.728746781 -2.263668283
```

28	-6.77924418	3.908153790	-6.28341647	-0.877015342	-3.704124445
29	-8.58134737	5.394455681	-3.14783142	2.372496625	-1.953650040
30	-8.45382093	5.610786617	-5.51442048	0.004237262	-3.705242879
31	8.39426870	6.275757086	-6.57220369	3.993451848	-1.507620428
32	-5.82029725	4.709312073	-6.31627213	-2.379783090	-3.392407411
33	-4.82767352	6.416828135	-0.40355284	1.152700985	-0.808863843
34	9.76169569	7.113480468	-6.00238195	1.623403585	-1.201571444
35	9.16869610	6.788223449	-5.90932264	1.075077005	0.866455095
36	-5.36185278	6.179497150	-1.18701539	1.801498966	-3.146254399
37	-5.34657663	3.330685481	-5.26265145	-1.095345290	-4.233995088
38	-0.42228789	7.525480477	1.84608495	1.432185488	-4.301792172
39	-8.45561707	5.930819616	-3.61479784	-0.786537674	-3.050203917
40	8.44203959	6.419059596	-6.99271503	3.308168450	-1.107791233
41	-5.59079438	5.404032152	-4.91806174	-2.917634844	-4.238210723
42	-5.42583288	3.519560832	-4.46477238	-2.019234715	-3.102148691
43	-7.07993673	3.606149828	-2.61278787	4.000139265	-1.860449240
44	-7.23319247	0.320528678	-6.39557833	2.411241178	-0.295575240
45	-4.86393820	6.379614358	-0.18583341	0.182731419	-0.889099308
46	-6.99573634	4.162026893	-3.25405978	0.128127271	-0.463393361
47	-7.10081926	4.641966567	-3.85737389	0.467137223	0.046537258
48	-3.42327910	7.884245543	-0.58667518	-0.355754603	-1.538442822
49	-6.75071608	1.893867983	-5.59430777	0.398633552	-3.941995359
50	-6.71480660	1.113405142	-5.50364133	2.869417145	-1.721849975
51	7.40932363	5.541663521	-6.75491828	2.941167498	-1.782396988
52	-7.08907548	2.241446212	-5.53498181	1.535049607	-2.688824047
53	-3.95888158	7.260487142	-0.07121571	-0.118767665	-2.750296492
54	-6.39393317	1.258137973	-5.98525225	3.300522392	-2.556830503
55	-7.55788958	3.873957914	-4.17180810	0.461759105	-2.278613479
56	-3.09926219	6.998800817	-0.14400207	1.692435089	-4.076786154
57	-7.06644109	3.116886664	-3.03097015	4.839082031	-0.113710733
58	-6.49247010	4.553569467	-1.14978149	1.652579471	-1.817648084
59	-7.38584878	1.333061312	-5.78986761	1.815875334	-1.531840303
60	-7.36834590	2.148144940	-5.83328266	0.195924981	-0.521142915
61	-5.43837352	5.028779078	-0.57140717	2.540756710	-1.147356517
62	-4.24795602	6.549345505	0.53317522	-0.391479897	-1.129717944
63	-4.05409459	6.607423804	-0.08012774	1.035762789	-0.956056536
64	-7.16828693	3.797154566	-3.06801540	1.583039465	0.238781641
65	-1.82830420	7.236377527	1.32253456	0.364601961	-2.798353204
66	-5.57803816	2.177947828	-6.47171151	-0.145548356	-0.671572470
67	-6.32398652	4.230098128	-1.35440803	1.573566622	0.491440020
68	-1.60539020	5.412632958	1.43045822	1.332976239	-3.563651244
69	-6.48928816	4.310538445	-1.40202586	-0.504340869	-1.990657430
70	-6.56821677	2.424875216	-5.76058063	0.861499367	-2.649628190
71	-6.77362275	1.664070824	-5.43206522	0.190668299	-1.642539826

72	8.93279576	6.309996005	-6.59348877	1.497513356	-1.439877365
73	-6.85834298	0.731748269	-6.23110109	1.313866702	-1.302430487
74	-6.81153622	2.270483919	-5.15842291	-0.089928587	-2.248923757
75	-0.04726220	6.549712514	2.22614118	3.043544795	-4.183220141
76	-2.92108957	5.929768405	0.63072491	0.812114823	-1.647124931
77	-5.84150648	2.748257549	-2.20467275	-0.460173419	1.766772364
78	9.36422055	6.387240211	-6.06763906	2.024555686	-1.565391927
79	-7.36672624	3.000299530	-5.16942897	0.790835426	-1.459012308
80	-6.19011098	0.466452065	-5.09233586	1.514515878	-0.035968423
81	-7.34732360	5.265296472	-1.29888648	0.220131678	-0.053875712
82	-7.15574970	3.023195697	-3.02804907	-0.616320349	-1.668756762
83	-3.99052631	4.255935959	-3.62507673	-3.397386796	-0.934592904
84	-5.53463848	6.246499116	-1.08499791	-1.429214746	-0.019005628
85	-6.95131425	4.202454903	-1.42066768	1.544577858	0.001108552
86	-2.72836665	5.658003971	0.73395093	-2.700587251	0.911110156
87	-5.88105555	4.573884841	-0.61152490	1.885967827	-0.689921322
88	-2.32062163	5.420733421	1.19096883	0.956549148	-0.775801164
89	-6.18524582	3.517738370	-5.72000824	-2.050588170	-2.144818172
90	-2.97529808	7.093269755	0.46117410	-0.543501199	-0.749190774
91	-1.77959492	5.046960023	1.61391201	1.532591268	-3.439003080
92	-6.51140100	3.026541156	-0.92331127	1.626555349	-0.469376874
93	-2.77204597	5.611907354	0.96531979	0.575193332	-1.901875577
94	-6.92990182	1.680681471	-2.96847139	0.945388287	-0.227069424
95	-6.15686914	2.102686648	-4.77790599	-0.081254522	-3.365220191
96	-6.85011631	3.868053962	-2.36885968	-0.198715579	-1.991359141
97	-5.73700193	4.884620620	-0.44214130	0.672540736	-1.500229961
98	-2.80353136	5.125497568	0.99459507	3.183574839	-2.597844051
99	-4.05332549	5.144834626	0.22738803	-0.538967461	-0.160410495
100	-6.09980761	2.921255621	-2.29949354	1.959528024	-0.182713220
101	-6.36751662	1.025442664	-4.20500577	0.041611588	-1.397407817
102	-4.68535361	4.298976340	-0.66802458	3.876097712	-1.952624560
103	-6.71885896	1.494595482	-5.04093326	-0.529714233	-1.999023037
104	-6.66959729	1.266904139	-5.10466105	0.203458152	-1.453271174
105	8.81556606	4.739424004	-5.56499348	3.266124487	-1.359135241
106	-1.93630671	5.536011981	1.05937964	1.676778717	-2.033197410
107	9.32771509	4.340808199	-6.06385378	3.064968898	-1.843241487
108	-7.08055098	2.192198156	-2.86522139	1.819625493	-0.828111264
109	-5.79653219	0.406370593	-4.54098125	-0.446549840	-0.057192235
110	9.12164760	5.841742681	-5.78857362	2.497849804	-1.379347276
111	-4.74347671	5.775574285	-0.37276810	-1.408482520	-0.168011913
112	11.00684497	4.775375029	-5.38836218	1.651714862	-0.503356746
113	-5.24115089	2.959681614	-1.21255631	1.497049855	1.115717443
114	-5.47759260	4.156239758	-0.70054330	1.852728335	-1.235481974
115	-6.53834522	3.432127240	-3.05060151	0.629927590	-0.008276635

116	-5.76731704	1.358387519	-4.16991081	0.522182697	-0.861387316
117	-6.76129921	1.842909600	-4.36612647	0.975748501	-0.205846255
118	-5.24904591	5.100394124	-1.56723521	-1.988645995	-1.405808322
119	-6.15410063	1.298819439	-4.96152051	1.472833386	-0.154340253
120	9.10328803	4.768764862	-5.47706445	2.438351177	-0.463823384
121	-6.42193910	1.936702133	-2.26149644	1.260092051	1.725056875
122	-6.58260582	4.443607787	-0.85224785	-0.715325618	-0.156875721
123	-6.03641026	1.023697601	-4.67972820	-0.592096056	0.903695115
124	-5.78208987	1.452589339	-4.26768236	-2.140920468	1.357303904
125	-1.63998216	5.911909268	1.46884581	0.720003249	-2.355244402
126	-5.62768713	2.653438879	-3.86897644	-1.574166061	-0.518642126
127	8.56351878	4.865859260	-5.79877264	1.931745160	0.024089458
128	-1.29465849	5.764763957	1.82990712	0.464778705	-0.384945155
129	-6.67556074	-0.185794255	-5.18578109	2.137490758	0.492367462
130	-1.50999645	5.684037856	1.82860477	0.478129484	-0.248987057
131	-6.32522021	0.287014591	-6.00945242	1.906436244	-2.413297506
132	-6.27613860	1.006272219	-4.65741039	0.600757103	-3.326401692
133	-2.85225215	4.708488497	1.06079045	1.984399349	-2.733032038
134	-6.07167847	1.335615237	-3.01899484	2.828440791	-0.155212095
135	-0.03793034	6.216104037	2.11457146	0.114810851	-3.367765384
136	-5.10077820	1.014372843	-4.38718478	-0.169990667	-3.025286990
137	-5.73485750	3.721751214	-0.96017605	-0.113218604	-1.496228868
138	-6.27959565	0.008133336	-4.37052775	0.186258137	-0.849687838
139	-6.29346554	0.861261141	-3.27411259	1.218491542	0.691958349
140	-6.16504839	1.217172472	-4.11341187	1.751917472	-0.594518151
141	-4.47714527	1.086448555	-2.90991221	-1.270402913	-1.581353857
142	-0.99437255	5.511010194	2.29831177	0.149101490	-1.542814274
143	-6.59088920	2.256216882	-2.61756734	2.129247860	-1.169581082
144	9.73058232	4.388857422	-5.58255546	2.233633670	-0.699778781
145	-5.81897258	2.140081400	-1.68462160	0.616825745	0.235911674
146	7.71309750	4.402576870	-5.90648484	2.568590033	-0.318633650
147	-2.80495582	4.739097732	0.86327854	2.339551892	-2.098835435
148	-5.64377951	-0.208690750	-5.77417886	0.783253080	-0.633961157
149	-6.37553679	2.251947904	-3.88203419	-0.229548801	-0.932950924
150	-6.24912176	0.513558028	-3.74427786	2.308393333	-0.406209436
151	-6.13539535	0.621274736	-4.12393190	1.547460379	-1.389869296
152	-3.88317674	4.628893036	0.51907721	-0.423941806	-0.835408225
153	-2.51892496	4.705446077	0.82266159	2.411832600	-2.107804546
154	-5.99053280	3.051528338	-0.47958315	2.169198154	-0.291396750
155	-1.97605636	5.729660027	1.66986797	-1.296669644	-0.121605088
156	-5.59707899	2.966736283	-1.64789712	2.516476366	-0.544529930
157	-5.64757796	1.106373664	-3.40154513	-0.903122801	-1.597959463
158	-6.09984370	1.615427650	-2.80803829	0.969825651	1.025735403
159	-5.67972345	0.890151485	-2.67027600	2.474336061	0.519633551

```
160   9.45208309  4.596206734  -4.89563927  2.012716183 -0.620322872
161  10.43020828  4.657187582  -4.46242576  1.933656754 -0.718740873
162  -4.76200377  3.277219977   0.33359628  2.906146133 -0.638333330
163  -6.57987539  1.680906174  -3.84684457  0.756362632 -0.296772863
164  -5.77653510  4.198135603  -0.53339035 -0.670063397 -0.449343601
165   9.70898930  5.551109516  -5.37584994  0.973772161  0.431060945
166   0.87467182  6.394248355   3.04818568  0.350790541 -1.720539764
167  -5.59939126  3.792900884  -0.56765588 -0.356514281  0.784835701
168  -1.99294672  4.156396176  -1.82545658 -5.640792197 -1.206479201
169  -5.22986096  2.004545550  -4.03143172 -3.978993993 -1.554755271
170  -3.93905737  4.744762057   0.60882576  0.275335637  0.028918710
171  -4.43252404  5.123008793  -0.21276670 -0.740701614  0.088658016
172  10.79888908  5.980078038  -4.14700471 -0.142950634 -0.851075197
173  -6.22054185  0.452663529  -5.40475718  1.490091821  1.063636460
174   8.86565093  5.358357902  -5.61854243  1.696076684 -0.444922496
175  -4.24040615  4.791664812   0.44060104  0.475485674 -0.245658807
176  -5.34027063  3.681545987  -0.30401206  1.306025903  0.633693336
177   9.87004843  4.911621705  -5.27995892  2.052158679  0.439377304
178  -5.53827721  3.273395965  -3.41528382 -0.188879335  1.551232489
179  -5.60493714  4.497651746  -1.26462895  0.095583461 -0.307431228
180  10.46575732  4.541294539  -4.79664380  1.141226322  0.739433746
181  -4.70282114  4.692179870   0.28679326 -0.374933829  1.117065844
182   9.69353238  5.256969943  -5.25919188  1.593784380  0.289814271
183  11.86242576  3.489347541  -4.46047050  0.379691508 -0.325723310
184  -2.61334789  4.533978888   1.24107869  0.070861468 -2.327164810
185  -5.95363989  1.939275011  -1.88863324  2.315259024 -1.070591705
186   9.52939902  5.077438528  -5.36210003  1.562821471 -2.670836815
187  -5.37082953 -0.545922658  -3.51203507  0.496152687 -1.377357836
188  -5.88786065  1.413966538  -0.77190325  3.654207275 -0.117664626
189  -5.88837899 -0.113028372  -5.56219794  0.451260625 -1.723486839
190   9.13979207  3.045554068  -5.01566417  2.541262560 -0.237730350
191  -1.52275474  4.594083578   1.58939603  1.889454944 -2.971348502
192  -5.44127241  0.081902014  -4.33043911 -1.405224994 -0.611373743
193  -4.40365818  1.879376591  -0.04746764  3.206442356 -1.137219003
194  -3.23491137  3.141521529   0.58292019  0.628533342  1.297698559
195  -6.05823696  1.107523191  -3.74220896  0.510692427 -0.728671690
196  -4.37286937  0.597723989  -4.57553171 -0.586264248 -1.844843506
197  -0.69625693  4.873669828   2.15437315  0.908099134 -2.695713272
198  -4.67532887  0.127451845  -4.19558092  1.438900737 -0.763592356
199  -5.97201332  2.528707982  -1.19157092  2.160513105 -1.260079501
200  -0.98597530  5.012320293   2.04446248  1.612200555 -2.798951863
 [ reached 'max' / getOption("max.print") -- omitted 3672 rows ]
> #Plot method
> summary(data_pca)
```

```
Importance of components:
                         PC1    PC2    PC3     PC4     PC5     PC6     PC7     PC8     PC9    PC10    PC11    PC12    PC13
PC14    PC15    PC16
Standard deviation     4.6273 2.4769 2.2962 1.51763 1.24600 1.15910 0.98973 0.90924 0.89752 0.75703
0.69782 0.59185 0.55554 0.52326 0.51767 0.48539
Proportion of Variance 0.4655 0.1334 0.1146 0.05007 0.03375 0.02921 0.02129 0.01797 0.01751
0.01246 0.01059 0.00761 0.00671 0.00595 0.00583 0.00512
Cumulative Proportion  0.4655 0.5988 0.7135 0.76354 0.79729 0.82649 0.84779 0.86576 0.88327
0.89573 0.90632 0.91393 0.92064 0.92659 0.93242 0.93754
                         PC17    PC18    PC19    PC20    PC21    PC22    PC23    PC24    PC25    PC26    PC27    PC28
PC29    PC30    PC31   PC32
Standard deviation     0.47184 0.45347 0.44193 0.43133 0.42248 0.41894 0.39579 0.38046 0.36147
0.34418 0.34199 0.32386 0.32319 0.31275 0.29004 0.2877
Proportion of Variance 0.00484 0.00447 0.00425 0.00404 0.00388 0.00382 0.00341 0.00315 0.00284
0.00258 0.00254 0.00228 0.00227 0.00213 0.00183 0.0018
Cumulative Proportion  0.94238 0.94685 0.95110 0.95514 0.95902 0.96284 0.96624 0.96939 0.97223
0.97481 0.97735 0.97963 0.98190 0.98403 0.98585 0.9877
                         PC33    PC34    PC35    PC36   PC37    PC38    PC39    PC40    PC41    PC42    PC43    PC44
PC45    PC46
Standard deviation     0.27297 0.26632 0.24813 0.24176 0.2251 0.22147 0.21331 0.18057 0.16434
0.15907 0.15542 0.14476 0.1364 0.09284
Proportion of Variance 0.00162 0.00154 0.00134 0.00127 0.0011 0.00107 0.00099 0.00071 0.00059
0.00055 0.00053 0.00046 0.0004 0.00019
Cumulative Proportion  0.98927 0.99082 0.99215 0.99342 0.9945 0.99559 0.99658 0.99729 0.99788
0.99843 0.99895 0.99941 0.9998 1.00000
> #Selecting 5 principal components
> data_final <- data_with_pca[,0:5]
> data_final
          PC1         PC2          PC3          PC4          PC5
1   -10.37151575  6.147904405 -11.26033150  2.694502132 -4.928103442
2    -9.32280116  7.023755019  -8.71100439 -0.823766672 -4.905720323
3    -9.50109278  4.311339068 -10.07240029  2.610463230 -4.461507817
4     6.83389697  8.455722811  -8.84177747  3.953419923 -3.060405336
5    -9.55848687  6.260124969  -7.12074247  2.023594038 -3.297437015
6    -9.02375298  4.004977409  -8.71030720  1.937465119 -3.905128241
7    -9.32657398  5.366431585  -6.18869269  3.920881421 -1.844468465
8    -7.86225118  7.820741740  -8.37111550  0.501872271 -3.762076060
9    -6.83254899  8.064057028  -2.12749246  1.704756898 -3.321272987
10   -7.98123670  6.164900996  -5.48511996  1.436842158 -1.359437987
11   -7.90941206  3.767611688  -6.48132487  2.449858047 -0.955387126
12   -6.69552177  5.679469373  -2.22525960  4.389718501 -2.395145571
13   -8.14738458  2.658595397  -7.52079879  0.461325534 -4.070270075
14   -7.23863956  5.741238435  -5.46658281 -1.655646889 -3.835411715
15    8.61543543  8.194210071  -7.64228019  2.708232248 -2.782813134
```

16	8.83262967	8.216708563	-7.62863384	2.019892341	-3.186838472
17	-5.74761186	7.674318988	-2.17072314	1.438785031	-1.559918304
18	-7.00138687	5.290122902	-4.95551272	-1.254892507	-2.847438842
19	-7.90106917	3.632098909	-7.97139007	0.327308944	-3.154246652
20	-2.88050721	7.812684734	0.91877550	0.800366037	-2.510303612
21	-7.87903905	2.341672381	-6.17997792	1.083637791	-4.698997893
22	-7.59797444	3.447927522	-6.32085082	2.110154503	-3.267704431
23	-5.63760709	7.704304261	-1.54281815	0.983022425	-2.757910194
24	-7.30352684	0.165506837	-7.04281905	3.229873194	-1.344711208
25	-7.54452125	3.917724495	-5.62500256	2.314954454	-2.531499224
26	-7.91301086	3.351778981	-6.30750679	1.267511721	-1.667931676
27	-8.67043820	3.048490511	-6.79359229	2.728746781	-2.263668283
28	-6.77924418	3.908153790	-6.28341647	-0.877015342	-3.704124445
29	-8.58134737	5.394455681	-3.14783142	2.372496625	-1.953650040
30	-8.45382093	5.610786617	-5.51442048	0.004237262	-3.705242879
31	8.39426870	6.275757086	-6.57220369	3.993451848	-1.507620428
32	-5.82029725	4.709312073	-6.31627213	-2.379783090	-3.392407411
33	-4.82767352	6.416828135	-0.40355284	1.152700985	-0.808863843
34	9.76169569	7.113480468	-6.00238195	1.623403585	-1.201571444
35	9.16869610	6.788223449	-5.90932264	1.075077005	0.866455095
36	-5.36185278	6.179497150	-1.18701539	1.801498966	-3.146254399
37	-5.34657663	3.330685481	-5.26265145	-1.095345290	-4.233995088
38	-0.42228789	7.525480477	1.84608495	1.432185488	-4.301792172
39	-8.45561707	5.930819616	-3.61479784	-0.786537674	-3.050203917
40	8.44203959	6.419059596	-6.99271503	3.308168450	-1.107791233
41	-5.59079438	5.404032152	-4.91806174	-2.917634844	-4.238210723
42	-5.42583288	3.519560832	-4.46477238	-2.019234715	-3.102148691
43	-7.07993673	3.606149828	-2.61278787	4.000139265	-1.860449240
44	-7.23319247	0.320528678	-6.39557833	2.411241178	-0.295575240
45	-4.86393820	6.379614358	-0.18583341	0.182731419	-0.889099308
46	-6.99573634	4.162026893	-3.25405978	0.128127271	-0.463393361
47	-7.10081926	4.641966567	-3.85737389	0.467137223	0.046537258
48	-3.42327910	7.884245543	-0.58667518	-0.355754603	-1.538442822
49	-6.75071608	1.893867983	-5.59430777	0.398633552	-3.941995359
50	-6.71480660	1.113405142	-5.50364133	2.869417145	-1.721849975
51	7.40932363	5.541663521	-6.75491828	2.941167498	-1.782396988
52	-7.08907548	2.241446212	-5.53498181	1.535049607	-2.688824047
53	-3.95888158	7.260487142	-0.07121571	-0.118767665	-2.750296492
54	-6.39393317	1.258137973	-5.98525225	3.300522392	-2.556830503
55	-7.55788958	3.873957914	-4.17180810	0.461759105	-2.278613479
56	-3.09926219	6.998800817	-0.14400207	1.692435089	-4.076786154
57	-7.06644109	3.116886664	-3.03097015	4.839082031	-0.113710733
58	-6.49247010	4.553569467	-1.14978149	1.652579471	-1.817648084
59	-7.38584878	1.333061312	-5.78986761	1.815875334	-1.531840303

60	-7.36834590	2.148144940	-5.83328266	0.195924981	-0.521142915
61	-5.43837352	5.028779078	-0.57140717	2.540756710	-1.147356517
62	-4.24795602	6.549345505	0.53317522	-0.391479897	-1.129717944
63	-4.05409459	6.607423804	-0.08012774	1.035762789	-0.956056536
64	-7.16828693	3.797154566	-3.06801540	1.583039465	0.238781641
65	-1.82830420	7.236377527	1.32253456	0.364601961	-2.798353204
66	-5.57803816	2.177947828	-6.47171151	-0.145548356	-0.671572470
67	-6.32398652	4.230098128	-1.35440803	1.573566622	0.491440020
68	-1.60539020	5.412632958	1.43045822	1.332976239	-3.563651244
69	-6.48928816	4.310538445	-1.40202586	-0.504340869	-1.990657430
70	-6.56821677	2.424875216	-5.76058063	0.861499367	-2.649628190
71	-6.77362275	1.664070824	-5.43206522	0.190668299	-1.642539826
72	8.93279576	6.309996005	-6.59348877	1.497513356	-1.439877365
73	-6.85834298	0.731748269	-6.23110109	1.313866702	-1.302430487
74	-6.81153622	2.270483919	-5.15842291	-0.089928587	-2.248923757
75	-0.04726220	6.549712514	2.22614118	3.043544795	-4.183220141
76	-2.92108957	5.929768405	0.63072491	0.812114823	-1.647124931
77	-5.84150648	2.748257549	-2.20467275	-0.460173419	1.766772364
78	9.36422055	6.387240211	-6.06763906	2.024555686	-1.565391927
79	-7.36672624	3.000299530	-5.16942897	0.790835426	-1.459012308
80	-6.19011098	0.466452065	-5.09233586	1.514515878	-0.035968423
81	-7.34732360	5.265296472	-1.29888648	0.220131678	-0.053875712
82	-7.15574970	3.023195697	-3.02804907	-0.616320349	-1.668756762
83	-3.99052631	4.255935959	-3.62507673	-3.397386796	-0.934592904
84	-5.53463848	6.246499116	-1.08499791	-1.429214746	-0.019005628
85	-6.95131425	4.202454903	-1.42066768	1.544577858	0.001108552
86	-2.72836665	5.658003971	0.73395093	-2.700587251	0.911110156
87	-5.88105555	4.573884841	-0.61152490	1.885967827	-0.689921322
88	-2.32062163	5.420733421	1.19096883	0.956549148	-0.775801164
89	-6.18524582	3.517738370	-5.72000824	-2.050588170	-2.144818172
90	-2.97529808	7.093269755	0.46117410	-0.543501199	-0.749190774
91	-1.77959492	5.046960023	1.61391201	1.532591268	-3.439003080
92	-6.51140100	3.026541156	-0.92331127	1.626555349	-0.469376874
93	-2.77204597	5.611907354	0.96531979	0.575193332	-1.901875577
94	-6.92990182	1.680681471	-2.96847139	0.945388287	-0.227069424
95	-6.15686914	2.102686648	-4.77790599	-0.081254522	-3.365220191
96	-6.85011631	3.868053962	-2.36885968	-0.198715579	-1.991359141
97	-5.73700193	4.884620620	-0.44214130	0.672540736	-1.500229961
98	-2.80353136	5.125497568	0.99459507	3.183574839	-2.597844051
99	-4.05332549	5.144834626	0.22738803	-0.538967461	-0.160410495
100	-6.09980761	2.921255621	-2.29949354	1.959528024	-0.182713220
101	-6.36751662	1.025442664	-4.20500577	0.041611588	-1.397407817
102	-4.68535361	4.298976340	-0.66802458	3.876097712	-1.952624560
103	-6.71885896	1.494595482	-5.04093326	-0.529714233	-1.999023037

104	-6.66959729	1.266904139	-5.10466105	0.203458152	-1.453271174
105	8.81556606	4.739424004	-5.56499348	3.266124487	-1.359135241
106	-1.93630671	5.536011981	1.05937964	1.676778717	-2.033197410
107	9.32771509	4.340808199	-6.06385378	3.064968898	-1.843241487
108	-7.08055098	2.192198156	-2.86522139	1.819625493	-0.828111264
109	-5.79653219	0.406370593	-4.54098125	-0.446549840	-0.057192235
110	9.12164760	5.841742681	-5.78857362	2.497849804	-1.379347276
111	-4.74347671	5.775574285	-0.37276810	-1.408482520	-0.168011913
112	11.00684497	4.775375029	-5.38836218	1.651714862	-0.503356746
113	-5.24115089	2.959681614	-1.21255631	1.497049855	1.115717443
114	-5.47759260	4.156239758	-0.70054330	1.852728335	-1.235481974
115	-6.53834522	3.432127240	-3.05060151	0.629927590	-0.008276635
116	-5.76731704	1.358387519	-4.16991081	0.522182697	-0.861387316
117	-6.76129921	1.842909600	-4.36612647	0.975748501	-0.205846255
118	-5.24904591	5.100394124	-1.56723521	-1.988645995	-1.405808322
119	-6.15410063	1.298819439	-4.96152051	1.472833386	-0.154340253
120	9.10328803	4.768764862	-5.47706445	2.438351177	-0.463823384
121	-6.42193910	1.936702133	-2.26149644	1.260092051	1.725056875
122	-6.58260582	4.443607787	-0.85224785	-0.715325618	-0.156875721
123	-6.03641026	1.023697601	-4.67972820	-0.592096056	0.903695115
124	-5.78208987	1.452589339	-4.26768236	-2.140920468	1.357303904
125	-1.63998216	5.911909268	1.46884581	0.720003249	-2.355244402
126	-5.62768713	2.653438879	-3.86897644	-1.574166061	-0.518642126
127	8.56351878	4.865859260	-5.79877264	1.931745160	0.024089458
128	-1.29465849	5.764763957	1.82990712	0.464778705	-0.384945155
129	-6.67556074	-0.185794255	-5.18578109	2.137490758	0.492367462
130	-1.50999645	5.684037856	1.82860477	0.478129484	-0.248987057
131	-6.32522021	0.287014591	-6.00945242	1.906436244	-2.413297506
132	-6.27613860	1.006272219	-4.65741039	0.600757103	-3.326401692
133	-2.85225215	4.708488497	1.06079045	1.984399349	-2.733032038
134	-6.07167847	1.335615237	-3.01899484	2.828440791	-0.155212095
135	-0.03793034	6.216104037	2.11457146	0.114810851	-3.367765384
136	-5.10077820	1.014372843	-4.38718478	-0.169990667	-3.025286990
137	-5.73485750	3.721751214	-0.96017605	-0.113218604	-1.496228868
138	-6.27959565	0.008133336	-4.37052775	0.186258137	-0.849687838
139	-6.29346554	0.861261141	-3.27411259	1.218491542	0.691958349
140	-6.16504839	1.217172472	-4.11341187	1.751917472	-0.594518151
141	-4.47714527	1.086448555	-2.90991221	-1.270402913	-1.581353857
142	-0.99437255	5.511010194	2.29831177	0.149101490	-1.542814274
143	-6.59088920	2.256216882	-2.61756734	2.129247860	-1.169581082
144	9.73058232	4.388857422	-5.58255546	2.233633670	-0.699778781
145	-5.81897258	2.140081400	-1.68462160	0.616825745	0.235911674
146	7.71309750	4.402576870	-5.90648484	2.568590033	-0.318633650
147	-2.80495582	4.739097732	0.86327854	2.339551892	-2.098835435

148	-5.64377951	-0.208690750	-5.77417886	0.783253080	-0.633961157
149	-6.37553679	2.251947904	-3.88203419	-0.229548801	-0.932950924
150	-6.24912176	0.513558028	-3.74427786	2.308393333	-0.406209436
151	-6.13539535	0.621274736	-4.12393190	1.547460379	-1.389869296
152	-3.88317674	4.628893036	0.51907721	-0.423941806	-0.835408225
153	-2.51892496	4.705446077	0.82266159	2.411832600	-2.107804546
154	-5.99053280	3.051528338	-0.47958315	2.169198154	-0.291396750
155	-1.97605636	5.729660027	1.66986797	-1.296669644	-0.121605088
156	-5.59707899	2.966736283	-1.64789712	2.516476366	-0.544529930
157	-5.64757796	1.106373664	-3.40154513	-0.903122801	-1.597959463
158	-6.09984370	1.615427650	-2.80803829	0.969825651	1.025735403
159	-5.67972345	0.890151485	-2.67027600	2.474336061	0.519633551
160	9.45208309	4.596206734	-4.89563927	2.012716183	-0.620322872
161	10.43020828	4.657187582	-4.46242576	1.933656754	-0.718740873
162	-4.76200377	3.277219977	0.33359628	2.906146133	-0.638333330
163	-6.57987539	1.680906174	-3.84684457	0.756362632	-0.296772863
164	-5.77653510	4.198135603	-0.53339035	-0.670063397	-0.449343601
165	9.70898930	5.551109516	-5.37584994	0.973772161	0.431060945
166	0.87467182	6.394248355	3.04818568	0.350790541	-1.720539764
167	-5.59939126	3.792900884	-0.56765588	-0.356514281	0.784835701
168	-1.99294672	4.156396176	-1.82545658	-5.640792197	-1.206479201
169	-5.22986096	2.004545550	-4.03143172	-3.978993993	-1.554755271
170	-3.93905737	4.744762057	0.60882576	0.275335637	0.028918710
171	-4.43252404	5.123008793	-0.21276670	-0.740701614	0.088658016
172	10.79888908	5.980078038	-4.14700471	-0.142950634	-0.851075197
173	-6.22054185	0.452663529	-5.40475718	1.490091821	1.063636460
174	8.86565093	5.358357902	-5.61854243	1.696076684	-0.444922496
175	-4.24040615	4.791664812	0.44060104	0.475485674	-0.245658807
176	-5.34027063	3.681545987	-0.30401206	1.306025903	0.633693336
177	9.87004843	4.911621705	-5.27995892	2.052158679	0.439377304
178	-5.53827721	3.273395965	-3.41528382	-0.188879335	1.551232489
179	-5.60493714	4.497651746	-1.26462895	0.095583461	-0.307431228
180	10.46575732	4.541294539	-4.79664380	1.141226322	0.739433746
181	-4.70282114	4.692179870	0.28679326	-0.374933829	1.117065844
182	9.69353238	5.256969943	-5.25919188	1.593784380	0.289814271
183	11.86242576	3.489347541	-4.46047050	0.379691508	-0.325723310
184	-2.61334789	4.533978888	1.24107869	0.070861468	-2.327164810
185	-5.95363989	1.939275011	-1.88863324	2.315259024	-1.070591705
186	9.52939902	5.077438528	-5.36210003	1.562821471	-2.670836815
187	-5.37082953	-0.545922658	-3.51203507	0.496152687	-1.377357836
188	-5.88786065	1.413966538	-0.77190325	3.654207275	-0.117664626
189	-5.88837899	-0.113028372	-5.56219794	0.451260625	-1.723486839
190	9.13979207	3.045554068	-5.01566417	2.541262560	-0.237730350
191	-1.52275474	4.594083578	1.58939603	1.889454944	-2.971348502

```
192  -5.44127241  0.081902014  -4.33043911 -1.405224994 -0.611373743
193  -4.40365818  1.879376591  -0.04746764  3.206442356 -1.137219003
194  -3.23491137  3.141521529   0.58292019  0.628533342  1.297698559
195  -6.05823696  1.107523191  -3.74220896  0.510692427 -0.728671690
196  -4.37286937  0.597723989  -4.57553171 -0.586264248 -1.844843506
197  -0.69625693  4.873669828   2.15437315  0.908099134 -2.695713272
198  -4.67532887  0.127451845  -4.19558092  1.438900737 -0.763592356
199  -5.97201332  2.528707982  -1.19157092  2.160513105 -1.260079501
200  -0.98597530  5.012320293   2.04446248  1.612200555 -2.798951863
 [ reached 'max' / getOption("max.print") -- omitted 3672 rows ]
>
> #KMeans
> set.seed(109)
> # Initialize total within sum of squares error: wss
> wss <- 0
> # For 1 to 30 cluster centers
> for (j in 1:15) {
+   km.out <- kmeans(data_final, centers = j, nstart = 20)
+   # Save total within sum of squares to wss variable
+   wss[j] <- km.out$tot.withinss
+ }
> # create a DF to use in a ggplot visualisation
> wss_df <- data.frame(num_cluster = 1:15, wgss = wss)
> # plot to determine optimal k
> ggplot(data = wss_df, aes(x=num_cluster, y= wgss)) +
+   geom_line(color = "lightgrey", size = 2) +
+   geom_point(color = "green", size = 4) +
+   theme_dark() +
+   geom_curve(x=15, xend=8, y=300000, yend= 290500, arrow = arrow(length = unit(0.2,"cm")), size =1,
colour = "purple")
> # Set k equal to the number of clusters corresponding to the elbow location
> k <- 4
> # Create a k-means model on wisc.data: wisc.km
> k_means <- kmeans(data_final, centers = k, nstart = 20)
> # add the cluster group back to the original DF for all players other than GK and Unknown
> cluster_data <- data_standardized %>%
+   mutate(Cluster = k_means$cluster)
> cluster_data
      ï..       Age  Overall Potential International.Reputation   Weak.Foot Skill.Moves Jersey.Number
Height       Weight   Crossing  Finishing
1  -1.087785  1.3717005 3.191279  3.050070             5.712935  1.39966159   1.6103701   -0.63100398 -
1.8365790 -0.553752087  1.4919060  2.1948857
2  -1.087585  1.8120458 3.191279  3.050070             5.712935  1.39966159   2.7646376   -0.79521543
0.9083646  0.988584356  1.4919060  2.1471341
```

3 -1.087386 0.2708374 2.909515 2.875068 5.712935 2.81949121 2.7646376 -0.63100398 -
1.0740947 -1.114601703 1.2407621 1.8128726
4 -1.087186 0.4910100 2.768633 2.875068 4.152847 -0.02016803 -1.8524324 -1.12363833
1.6708490 0.007097529 -1.8734215 -1.7207485
5 -1.086986 0.4910100 2.768633 2.700065 4.152847 2.81949121 1.6103701 -0.79521543 -
0.3116103 -0.834176895 1.9439649 1.5741144
6 -1.086787 0.4910100 2.768633 2.525063 4.152847 1.39966159 1.6103701 -0.63100398 -
1.3790884 -0.273327279 1.3412197 1.6696177
7 -1.086587 1.5918731 2.768633 2.525063 4.152847 1.39966159 1.6103701 -0.63100398 -
1.3790884 -1.395026511 1.5923635 1.0965981
8 -1.086388 1.3717005 2.768633 2.525063 5.712935 1.39966159 0.4561026 -0.68574113
0.1458803 1.409221568 1.1403046 2.0993824
9 -1.086188 1.5918731 2.768633 2.525063 4.152847 -0.02016803 0.4561026 -0.35731823
0.1458803 0.848371952 0.5877882 0.5235784
10 -1.085590 0.7111826 2.627750 2.350060 4.152847 2.81949121 0.4561026 -0.74047828
0.1458803 0.007097529 1.6928211 1.2876046
11 -1.085190 1.5918731 2.627750 2.350060 4.152847 -1.43999766 1.6103701 -0.02889533
-1.3790884 -1.254814107 1.4919060 1.2876046
12 -1.084991 0.4910100 2.486868 2.350060 2.592759 -0.02016803 -0.6981649 -0.46679253
-2.1415728 -0.553752087 0.6882457 0.7623366
13 -1.084791 -0.1695079 2.486868 3.050070 2.592759 -0.02016803 1.6103701 -0.02889533
-0.6166041 -0.133114875 1.3914484 1.6696177
14 -1.084592 -0.1695079 2.486868 2.525063 2.592759 1.39966159 0.4561026 -0.68574113
0.9083646 1.829858780 1.0398471 2.1471341
15 -1.084193 0.2708374 2.486868 2.700065 2.592759 1.39966159 -1.8524324 0.02584182
0.9083646 1.269009164 -1.9738790 -1.6729968
16 -1.083993 0.2708374 2.486868 2.350060 4.152847 -1.43999766 -1.8524324 -1.12363833
2.4333333 2.811345608 -2.0241078 -1.6729968
17 -1.083794 0.9313553 2.486868 2.175057 4.152847 -0.02016803 0.4561026 -0.90468973
0.9083646 0.007097529 0.3868731 0.8578399
18 -1.083594 1.3717005 2.486868 2.175057 4.152847 1.39966159 0.4561026 -0.02889533
0.4508740 0.147309933 0.7887032 1.9083759
19 -1.083195 1.1515279 2.486868 2.175057 4.152847 1.39966159 1.6103701 -0.63100398
-1.3790884 -0.834176895 0.7887032 2.0993824
20 -1.082995 1.8120458 2.486868 2.175057 4.152847 -0.02016803 -0.6981649 -1.01416403
0.9083646 1.269009164 0.1859580 -0.7657157
21 -1.082796 -1.2703711 2.345986 3.225073 2.592759 1.39966159 2.7646376 -0.63100398
-0.6166041 -0.413539683 1.1403046 1.8606242
Heading.Accuracy Short.Passing Volleys Dribbling Curve FK.Accuracy Long.Passing Ball.Control
Acceleration Sprint.Speed Agility Reactions Balance
1 0.75369099 1.6848656 1.9863077 1.7997488 2.0295349 2.5052292 1.83445120 1.8047771
1.7115729 1.3810617 1.79984320 2.985502 2.1378487
2 1.76105880 1.1055559 2.0378096 1.3536297 1.4302396 1.5539810 1.21248899 1.6924545
1.5763900 1.7225829 1.52352339 3.093794 0.3973190

3 0.32953612 1.2986591 1.8833039 1.7501800 1.7798285 2.1352993 1.27468521 1.7486158
1.9143474 1.6542787 2.14524296 2.877210 1.3720156
4 -1.84425756 -0.8898441 -1.7733304 -2.1161850 -1.5662367 -1.4583048 -0.40461276 -
1.2279342 -0.5865377 -0.5314571 -0.34163533 2.444041 -1.4824530
5 -0.04159938 1.8136010 1.7803002 1.2544922 1.6300047 1.9239108 2.08323609
1.5239705 0.8328836 0.6980193 0.97088377 2.552333 0.8846673
6 0.27651677 1.6204978 1.6772964 1.7006112 1.5301222 1.7125224 1.58566632 1.6924545
1.9143474 1.5176702 2.07616301 2.444041 2.0682275
7 -0.04159938 1.8779688 1.4712888 1.4527673 1.6300047 1.6596753 1.89664743
1.6362932 0.9680665 0.4248023 1.93800310 2.444041 2.0682275
8 1.12482650 1.1699236 2.0893115 1.3040610 1.6799460 1.9767580 0.40393811 1.4678092
1.3736155 0.6297150 1.17812362 2.660625 1.3023945
9 1.86709751 0.9124526 0.9562699 0.1144102 1.0806507 1.3425925 1.21248899 1.1308413
0.6977006 0.6297150 0.90180381 1.902580 0.1188343
10 -0.09461874 1.8136010 1.7803002 1.0066483 1.6799460 1.9767580 2.20762853
1.4678092 -0.1133973 -0.2582401 0.34916419 2.335749 0.4669402
11 -0.09461874 1.8779688 1.7803002 1.4031985 1.4801809 1.6068281 1.83445120
1.6924545 0.2921516 -0.1216317 1.86892315 2.444041 1.7897428
12 -0.09461874 1.4273946 0.4412510 0.9075107 -0.1678811 0.1271088 1.46127388
0.9061960 1.1032495 0.8346277 1.17812362 2.768917 1.9289851
13 0.64765227 1.4917623 2.0893115 1.5519049 1.7798285 2.1881464 1.08809655
1.5801318 1.4412070 1.1761490 1.79984320 2.010872 1.4416368
14 1.54898136 1.0411881 1.8833039 0.9570795 1.2804158 1.1312041 1.52347010
1.1308413 0.1569687 0.4248023 0.41824414 2.552333 0.4669402
15 -2.37445114 -1.7909925 -1.7218285 -2.1657538 -1.7160605 -1.8282346 -0.96437876 -
2.5758059 -1.8707760 -1.0778910 -1.93047424 1.902580 -1.4824530
16 -2.26841243 -1.9840958 -1.8248323 -2.3640289 -1.6661193 -1.4054577 -1.39975230 -
2.2949993 -1.3300441 -0.9412826 -0.27255538 1.794288 -1.3432106
17 0.64765227 1.6204978 -0.1767717 0.9570795 0.6811205 1.1312041 1.52347010
1.3554865 -1.0596782 -0.9412826 0.07284438 2.119164 -0.8558623
18 1.76105880 0.9124526 2.1923153 0.9570795 1.2304745 1.5539810 -0.34241654
1.0185186 0.6301091 0.6980193 0.83272386 2.552333 -0.3685140
19 1.12482650 1.1055559 1.9348058 1.4031985 1.4801809 1.3954397 0.40393811
1.4116479 1.5087985 0.9712362 1.45444343 2.444041 1.8593639
20 1.44294265 -0.3105345 -0.1252698 -0.1334337 0.3814729 -0.8241394 0.09295701 -
0.3855144 -0.1809888 0.6297150 -0.75611505 1.577704 -0.6469987
21 1.12482650 1.1699236 1.5742926 1.4527673 1.2304745 0.8669685 0.96370411
1.5239705 2.0495304 2.0641041 1.86892315 2.119164 1.3023945
Shot.Power Jumping Stamina Strength Long.Shots Aggression Interceptions Positioning Vision
Penalties Composure Marking Standing.Tackle
1 1.34352854 0.15177264 0.48906217 -0.60256102 2.0376206 -0.64395656 -1.26756793
1.8930605 2.4020311 1.41774311 2.7751992 -0.81382432 -0.99693438
2 1.89288042 2.42074430 1.48199590 1.00233410 1.9893135 0.18980770 -0.95952933
1.9401677 1.6035537 2.02215398 2.6874373 -1.04145761 -0.87043634

3   1.06885260 -0.43647927 1.04758739 -1.40500859 1.4579349 -0.19928229   -0.65149073
1.6575244 1.9362526 1.78038963 2.5996753 -1.08698427   -1.16559844
4 -1.62297158 0.06773665 -1.31063020 -0.20133724 -1.9235649 -1.19979940   -0.91552382 -
1.9697314 0.6719968 -0.69769495 0.3178650 -1.63330419   -1.29209648
5   1.67313967 -0.26840730 1.60611261 0.68135508 1.8926992 0.91240339   0.44864713
1.5633100 2.4020311 1.65950746 2.0731037 0.77960877   0.26804604
6   1.17872298 -0.85665921 1.17170411 -0.04084773 1.3613207 -0.31045086   -0.43146316
1.5633100 2.0693322 2.08259507 2.3363895 -0.76829766   -1.03910039
7   1.01391742 0.15177264 1.54405425 -0.68280578 1.4579349 0.13422341   1.41676844
1.1864522 2.2689515 1.84083072 1.7220560 0.41539550   1.02703429
8   1.39846373 0.23580863 1.60611261 1.32331312 1.6028563 1.52383051   -0.43146316
1.7988461 1.7366333 2.02215398 1.8098179 0.50644882   -0.28011214
9   1.01391742 2.25267233 1.23376247 1.32331312 0.3468707 1.57941480   1.72480704   0.2914151
0.3392979 1.41774311 1.5465321 1.64461531   1.70169052
10 1.45339892 -3.04159488 0.67523725 0.52086556 1.9410063 0.02305485   1.37276293
1.1864522 1.8697129 1.29686093 1.8098179 0.96171541   1.15353234
11 0.62937110 -0.18437131 0.86141232 -1.16427432 1.1197850 -0.14369801   -0.03541353
1.6575244 2.2689515 1.41774311 2.5119134 0.36986884   0.05721597
12 0.57443592 0.90809653 1.97846276 0.76159983 0.8299421 1.69058337   1.81281807
0.8095945 1.4039344 0.14848027 1.8098179 1.78119529   1.65952450
13 1.17872298 0.74002455 0.98552903 -0.12109249 1.7477778 -0.64395656   -0.82751279
1.4219883 1.9362526 2.08259507 1.7220560 -1.26909091   -1.33426249
14 1.50833410 0.99213251 1.54405425 1.40355788 1.6028563 0.91240339   -0.69549624
1.8459533 1.4704742 2.32435941 2.1608657 0.23328886   -0.65960627
15 -2.11738827 1.07616850 -1.80709707 1.00233410 -2.0201791 -0.92187798   -1.26756793 -
2.0168386 0.7385366 -1.60431126 0.4056270 -1.17803759   -1.62942459
16 -1.34829564 0.15177264 -1.62092199 0.28013129 -1.6820292 -2.03356367   -1.57560653 -
1.9226242 -0.9249579 -1.48342908 0.1423412 -1.40567089   -1.41859452
17 0.02508404 -0.01629934 1.35787918 0.84184459 0.1053350 1.41266195   1.59279050
1.0922378 1.9362526 0.51112680 2.2486276 1.78119529   1.44869443
18 1.45339892 1.83249239 1.73022933 0.92208934 1.3130135 1.35707766   -0.12342456
1.8459533 1.2708548 2.02215398 1.5465321 0.05118222   -0.28011214
19 1.50833410 1.24424048 0.73729560 0.52086556 1.5062421 0.30097627   -1.17955690
1.7988461 1.6700935 1.90127180 2.2486276 -0.95040430   -1.33426249
20 0.95898223 1.91652838 0.05465367 1.80478166 -0.1362007 1.80175193   1.63679602 -
1.2160159 -0.5257193 -0.09328407 1.7220560 1.91777527   1.74385653
21 1.01391742 0.74002455 1.17170411 0.36037605 1.2647064 0.13422341   -0.56347970
1.6104172 1.6035537 1.11553767 1.8975799 -0.76829766   -0.74393830
  Sliding.Tackle GK.Diving GK.Handling GK.Kicking GK.Positioning GK.Reflexes   Values   Wages Cluster
1   -0.98817606 -0.5749064 -0.3216783 -0.08855293   -0.1573438 -0.4700254 10.587303 13.398194
3
2   -1.11591110 -0.5236341 -0.3216783 -0.08855293   -0.1573438 -0.3181741 7.179220 9.419694
3

```
3    -0.69012766 -0.4210895  -0.4295405 -0.08855293     -0.1039817  -0.3181741 11.401173  6.560146
3
4    -1.54169454  3.7319654   3.6692221  3.96449911      3.7914527   3.8830465  6.670551  5.814177
2
5     0.07628253 -0.1134558  -0.2138161 -0.65147682     -0.3707923  -0.2169398  9.722565  8.176412
3
6    -1.15848944 -0.3185450  -0.2677472 -0.59518444     -0.4775165  -0.4700254  8.806961  7.803428
3
7     1.01300610 -0.2160004  -0.4295405 -0.53889205     -0.1573438  -0.4194083  6.161882  9.792678
3
8    -0.47723594  0.5018116   0.4333569  0.81212530      0.8565364   0.9978709  7.484422 10.662975
3
9     1.77941629 -0.3185450  -0.4834716 -0.42630727     -0.5308786  -0.3181741  4.534142  8.798053
3
10    0.84269272 -0.3698172  -0.3216783 -0.20113771     -0.5308786  -0.3687912  7.128354  8.176412
3
11   -0.86044103 -0.5749064  -0.1059540 -0.53889205     -0.5842407  -0.2675569  5.449746  6.435818
3
12    1.52394623 -0.1134558  -0.2677472 -0.37001488     -0.5308786  -0.3687912  5.754947  4.943880
3
13   -1.24364613 -0.6261787  -0.6991959 -0.70776921     -0.6376029  -0.4700254  8.400026  4.446568
3
14   -0.47723594 -0.4723618  -0.3756094 -0.31372249     -0.1573438  -0.3181741  7.840490  4.446568
3
15   -1.66942957  3.5781486   3.6692221  4.02079150      3.6313664   3.6805780  5.246278  5.316865
2
16   -1.41395950  3.4756040   3.9928086  3.12011327      3.6847285   3.5793438  4.788476  5.316865
2
17    1.31105451 -0.6261787  -0.4834716 -0.20113771     -0.4241544  -0.2169398  4.585009  7.181787
3
18   -0.43465759 -0.2672727  -0.6452648 -0.20113771     -0.2107059  -0.3687912  5.449746  4.322240
3
19   -1.58427288 -0.2160004  -0.1059540 -0.59518444     -0.3174302  -0.1663227  5.907548  6.808803
3
20    1.73683795 -0.7287232  -0.7531270 -0.82035399     -0.6909650  -0.7231110  2.092531  4.695224
3
21   -0.73270600 -0.2160004  -0.6452648 -0.53889205     -0.3174302  -0.5712596  7.586156  1.835677
3
 [ reached 'max' / getOption("max.print") -- omitted 3851 rows ]
> #Conclusion : With the help of kmeans we clustered playes into 4 different categories
```