NAME: SAKSHI AGRAWAL

DEPARTMENT: IT

CLASS: B.E.

SEMESTER: 8TH

ROLL NUMBER: 01

BATCH: B1

DIVISION: B

STUDENT-ID: 2017DSIT067

Experiment Number: 01

Title: Installation of R and R-Studio.

Date of Performance:

Date of Submission:

Grade:

Signature:

Aim: Installation of R and R studio.

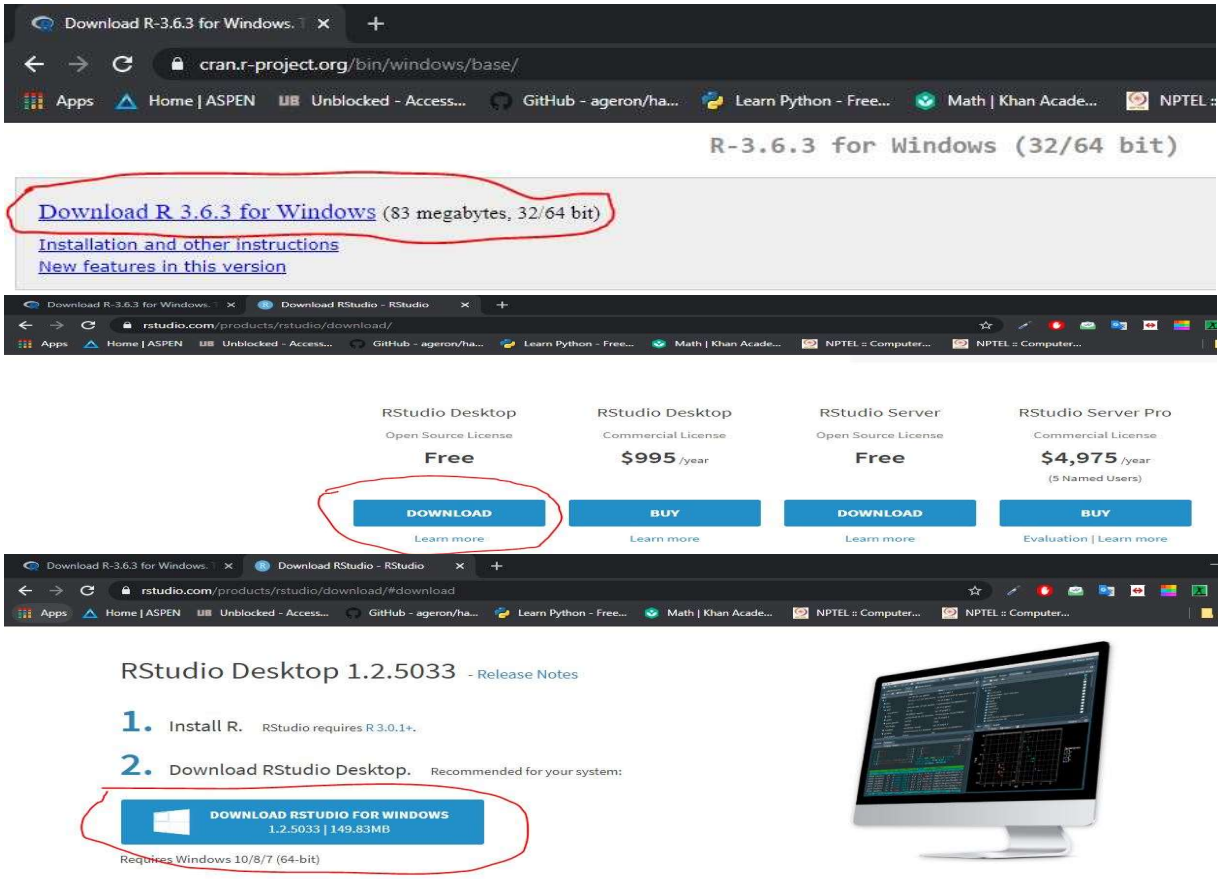
Theory:

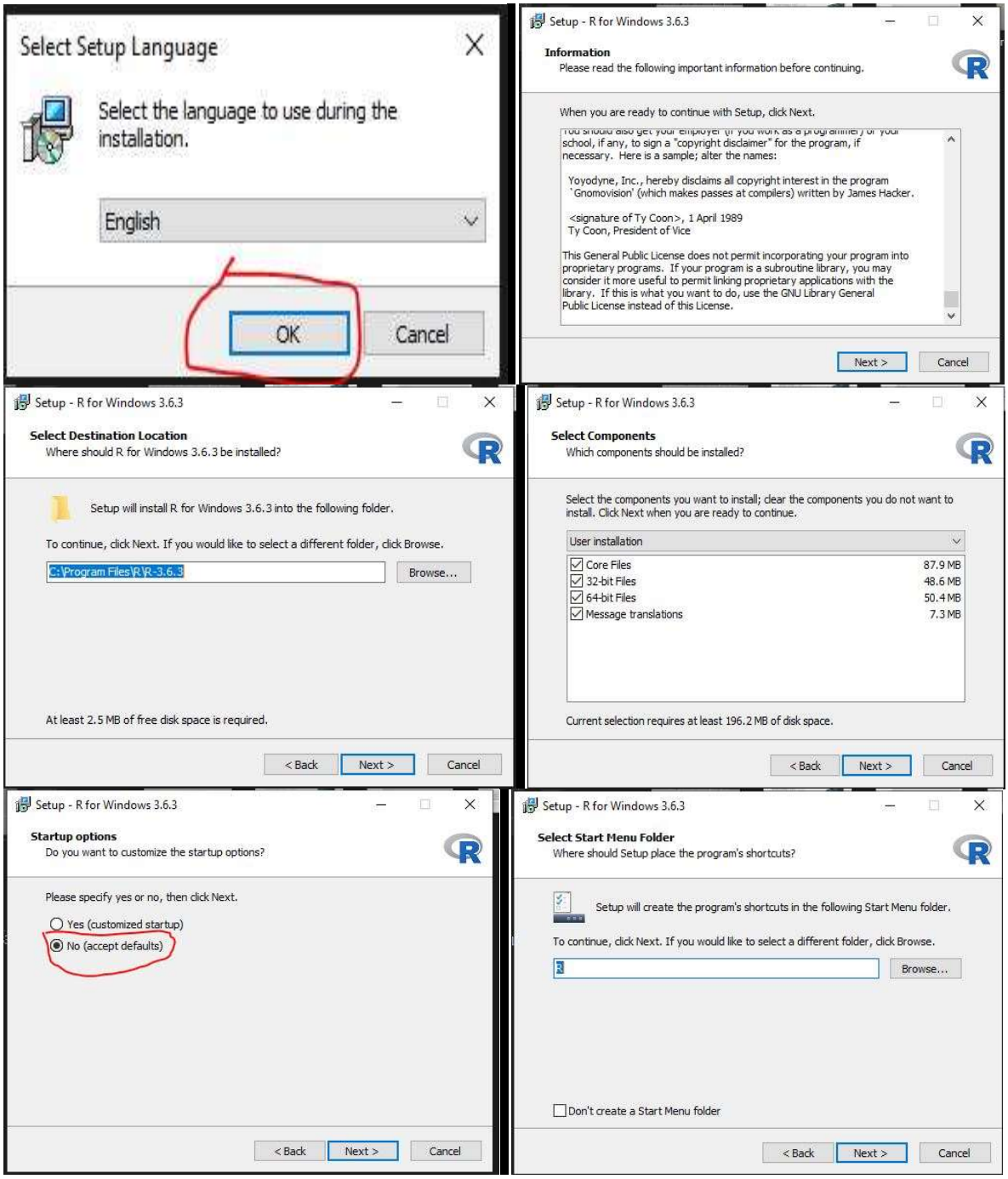
- To install R in system visit following link: https://cran.r-project.org/bin/windows/base/

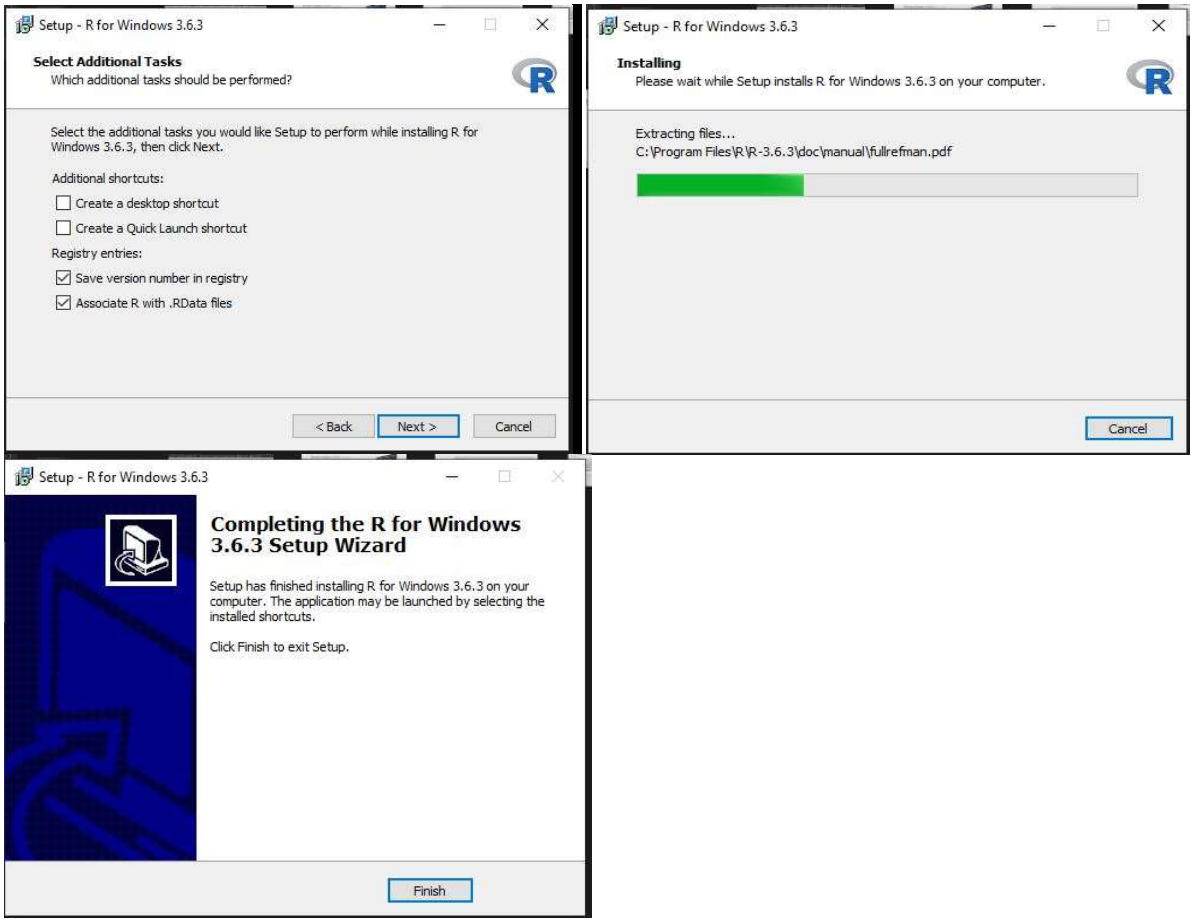
- To install R-Studio visit following link: https://rstudio.com/products/rstudio/download/

Outputs:

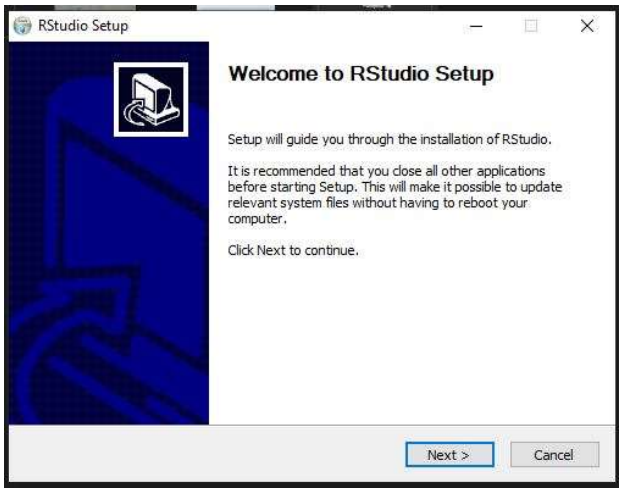
R installation:



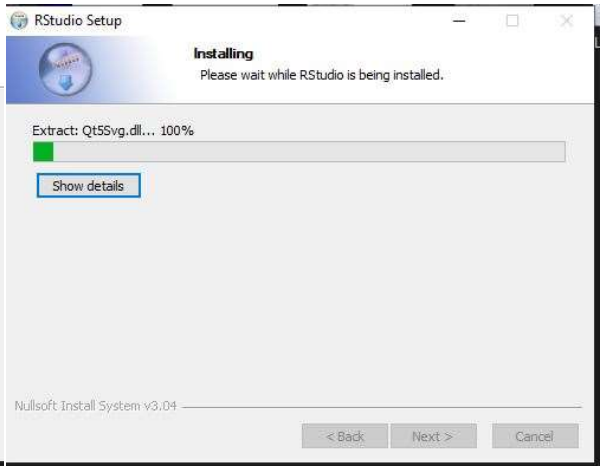

Select Setup Language
Select the language to use during the installation.
English
OK
Cancel
Setup - R for Windows 3.6.3
Information
Please read the following important information before continuing.
When you are ready to continue with Setup, click Next.
Yoyodyne, Inc., hereby disclaims all copyright interest in the program 'Gnomovision' (which makes passes at compilers) written by James Hacker.
<signature of Ty Coon>, 1 April 1989
Ty Coon, President of Vice
This General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Library General Public License instead of this License.
Next >
Cancel
Setup - R for Windows 3.6.3
Select Destination Location
Where should R for Windows 3.6.3 be installed?
Setup will install R for Windows 3.6.3 into the following folder.
To continue, click Next. If you would like to select a different folder, click Browse.
C:\Program Files\R\R-3.6.3
Browse...
At least 2.5 MB of free disk space is required.
< Back
Next >
Cancel
Setup - R for Windows 3.6.3
Select Components
Which components should be installed?
Select the components you want to install; clear the components you do not want to install. Click Next when you are ready to continue.
User installation
Core Files 87.9 MB
32-bit Files 48.6 MB
64-bit Files 50.4 MB
Message translations 7.3 MB
Current selection requires at least 196.2 MB of disk space.
< Back
Next >
Cancel
Setup - R for Windows 3.6.3
Startup options
Do you want to customize the startup options?
Please specify yes or no, then click Next.
Yes (customized startup)
No (accept defaults)
< Back
Next >
Cancel
Setup - R for Windows 3.6.3
Select Start Menu Folder
Where should Setup place the program's shortcuts?
Setup will create the program's shortcuts in the following Start Menu folder.
To continue, click Next. If you would like to select a different folder, click Browse.
Browse...
Don't create a Start Menu folder
< Back
Next >
Cancel

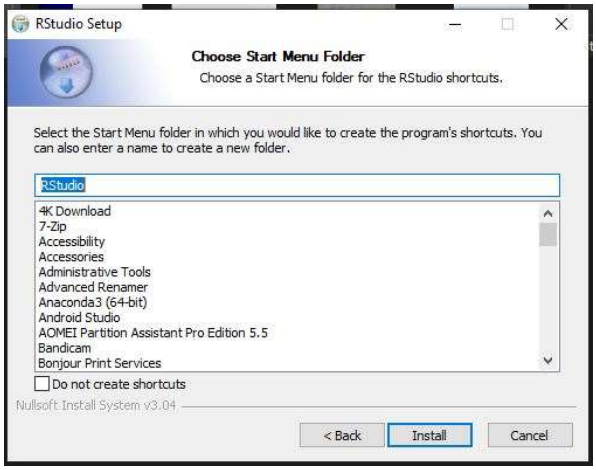


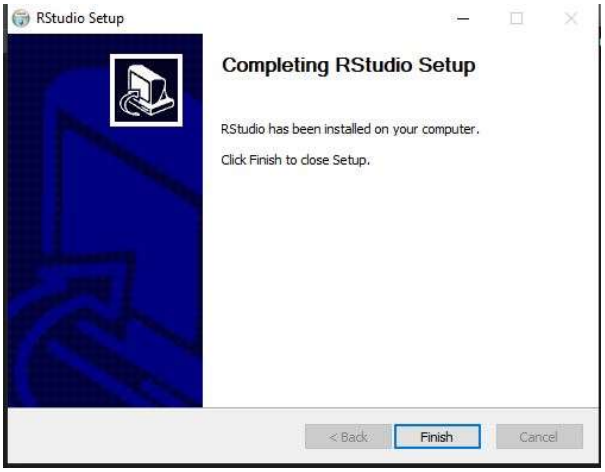
R-Studio Installation:

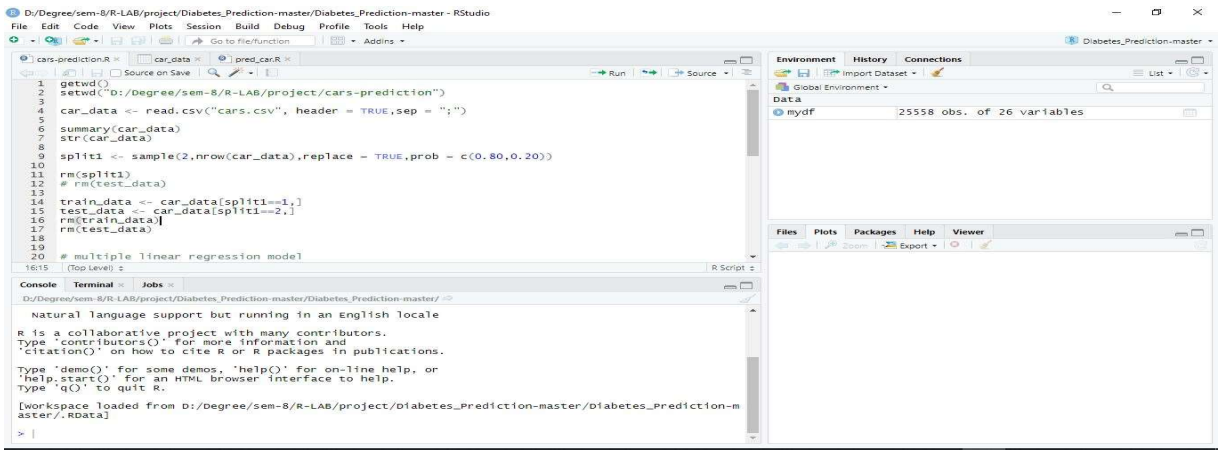



RStudio Setup
Choose Install Location
Choose the folder in which to install RStudio.
Setup will install RStudio in the following folder. To install in a different folder, click Browse and select another folder. Click Next to continue.
Destination Folder
C:\Program Files\RStudio
Browse...
Space required: 777.5 MB
Space available: 43.5 GB
Nullsoft Install System v3.04
< Back
Next >
Cancel


RStudio Setup
Installing
Please wait while RStudio is being installed.
Extract: Qt5Svg.dll... 100%
Show details
Nullsoft Install System v3.04
< Back
Next >
Cancel


RStudio Setup
Choose Start Menu Folder
Choose a Start Menu folder for the RStudio shortcuts.
Select the Start Menu folder in which you would like to create the program's shortcuts. You can also enter a name to create a new folder.
RStudio
4K Download
7-Zip
Accessibility
Accessories
Administrative Tools
Advanced Renamer
Anaconda3 (64-bit)
Android Studio
AOMEI Partition Assistant Pro Edition 5.5
Bandicam
Bonjour Print Services
Do not create shortcuts
Nullsoft Install System v3.04
< Back
Install
Cancel


RStudio Setup
Completing RStudio Setup
RStudio has been installed on your computer.
Click Finish to close Setup.
< Back
Finish
Cancel


D:/Degree/sem-8/R-LAB/project/Diabetes_Prediction-master/Diabetes_Prediction-master - RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
cars-prediction.R
car_data
pred_car.R
getwd()
setwd("D:/Degree/sem-8/R-LAB/project/cars-prediction")
car_data <- read.csv("cars.csv", header = TRUE,sep = ";")
summary(car_data)
str(car_data)
split1 <- sample(2,nrow(car_data),replace = TRUE,prob = c(0.80,0.20))
rm(split1)
# rm(test_data)
train_data <- car_data[split1==1,]
test_data <- car_data[split1==2,]
rm(train_data)
rm(test_data)
# multiple linear regression model
Environment History Connections
Import Dataset
Global Environment
Data
mydf 25558 obs. of 26 variables
Files Plots Packages Help Viewer
Console Terminal Jobs
Natural language support but running in an English locale
R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.
Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.
[workspace loaded from D:/Degree/sem-8/R-LAB/project/Diabetes_Prediction-master/Diabetes_Prediction-master/.RData]

Conclusion:

In this experiment we have installed R and R-studio, to run the R-Studio first we need to install R.

Experiment Number: 02

Title: Basic Functionality of R-Variables, Basic Arithmetic, Help command

Date of Performance:

Date of Submission:

Grade:

Signature:

Aim: Basic Functionality of R- Variables, Basic Arithmetic,
Help command

Theory:

R-Variable:
A variable provides us with named storage that our programs can manipulate. A variable in R can store an atomic vector, group of atomic vectors or a combination of many R objects. A valid variable name consists of letters, numbers and the dot or underline characters.

Variable Assignment:
The variables can be assigned values using leftward, rightward and equal to operator.

Data type of variable:
In R, a variable itself is not declared of any data type, rather it gets the data type of the R - object assigned to it. So R is called a dynamically typed language, which means that we can change a variable's data type of the same variable again and again when using it in a program.

Basic Arithmetic:
The R Arithmetic operators include operators like Arithmetic Addition, Subtraction, Division, Multiplication, Exponent, Integer Division, and Modulus. All these R arithmetic operators are binary operators, which means they operate on two operands. The table below shows all the Arithmetic Operators in R Programming language with examples.

R ARITHMETIC OPERATORS	OPERATION	EXAMPLE
+	Addition	15 + 5 = 20
–	Subtraction	15 – 5 = 10
*	Multiplication	15 * 5 = 75
/	Division	15 / 5 = 3
%/%	Integer Division – Same as Division. but it returns the integer value by flooring the extra decimals	16 %/% 3 = 5. If you divide 16 with 3 you get 5.333, but the Integer division operator trims the decimal values and outputs the integer
^	Exponent – It returns the Power of One variable against the other	15 ^ 3 = 3375 (It means 15 Power 3 or 10^3).
%%	Modulus – It returns the remainder after the division	15 %% 5 = 0 (Here remainder is zero). If it is 17 %% 4, then result = 1.

Help Command:

The help() function and ? help operator in R provide access to the documentation pages for R functions, data sets, and other objects, both for packages in the standard R distribution and for contributed packages. To access documentation for the standard lm (linear model) function, for example, enter the command help(lm) or help("lm"), or ?lm or ?"lm" (i.e., the quotes are optional)

Program/Code:

```
>#Basic Arithmetic Operations.
> 2 - 5
[1] -3
> 6 / 3
[1] 2
> 3 + 2 * 5
[1] 13
> (3+2) * 5
[1] 25
> 4^3
[1] 64
```

```
> exp(4)
[1] 54.59815
> log(2.742)
[1] 1.008688
> log10(1000)
[1] 3
> pi
[1] 3.141593
>
> #Variable Name.
>
> x <- 5
> x
[1] 5
> var1 <- 7/2
> print(var1)
[1] 3.5
> valid.variable.name <- 18.6
> valid.variable.name
[1] 18.6
>
> #Variable and arithmetic operations.
>
> a <- 3
> a
[1] 3
>#Some common operations.
> sqrt(a)
[1] 1.732051
> a^4
[1] 81
> log(a)
[1] 1.098612
```

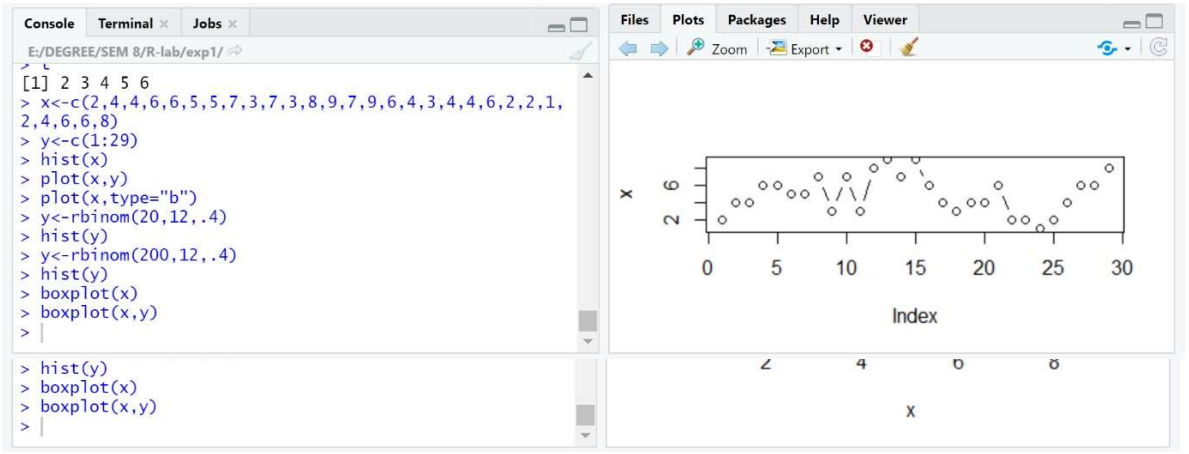
```
> log10(a)
[1] 0.4771213
> exp(a)
[1] 20.08554
> tan(a)
[1] -0.1425465
> b <- pi
> b
[1] 3.141593
>
> #Vectors and Matrices.
> #Vectors.
> x <- c(2, 3, 5, 1, 4, 4)
> x
[1] 2 3 5 1 4 4
> sum(x)
[1] 19
> mean(x)
[1] 3.166667
> sd(x)
[1] 1.47196
> median(x)
[1] 3.5
> sqrt(x)
[1] 1.414214 1.732051 2.236068 1.000000
2.000000 2.000000
> x^2
[1] 4 9 25 1 16 16
> seq(1, 10)
 [1] 1 2 3 4 5 6 7 8 9 10
> seq(1, 10, 2)
[1] 1 3 5 7 9
> seq(1:10)
```

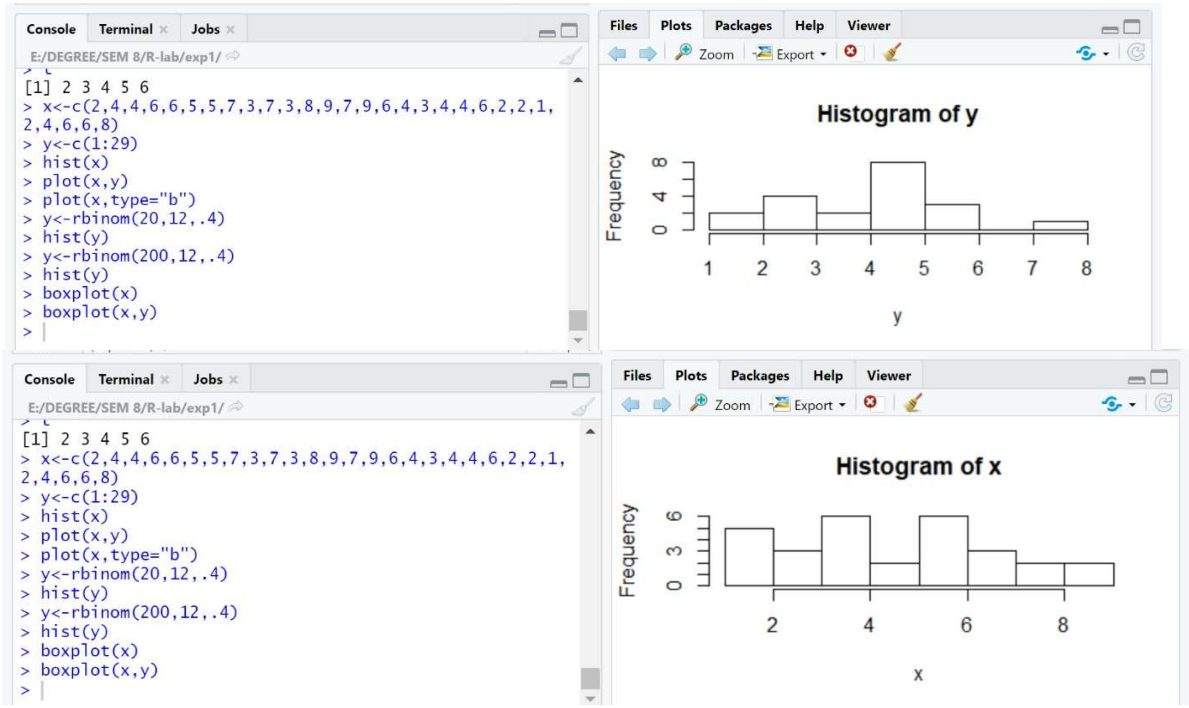
```
 [1] 1 2 3 4 5 6 7 8 9 10
> seq(1, 10, by=2)
[1] 1 3 5 7 9
> y <- c(1:7)
> y
[1] 1 2 3 4 5 6 7
> z <- 1:7
> z
[1] 1 2 3 4 5 6 7
> w <- c(1:12, 0, -6)
> w
 [1] 1 2 3 4 5 6 7 8 9 10 11 12 0 -6
>
> #Some operations with Numerical vectors and Logical vectors.
>
> x <- c(-5, 0, 7, -6, 14, 27)
> x[1]
[1] -5
> x[length(x)]
[1] 27
> x[c(2, 3)]
[1] 0 7
> x[-c(2, 3)]
[1] -5 -6 14 27
> x[c(1, 4)] <- c(2, 3)
> x < 3
[1] TRUE TRUE FALSE FALSE FALSE FALSE
> x[x < 3]
[1] 2 0
>
> #Intro to Graphing.
 > x <- c(2,4,4,6,6,5,5,7,3,7,3,8,9,7,9,6,4,3,4,4,6,2,2,1, 2,4,6,6,8)
> y <- c(1:29)
```

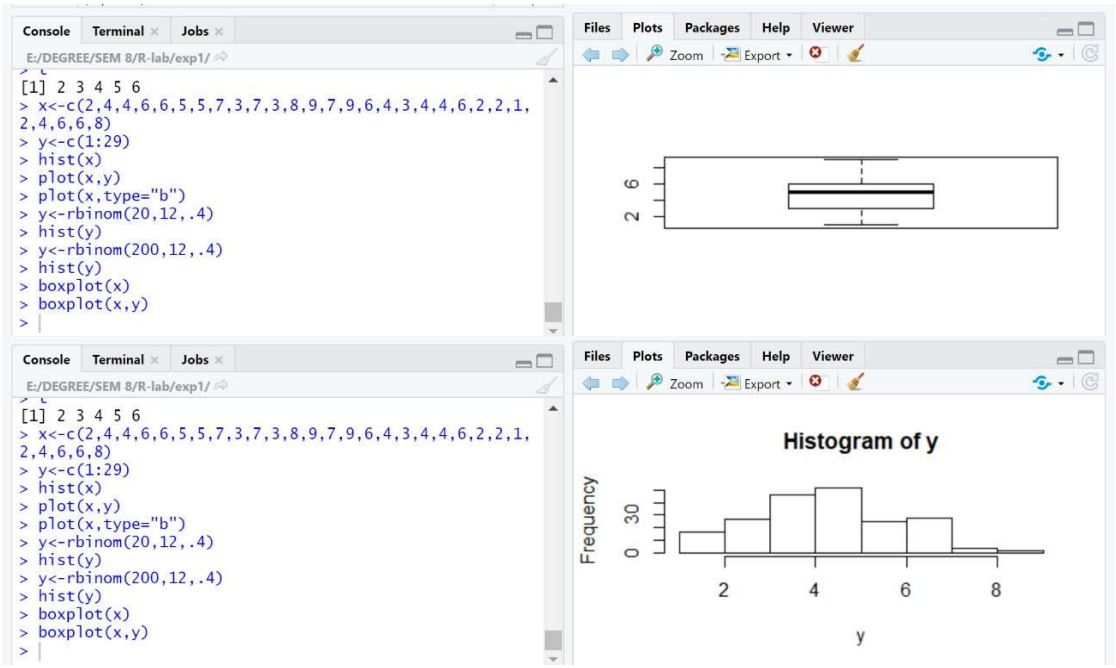
```
> hist(x)
> plot(x, y)
> plot(x, type = "b")
> y <- rbinom(20, 12, .4)
> hist(y)
> y <- rbinom(200, 12, .4)
> hist(y)
> boxplot(x)
> boxplot(x, y)

>#Logical operators.
> x <- 1:5
> test <- (x < 4)
> test
[1] TRUE TRUE TRUE FALSE FALSE
> test <- (x == 3)
> test
[1] FALSE FALSE TRUE FALSE FALSE
> test <- (x > 1 & x < 4)
> test
[1] FALSE TRUE TRUE FALSE FALSE
> test <- (x > 4|x == 2)
> test
[1] FALSE TRUE FALSE FALSE TRUE
> test <- (x != 4)
> test
[1] TRUE TRUE TRUE FALSE TRUE
> test <- (x %in% c(2, 4))
> test
[1] FALSE TRUE FALSE TRUE FALSE
> #Normal Distribution.
> pnorm(1.43)
[1] 0.9236415
```

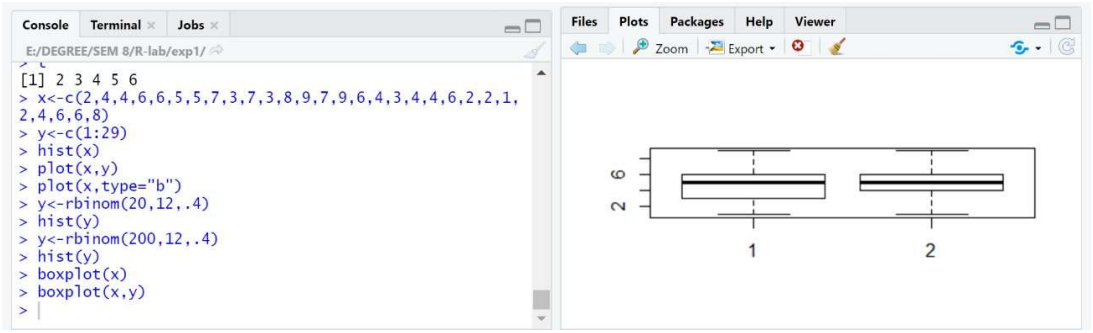
```
> pnorm(129, 100, 16)
[1] 0.9650455
> qnorm(.994)
[1] 2.512144
> qnorm(.95, 100, 16)
[1] 126.3177
> #Things to know.
> x <- runif(30, 0, 1)
> y <- c()
> for(i in 1:10){
+ y[i] <- rbinom(1, 3, .2)
+ }
> y
 [1] 0 0 0 1 0 2 1 1 0 1
> s <- c(1:5)
> s
[1] 1 2 3 4 5
> t = c(2:6)
> t
[1] 2 3 4 5 6
```

Outputs:


Console
Terminal
Jobs
E:/DEGREE/SEM 8/R-lab/exp1/
[1] 2 3 4 5 6
> x<-c(2,4,4,6,6,5,5,7,3,7,3,8,9,7,9,6,4,3,4,4,6,2,2,1,
2,4,6,6,8)
> y<-c(1:29)
> hist(x)
> plot(x,y)
> plot(x,type="b")
> y<-rbinom(20,12,.4)
> hist(y)
> y<-rbinom(200,12,.4)
> hist(y)
> boxplot(x)
> boxplot(x,y)
Files
Plots
Packages
Help
Viewer
Zoom
Export
x
Index
x


Console
Terminal
Jobs
E:/DEGREE/SEM 8/R-lab/exp1/
[1] 2 3 4 5 6
> x<-c(2,4,4,6,6,5,5,7,3,7,3,8,9,7,9,6,4,3,4,4,6,2,2,1,
2,4,6,6,8)
> y<-c(1:29)
> hist(x)
> plot(x,y)
> plot(x,type="b")
> y<-rbinom(20,12,.4)
> hist(y)
> y<-rbinom(200,12,.4)
> hist(y)
> boxplot(x)
> boxplot(x,y)
Files
Plots
Packages
Help
Viewer
Zoom
Export
Histogram of y
Frequency
y
Histogram of x
Frequency
x





Conclusion:

In the above experiment we have seen that how the Datatype can be defined in r, all the basic arithmetic and Help function.

Experiment Number: 03

Title: Basic Data types and data structures in R- vector, matrices, list, data frame.

Date of Performance:

Date of Submission:

Grade:

Signature:

Aim: Basic Data types and data structures in R- vector, matrices, list, data frame.

Theory:

Basic Datatypes:

R has 6 basic data types:

- character
- numeric (real or decimal)
- integer
- logical
- complex

Data Structures:

R has following data structures:

- atomic vector
- list
- matrix
- data frame
- factors

Vector:
A vector is the most common and basic data structure in R and is pretty much the workhorse of R. Technically, vectors can be one of two types:

- atomic vectors
- lists

Matrices:
In R matrices are an extension of the numeric or character vectors. They are not a separate type of object but simply an atomic vector with

dimensions; the number of rows and columns. As with atomic vectors, the elements of a matrix must be of the same data type.

Data Frame:
A data frame is a very important data type in R. It's pretty much the *de facto* data structure for most tabular data and what we use for statistics.
A data frame is a *special type of list* where every element of the list has same length (i.e. data frame is a "rectangular" list).
Data frames can have additional attributes such as rownames(), which can be useful for annotating data, like subject_id or sample_id. But most of the time they are not used.

Program/Code with outputs:

```
>#Example.
> x <- "dataset"
> typeof(x)
[1] "character"
> attributes(x)
NULL
>
> y<-1:10
> y
[1] 1 2 3 4 5 6 7 8 9 10
> typeof(y)
[1] "integer"
> length(y)
[1] 10
> z <- as.numeric(y)
> z
[1] 1 2 3 4 5 6 7 8 9 10
> typeof(z)
[1] "double"
```

```
>
> #Vector Modes.
>
> vector()
logical(0)
> vector("character", length=5)
[1] "" "" "" "" ""
> character(5)
[1] "" "" "" "" ""
> numeric(5)
[1] 0 0 0 0 0
> logical(5)
[1] FALSE FALSE FALSE FALSE FALSE
>
> x <- c(1, 2, 3)
> x1 <- c(1L, 2L, 3L)
> y <- c(TRUE, TRUE, FALSE, FALSE)
> z <- c("Sarah", "Tracy", "Jon")
> typeof(z)
[1] "character"
> length(z)
[1] 3
> class(z)
[1] "character"
> str(z)
chr [1:3] "Sarah" "Tracy" "Jon"
> z <- c(z, "Annette")
> z
[1] "Sarah" "Tracy" "Jon" "Annette"
> z <- c("Greg", z)
> z
[1] "Greg" "Sarah" "Tracy" "Jon" "Annette"
>
```

```
> #Vectors for sequence number.
>
> series <- 1:10
> seq(10)
[1] 1 2 3 4 5 6 7 8 9 10
> seq(from=1, to=10, by=0.1)
[1] 1.0 1.1 1.2 1.3 1.4 1.5 1.6 1.7 1.8 1.9 2.0 2.1 2.2 2.3 2.4 2.5 2.6 2.7 2.8
2.9 3.0
[22] 3.1 3.2 3.3 3.4 3.5 3.6 3.7 3.8 3.9 4.0 4.1 4.2 4.3 4.4 4.5 4.6 4.7 4.8
4.9 5.0 5.1
[43] 5.2 5.3 5.4 5.5 5.6 5.7 5.8 5.9 6.0 6.1 6.2 6.3 6.4 6.5 6.6 6.7 6.8 6.9
7.0 7.1 7.2
[64] 7.3 7.4 7.5 7.6 7.7 7.8 7.9 8.0 8.1 8.2 8.3 8.4 8.5 8.6 8.7 8.8 8.9 9.0
9.1 9.2 9.3
[85] 9.4 9.5 9.6 9.7 9.8 9.9 10.0
>
> #Missing Data.
>
> x <- c(0.5, NA, 0.7)
> x <- c(TRUE, FALSE, NA)
> x <- c("a", NA, "c", "d", "e")
> x <- c(1+5i, 2-3i, NA)
> x <- c("a", NA, "c", "d", NA)
> y <- c("a", "b", "c", "d", "e")
> is.na(x)
[1] FALSE TRUE FALSE FALSE TRUE
> is.na(y)
[1] FALSE FALSE FALSE FALSE FALSE
> anyNA(x)
[1] TRUE
> anyNA(y)
[1] FALSE
>
```

```
> #Other special values.
>
> 1 / 0
[1] Inf
> 0 / 0
[1] NaN
>
> #Mix types inside a vector.
>
> xx <- c(1.7, "a")
>xx
[1] 1.7 “a”
> xx <- c(TRUE, 2)
>xx
[1] TRUE 2
> xx <- c("a", TRUE)
>xx
[1] “a” TRUE
> as.numeric("1")
[1] 1
> as.character(1:2)
[1] "1" "2"
>
> #Object attributes.
>
> length(1:10)
[1] 10
> nchar("Software Carpentry")
[1] 18
>
> #Matrix.
>
> m <-matrix(nrow  = 2, ncol = 2)
```

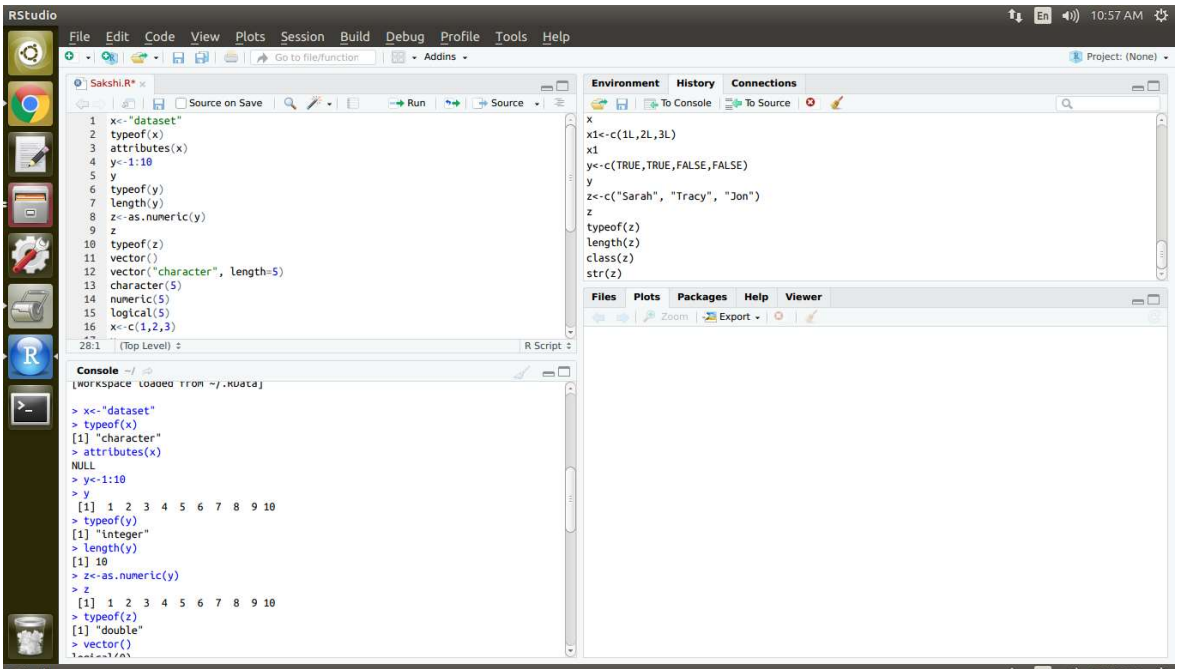
```
> m
[,1] [,2]
[1,] NA NA
[2,] NA NA
> dim(m)
[1] 2 2
> m <- matrix(c( 1:3 ))
> class(m)
[1] "matrix"
> typeof(m)
[1] "integer"
>
> #Data Types of matrix elements.
>
> FOURS <- matrix(
+ c(4,  4, 4, 4),
+ nrow = 2,
+ ncol = 2)
> FOURS
      [,1]  [,2]
[1,]    4     4
[2,]    4     4
>
> m <- matrix(1:6, nrow = 2, ncol = 3)
> m
      [,1]  [,2]  [,3]
[1,]    1     3     5
[2,]    2     4     6
> m <- 1:10
> dim(m) <- c(2, 5)
>m
      [,1]  [,2]  [,3]  [,4]  [,5]
[1,]    1     3     5     7     9
```

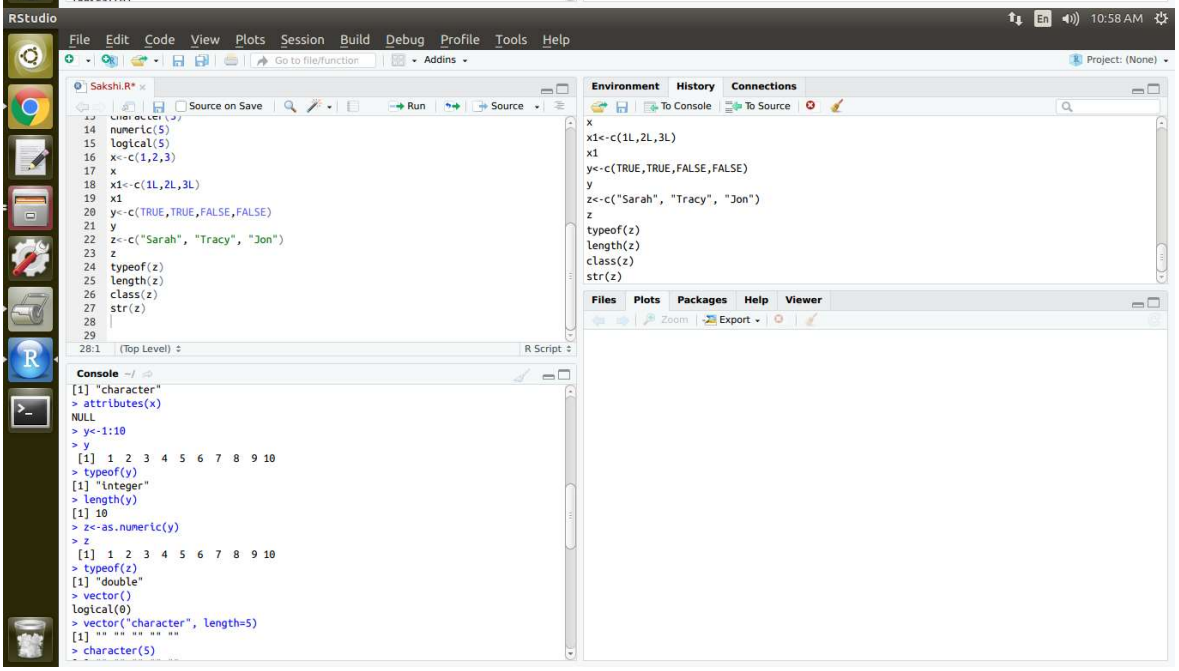
```
[2,]    2    4    6    8   10
>
> x <- 1:3
> y <- 10:12
> cbind(x, y)
     x  y
[1,] 1 10
[2,] 2 11
[3,] 3 12
> rbind(x, y)
  [,1] [,2] [,3]
x    1    2    3
y   10   11   12
> mdat <- matrix(c(1, 2, 3, 11, 12, 13),
+ nrow = 2,
+ ncol = 3,
+ byrow = TRUE)
> mdat
     [,1] [,2] [,3]
[1,]    1    2    3
[2,]   11   12   13
> mdat[2, 3]
[1] 13
>
> #List.
>
> x <- list(1, "a", TRUE, 1+4i)
> x
[[1]]
[1] 1
[[2]]
[1] "a"
[[3]]
```

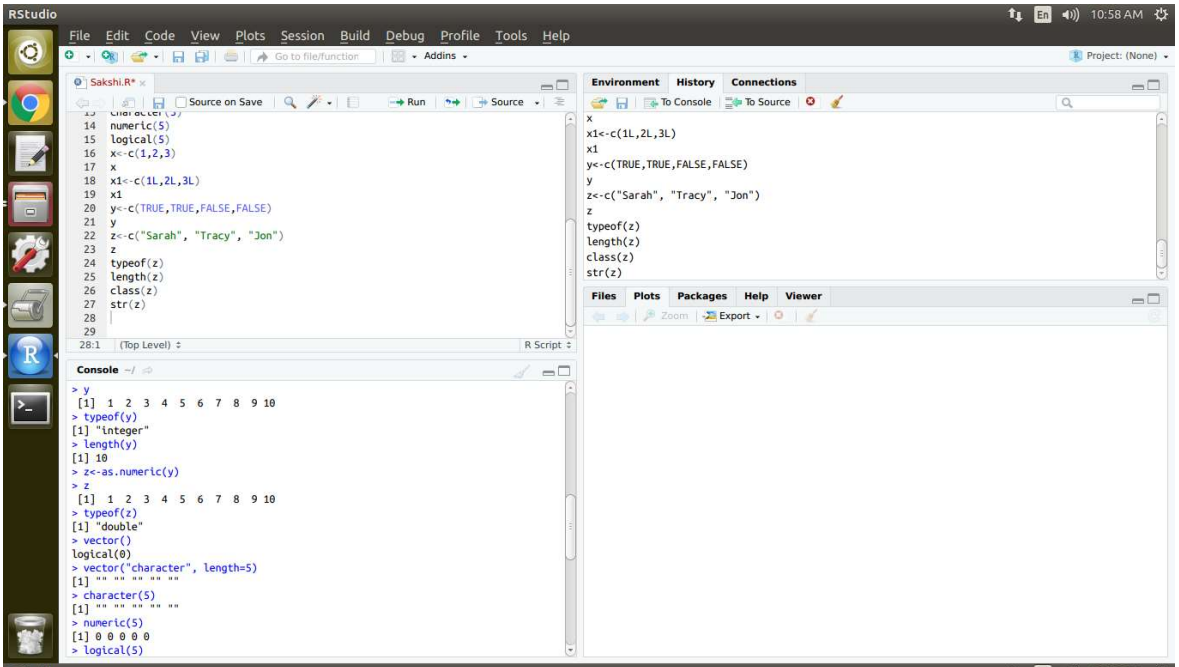
```
[1] TRUE
[[4]]
[1] 1+4i
> x <- vector("list", length = 5)
> length(x)
[1] 5
> x[[1]]
NULL
> x <- 1:10
> x <- as.list(x)
> length(x)
[1] 10
>
> xlist <- list(a = "Sakshi Agrawal", b = 1:10, data = head(iris))
> xlist
$a
[1] "Sakshi Agrawal"
$b
[1] 1 2 3 4 5 6 7 8 9 10
$data
  Sepal.Length Sepal.Width 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
> names(xlist)
[1] "a" "b" "data"
>
> #Data Frame.
>
> dat<-data.frame(id=letters[1:10],x=1:10,y=11:20)
```

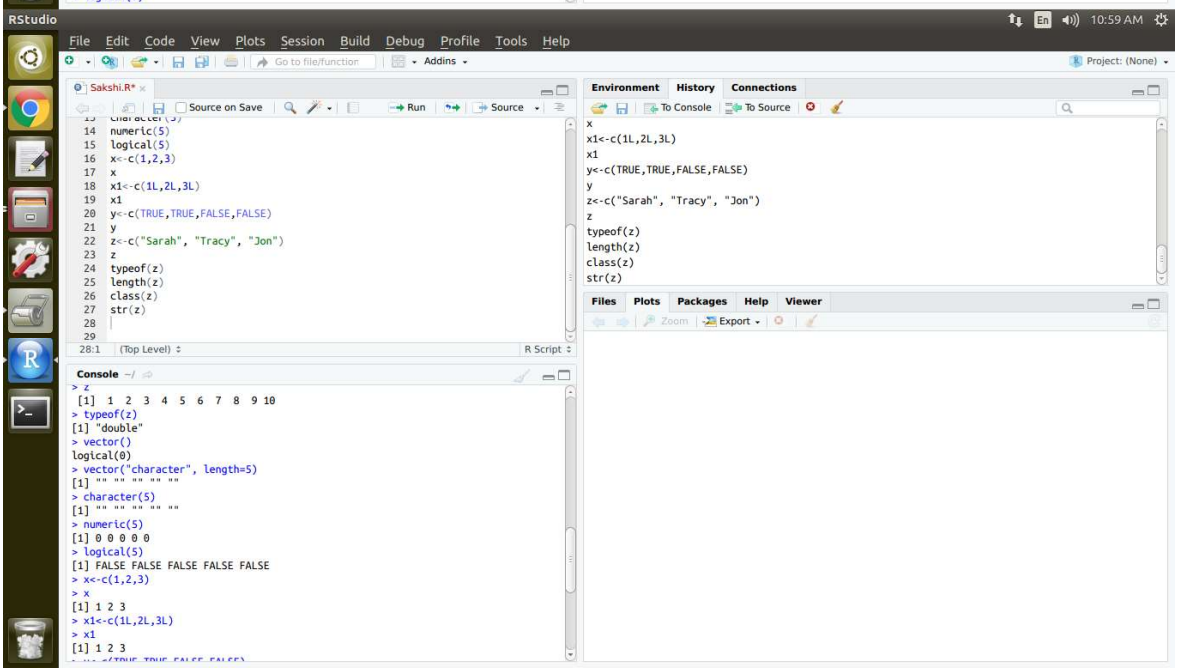
```
> dat
id      x       y
1       a1      11
2       b2      12
3       c3      13
4       d4      14
5       e5      15
6       f6      16
7       g7      17
8       h8      18
9       i9      19
10      j10     20
> is.list(dat)
[1] TRUE
> class(dat)
[1] "data.frame"
> dat[1,3]
[1] 11
> dat[["y"]]
[1] 11 12 13 14 15 16 17 18 19 20
> dat$y
[1] 11 12 13 14 15 16 17 18 19 20
> 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
...
```

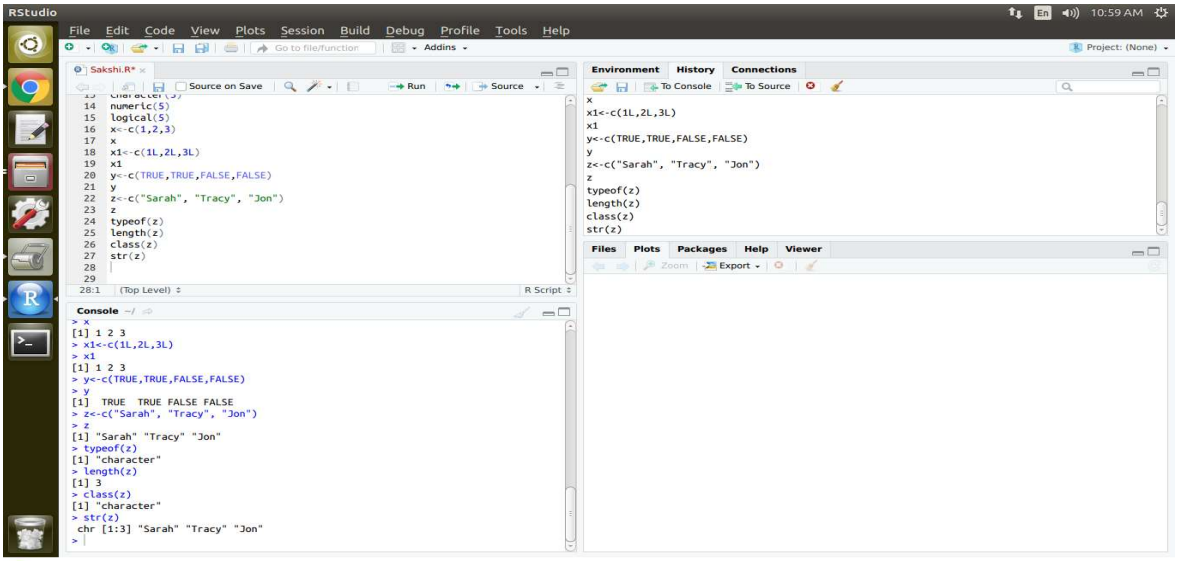
OUTPUTS:

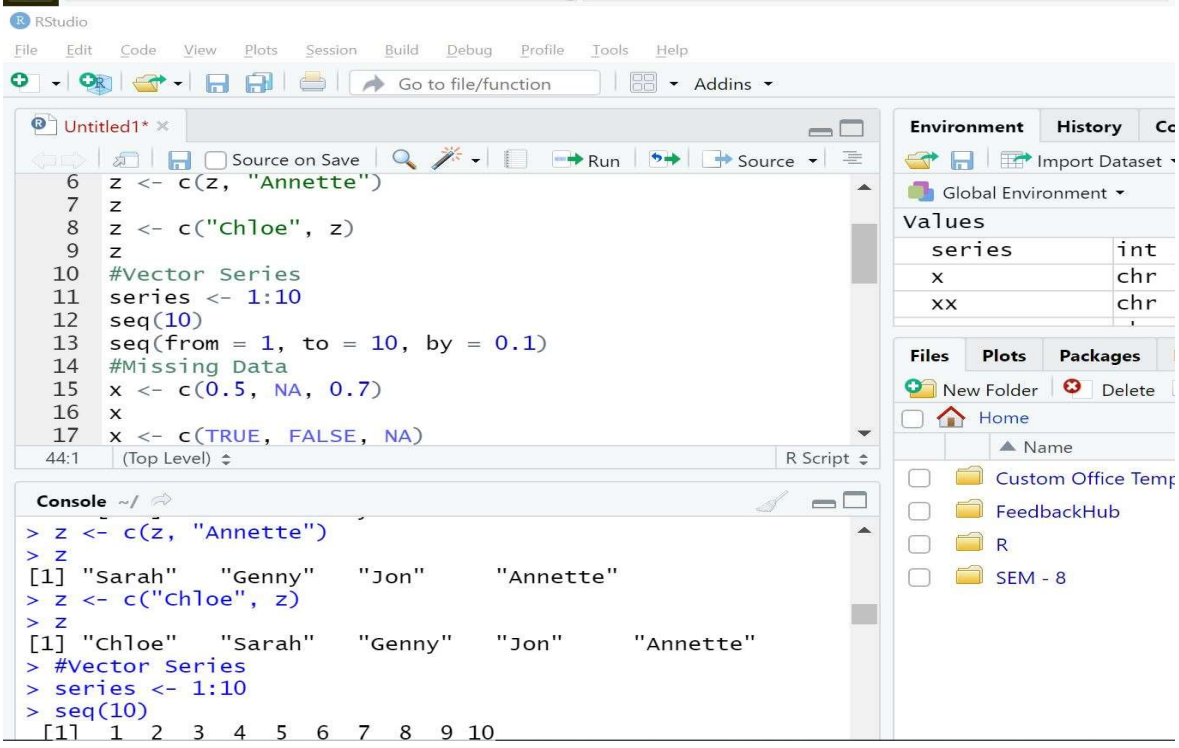

RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
Sakshi.R*
Source on Save
Run
Source
1 x<-"dataset"
2 typeof(x)
3 attributes(x)
4 y<-1:10
5 y
6 typeof(y)
7 length(y)
8 z<-as.numeric(y)
9 z
10 typeof(z)
11 vector()
12 vector("character", length=5)
13 character(5)
14 numeric(5)
15 logical(5)
16 x<-c(1,2,3)
28:1 (Top Level) R Script
Console ~/
[Workspace loaded from ~/.RData]
> x<-"dataset"
> typeof(x)
[1] "character"
> attributes(x)
NULL
> y<-1:10
> y
[1] 1 2 3 4 5 6 7 8 9 10
> typeof(y)
[1] "integer"
> length(y)
[1] 10
> z<-as.numeric(y)
> z
[1] 1 2 3 4 5 6 7 8 9 10
> typeof(z)
[1] "double"
> vector()
Environment History Connections
To Console To Source
x
x1<-c(1L,2L,3L)
x1
y<-c(TRUE,TRUE,FALSE,FALSE)
y
z<-c("Sarah", "Tracy", "Jon")
z
typeof(z)
length(z)
class(z)
str(z)
Files Plots Packages Help Viewer
Zoom Export
10:57 AM
Project: (None)

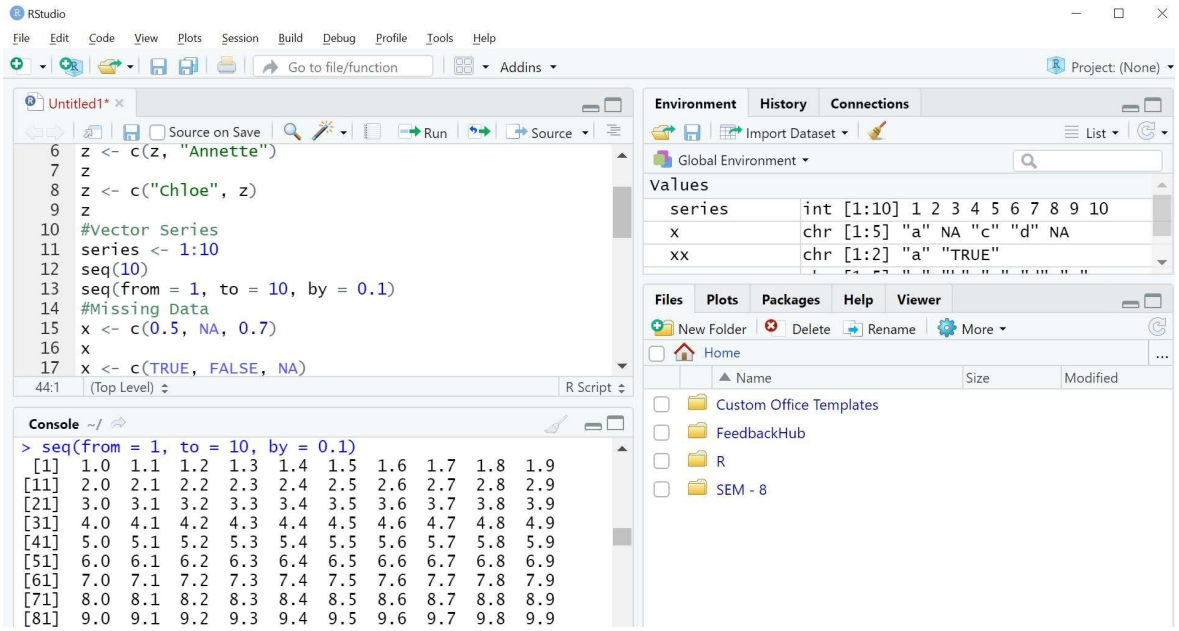

RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
Sakshi.R*
Source on Save
Run
Source
14 numeric(5)
15 logical(5)
16 x<-c(1,2,3)
17 x
18 x1<-c(1L,2L,3L)
19 x1
20 y<-c(TRUE,TRUE,FALSE,FALSE)
21 y
22 z<-c("Sarah", "Tracy", "Jon")
23 z
24 typeof(z)
25 length(z)
26 class(z)
27 str(z)
28
29
28:1 (Top Level) R Script
Console ~/
[1] "character"
> attributes(x)
NULL
> y<-1:10
> y
[1] 1 2 3 4 5 6 7 8 9 10
> typeof(y)
[1] "integer"
> length(y)
[1] 10
> z<-as.numeric(y)
> z
[1] 1 2 3 4 5 6 7 8 9 10
> typeof(z)
[1] "double"
> vector()
logical(0)
> vector("character", length=5)
[1] "" "" "" "" ""
> character(5)
Environment History Connections
To Console To Source
x
x1<-c(1L,2L,3L)
x1
y<-c(TRUE,TRUE,FALSE,FALSE)
y
z<-c("Sarah", "Tracy", "Jon")
z
typeof(z)
length(z)
class(z)
str(z)
Files Plots Packages Help Viewer
Zoom Export
10:58 AM
Project: (None)

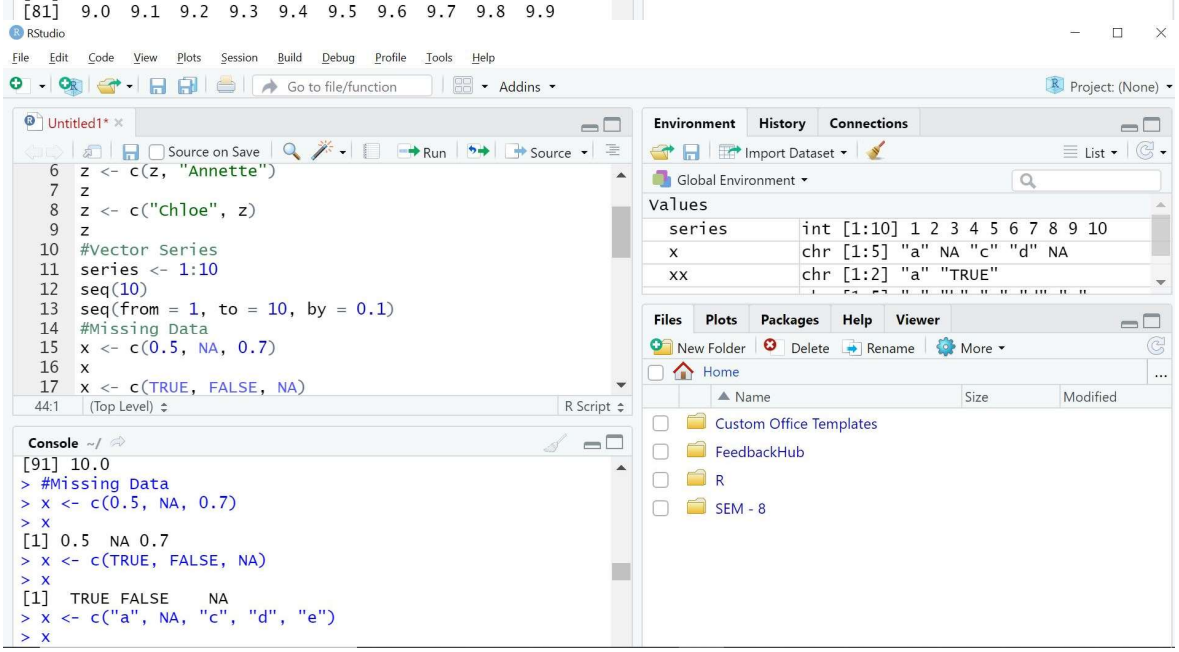

RStudio
10:58 AM
File Edit Code View Plots Session Build Debug Profile Tools Help
Sakshi.R*
Source on Save
Run
Source
numeric(5)
logical(5)
x<-c(1,2,3)
x
x1<-c(1L,2L,3L)
x1
y<-c(TRUE,TRUE,FALSE,FALSE)
y
z<-c("Sarah", "Tracy", "Jon")
z
typeof(z)
length(z)
class(z)
str(z)
28:1 (Top Level) R Script
Console
> y
[1] 1 2 3 4 5 6 7 8 9 10
> typeof(y)
[1] "integer"
> length(y)
[1] 10
> z<-as.numeric(y)
> z
[1] 1 2 3 4 5 6 7 8 9 10
> typeof(z)
[1] "double"
> vector()
logical(0)
> vector("character", length=5)
[1] "" "" "" "" ""
> character(5)
[1] "" "" "" "" ""
> numeric(5)
[1] 0 0 0 0 0
> logical(5)
Environment History Connections
To Console
To Source
x
x1<-c(1L,2L,3L)
x1
y<-c(TRUE,TRUE,FALSE,FALSE)
y
z<-c("Sarah", "Tracy", "Jon")
z
typeof(z)
length(z)
class(z)
str(z)
Files Plots Packages Help Viewer
Zoom
Export

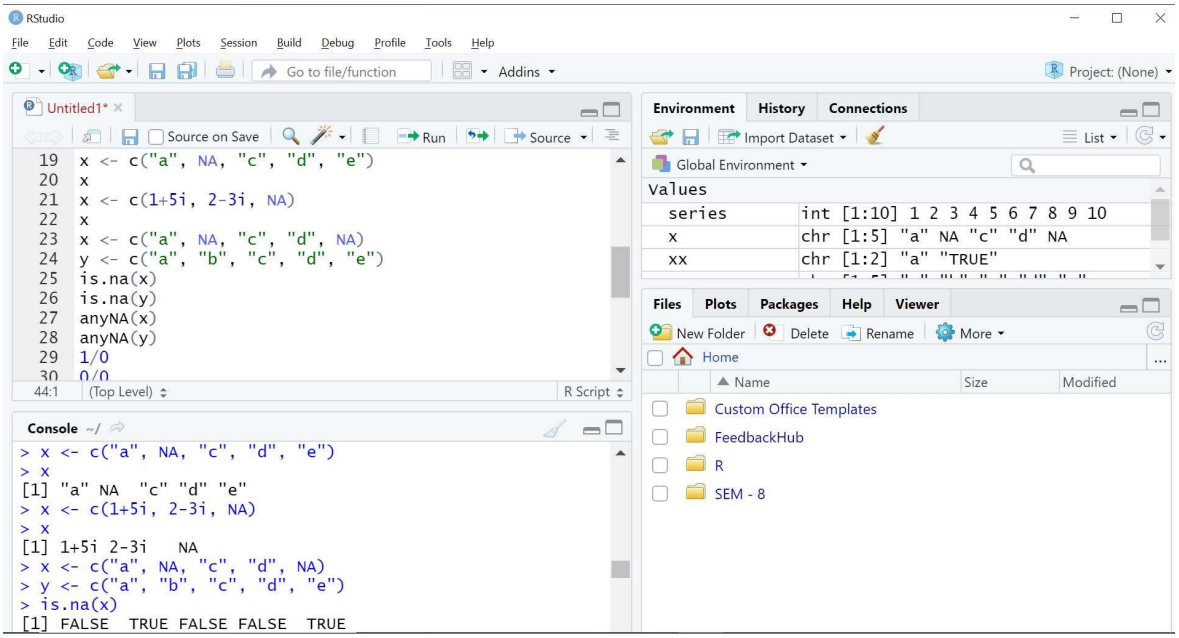

RStudio
10:59 AM
File Edit Code View Plots Session Build Debug Profile Tools Help
Sakshi.R*
Source on Save
Run
Source
numeric(5)
logical(5)
x<-c(1,2,3)
x
x1<-c(1L,2L,3L)
x1
y<-c(TRUE,TRUE,FALSE,FALSE)
y
z<-c("Sarah", "Tracy", "Jon")
z
typeof(z)
length(z)
class(z)
str(z)
28:1 (Top Level) R Script
Console
> z
[1] 1 2 3 4 5 6 7 8 9 10
> typeof(z)
[1] "double"
> vector()
logical(0)
> vector("character", length=5)
[1] "" "" "" "" ""
> character(5)
[1] "" "" "" "" ""
> numeric(5)
[1] 0 0 0 0 0
> logical(5)
[1] FALSE FALSE FALSE FALSE FALSE
> x<-c(1,2,3)
> x
[1] 1 2 3
> x1<-c(1L,2L,3L)
> x1
[1] 1 2 3
Environment History Connections
To Console
To Source
x
x1<-c(1L,2L,3L)
x1
y<-c(TRUE,TRUE,FALSE,FALSE)
y
z<-c("Sarah", "Tracy", "Jon")
z
typeof(z)
length(z)
class(z)
str(z)
Files Plots Packages Help Viewer
Zoom
Export

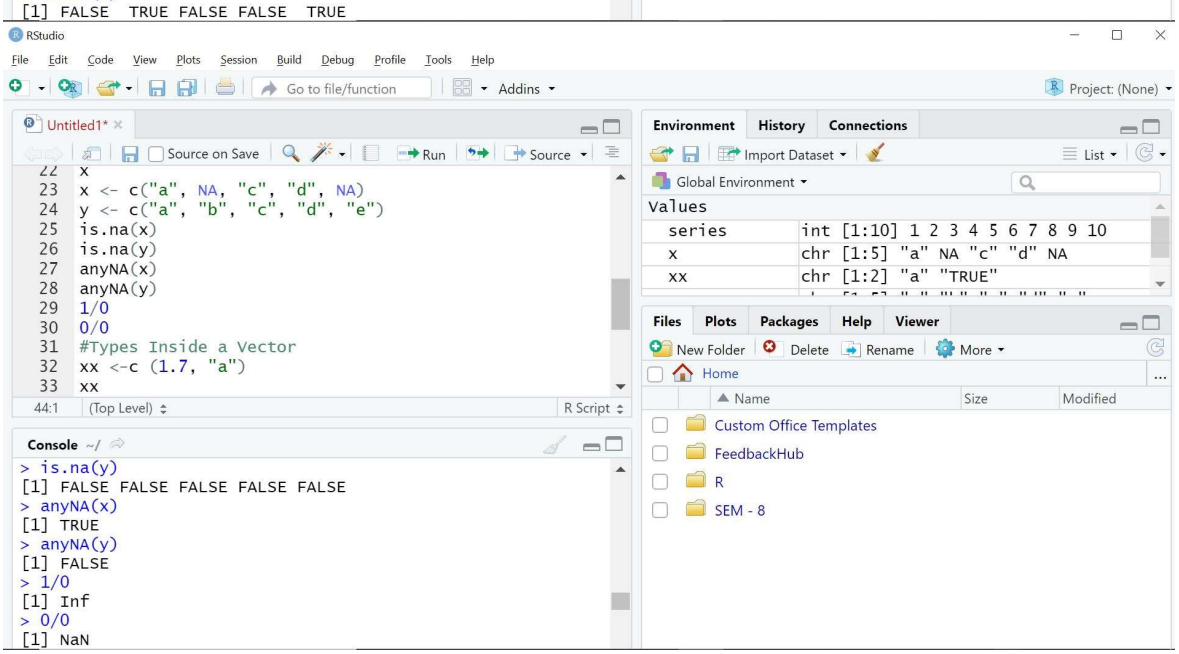

RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
Sakshi.R*
Source on Save
Run
Source
numeric(5)
logical(5)
x<-c(1,2,3)
x
x1<-c(1L,2L,3L)
x1
y<-c(TRUE,TRUE,FALSE,FALSE)
y
z<-c("Sarah", "Tracy", "Jon")
z
typeof(z)
length(z)
class(z)
str(z)
28:1 (Top Level) R Script
Console ~/
> x
[1] 1 2 3
> x1<-c(1L,2L,3L)
> x1
[1] 1 2 3
> y<-c(TRUE,TRUE,FALSE,FALSE)
> y
[1] TRUE TRUE FALSE FALSE
> z<-c("Sarah", "Tracy", "Jon")
> z
[1] "Sarah" "Tracy" "Jon"
> typeof(z)
[1] "character"
> length(z)
[1] 3
> class(z)
[1] "character"
> str(z)
chr [1:3] "Sarah" "Tracy" "Jon"
Environment History Connections
To Console To Source
Files Plots Packages Help Viewer
Zoom Export

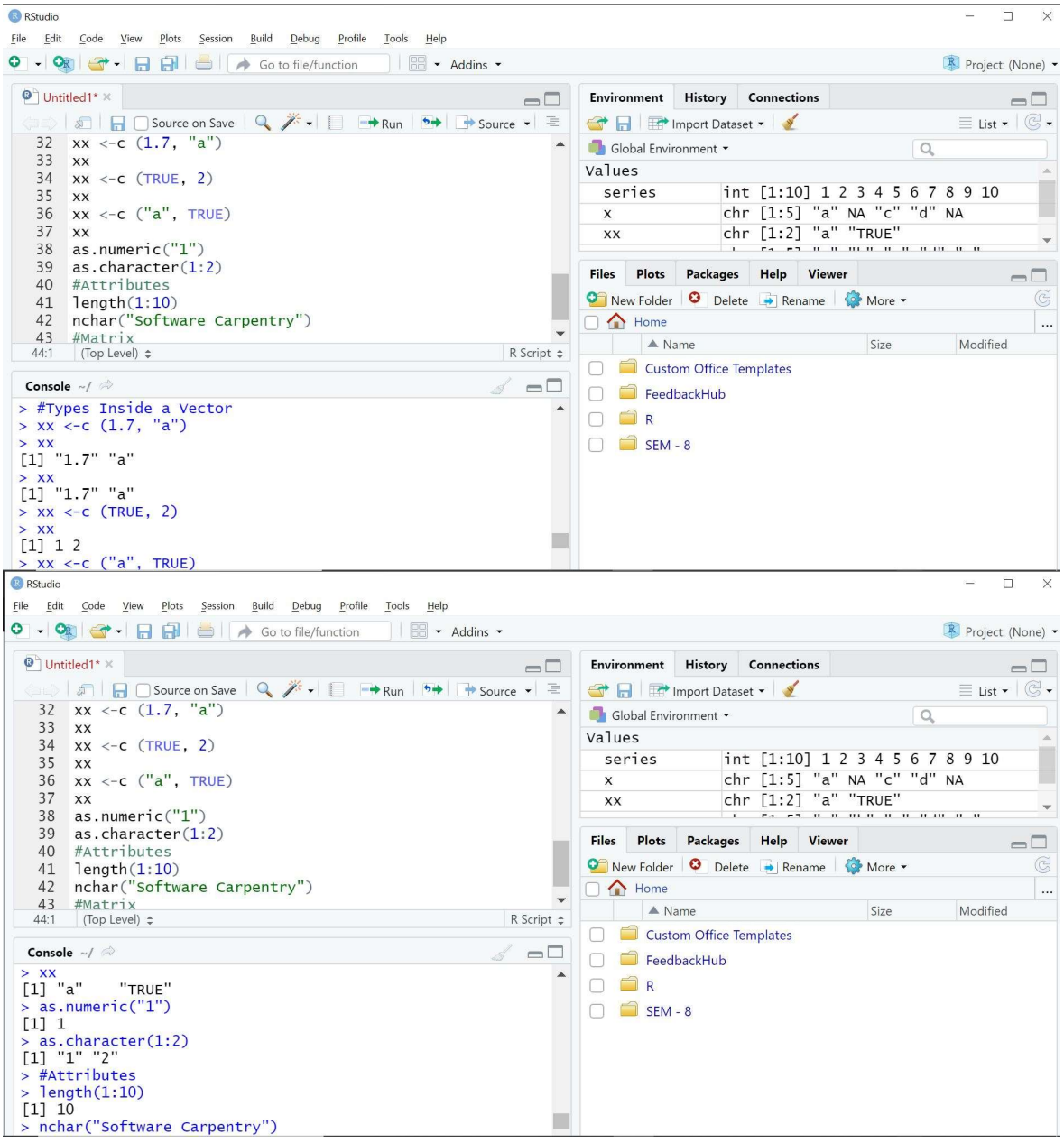

RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
Go to file/function
Addins
Untitled1*
Source on Save
Run
Source
6 z <- c(z, "Annette")
7 z
8 z <- c("Chloe", z)
9 z
10 #Vector Series
11 series <- 1:10
12 seq(10)
13 seq(from = 1, to = 10, by = 0.1)
14 #Missing Data
15 x <- c(0.5, NA, 0.7)
16 x
17 x <- c(TRUE, FALSE, NA)
44:1 (Top Level) R Script
Console ~/
> z <- c(z, "Annette")
> z
[1] "Sarah" "Genny" "Jon" "Annette"
> z <- c("Chloe", z)
> z
[1] "Chloe" "Sarah" "Genny" "Jon" "Annette"
> #Vector Series
> series <- 1:10
> seq(10)
[1] 1 2 3 4 5 6 7 8 9 10
Environment History
Import Dataset
Global Environment
Values
series int
x chr
xx chr
Files Plots Packages
New Folder Delete
Home
Name
Custom Office Temp
FeedbackHub
R
SEM - 8


RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
Go to file/function
Addins
Project: (None)
Untitled1*
Source on Save
Run
Source
6 z <- c(z, "Annette")
7 z
8 z <- c("Chloe", z)
9 z
10 #Vector Series
11 series <- 1:10
12 seq(10)
13 seq(from = 1, to = 10, by = 0.1)
14 #Missing Data
15 x <- c(0.5, NA, 0.7)
16 x
17 x <- c(TRUE, FALSE, NA)
44:1 (Top Level) R Script
Console ~/
> seq(from = 1, to = 10, by = 0.1)
[1] 1.0 1.1 1.2 1.3 1.4 1.5 1.6 1.7 1.8 1.9
[11] 2.0 2.1 2.2 2.3 2.4 2.5 2.6 2.7 2.8 2.9
[21] 3.0 3.1 3.2 3.3 3.4 3.5 3.6 3.7 3.8 3.9
[31] 4.0 4.1 4.2 4.3 4.4 4.5 4.6 4.7 4.8 4.9
[41] 5.0 5.1 5.2 5.3 5.4 5.5 5.6 5.7 5.8 5.9
[51] 6.0 6.1 6.2 6.3 6.4 6.5 6.6 6.7 6.8 6.9
[61] 7.0 7.1 7.2 7.3 7.4 7.5 7.6 7.7 7.8 7.9
[71] 8.0 8.1 8.2 8.3 8.4 8.5 8.6 8.7 8.8 8.9
[81] 9.0 9.1 9.2 9.3 9.4 9.5 9.6 9.7 9.8 9.9
Environment History Connections
Import Dataset
List
Global Environment
Values
series int [1:10] 1 2 3 4 5 6 7 8 9 10
x chr [1:5] "a" NA "c" "d" NA
xx chr [1:2] "a" "TRUE"
Files Plots Packages Help Viewer
New Folder Delete Rename More
Home
Name Size Modified
Custom Office Templates
FeedbackHub
R
SEM - 8

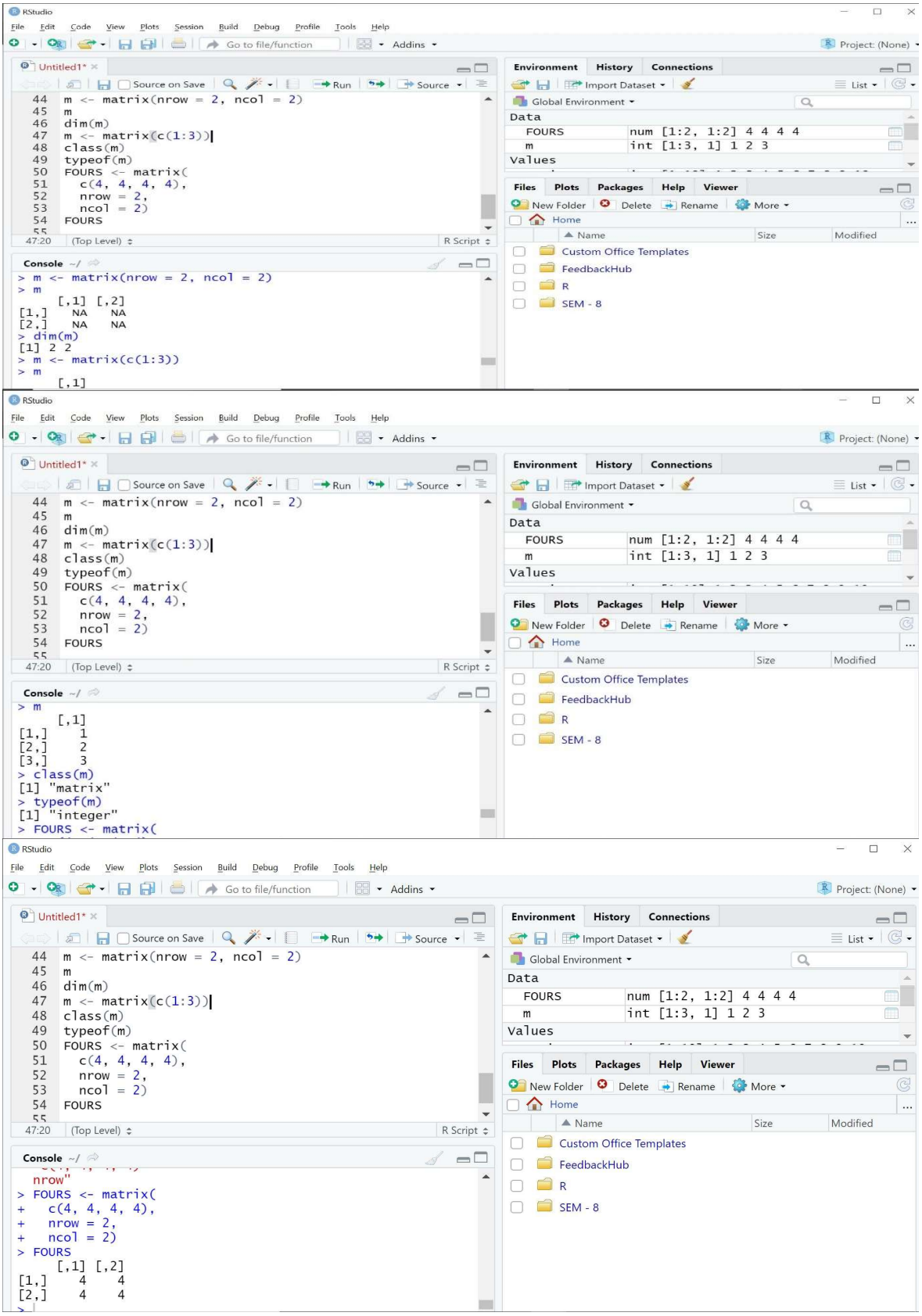

RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
Go to file/function
Addins
Project: (None)
Untitled1*
Source on Save
Run
Source
6 z <- c(z, "Annette")
7 z
8 z <- c("Chloe", z)
9 z
10 #Vector Series
11 series <- 1:10
12 seq(10)
13 seq(from = 1, to = 10, by = 0.1)
14 #Missing Data
15 x <- c(0.5, NA, 0.7)
16 x
17 x <- c(TRUE, FALSE, NA)
44:1 (Top Level) R Script
Console ~/
[91] 10.0
> #Missing Data
> x <- c(0.5, NA, 0.7)
> x
[1] 0.5 NA 0.7
> x <- c(TRUE, FALSE, NA)
> x
[1] TRUE FALSE NA
> x <- c("a", NA, "c", "d", "e")
> x
Environment History Connections
Import Dataset
List
Global Environment
Values
series int [1:10] 1 2 3 4 5 6 7 8 9 10
x chr [1:5] "a" NA "c" "d" NA
xx chr [1:2] "a" "TRUE"
Files Plots Packages Help Viewer
New Folder Delete Rename More
Home
Name Size Modified
Custom Office Templates
FeedbackHub
R
SEM - 8

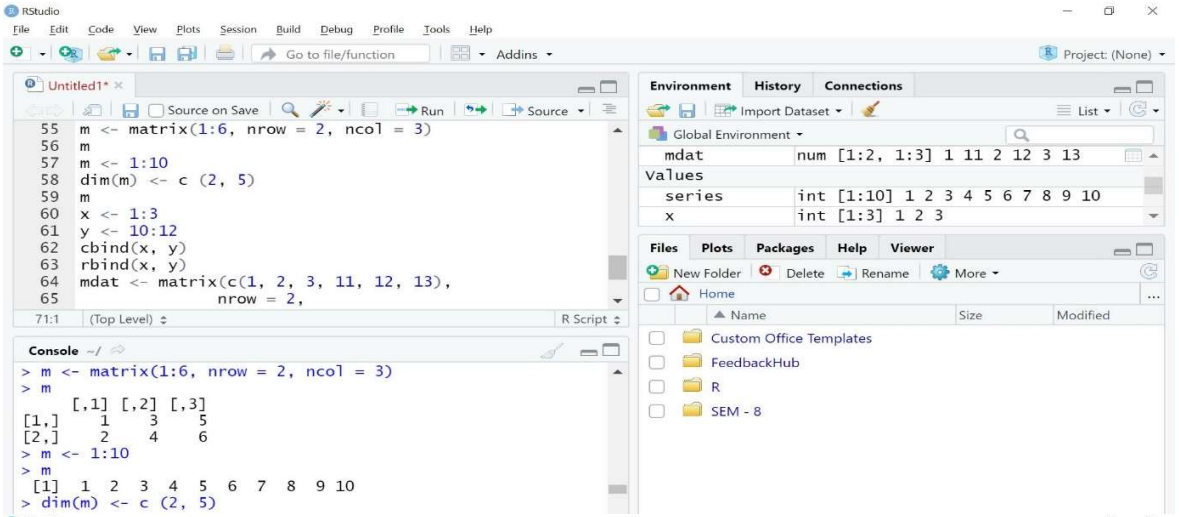

RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
Go to file/function
Addins
Project: (None)
Untitled1*
Source on Save
Run
Source
19 x <- c("a", NA, "c", "d", "e")
20 x
21 x <- c(1+5i, 2-3i, NA)
22 x
23 x <- c("a", NA, "c", "d", NA)
24 y <- c("a", "b", "c", "d", "e")
25 is.na(x)
26 is.na(y)
27 anyNA(x)
28 anyNA(y)
29 1/0
30 0/0
44:1 (Top Level)
R Script
Console ~/
> x <- c("a", NA, "c", "d", "e")
> x
[1] "a" NA "c" "d" "e"
> x <- c(1+5i, 2-3i, NA)
> x
[1] 1+5i 2-3i NA
> x <- c("a", NA, "c", "d", NA)
> y <- c("a", "b", "c", "d", "e")
> is.na(x)
[1] FALSE TRUE FALSE FALSE TRUE
Environment History Connections
Import Dataset
List
Global Environment
Values
series int [1:10] 1 2 3 4 5 6 7 8 9 10
x chr [1:5] "a" NA "c" "d" NA
xx chr [1:2] "a" "TRUE"
Files Plots Packages Help Viewer
New Folder Delete Rename More
Home
Name Size Modified
Custom Office Templates
FeedbackHub
R
SEM - 8

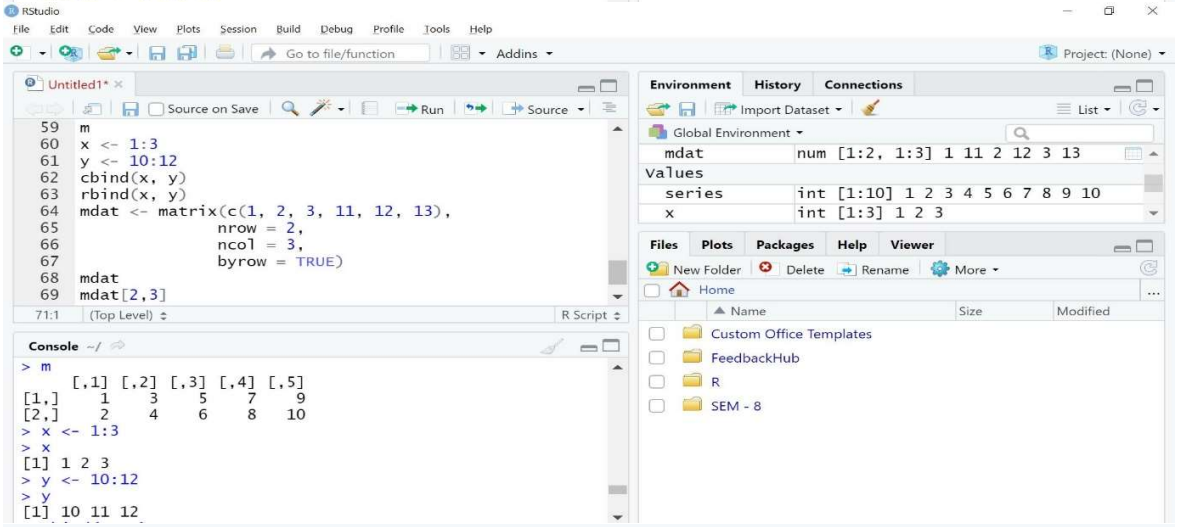

RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
Go to file/function
Addins
Project: (None)
Untitled1*
Source on Save
Run
Source
22 x
23 x <- c("a", NA, "c", "d", NA)
24 y <- c("a", "b", "c", "d", "e")
25 is.na(x)
26 is.na(y)
27 anyNA(x)
28 anyNA(y)
29 1/0
30 0/0
31 #Types Inside a Vector
32 xx <-c (1.7, "a")
33 xx
44:1 (Top Level)
R Script
Console ~/
> is.na(y)
[1] FALSE FALSE FALSE FALSE FALSE
> anyNA(x)
[1] TRUE
> anyNA(y)
[1] FALSE
> 1/0
[1] Inf
> 0/0
[1] NaN
Environment History Connections
Import Dataset
List
Global Environment
Values
series int [1:10] 1 2 3 4 5 6 7 8 9 10
x chr [1:5] "a" NA "c" "d" NA
xx chr [1:2] "a" "TRUE"
Files Plots Packages Help Viewer
New Folder Delete Rename More
Home
Name Size Modified
Custom Office Templates
FeedbackHub
R
SEM - 8

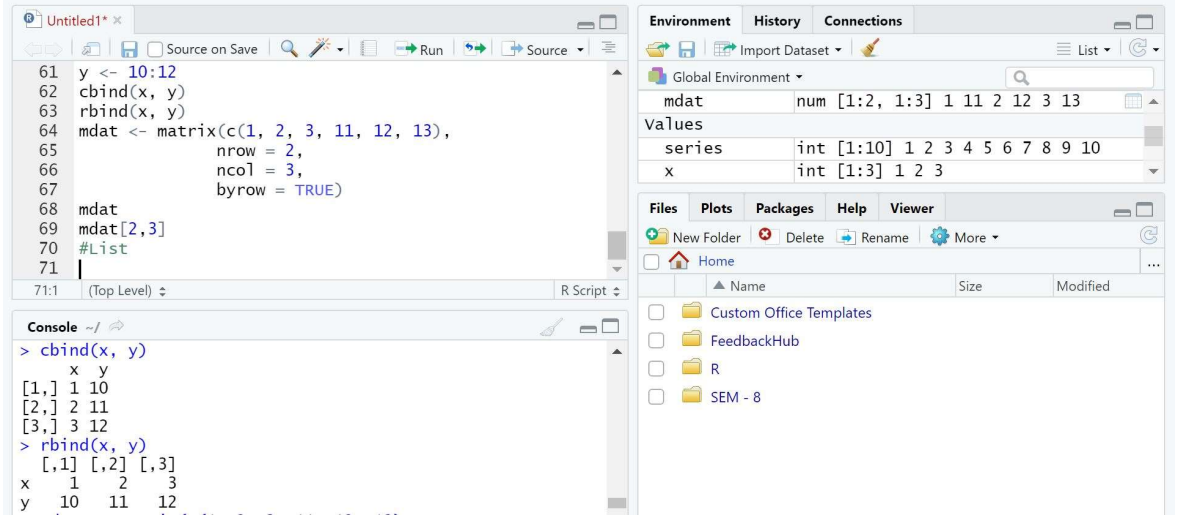

RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
Go to file/function
Addins
Project: (None)
Untitled1*
Source on Save
Run
Source
32 xx <-c (1.7, "a")
33 xx
34 xx <-c (TRUE, 2)
35 xx
36 xx <-c ("a", TRUE)
37 xx
38 as.numeric("1")
39 as.character(1:2)
40 #Attributes
41 length(1:10)
42 nchar("Software Carpentry")
43 #Matrix
44:1 (Top Level) R Script
Console ~/
> #Types Inside a Vector
> xx <-c (1.7, "a")
> xx
[1] "1.7" "a"
> xx
[1] "1.7" "a"
> xx <-c (TRUE, 2)
> xx
[1] 1 2
> xx <-c ("a", TRUE)
Environment History Connections
Import Dataset
List
Global Environment
Values
series int [1:10] 1 2 3 4 5 6 7 8 9 10
x chr [1:5] "a" NA "c" "d" NA
xx chr [1:2] "a" "TRUE"
Files Plots Packages Help Viewer
New Folder Delete Rename More
Home
Name Size Modified
Custom Office Templates
FeedbackHub
R
SEM - 8

RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
Go to file/function
Addins
Project: (None)
Untitled1*
Source on Save
Run
Source
32 xx <-c (1.7, "a")
33 xx
34 xx <-c (TRUE, 2)
35 xx
36 xx <-c ("a", TRUE)
37 xx
38 as.numeric("1")
39 as.character(1:2)
40 #Attributes
41 length(1:10)
42 nchar("Software Carpentry")
43 #Matrix
44:1 (Top Level) R Script
Console ~/
> xx
[1] "a" "TRUE"
> as.numeric("1")
[1] 1
> as.character(1:2)
[1] "1" "2"
> #Attributes
> length(1:10)
[1] 10
> nchar("Software Carpentry")
Environment History Connections
Import Dataset
List
Global Environment
Values
series int [1:10] 1 2 3 4 5 6 7 8 9 10
x chr [1:5] "a" NA "c" "d" NA
xx chr [1:2] "a" "TRUE"
Files Plots Packages Help Viewer
New Folder Delete Rename More
Home
Name Size Modified
Custom Office Templates
FeedbackHub
R
SEM - 8


RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
Go to file/function
Addins
Project: (None)
Untitled1*
Source on Save
Run
Source
44 m <- matrix(nrow = 2, ncol = 2)
45 m
46 dim(m)
47 m <- matrix(c(1:3))
48 class(m)
49 typeof(m)
50 FOURS <- matrix(
51 c(4, 4, 4, 4),
52 nrow = 2,
53 ncol = 2)
54 FOURS
47:20 (Top Level) R Script
Console ~/
> m <- matrix(nrow = 2, ncol = 2)
> m
[,1] [,2]
[1,] NA NA
[2,] NA NA
> dim(m)
[1] 2 2
> m <- matrix(c(1:3))
> m
[,1]
[1,] 1
[2,] 2
[3,] 3
> class(m)
[1] "matrix"
> typeof(m)
[1] "integer"
> FOURS <- matrix(
+ c(4, 4, 4, 4),
+ nrow = 2,
+ ncol = 2)
> FOURS
[,1] [,2]
[1,] 4 4
[2,] 4 4
Environment History Connections
Import Dataset
List
Global Environment
Data
FOURS num [1:2, 1:2] 4 4 4 4
m int [1:3, 1] 1 2 3
Values
Files Plots Packages Help Viewer
New Folder Delete Rename More
Home
Name Size Modified
Custom Office Templates
FeedbackHub
R
SEM - 8


RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
Go to file/function
Addins
Project: (None)
Untitled1*
Source on Save
Run
Source
55 m <- matrix(1:6, nrow = 2, ncol = 3)
56 m
57 m <- 1:10
58 dim(m) <- c (2, 5)
59 m
60 x <- 1:3
61 y <- 10:12
62 cbind(x, y)
63 rbind(x, y)
64 mdat <- matrix(c(1, 2, 3, 11, 12, 13),
65 nrow = 2,
71:1 (Top Level) R Script
Console ~/
> m <- matrix(1:6, nrow = 2, ncol = 3)
> m
[,1] [,2] [,3]
[1,] 1 3 5
[2,] 2 4 6
> m <- 1:10
> m
[1] 1 2 3 4 5 6 7 8 9 10
> dim(m) <- c (2, 5)
Environment History Connections
Import Dataset
List
Global Environment
mdat num [1:2, 1:3] 1 11 2 12 3 13
Values
series int [1:10] 1 2 3 4 5 6 7 8 9 10
x int [1:3] 1 2 3
Files Plots Packages Help Viewer
New Folder Delete Rename More
Home
Name Size Modified
Custom Office Templates
FeedbackHub
R
SEM - 8


RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
Go to file/function
Addins
Project: (None)
Untitled1*
Source on Save
Run
Source
59 m
60 x <- 1:3
61 y <- 10:12
62 cbind(x, y)
63 rbind(x, y)
64 mdat <- matrix(c(1, 2, 3, 11, 12, 13),
65 nrow = 2,
66 ncol = 3,
67 byrow = TRUE)
68 mdat
69 mdat[2,3]
71:1 (Top Level) R Script
Console ~/
> m
[,1] [,2] [,3] [,4] [,5]
[1,] 1 3 5 7 9
[2,] 2 4 6 8 10
> x <- 1:3
> x
[1] 1 2 3
> y <- 10:12
> y
[1] 10 11 12
Environment History Connections
Import Dataset
List
Global Environment
mdat num [1:2, 1:3] 1 11 2 12 3 13
Values
series int [1:10] 1 2 3 4 5 6 7 8 9 10
x int [1:3] 1 2 3
Files Plots Packages Help Viewer
New Folder Delete Rename More
Home
Name Size Modified
Custom Office Templates
FeedbackHub
R
SEM - 8


Untitled1*
Source on Save
Run
Source
61 y <- 10:12
62 cbind(x, y)
63 rbind(x, y)
64 mdat <- matrix(c(1, 2, 3, 11, 12, 13),
65 nrow = 2,
66 ncol = 3,
67 byrow = TRUE)
68 mdat
69 mdat[2,3]
70 #List
71
71:1 (Top Level) R Script
Console ~/
> cbind(x, y)
x y
[1,] 1 10
[2,] 2 11
[3,] 3 12
> rbind(x, y)
[,1] [,2] [,3]
x 1 2 3
y 10 11 12
Environment History Connections
Import Dataset
List
Global Environment
mdat num [1:2, 1:3] 1 11 2 12 3 13
Values
series int [1:10] 1 2 3 4 5 6 7 8 9 10
x int [1:3] 1 2 3
Files Plots Packages Help Viewer
New Folder Delete Rename More
Home
Name Size Modified
Custom Office Templates
FeedbackHub
R
SEM - 8

Conclusion:

We have executed Vectors, Matrices, Data frames in R studio.

Experiment No: 4

Title: Programming constructs in R – loops, conditional executives.

Date of Performance:

Date of Submission:

Grade:

Signature:

Aim: Programming constructs in R – loops, conditional executives.

Theory:

Loops:

Sr.No.	Loop Type & Description
1	repeat loop Executes a sequence of statements multiple times and abbreviates the code that manages the loop variable.
2	while loop Repeats a statement or group of statements while a given condition is true. It tests the condition before executing the loop body.
3	for loop Like a while statement, except that it tests the condition at the end of the loop body.

Conditional Executives:

Sr.No.	Statement & Description
1	if statement An **if** statement consists of a Boolean expression followed by one or more statements.
2	if...else statement An **if** statement can be followed by an optional **else** statement, which executes when the Boolean expression is false.
3	switch statement A **switch** statement allows a variable to be tested for equality against a list of values.

Program:

```
 >#Functions.
 >fahrenheit_to_celsius <- function(temp_F){
 + temp_C <- (temp_F-32)*5/9
 + return(temp_C)
 + }
 > fahrenheit_to_celsius(212)
 [1] 100
 >
 >
 > celsius_to_kelvin <- function(temp_C){
 + temp_K <- temp_C + 273.15
 + return(temp_K)
 + }
 > celsius_to_kelvin(0)
 [1] 273.15
 >
 >celsius_to_kelvin(fahrenheit_to_celsius(32.0))
 [1] 273.15
 >
 >highlight <- function(content, wrapper)
 +{
 + answer <- c(wrapper, content, wrapper)
 + return(answer)
 + }
 >best_practice <- c("Write", "programs", "for", "people", "not", "co
mputers")
 >asterisk <- “***”
 >highlight(best_practice, asterisk)
 [1] "***"          "Write"     "programs"  "for"

 [5] "people"    "not"   "computers" "***"
 >
```

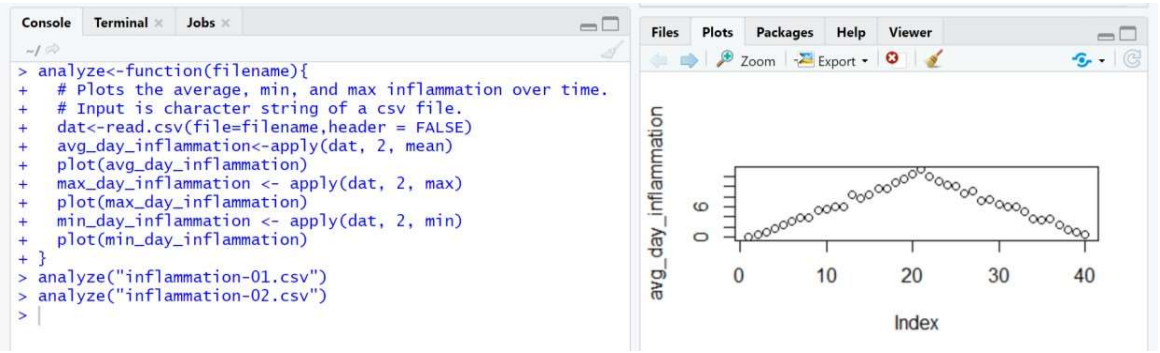
```
>edges <- function(v)
+{
+ first <- v[1]
+ last <- v[length(v)]
+ answer <- c(first,last)
+ return(answer)
+ }
>dry_principle<- c("Don't","repeat","yourself","or","others")
>edges(dry_principle)
[1] "Don't" "others"
>
>
>input_1 <- 20
>mySum <- function(input_1, input_2 = 10)
+{
+ output <- input_1+input_2
+ return(output)
+ }
>mySum(input_1 = 1, 3)
[1] 4
>mySum(3)
[1] 13
>
>
>center <- function(data, midpoint)
+{
+ new_data <- (data-mean(data)) + midpoint
+ return(new_data)
+ }
> z <- c(0,0,0,0)
> z
[1] 0 0 0 0
> center(z, 3)
```

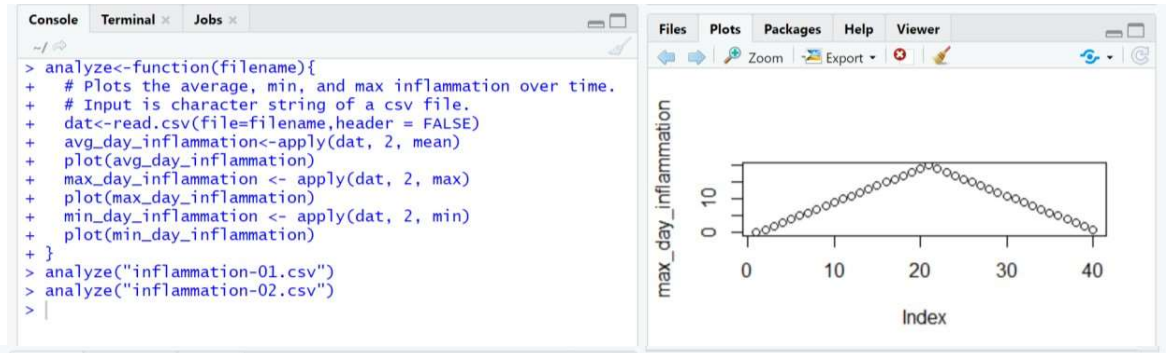
```
[1] 3 3 3 3
>
>
> setwd("C:/Users/Akshata/Documents")
> dat <-  read.csv(file = "inflammation- 01.csv", header = FALSE)
> centered <- center(dat[,4], 0)
> head(centered)
[1] 1.25 -0.75 1.25 -1.75 1.25 0.25
> min(dat[,4])
[1] 0
> mean(dat[,4])
[1] 1.75
> max(dat[,4])
[1] 3
> min(centered)
[1] -1.75
> mean(centered)
[1] 0
> max(centered)
[1] 1.25
> sd(dat[,4])
[1] 1.067628
> sd(centered)
[1] 1.067628
> sd(dat[,4]) - sd(centered)
[1] 0
> all.equal(sd(dat[,4]), sd(centered))
[1] TRUE
>
>
> center <- function(data, midpoint = 0) {
+ # return a new vector containing the original data centered around
the
```

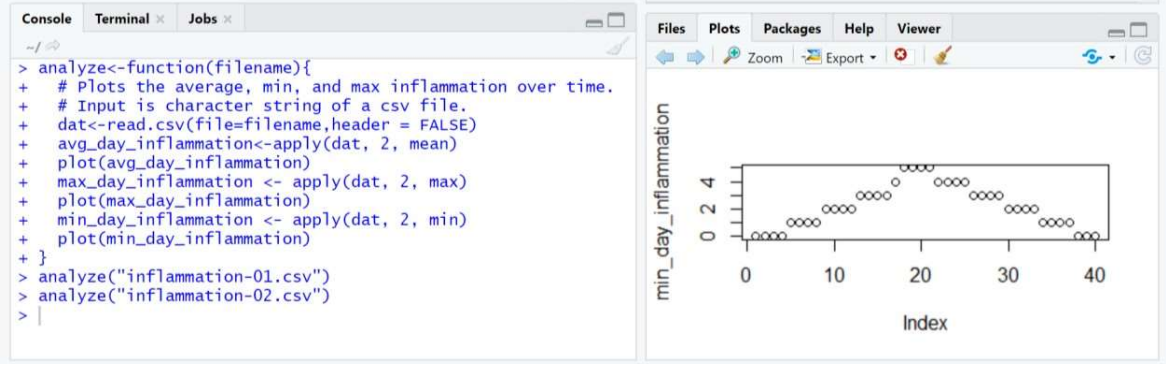
```
+ # midpoint (0 by default).
+ # Example: center(c(1, 2, 3), 0) => c(-1, 0, 1)
+ new_data <- (data - mean(data)) + midpoint
+ return(new_data)
+ }
> test_data <- c(0, 0, 0, 0)
> center(test_data, 3)
[1] 3 3 3 3
> more_data <- 5 + test_data
> more_data [1] 5 5 5 5
> center(more_data) [1] 0 0 0 0
>
> display <- function(a = 1, b = 2, c = 3) {
+ result <- c(a, b, c)
+ names(result) <- c("a", "b", "c") # This names each element of the
vector
+ return(result)
+ }
> display() a b c
1 2 3
> display(55) a b  c
55 2 3
> display(55,66) a b  c
55 66 3
> display(55,66,77) a b  c
55 66 77
> display(c=77) a b  c
 1 2 77
> ##Loops

> a <- 1:10
> b <- 1:10
> res <- numeric(length = length(a))
```

```
> for (i in seq_along(a)) {
+ res[i] <- a[i] + b[i]
+ }
> res
 [1] 2 4 6 8 10 12 14 16 18 20
> res2 <- a + b
> all.equal(res, res2)
[1] TRUE
>
>
> a <- 1:10
> b <- 1:5
> a + b
 [1] 2 4 6 8 10 7 9 11 13 15
> a <- 1:10
> b <- 5
> a * b
 [1] 5 10 15 20 25 30 35 40 45 50
>
>
> a <- 1:10
> b <- 1:7
> a + b
 [1] 2 4 6 8 10 12 14 9 11 13
```

Output:







Conclusion:

We have seen how loops and conditional statements in R works.

Experiment No: 5

Title: Tables with labels in R- Multiple response variables, Excel function translations.

Date of Performance:

Date of Submission:

Grade:

Signature:

Aim: Tables with labels in R- Multiple response variables, Excel function translations.

Theory, Program, Outputs:

Tables in R

```
The downloaded binary packages are in
        C:\Users\SAYALI\AppData\Local\Temp\RtmpAZuPqf\downloaded_packages
> library(expss)
>
> data(mtcars)
> mtcars = apply_labels(mtcars,
+                       mpg = "Miles/(US) gallon",
+                       cyl = "Number of cylinders",
+                       disp = "Displacement (cu.in.)",
+                       hp = "Gross horsepower",
+                       drat = "Rear axle ratio",
+                       wt = "Weight (1000 lbs)",
+                       qsec = "1/4 mile time",
+                       vs = "Engine",
+                       vs = c("V-engine" = 0,
+                              "Straight engine" = 1),
+                       am = "Transmission",
+                       am = c("Automatic" = 0,
+                              "Manual"=1),
+                       gear = "Number of forward gears",
+                       carb = "Number of carburetors"
+ )
> cro(mtcars$am, mtcars$vs)

 |             |             |   Engine |                |
 |             |             | V-engine | Straight engine |
 | ----------- | ----------- | -------- | --------------- |
 | Transmission |    Automatic |       12 |               7 |
 |             |       Manual |        6 |               7 |
 |             | #Total cases |       18 |              14 |
> cro_cpct(mtcars$cyl, list(total(), mtcars$am, mtcars$vs))

 |                   |             | #Total | Transmission |        |   Engine |                |
 |                   |             |        |    Automatic | Manual | V-engine | Straight engine |
 | ----------------- | ----------- | ------ | ------------ | ------ | -------- | --------------- |
 | Number of cylinders |           4 |   34.4 |         15.8 |   61.5 |      5.6 |            71.4 |
 |                   |           6 |   21.9 |         21.1 |   23.1 |     16.7 |            28.6 |
 |                   |           8 |   43.8 |         63.2 |   15.4 |     77.8 |                 |
 |                   | #Total cases |   32.0 |         19.0 |   13.0 |     18.0 |            14.0 |
> mtcars %>% calc_cro_cpct(cyl, list(total(), am, vs))

 |                   |             | #Total | Transmission |        |   Engine |                |
 |                   |             |        |    Automatic | Manual | V-engine | Straight engine |
 | ----------------- | ----------- | ------ | ------------ | ------ | -------- | --------------- |
 | Number of cylinders |           4 |   34.4 |         15.8 |   61.5 |      5.6 |            71.4 |
 |                   |           6 |   21.9 |         21.1 |   23.1 |     16.7 |            28.6 |
 |                   |           8 |   43.8 |         63.2 |   15.4 |     77.8 |                 |
 |                   | #Total cases |   32.0 |         19.0 |   13.0 |     18.0 |            14.0 |
> mtcars %>% calc_cro_cpct(cyl, list(total(), am %nest% vs))

 |                   |             | #Total | Transmission |                |        |                |
 |                   |             |        |    Automatic |                | Manual |                |
 |                   |             |        |       Engine |                | Engine |                |
 |                   |             |        |    V-engine | Straight engine | V-engine | Straight engine |
 | ----------------- | ----------- | ------ | ------------ | --------------- | -------- | --------------- |
 | Number of cylinders |           4 |   34.4 |              |            42.9 |     16.7 |             100 |
 |                   |           6 |   21.9 |              |            57.1 |     50.0 |                 |
```

```
> mtcars %>%
+     tab_cells(cyl) %>%
+     tab_cols(total(), am) %>%
+     tab_stat_cpct() %>%
+     tab_pivot()
```

		#Total	Transmission	
			Automatic	Manual
Number of cylinders	4	34.4	15.8	61.5
	6	21.9	21.1	23.1
	8	43.8	63.2	15.4
#Total cases		32.0	19.0	13.0

```
> mtcars %>%
+     tab_cells(mpg, disp, hp, wt, qsec) %>%
+     tab_cols(total(), am) %>%
+     tab_stat_mean_sd_n() %>%
+     tab_last_sig_means(subtable_marks = "both") %>%
+     tab_pivot() %>%
+     set_caption("Table with summary statistics and significance marks.")
```

Table with summary statistics and significance marks.

		#Total	Transmission	
			Automatic	Manual
			A	B
Miles/(US) gallon	Mean	20.1	17.1 < B	24.4 > A
	Std. dev.	6.0	3.8	6.2
	Unw. valid N	32.0	19.0	13.0
Displacement (cu.in.)	Mean	230.7	290.4 > B	143.5 < A
	Std. dev.	123.9	110.2	87.2
	Unw. valid N	32.0	19.0	13.0
Gross horsepower	Mean	146.7	160.3	126.8
	Std. dev.	68.6	53.9	84.1
	Unw. valid N	32.0	19.0	13.0
Weight (1000 lbs)	Mean	3.2	3.8 > B	2.4 < A
	Std. dev.	1.0	0.8	0.6
	Unw. valid N	32.0	19.0	13.0
1/4 mile time	Mean	17.8	18.2	17.4
	Std. dev.	1.8	1.8	1.8
	Unw. valid N	32.0	19.0	13.0

```
> mtcars %>%
+     tab_cells(mpg, disp, hp, wt, qsec) %>%
+     tab_cols(total(label = "#Total| |"), am) %>%
+     tab_stat_fun(Mean = w_mean, "Std. dev." = w_sd, "Valid N" = w_n, method = list) %>%
+     tab_pivot()
```

	#Total			Transmission				
				Automatic			Manual	
	Mean	Std. dev.	Valid N	Mean	Std. dev.	Valid N	Mean	Std. dev.
Miles/(US) gallon	20.1	6.0	32	17.1	3.8	19	24.4	6.2
Displacement (cu.in.)	230.7	123.9	32	290.4	110.2	19	143.5	87.2
Gross horsepower	146.7	68.6	32	160.3	53.9	19	126.8	84.1
Weight (1000 lbs)	3.2	1.0	32	3.8	0.8	19	2.4	0.6
1/4 mile time	17.8	1.8	32	18.2	1.8	19	17.4	1.8

```
> mtcars %>%
+     tab_cols(total(), vs) %>%
+     tab_cells(mpg) %>%
+     tab_stat_mean() %>%
+     tab_stat_valid_n() %>%
+     tab_cells(am) %>%
+     tab_stat_cpct(total_row_position = "none", label = "col %") %>%
+     tab_stat_rpct(total_row_position = "none", label = "row %") %>%
+     tab_stat_tpct(total_row_position = "none", label = "table %") %>%
+     tab_pivot(stat_position = "inside_rows")
```

			#Total	Engine	
				V-engine	Straight engine
Miles/(US) gallon	Mean		20.1	16.6	24.6
	Valid N		32.0	18.0	14.0
Transmission	Automatic	col %	59.4	66.7	50.0
		row %	100.0	63.2	36.8
		table %	59.4	37.5	21.9
	Manual	col %	40.6	33.3	50.0
		row %	100.0	46.2	53.8
		table %	40.6	18.8	21.9

```
> mtcars %>%
+     tab_cells(cyl) %>%
+     tab_cols(total(), vs) %>%
+     tab_rows(am) %>%
+     tab_stat_cpct(total_row_position = "above",
+                   total_label = c("number of cases", "row %"),
+                   total_statistic = c("u_cases", "u_rpct")) %>%
+     tab_pivot()
```

				#Total	Engine	
					V-engine	Straight engine
Transmission	Automatic	Number of cylinders	#number of cases	19.0	12.0	7.0
			#row %	100.0	63.2	36.8
			4	15.8		42.9
			6	21.1		57.1
			8	63.2	100.0	
	Manual	Number of cylinders	#number of cases	13.0	6.0	7.0
			#row %	100.0	46.2	53.8
			4	61.5	16.7	100.0
			6	23.1	50.0	
			8	15.4	33.3	

```
> mtcars %>%
+     tab_cells(sheet(mpg, disp, hp, wt, qsec)) %>%
+     tab_cols(total(label = "#Total| |"), am) %>%
+     tab_stat_fun_df(
+         function(x){
+             frm = reformulate(".", response = as.name(names(x)[1]))
+             model = lm(frm, data = x)
+             sheet('Coef.' = coef(model),
+                   confint(model)
+             )
+         }
+     ) %>%
```

Multiple response variables:

```
|                          | #Total |         |         | Transmission |         |         |        |         |         |
|                          |        |         |         |    Automatic |         |         | Manual |         |         |
|                          |  Coef. | 2.5 %   | 97.5 %  |        Coef. | 2.5 %   | 97.5 %  |  Coef. | 2.5 %   | 97.5 %  |
| ------------------------ | ------ | ------- | ------- | ------------ | ------- | ------- | ------ | ------- | ------- |
|              (Intercept) |   27.3 |     9.6 |    45.1 |         21.8 |    -1.9 |    45.5 |   13.3 |   -21.9 |    48.4 |
| `Displacement (cu.in.)`  |    0.0 |     0.0 |     0.0 |          0.0 |     0.0 |     0.0 |    0.0 |    -0.1 |     0.1 |
|     `Gross horsepower`   |    0.0 |    -0.1 |     0.0 |          0.0 |    -0.1 |     0.0 |    0.0 |     0.0 |     0.1 |
|      `Weight (1000 lbs)` |   -4.6 |    -7.2 |    -2.0 |         -2.3 |    -5.0 |     0.4 |   -7.7 |   -12.5 |    -2.9 |
|          `1/4 mile time` |    0.5 |    -0.4 |     1.5 |          0.4 |    -0.7 |     1.6 |    1.6 |    -0.2 |     3.4 |
>
> data(product_test)
> w = product_test
> w = w %>%
+     do_if(cell == 1, {
+         recode(a1_1 %to% a1_6, other ~ copy) %into% (h1_1 %to% h1_6)
+         recode(b1_1 %to% b1_6, other ~ copy) %into% (p1_1 %to% p1_6)
+         recode(a22, other ~ copy) %into% h22
+         recode(b22, other ~ copy) %into% p22
+         c1r = c1
+     }) %>%
+     do_if(cell == 2, {
+         recode(a1_1 %to% a1_6, other ~ copy) %into% (p1_1 %to% p1_6)
+         recode(b1_1 %to% b1_6, other ~ copy) %into% (h1_1 %to% h1_6)
+         recode(a22, other ~ copy) %into% p22
+         recode(b22, other ~ copy) %into% h22
+         recode(c1, 1 ~ 2, 2 ~ 1, other ~ copy) %into% c1r
+     }) %>%
+     compute({
+         # recode age by groups
+         age_cat = recode(s2a, lo %thru% 25 ~ 1, lo %thru% hi ~ 2)
+         # count number of likes
+         # codes 2 and 99 are ignored.
+         h_likes = count_row_if(1 | 3 %thru% 98, h1_1 %to% h1_6)
+         p_likes = count_row_if(1 | 3 %thru% 98, p1_1 %to% p1_6)
+     })
>
> codeframe_likes = num_lab("
+     1 Liked everything
+     2 Disliked everything
+     3 Chocolate
+     4 Appearance
+     5 Taste
+     6 Stuffing
+     7 Nuts
+     8 Consistency
+     98 Other
+     99 Hard to answer
+ ")
> overall_liking_scale = num_lab("
+     1 Extremely poor
+     2 Very poor
+     3 Quite poor
+     4 Neither good, nor poor
+     5 Quite good
+     6 Very good
+     7 Excellent
+ ")
> w = apply_labels(w,
+                  c1r = "Preferences",
+                  c1r = num_lab("
+         1 VSX123
+         2 SDF456
+         3 Hard to say
+     "),
+
+                  age_cat = "Age",
+                  age_cat = c("18 - 25" = 1, "26 - 35" = 2),
+
+                  h1_1 = "Likes. VSX123",
+                  p1_1 = "Likes. SDF456",
+                  h1_1 = codeframe_likes,
+                  p1_1 = codeframe_likes,
+
+                  h_likes = "Number of likes. VSX123",
+                  p_likes = "Number of likes. SDF456",
+
+                  h22 = "Overall quality. VSX123",
+                  p22 = "Overall quality. SDF456",
+                  h22 = overall_liking_scale,
+                  p22 = overall_liking_scale
+ )
> w %>% tab_cols(total(), age_cat) %>%
+     tab_cells(c1r) %>%
+     tab_mis_val(3) %>%
+     tab_stat_cases() %>%
+     tab_last_sig_cases() %>%
+     tab_pivot()

|             |                       | #Total |      Age |          |
|             |                       |        |  18 - 25 | 26 - 35  |
| ----------- | --------------------- | ------ | -------- | -------- |
| Preferences |                VSX123 |   94.0 |     46.0 |     48.0 |
|             |                SDF456 |   50.0 |     22.0 |     28.0 |
|             |          Hard to say |        |          |          |
|             | #Chi-squared p-value |  <0.05 |  (warn.) |          |
|             |          #Total cases |  144.0 |     68.0 |     76.0 |
> banner = w %>% tab_cols(total(), age_cat, c1r)
> tab_cpct_sig = . %>% tab_stat_cpct() %>%
+     tab_last_sig_cpct(sig_labels = paste0("<b>",LETTERS, "</b>"))
> tab_means_sig = . %>% tab_stat_mean_sd_n(labels = c("<b><u>Mean</u></b>", "sd", "N")) %>%
+     tab_last_sig_means(
+         sig_labels = paste0("<b>",LETTERS, "</b>"),
+         keep = "means")
>
> banner %>%
+     tab_cells(c1r) %>%
+     tab_cpct_sig() %>%
+     tab_pivot()
```

```
+     tab_pivot()
```

		#Total	Age		Preferences		
			18 - 25	26 - 35	VSX123	SDF456	Hard to say
			A	B	A	B	C
Preferences	VSX123	62.7	65.7	60.0	100.0		
	SDF456	33.3	31.4	35.0		100.0	
	Hard to say	4.0	2.9	5.0			100.0
	#Total cases	150	70	80	94	50	6

```
> banner %>%
+     tab_cells(h22) %>%
+     tab_means_sig() %>%
+     tab_cpct_sig() %>%
+     tab_cells(p22) %>%
+     tab_means_sig() %>%
+     tab_cpct_sig() %>%
+     tab_pivot()
```

		#Total	Age		Preferences
			18 - 25	26 - 35	VSX123
			A	B	A
Overall quality. VSX123	<u>Mean</u>	5.5	5.4	5.6	5.3
	Extremely poor				
	Very poor				
	Quite poor	2.0	2.9	1.2	3.2
	Neither good, nor poor	10.7	11.4	10.0	14.9 B
	Quite good	39.3	45.7	33.8	40.4
	Very good	33.3	24.3	41.2 A	30.9
	Excellent	14.7	15.7	13.8	10.6
	#Total cases	150	70	80	94
Overall quality. SDF456	<u>Mean</u>	5.4	5.3	5.4	5.4
	Extremely poor				
	Very poor	0.7		1.2	1.1
	Quite poor	2.7	4.3	1.2	2.1
	Neither good, nor poor	16.7	20.0	13.8	18.1
	Quite good	31.3	27.1	35.0	28.7
	Very good	35.3	35.7	35.0	35.1
	Excellent	13.3	12.9	13.8	14.9
	#Total cases	150	70	80	94

SDF456	Hard to say
B	C
5.8 A	5.5
2.0	16.7
38.0	33.3
38.0	33.3
22.0	16.7
50	6
5.3	5.7
50	6

```
> banner %>%
+     tab_cells(h_likes) %>%
+     tab_means_sig() %>%
+     tab_cells(mrset(h1_1 %to% h1_6)) %>%
+     tab_cpct_sig() %>%
+     tab_cells(p_likes) %>%
+     tab_means_sig() %>%
+     tab_cells(mrset(p1_1 %to% p1_6)) %>%
+     tab_cpct_sig() %>%
+     tab_pivot()
```

		#Total	Age		Preferences	
			18 - 25	26 - 35	VSX123	SDF456
			A	B	A	B
Number of likes. VSX123	<u>Mean</u>	2.0	2.0	2.1	1.9	2.2
Likes. VSX123	Liked everything					
	Disliked everything	3.3	1.4	5.0	4.3	2.0
	Chocolate	34.0	38.6	30.0	35.1	32.0
	Appearance	29.3	21.4	36.2 A	25.5	38.0
	Taste	32.0	38.6	26.2	23.4	48.0 A
	Stuffing	27.3	20.0	33.8	28.7	26.0
	Nuts	66.7	72.9	61.3	69.1	60.0
	Consistency	12.0	4.3	18.8 A	8.5	14.0
	Other					
	Hard to answer					
	#Total cases	150	70	80	94	50
Number of likes. SDF456	<u>Mean</u>	2.0	2.0	2.1	2.0	2.0
Likes. SDF456	Liked everything					
	Disliked everything	1.3	1.4	1.2	2.1	
	Chocolate	32.0	27.1	36.2	29.8	34.0
	Appearance	32.0	35.7	28.7	34.0	30.0
	Taste	39.3	42.9	36.2	36.2	44.0
	Stuffing	27.3	24.3	30.0	31.9	20.0
	Nuts	61.3	60.0	62.5	58.5	68.0
	Consistency	10.0	5.7	13.8	11.7	6.0
	Other	0.7		1.2	1.1	
	Hard to answer					
	#Total cases	150	70	80	94	50

Hard to say
C
2.3
33.3
16.7
33.3
16.7
83.3
50.0 A B
6

```
83.3
50.0 <b>A</b> <b>B</b>


   6
                  2.0


                 50.0
                 16.7
                 50.0
                 16.7
                 50.0
                 16.7


                    6
```

```
> w %>%
+     tab_cols(total(label = "#Total| |"), c1r) %>%
+     tab_cells(list(unvr(mrset(h1_1 %to% h1_6)))) %>%
+     tab_stat_cpct(label = var_lab(h1_1)) %>%
+     tab_cells(list(unvr(mrset(p1_1 %to% p1_6)))) %>%
+     tab_stat_cpct(label = var_lab(p1_1)) %>%
+     tab_pivot(stat_position = "inside_columns")
```

	#Total		Preferences					
			VSX123		SDF456			
	Likes. VSX123	Likes. SDF456	Likes. VSX123	Likes. SDF456	Likes. VSX123	Likes. SDF456	Hard to Say Likes. VSX123	Likes. SDF456
Liked everything								
Disliked everything	3.3	1.3	4.3	2.1	2			
Chocolate	34.0	32.0	35.1	29.8	32	34	33.3	50.0
Appearance	29.3	32.0	25.5	34.0	38	30	16.7	16.7
Taste	32.0	39.3	23.4	36.2	48	44	33.3	50.0
Stuffing	27.3	27.3	28.7	31.9	26	20	16.7	16.7
Nuts	66.7	61.3	69.1	58.5	60	68	83.3	50.0
Consistency	12.0	10.0	8.5	11.7	14	6	50.0	16.7
Other		0.7		1.1				
Hard to answer								
#Total cases	150.0	150.0	94.0	94.0	50	50	6.0	6.0

```
> library(expss)
> w = text_to_columns("
+         a  b  c
+         2 15 50
+         1 70 80
+         3 30 40
+         2 30 40
+ ")
> w$d = ifelse(w$b>60, 1, 0)
> w = compute(w, {
+     d = ifelse(b>60, 1, 0)
+     e = 42
+     abc_sum = sum_row(a, b, c)
+     abc_mean = mean_row(a, b, c)
+ })
> count_if(1, w)
[1] 2
> calculate(w, count_if(1, a, b, c))
[1] 1
> w$d = count_row_if(gt(1), w)
> w = compute(w, {
+     d = count_row_if(gt(1), a, b, c)
+ })
> count_col_if(le(1), w$a)
[1] 1
> sum(w, na.rm = TRUE)
[1] 1026
> w$d = mean_row(w)
> w = compute(w, {
+     d = mean_row(a, b, c)
+ })
> w$d = mean_row(w)
> w = compute(w, {
+     d = mean_row(a, b, c)
+ })
> sum_col(w$a)
[1] 8
> sum_if(gt(40), w)
[1] 831.6667
> calculate(w, sum_if(gt(40), a, b, c))
[1] 200
> w$d = sum_row_if(lt(40), w)
>
> w = compute(w, {
+     d = sum_row_if(lt(40), a, b, c)
+ })
> mean_col_if(lt(3), w$a, data = w$b)
[1] 38.33333
> calculate(w, mean_col_if(lt(3), a, data = sheet(b, c)))
       b        c
38.33333 56.66667
> dict = text_to_columns("
+     x  y
+     1  apples
+     2  oranges
+     3  peaches
+ ")
> w$d = vlookup(w$a, dict, 2)
> w$d = vlookup(w$a, dict, "y")
```

```
> w = apply_labels(w,
+                  a = num_lab("
+                        1 apples
+                        2 oranges
+                        3 peaches
+                    ")
+ )
> fre(w$a)
```

w$a	Count	Valid percent	Percent	Responses, %	Cumulative responses, %
apples	1	25	25	25	25
oranges	2	50	50	50	75
peaches	1	25	25	25	100
#Total	4	100	100	100	
<NA>	0		0		

```
> fre(w$a)
```

w$a	Count	Valid percent	Percent	Responses, %	Cumulative responses, %
apples	1	25	25	25	25
oranges	2	50	50	50	75
peaches	1	25	25	25	100
#Total	4	100	100	100	
<NA>	0		0		

```
> cro_mean(sheet(w$b, w$c), w$a)
```

	w$a		
	apples	oranges	peaches
w$b	70	22.5	30
w$c	80	45.0	40

```
>
```

Conclusion:

We have executed tables, Multiple response variables, Excel function translations using R studio

Experiment No: 6

Title: Working with large datasets, with dplyrand & data table.

Date of Performance:

Date of Submission:

Grade:

Signature:

Aim: Working with large datasets, with dplyrand & data table.

Theory, Program and Outputs:

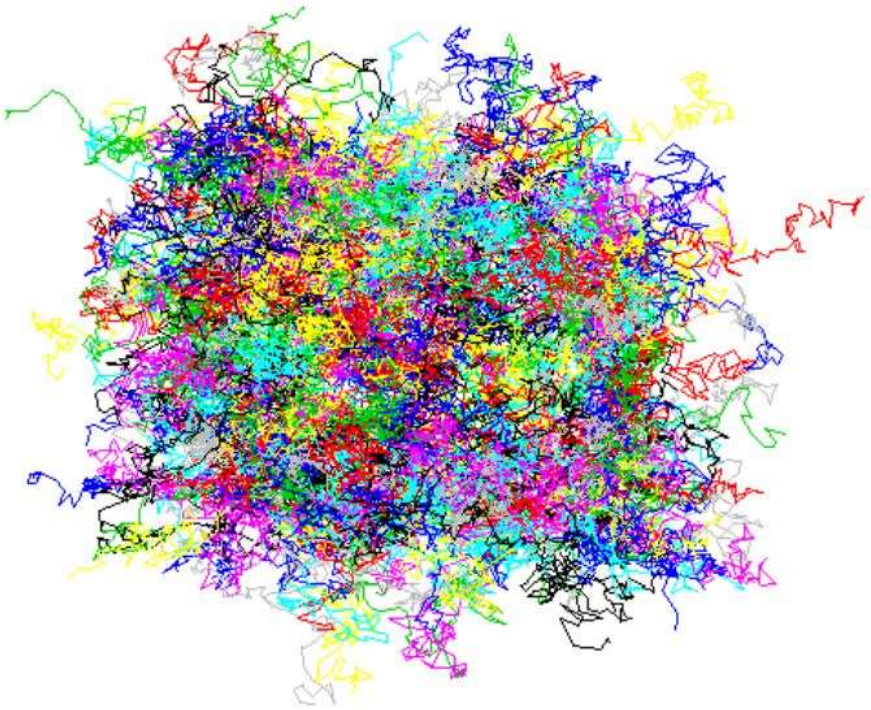
In order to illustrate, let us generate our "large" telematic dataset. Assume that we have 10,000 drivers, each of them drives about 200 times, and each time, we have, say, 80 locations. That mean around 160 million observations. It is "large", but not huge.

```
> rm(list=ls())
> N_id=10000
> N_tr=200
> T_tr=80
```

```
> set.seed(1)
> N=rpois(N_id,N_tr)
> N_traj=rpois(sum(N),T_tr)
```

```
> origin_lat=runif(N_id,-5,5)
> origin_lon=runif(N_id,-5,5)
```

```
> lat=lon=Traj_Id=rep(NA,sum(N_traj))
> Pers_Id=rep(NA,length(N_traj))
> s=1
> for(i in 1:N_id){Pers_Id[s:(s+N[i])]=i;s=s+N[i]}
> s=1
> for(i in 1:length(N_traj)){lat[s:
(s+N_traj[i])]=origin_lat[Pers_Id[i]]+
+  cumsum(c(0,rnorm(N_traj[i]-1,0,sd=.2)));
+  lon[s:(s+N_traj[i])]=origin_lon[Pers_Id[i]]+
+  cumsum(c(0,rnorm(N_traj[i]-1,0,sd=.2)));
+  s=s+N_traj[i]}
```



```
> Pers_Id=rep(NA,sum(N_traj))
> N0=cumsum(c(0,N))
> s=1
> for(i in 1:N_id){Pers_Id[s:
(s+sum(N_traj[(N0[i]+1):N0[i+1]]))]=i;s=s+sum(N_traj[(
N0[i]+1):N0[i+1]])}
> s=1
> for(i in 1:sum(N)){Traj_Id[s:
(s+N_traj[i])]=i;s=s+N_traj[i]}
```

```
> df=data.frame(Pers_Id,Traj_Id,lat,lon)
```

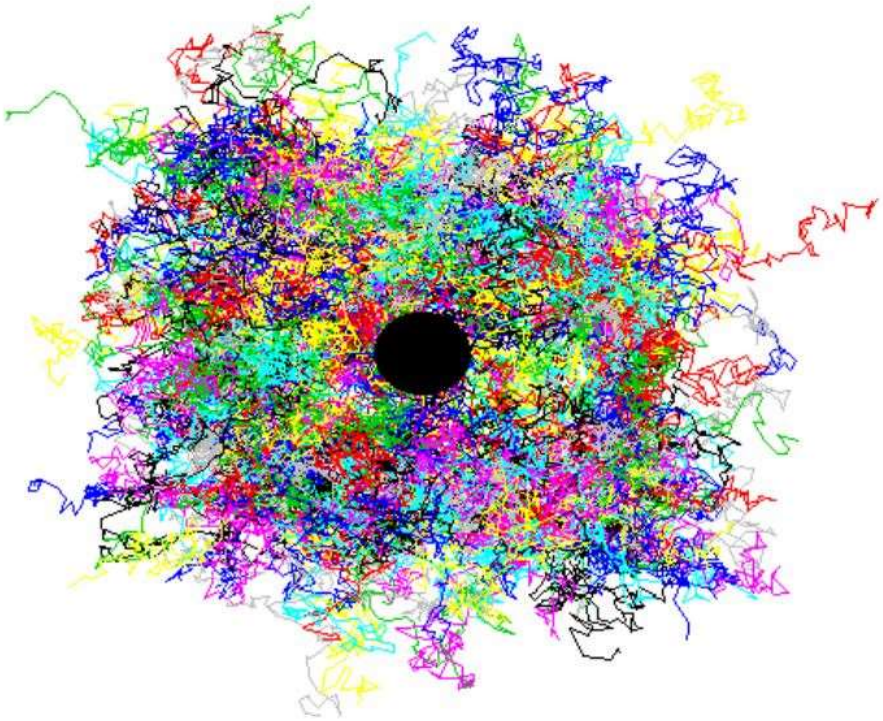
```
> library(dplyr)
> ldf=data_frame(Pers_Id,Traj_Id,lat,lon)
```

```
> library(data.table)
> dt=data.table(Pers_Id,Traj_Id,lat,lon)
```

```
> object.size(df)
3839908904 bytes
```

```
> rm(list=ls())
> library(data.table)
> system.time( load("dt_json_2.RData") )
       user      system     elapsed
      21.53        1.33       27.71
```

```
> dropbox_ldf <-
"https://dl.dropboxusercontent.com/s/l7rolinwojn37e2/l
df_json_2.RData"
> dropbox_df  <-
"https://dl.dropboxusercontent.com/s/wzr19v9pyl7ah0j/d
f_json_2.RData"
> dropbox_dt  <-
"https://dl.dropboxusercontent.com/s/nlwi1mmsxr5f23k/d
t_json_2.RData"

> source_https <- function(loc,...){
+  curl=getCurlHandle()
+  curlSetOpt(cookiejar="cookies.txt",
+ useragent="Mozilla/5.0", followlocation=TRUE)
+  tmp <- getURLContent(loc,
.opts=list(ssl.verifypeer=FALSE),curl=curl)
+  fch <- source(tmp)
+ fch
+ }
```



```
> system.time(load("df_json_2.RData"))
       user      system     elapsed
      21.76        1.39       29.59
```

```
> system.time(n <- nrow(df))
       user      system     elapsed
          0           0           0
> system.time(df$first<-c(1,df$Traj_Id[2:n]!=
df$Traj_Id[1:(n-1)]))
       user      system     elapsed
       3.12        1.23        4.37
> system.time(df$last<-c(df$Traj_Id[2:n]!=
df$Traj_Id[1:(n-1)],1))
       user      system     elapsed
       2.90        6.23       31.58
```

```
> object.size(df)
6399847720 bytes
```

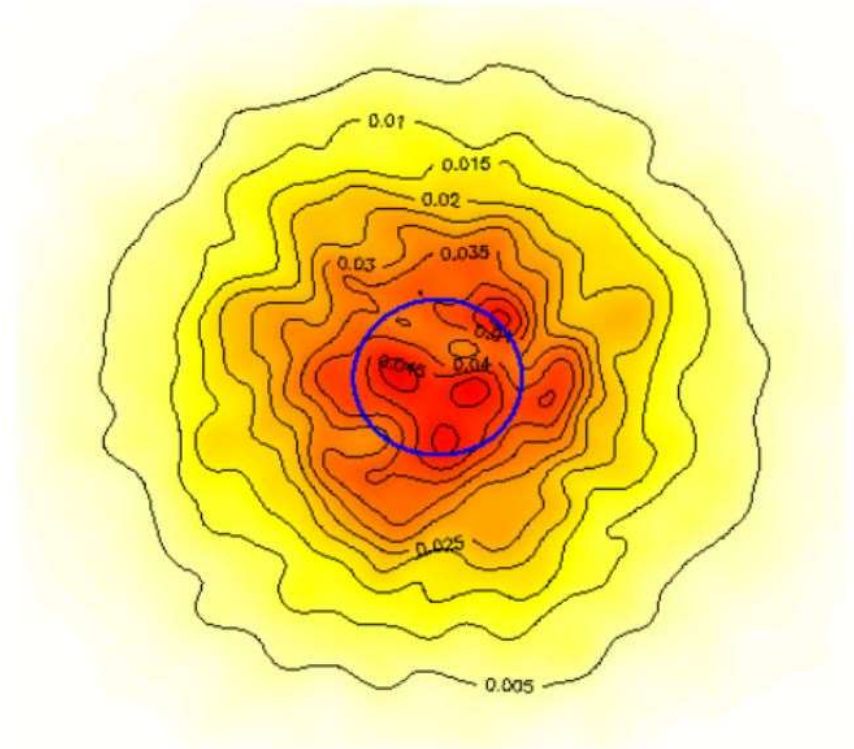
```
> lat_0=0
> lon_0=0
> system.time(df$test <-(lat_0-df$lat)^2+(lon_0-
df$lon)^2) <=1)&(df$last==1)
Error: cannot allocate vector of size 1.2 Gb/pre>
```

```
> df$first <- NULL
> df$last <- NULL
> object.size(df)
3839908904 bytes
> system.time(df$test<-(lat_0-df$lat)^2+(lon_0-
df$lon)^2) <=1)&(c(df$Traj_Id[2:n]!=
df$Traj_Id[1:(n-1)],1)==1))
      user    system    elapsed
      9.39     11.10      41.17
```

```
> system.time(list_Traj <-
unique(df$Traj_Id[df$test==TRUE]))
       user      system     elapsed
       0.72        0.25        0.98
```

```
> system.time(base <- df[(df$Traj_Id %in% list_Traj)&
(c(1,df$Traj_Id[2:n]!=df$Traj_Id[1:(n-
1)])==1),c("lat","lon")])
       user      system     elapsed
       11.7         2.7        14.4
> head(base)
             lat      lon
556   -0.9597891 2.469243
2866  -0.9597891 2.469243
4321  -0.9597891 2.469243
5677  -0.9597891 2.469243
9403  -0.9597891 2.469243
10432 -0.9597891 2.469243
> nrow(base0
[1]  63453
```

```
> X <- base[,c("lon","lat")]
> library(KernSmooth)
> kde2d <- bkde2D(X,
bandwidth=c(bw.ucv(X[,1]),bw.ucv(X[,2])),gridsize =
c(251L, 251L))
> image(x=kde2d$x1, y=kde2d$x2,z=kde2d$fhat,col=
rev(heat.colors(100)))
> contour(x=kde2d$x1, y=kde2d$x2,z=kde2d$fhat,
add=TRUE)
```




```
> rm(list=ls())
> library(data.table)
> system.time( load("dt_json_2.RData") )
       user      system     elapsed
      21.53        1.33       27.71
```

```
> system.time( setkey(dt,Traj_Id) )
       user      system     elapsed
       0.38        0.09        0.47
```

```
> system.time(depart <-dt[!duplicated(Traj_Id)])
       user      system     elapsed
       2.48        0.65        3.14
```

```
> system.time( depart <- dt[J(unique(Traj_Id)), mult =
"first"])
       user      system     elapsed
       2.64        0.75        3.39
> system.time( arrivee <- dt[J(unique(Traj_Id)), mult
= "last"] )
       user      system     elapsed
       2.81        0.51        3.33
```

```
> lat_0=0
> lon_0=0
> system.time( arrivee[,dist:=(lat-lat_0)^2+(lon-
lon_0)^2] )
       user      system     elapsed
       0.03        0.08        1.60
```

```
> system.time( fin <- subset(arrivee,dist <= 1) )
       user      system     elapsed
       0.04        0.00        0.78
```

```
> system.time( fin[,Pers_Id:=NULL] )
       user      system     elapsed
       0.00        0.00        0.07
> system.time( fin[,lat:=NULL] )
       user      system     elapsed
        0.0         0.0         0.2
> system.time( fin[,lon:=NULL] )
       user      system     elapsed
          0           0           0
```

```
> system.time( setkey(fin, Traj_Id) )
       user      system     elapsed
       0.00        0.00        0.42
```

```
> system.time( base <<- merge(fin,depart,all.x=TRUE) )
       user      system     elapsed
       0.02        0.00        0.17
```

```
> system.time( base <<- merge(fin,depart,all.x=TRUE) )
       user      system     elapsed
       0.02        0.00        0.17
>
> head(base)
   Traj_Id       dist Pers_Id        lat      lon
1:       8 0.41251163       1 -0.9597891 2.469243
2:      36 0.34545373       1 -0.9597891 2.469243
3:      54 0.24766671       1 -0.9597891 2.469243
4:      71 0.00210023       1 -0.9597891 2.469243
5:     117 0.00755432       1 -0.9597891 2.469243
6:     130 0.82806342       1 -0.9597891 2.469243
```

```
> rm(list=ls())
> library(dplyr)
> library(data.table)
> system.time( load("ldf_json_2.RData") )
       user      system     elapsed
      27.53        1.77       69.84
```

```
> system.time( ldepart <<- ldf %>% group_by(Traj_Id)
%>%
+ summarise(first_lat=head(lat,1),
first_lon=head(lon,1)) )
       user      system     elapsed
      60.82        1.46       80.54

> system.time( larrive <<- ldf %>% group_by(Traj_Id)
%>%
+ summarise(last_lat=tail(lat,1),last_lon=tail(lon,1))
)
       user      system     elapsed
      60.81        0.31       62.15

> lat_0=0
> lon_0=0
> system.time( system.time( larrive <<-
mutate(larrive,dist=(last_lat-lat_0)^2+(last_lon-
lon_0)^2) ))
```

```
> system.time( lfin <<- filter(larrive,dist<=1) )
       user      system     elapsed
       0.05        0.00        0.04
```

```
> system.time( lbase <- left_join(lfin,ldepart) )
Joining by: "Traj_Id"
       user      system     elapsed
       0.53        0.05        0.66
```

```
> head(lbase)
Source: local data frame [63,453 x 6]

  Traj_Id    last_lat     last_lon        dist
first_lat first_lon
1       8  0.41374639  0.491248980 0.412511638
-0.9597891  2.469243
2      36  0.58774352  0.003360806 0.345453735
-0.9597891  2.469243
3      54  0.34479069 -0.358867800 0.247666719
-0.9597891  2.469243
4      71 -0.04341135  0.014686416 0.002100236
-0.9597891  2.469243
5     117 -0.05103682 -0.070353141 0.007554322
-0.9597891  2.469243
6     130 -0.56196768 -0.715720445 0.828063425
-0.9597891  2.469243
```

Conclusion:

The longest part was to extract those first and last observations. So far, it looks like data.table is just perfect to deal with those “large” datasets.

Experiment No: 7

Title: EDA with R.

Date of Performance:

Date of Submission:

Grade:

Signature:

Aim: EDA with R – Range, Summary, Mean, Variance, Median, Standard daviation, histogram, boxplot, scatterplot.

Theory, Program:

```
install.packages('soilDB',dep=TRUE)
library(aqp)
library(soilDB)
data("loafercreek")

n<-c("A",
    "BAt",
    "Bt1",
    "Bt2",
    "Cr",
    "R")
p<-c("A",
    "BA|AB",
    "Bt|Bw",
    "Bt3|Bt4|2B|C",
    "Cr",
    "R")
loafercreek$genhz<-generalize.hz(loafercreek$hzname,n,p)
h<-horizons(loafercreek)
table(h$genhz,h$hzname)
View(h)
vars <- c("genhz", "clay", "total_frags_pct", "phfield", "effclass")
summary(h[, vars])
# just for factors
levels(h$genhz)
# for characters and factors
sort(unique(h$hzname))
h$hzname <- ifelse(h$hzname == "BT", "Bt", h$hzname)
# or
```

```
h$hzname[h$hzname == "BT"] <- "Bt"
# or as a last resort we could manually edit the spreadsheet in R
edit(h)
# gopheridge rules
n <- c('A', 'Bt1', 'Bt2', 'Bt3','Cr','R')
p <- c('^A|BA$', 'Bt1|Bw','Bt$|Bt2', 'Bt3|CBt$|BCt','Cr','R')
# first remove missing values and create a new vector
clay <- na.exclude(h$clay)
mean(clay)

# or use the additional na.rm argument
mean(h$clay, na.rm = TRUE)
median(clay)
sort(table(round(h$clay)), decreasing = TRUE)[1] # sort and select the 1st
value, which will be the mode
table(h$genhz)
table(h$genhz, h$texcl)
# append the table with row and column sums
addmargins(table(h$genhz, h$texcl))

# calculate the proportions relative to the rows, margin = 1 calculates for
rows, margin = 2 calculates for columns, margin = NULL calculates for
total observations
round(prop.table(table(h$genhz, h$texture_class), margin = 1) * 100)

knitr::kable(addmargins(table(h$genhz, h$texcl)))
knitr::kable(round(prop.table(table(h$genhz, h$texture_class), margin =
1) * 100))
aggregate(clay ~ genhz, data = h, mean)
aggregate(clay ~ genhz, data = h, median)
aggregate(clay ~ genhz, data = h, summary)
var(h$clay, na.rm=TRUE)
sd(h$clay, na.rm = TRUE)
```

```
sqrt(var(clay))
cv <- sd(clay) / mean(clay) * 100
cv
quantile(clay)
quantile(clay, c(0.05, 0.5, 0.95))
range(clay)
diff(range(clay))
max(clay) - min(clay)
IQR(clay)
diff(quantile(clay, c(0.25, 0.75)))
h$hzdepm <- (h$hzdepb + h$hzdept) / 2 # Compute the middle horizon
depth
vars <- c("hzdepm", "clay", "sand", "total_frags_pct", "phfield")
round(cor(h[, vars], use = "complete.obs"), 2)
library(ggplot2)
# bar plot
ggplot(h, aes(x = texcl)) +
      geom_bar()
ggplot(h, aes(x = clay)) +
      geom_histogram(bins = nclass.Sturges(h$clay))
ggplot(h, aes(x = clay)) +
      geom_density()
ggplot(h, (aes(x = genhz, y = clay))) +
      geom_boxplot()
# QQ Plot for Clay
ggplot(h, aes(sample = clay)) +
      geom_qq() +
      geom_qq_line()
# QQ Plot for Frags
ggplot(h, aes(sample = total_frags_pct)) +
      geom_qq() +
      geom_qq_line()
# scatter plot
```

```
ggplot(h, aes(x = clay, y = hzdepm)) +
      geom_point() +
      ylim(100, 0)
```

Outputs:

```
> library(soilDB)
> data("loafercreek")
> n<-c("A",
+       "BAt",
+       "Bt1",
+       "Bt2",
+       "Cr",
+       "R")
> p<-c("A",
+       "BA|AB",
+       "Bt|Bw",
+       "Bt3|Bt4|2B|C",
+       "Cr",
+       "R")
> loafercreek$genhz<-generalize.hz(loafercreek$hzname,n,p)
> h<-horizons(loafercreek)
> table(h$genhz,h$hzname)

           2BCt 2Bt1 2Bt2 2Bt3 2Bt4 2Bt5 2CB 2CBt 2Cr 2Crt 2R  A A1 A2 AB ABt Ad Ap
  A           0    0    0    0    0    0   0    0   0    0  0 97  7  7  0   0  1  1
  BAt         0    0    0    0    0    0   0    0   0    0  0  0  0  0  1   0  0  0
  Bt1         0    0    0    0    0    0   0    0   0    0  0  0  0  0  0   2  0  0
  Bt2         1    2    7    8    6    1   1    1   0    0  0  0  0  0  0   0  0  0
  Cr          0    0    0    0    0    0   0    0   4    2  0  0  0  0  0   0  0  0
  R           0    0    0    0    0    0   0    0   0    0  5  0  0  0  0   0  0  0
  not-used    0    0    0    0    0    0   0    0   0    0  0  0  0  0  0   0  0  0

           B BA BAt BC BCt Bt Bt1 Bt2 Bt3 Bt4 Bw Bw1 Bw2 Bw3  C CBt Cd Cr Cr/R Crt
  A        0  0   0  0   0  0   0   0   0   0  0   0   0   0  0   0  0  0    0   0
  BAt      0 31   8  0   0  0   0   0   0   0  0   0   0   0  0   0  0  0    0   0
  Bt1      0  0   0  0   0  8  94  89   0   0 10   2   2   1  0   0  0  0    0   0
  Bt2      0  0   0  5  16  0   0   0  47   8  0   0   0   0  6   6  1  0    0   0
  Cr       0  0   0  0   0  0   0   0   0   0  0   0   0   0  0   0  0 49    0  20
  R        0  0   0  0   0  0   0   0   0   0  0   0   0   0  0   0  0  0    1   0
  not-used 1  0   0  0   0  0   0   0   0   0  0   0   0   0  0   0  0  0    0   0

           H1 Oi  R Rt
  A         0  0  0  0
  BAt       0  0  0  0
  Bt1       0  0  0  0
  Bt2       0  0  0  0
  Cr        0  0  0  0
  R         0  0 41  1
  not-used  1 24  0  0
```

```
> view(h)
> vars <- c("genhz", "clay", "total_frags_pct", "phfield", "effclass")
> summary(h[, vars])
      genhz          clay       total_frags_pct    phfield            effclass
 A       :113   Min.   :10.00   Min.   : 0.00   Min.   :4.90   very slight:  0
 BAt     : 40   1st Qu.:18.00   1st Qu.: 0.00   1st Qu.:6.00   slight     :  0
 Bt1     :208   Median :22.00   Median : 5.00   Median :6.30   strong     :  0
 Bt2     :116   Mean   :23.67   Mean   :14.18   Mean   :6.18   violent    :  0
 Cr      : 75   3rd Qu.:28.00   3rd Qu.:20.00   3rd Qu.:6.50   none       : 86
 R       : 48   Max.   :60.00   Max.   :95.00   Max.   :7.00   NA's       :540
 not-used: 26   NA's   :173                     NA's   :381
> # just for factors
> levels(h$genhz)
[1] "A"        "BAt"      "Bt1"      "Bt2"      "Cr"       "R"        "not-used"
> # for characters and factors
> sort(unique(h$hzname))
 [1] "2BCt" "2Bt1" "2Bt2" "2Bt3" "2Bt4" "2Bt5" "2CB"  "2CBt" "2Cr"  "2Crt" "2R"
[12] "A"    "A1"   "A2"   "AB"   "ABt"  "Ad"   "Ap"   "B"    "BA"   "BAt"  "BC"
[23] "BCt"  "Bt"   "Bt1"  "Bt2"  "Bt3"  "Bt4"  "Bw"   "Bw1"  "Bw2"  "Bw3"  "C"
[34] "CBt"  "Cd"   "Cr"   "Cr/R" "Crt"  "H1"   "Oi"   "R"    "Rt"
> h$hzname <- ifelse(h$hzname == "BT", "Bt", h$hzname)
> h$hzname[h$hzname == "BT"] <- "Bt"
> edit(h)
     peiid  phiid hzname    genhz hzdept hzdepb clay silt sand fragvoltot texture
557  64505 299222     Oi not-used      0      1   NA   NA   NA          0     SPM
558  64505 299223      A        A      1      5 16.7 46.8 36.5          0    GR-L
559  64505 299224    BAt      BAt      5     15 17.0 48.0 35.0          0    GR-L
560  64505 299225    Bt1      Bt1     15     30 19.6 47.4 33.0          0       L
561  64505 299226    Bt2      Bt1     30     58 23.6 44.7 31.7          0    GR-L
562  64505 299227    Bt3      Bt2     58     79 21.5 50.5 28.0          0    GR-L
563  64505 299228    Crt       Cr     79    107   NA   NA   NA          0      BR
564  64505 299229      R        R    107    108   NA   NA   NA          0      BR
538 115595 579662      A        A      0      5 15.0   NA   NA          0       L
539 115595 579661     BA      BAt      5     10 20.0   NA   NA          0       L
540 115595 579660    BAt      BAt     10     28 25.0   NA   NA          0       L
541 115595 579659    Bt1      Bt1     28     51 29.0   NA   NA          0      CL
542 115595 579658    Bt2      Bt1     51     74 35.0   NA   NA          0      CL
543 115595 579657     Cr       Cr     74     81   NA   NA   NA          0      BR
    texcl lieutex phfield effclass labsampnum    rupresblkdry        stickiness
```

557	<NA>	spm	NA	<NA>	<NA>		<NA>	<NA>
558	l	<NA>	6.1	<NA>	01P1582S		hard	nonsticky
559	l	<NA>	6.2	<NA>	01P1583S		hard	nonsticky
560	l	<NA>	6.2	<NA>	01P1584S		hard	nonsticky
561	l	<NA>	6.3	<NA>	01P1585S		hard	slightly sticky
562	l	<NA>	6.2	<NA>	01P1586S		hard	slightly sticky
563	<NA>	br	NA	<NA>	<NA>		<NA>	<NA>
564	<NA>	br	NA	<NA>	<NA>		<NA>	<NA>
538	l	<NA>	5.5	none	01P1566	moderately	hard	nonsticky
539	l	<NA>	4.9	none	01P1567	moderately	hard	nonsticky
540	l	<NA>	5.5	none	01P1568	slightly	hard	slightly sticky
541	cl	<NA>	6.0	none	01P1569	moderately	hard	slightly sticky
542	cl	<NA>	6.0	none	01P1570	moderately	hard	moderately sticky
543	<NA>	br	NA	<NA>	<NA>		<NA>	<NA>

	plasticity	ksatpedon	texture_class	d_r	d_g	d_b	d_hue
557	<NA>	NA	spm	NA	NA	NA	<NA>
558	slightly plastic	NA	l	0.6032814	0.4497634	0.3499322	5YR
559	slightly plastic	NA	l	0.6032814	0.4497634	0.3499322	5YR
560	slightly plastic	NA	l	0.6516906	0.4320947	0.2762389	5YR
561	moderately plastic	NA	l	0.6516906	0.4320947	0.2762389	5YR
562	moderately plastic	NA	l	0.6516906	0.4320947	0.2762389	5YR
563	<NA>	NA	br	NA	NA	NA	<NA>
564	<NA>	NA	br	NA	NA	NA	<NA>
538	slightly plastic	NA	l	0.5801129	0.4622333	0.3125959	10YR
539	slightly plastic	NA	l	0.6157417	0.4523969	0.2177761	10YR
540	slightly plastic	NA	l	0.6373120	0.4408493	0.2484700	7.5YR
541	moderately plastic	NA	cl	0.6373120	0.4408493	0.2484700	7.5YR
542	moderately plastic	NA	cl	0.6373120	0.4408493	0.2484700	7.5YR
543	<NA>	NA	br	NA	NA	NA	<NA>

	d_value	d_chroma	d_sigma	m_r	m_g	m_b	m_hue	m_value	m_chroma
557	NA	NA	NA	NA	NA	NA	<NA>	NA	NA
558	5	4	NA	0.4789920	0.3545817	0.2821670	5YR	4	3
559	5	4	NA	0.4789920	0.3545817	0.2821670	5YR	4	3
560	5	6	NA	0.3940324	0.2499977	0.1668267	5YR	3	4
561	5	6	NA	0.5409717	0.3305930	0.1852933	5YR	4	6
562	5	6	NA	0.5409717	0.3305930	0.1852933	5YR	4	6
563	NA	NA	NA	NA	NA	NA	<NA>	NA	NA
564	NA	NA	NA	NA	NA	NA	<NA>	NA	NA
538	5	4	NA	0.4713544	0.3589827	0.2681530	7.5YR	4	3
539	5	6	NA	0.3827022	0.2567391	0.1506936	7.5YR	3	4
540	5	6	NA	0.5409717	0.3305930	0.1852933	5YR	4	6
541	5	6	NA	0.5409717	0.3305930	0.1852933	5YR	4	6
542	5	6	NA	0.5409717	0.3305930	0.1852933	5YR	4	6
543	NA	NA	NA	NA	NA	NA	<NA>	NA	NA

	m_sigma	moist_soil_color	dry_soil_color	fine_gravel	gravel	cobbles	stones
557	NA	<NA>	<NA>	0	0	0	0
558	NA	#7A5A48	#9A7359	20	20	0	0
559	NA	#7A5A48	#9A7359	25	25	0	0
560	NA	#64402B	#A66E46	0	10	0	0
561	NA	#8A542F	#A66E46	0	20	5	0
562	NA	#8A542F	#A66E46	0	20	0	0
563	NA	<NA>	<NA>	0	0	0	0
564	NA	<NA>	<NA>	0	0	0	0
538	NA	#785C44	#947650	0	0	0	0
539	NA	#624126	#9D7338	0	5	0	0
540	NA	#8A542F	#A3703F	0	5	0	0
541	NA	#8A542F	#A3703F	0	5	0	0
542	NA	#8A542F	#A3703F	0	0	0	0
543	NA	<NA>	<NA>	0	0	0	0

	boulders	channers	flagstones	parafine_gravel	paragravel	paracobbles	parastones
557	0	0	0	0	0	0	0
558	0	0	0	0	0	0	0
559	0	0	0	0	0	0	0
560	0	0	0	0	0	0	0
561	0	0	0	0	0	0	0
562	0	0	0	0	0	0	0
563	0	0	0	0	0	0	0
564	0	0	0	0	0	0	0
538	0	0	0	0	0	0	0
539	0	0	0	0	0	0	0
540	0	0	0	0	0	0	0
541	0	0	0	0	0	0	0
542	0	0	0	0	0	0	0
543	0	0	0	0	0	0	0

	paraboulders	parachanners	paraflagstones	unspecified	total_frags_pct_nopf
557	0	0	0	0	0
558	0	0	0	0	20
559	0	0	0	0	25
560	0	0	0	0	10
561	0	0	0	0	25
562	0	0	0	0	20
563	0	0	0	0	0
564	0	0	0	0	0
538	0	0	0	0	0
539	0	0	0	0	5
540	0	0	0	0	5
541	0	0	0	0	5
542	0	0	0	0	0
543	0	0	0	0	0

	total_frags_pct	art_fgr	art_gr	art_cb	art_st	art_by	art_ch	art_fl
557	0	0	0	0	0	0	0	0
558	20	0	0	0	0	0	0	0
559	25	0	0	0	0	0	0	0
560	10	0	0	0	0	0	0	0
561	25	0	0	0	0	0	0	0
562	20	0	0	0	0	0	0	0
563	0	0	0	0	0	0	0	0
564	0	0	0	0	0	0	0	0
538	0	0	0	0	0	0	0	0
539	5	0	0	0	0	0	0	0
540	5	0	0	0	0	0	0	0
541	5	0	0	0	0	0	0	0
542	0	0	0	0	0	0	0	0
543	0	0	0	0	0	0	0	0

	art_unspecified	total_art_pct	huartvol_cohesive	huartvol_penetrable
557	0	0	0	0
558	0	0	0	0
559	0	0	0	0
560	0	0	0	0
561	0	0	0	0
562	0	0	0	0
563	0	0	0	0
564	0	0	0	0
538	0	0	0	0
539	0	0	0	0
540	0	0	0	0
541	0	0	0	0
542	0	0	0	0
543	0	0	0	0

```
    huartvol_innocuous huartvol_persistent soil_color hzID
557                  0                   0       <NA>    1
558                  0                   0    #7A5A48    2
559                  0                   0    #7A5A48    3
560                  0                   0    #64402B    4
561                  0                   0    #8A542F    5
562                  0                   0    #8A542F    6
563                  0                   0       <NA>    7
564                  0                   0       <NA>    8
538                  0                   0    #785C44    9
539                  0                   0    #624126   10
540                  0                   0    #8A542F   11
541                  0                   0    #8A542F   12
542                  0                   0    #8A542F   13
543                  0                   0       <NA>   14
 [ reached 'max' / getOption("max.print") -- omitted 612 rows ]
> # gopheridge rules
> n <- c('A', 'Bt1', 'Bt2', 'Bt3','Cr','R')
> p <- c('^A|BA$', 'Bt1|Bw','Bt$|Bt2', 'Bt3|CBt$|BCt','Cr','R')
> # first remove missing values and create a new vector
> clay <- na.exclude(h$clay)
> mean(clay)
[1] 23.6713
> # or use the additional na.rm argument
> mean(h$clay, na.rm = TRUE)
[1] 23.6713
> median(clay)
[1] 22
> sort(table(round(h$clay)), decreasing = TRUE)[1] # sort and select the 1st value, whi
ch will be the mode
25
41
> table(h$genhz)

       A      BAt      Bt1      Bt2       Cr        R not-used
     113       40      208      116       75       48       26
> table(h$genhz, h$texcl)

          cos   s  fs vfs lcos  ls lfs lvfs cosl  sl fsl vfsl   l sil  si scl  cl
  A         0   0   0   0    0   0   0    0    0   6   0    0  78  27   0   0   0
  BAt       0   0   0   0    0   0   0    0    0   1   0    0  31   4   0   0   2
  Bt1       0   0   0   0    0   0   0    0    0   1   0    0 125  20   0   4  46
  Bt2       0   0   0   0    0   0   0    0    0   0   0    0  28   5   0   5  52
  Cr        0   0   0   0    0   0   0    0    0   0   0    0   0   0   0   0   0
  R         0   0   0   0    0   0   0    0    0   0   0    0   0   0   0   0   0
  not-used  0   0   0   0    0   0   0    0    0   0   0    0   0   0   0   0   1

          sicl  sc sic   c
  A          0   0   0   0
  BAt        1   0   0   0
  Bt1        5   0   1   2
  Bt2        3   0   1  16
  Cr         0   0   0   1
  R          0   0   0   0
  not-used   0   0   0   0
> # append the table with row and column sums
> addmargins(table(h$genhz, h$texcl))
```

```
           cos   s  fs vfs lcos  ls lfs lvfs cosl  sl fsl vfsl   l sil  si scl  cl
A            0   0   0   0    0   0   0    0    0   6   0    0  78  27   0   0   0
BAt          0   0   0   0    0   0   0    0    0   1   0    0  31   4   0   0   2
Bt1          0   0   0   0    0   0   0    0    0   1   0    0 125  20   0   4  46
Bt2          0   0   0   0    0   0   0    0    0   0   0    0  28   5   0   5  52
Cr           0   0   0   0    0   0   0    0    0   0   0    0   0   0   0   0   0
R            0   0   0   0    0   0   0    0    0   0   0    0   0   0   0   0   0
not-used     0   0   0   0    0   0   0    0    0   0   0    0   0   0   0   0   1
Sum          0   0   0   0    0   0   0    0    0   8   0    0 262  56   0   9 101

           sicl  sc sic   c Sum
A             0   0   0   0 111
BAt           1   0   0   0  39
Bt1           5   0   1   2 204
Bt2           3   0   1  16 110
Cr            0   0   0   1   1
R             0   0   0   0   0
not-used      0   0   0   0   1
Sum           9   0   2  19 466
> # calculate the proportions relative to the rows, margin = 1 calculates for rows, mar
gin = 2 calculates for columns, margin = NULL calculates for total observations
> round(prop.table(table(h$genhz, h$texture_class), margin = 1) * 100)

             br   c  cb  cl  gr   l  pg scl sic sicl sil  sl spm
A             0   0   0   0   0  70   0   0   0    0  24   5   0
BAt           0   0   0   5   0  79   0   0   0    3  10   3   0
Bt1           0   1   0  23   0  61   0   2   0    2  10   0   0
Bt2           0  14   1  46   2  25   1   4   1    3   4   0   0
Cr           98   2   0   0   0   0   0   0   0    0   0   0   0
R           100   0   0   0   0   0   0   0   0    0   0   0   0
not-used      0   0   0   4   0   0   0   0   0    0   0   0  96
> knitr::kable(addmargins(table(h$genhz, h$texcl)))
```

	cos	s	fs	vfs	lcos	ls	lfs	lvfs	cosl	sl	fsl	vfsl	l	sil	si	scl	cl	sicl	sc	sic	c	Sum
A	0	0	0	0	0	0	0	0	0	6	0	0	78	27	0	0	0	0	0	0	0	111
BAt	0	0	0	0	0	0	0	0	0	1	0	0	31	4	0	0	2	1	0	0	0	39
Bt1	0	0	0	0	0	0	0	0	0	1	0	0	125	20	0	4	46	5	0	1	2	204
Bt2	0	0	0	0	0	0	0	0	0	0	0	0	28	5	0	5	52	3	0	1	16	110
Cr	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
not-used	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	1
Sum	0	0	0	0	0	0	0	0	0	8	0	0	262	56	0	9	101	9	0	2	19	466

```
> knitr::kable(round(prop.table(table(h$genhz, h$texture_class), margin = 1) * 100))
```

	br	c	cb	cl	gr	l	pg	scl	sic	sicl	sil	sl	spm
A	0	0	0	0	0	70	0	0	0	0	24	5	0
BAt	0	0	0	5	0	79	0	0	0	3	10	3	0
Bt1	0	1	0	23	0	61	0	2	0	2	10	0	0
Bt2	0	14	1	46	2	25	1	4	1	3	4	0	0
Cr	98	2	0	0	0	0	0	0	0	0	0	0	0
R	100	0	0	0	0	0	0	0	0	0	0	0	0
not-used	0	0	0	4	0	0	0	0	0	0	0	0	96

```
> aggregate(clay ~ genhz, data = h, mean)
  genhz     clay
1     A 16.23113
2   BAt 19.53889
3   Bt1 24.14221
4   Bt2 31.35045
5    Cr 15.00000
```

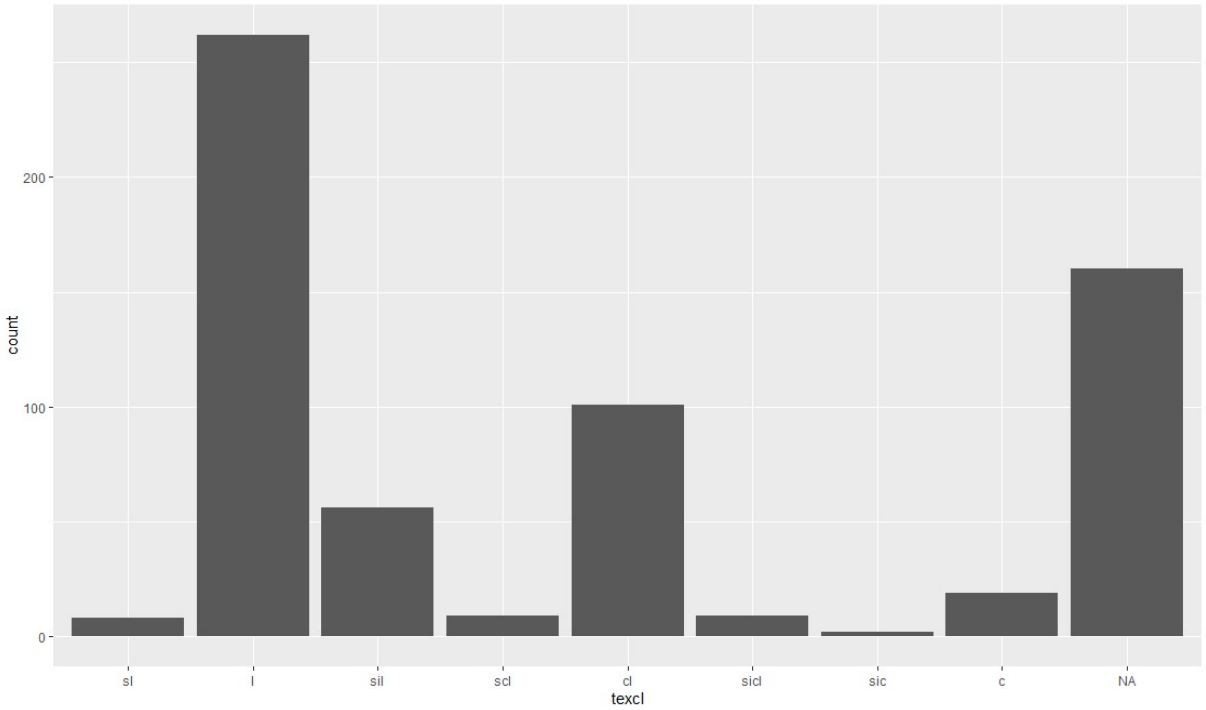
```
> aggregate(clay ~ genhz, data = h, summary)
  genhz clay.Min. clay.1st Qu. clay.Median clay.Mean clay.3rd Qu. clay.Max.
1     A  10.00000     14.00000    16.00000  16.23113     18.00000  25.00000
2   BAt  14.00000     17.00000    19.50000  19.53889     20.00000  28.00000
3   Bt1  12.00000     20.00000    24.00000  24.14221     28.00000  51.40000
4   Bt2  10.00000     26.00000    30.00000  31.35045     35.00000  60.00000
5    Cr  15.00000     15.00000    15.00000  15.00000     15.00000  15.00000
> var(h$clay, na.rm=TRUE)
[1] 64.89187
> sd(h$clay, na.rm = TRUE)
[1] 8.055549
> sqrt(var(clay))
[1] 8.055549
> cv <- sd(clay) / mean(clay) * 100
> cv
[1] 34.03087
> quantile(clay)
  0%  25%  50%  75% 100%
  10   18   22   28   60
> quantile(clay, c(0.05, 0.5, 0.95))
  5%  50%  95%
13.0 22.0 38.4
> range(clay)
[1] 10 60
> diff(range(clay))
[1] 50
> max(clay) - min(clay)
[1] 50
> IQR(clay)
[1] 10
> diff(quantile(clay, c(0.25, 0.75)))
75%
 10
> h$hzdepm <- (h$hzdepb + h$hzdept) / 2 # Compute the middle horizon depth
> vars <- c("hzdepm", "clay", "sand", "total_frags_pct", "phfield")
> round(cor(h[, vars], use = "complete.obs"), 2)
               hzdepm  clay  sand total_frags_pct phfield
hzdepm           1.00  0.59 -0.08           0.50   -0.03
clay             0.59  1.00 -0.17           0.28    0.13
sand            -0.08 -0.17  1.00          -0.05    0.12
total_frags_pct  0.50  0.28 -0.05           1.00   -0.16
phfield         -0.03  0.13  0.12          -0.16    1.00
> library(ggplot2)
> # bar plot
> ggplot(h, aes(x = texcl)) +
+         geom_bar()
> ggplot(h, aes(x = clay)) +
+         geom_histogram(bins = nclass.Sturges(h$clay))
Warning message:
Removed 173 rows containing non-finite values (stat_bin).
> ggplot(h, aes(x = clay)) +
+         geom_histogram(bins = nclass.Sturges(h$clay))
```

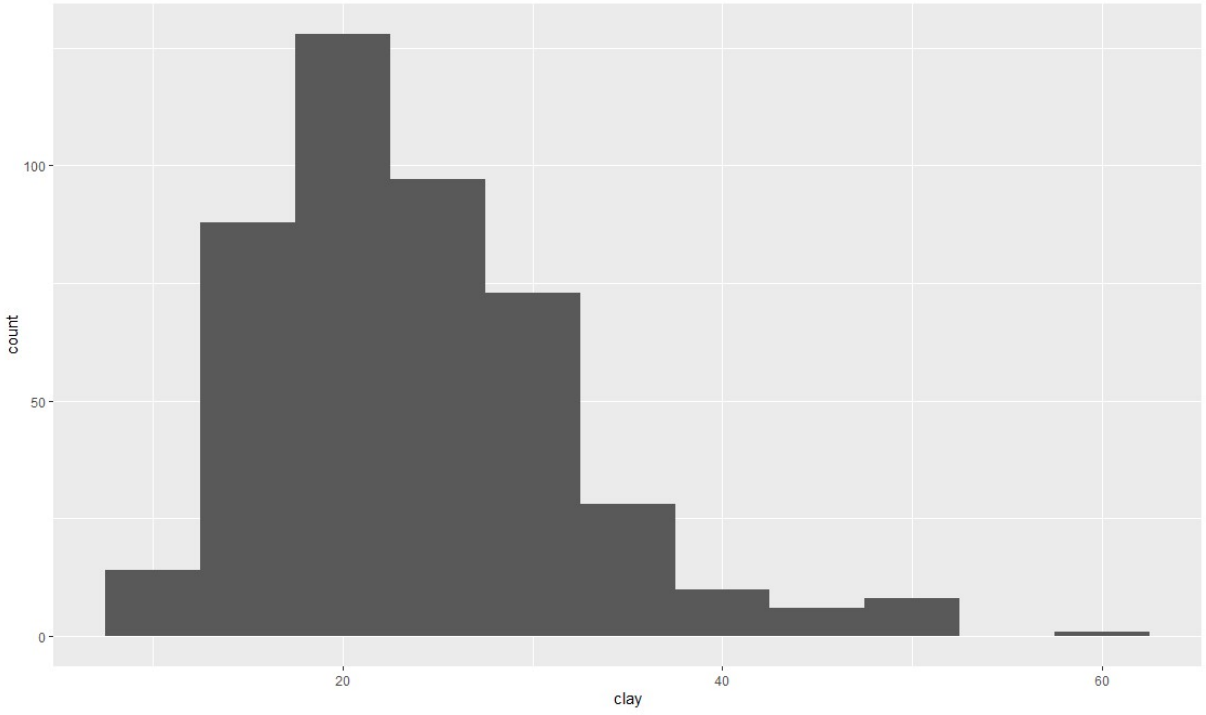
```
> ggplot(h, aes(x = clay)) +
+         geom_density()
Warning message:
Removed 173 rows containing non-finite values (stat_density).
> ggplot(h, (aes(x = genhz, y = clay))) +
+         geom_boxplot()
Warning message:
Removed 173 rows containing non-finite values (stat_boxplot).
> # QQ Plot for Clay
> ggplot(h, aes(sample = clay)) +
+         geom_qq() +
+         geom_qq_line()
Warning messages:
1: Removed 173 rows containing non-finite values (stat_qq).
2: Removed 173 rows containing non-finite values (stat_qq_line).
> # QQ Plot for Frags
> ggplot(h, aes(sample = total_frags_pct)) +
+         geom_qq() +
+         geom_qq_line()
> # scatter plot
> ggplot(h, aes(x = clay, y = hzdepm)) +
+         geom_point() +
+         ylim(100, 0)
Warning message:
Removed 174 rows containing missing values (geom_point).
>
```

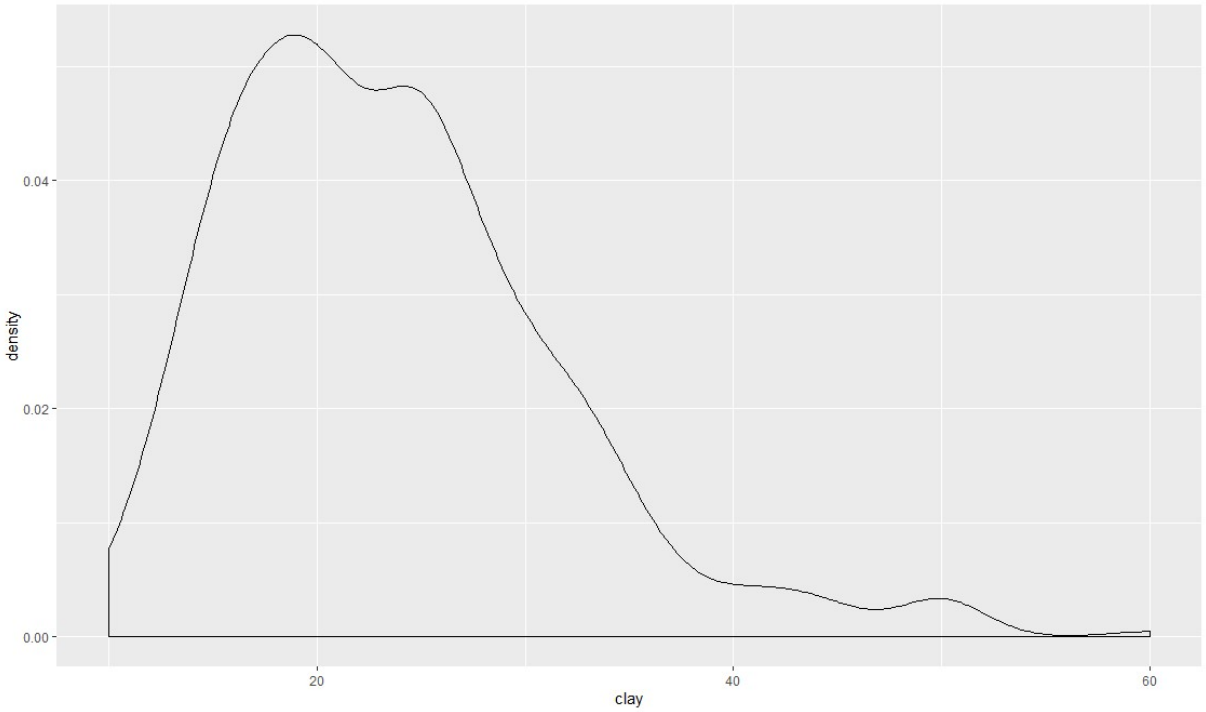
Data Editor

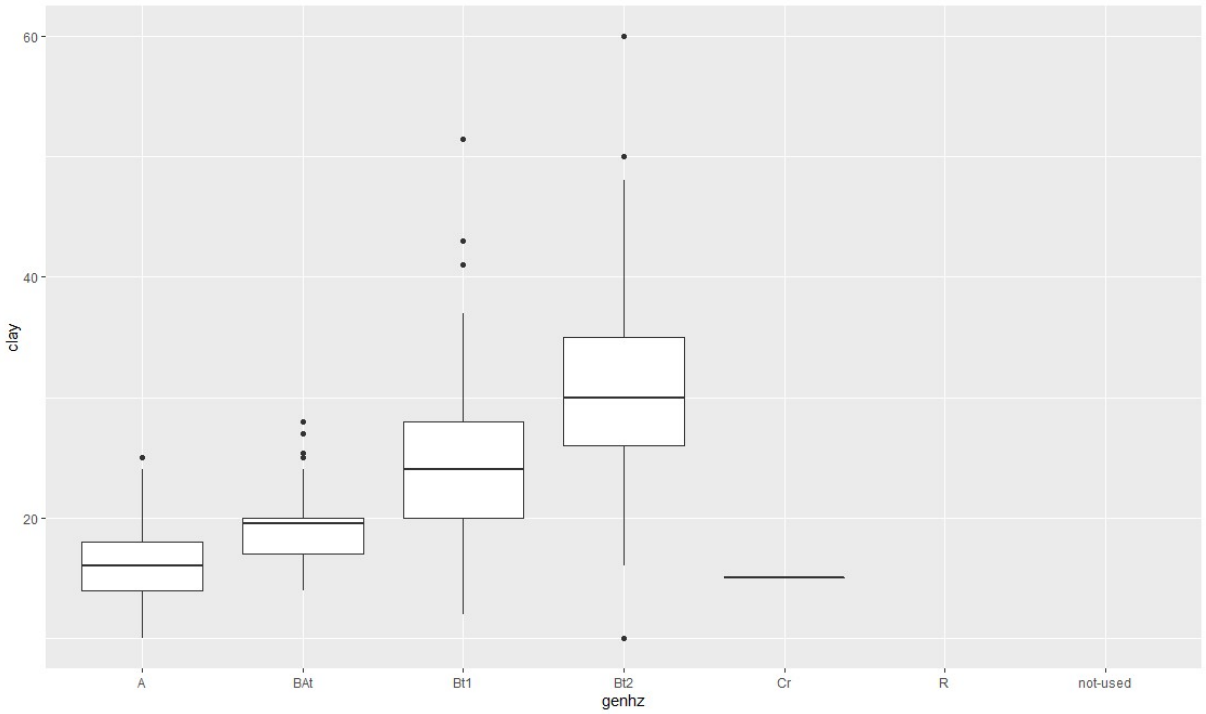
File Edit Help

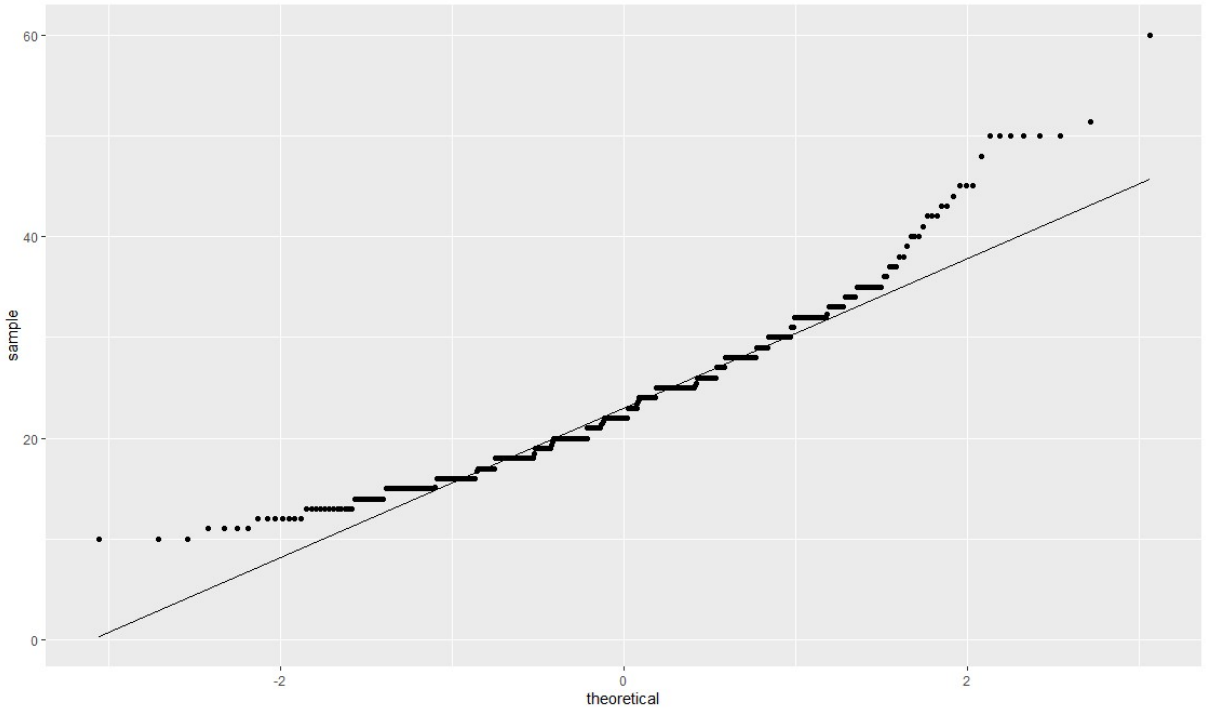
	row.names	peiid	phiid	hzname	genhz	hzdept	hzdepb	clay	silt
1	557	64505	299222	Oi	not-used	0	1	NA	NA
2	558	64505	299223	A	A	1	5	16.7	46.8
3	559	64505	299224	BAt	BAt	5	15	17	48
4	560	64505	299225	Bt1	Bt1	15	30	19.6	47.4
5	561	64505	299226	Bt2	Bt1	30	58	23.6	44.7
6	562	64505	299227	Bt3	Bt2	58	79	21.5	50.5
7	563	64505	299228	Crt	Cr	79	107	NA	NA
8	564	64505	299229	R	R	107	108	NA	NA
9	538	115595	579662	A	A	0	5	15	NA
10	539	115595	579661	BA	BAt	5	10	20	NA
11	540	115595	579660	BAt	BAt	10	28	25	NA
12	541	115595	579659	Bt1	Bt1	28	51	29	NA
13	542	115595	579658	Bt2	Bt1	51	74	35	NA
14	543	115595	579657	Cr	Cr	74	81	NA	NA
15	596	115819	580839	A	A	0	5	15	NA
16	597	115819	580840	BA	BAt	5	10	20	NA
17	598	115819	580841	Bt1	Bt1	10	28	25	NA
18	599	115819	580842	Bt2	Bt1	28	51	29	NA
19	600	115819	580843	Bt3	Bt2	51	74	35	NA

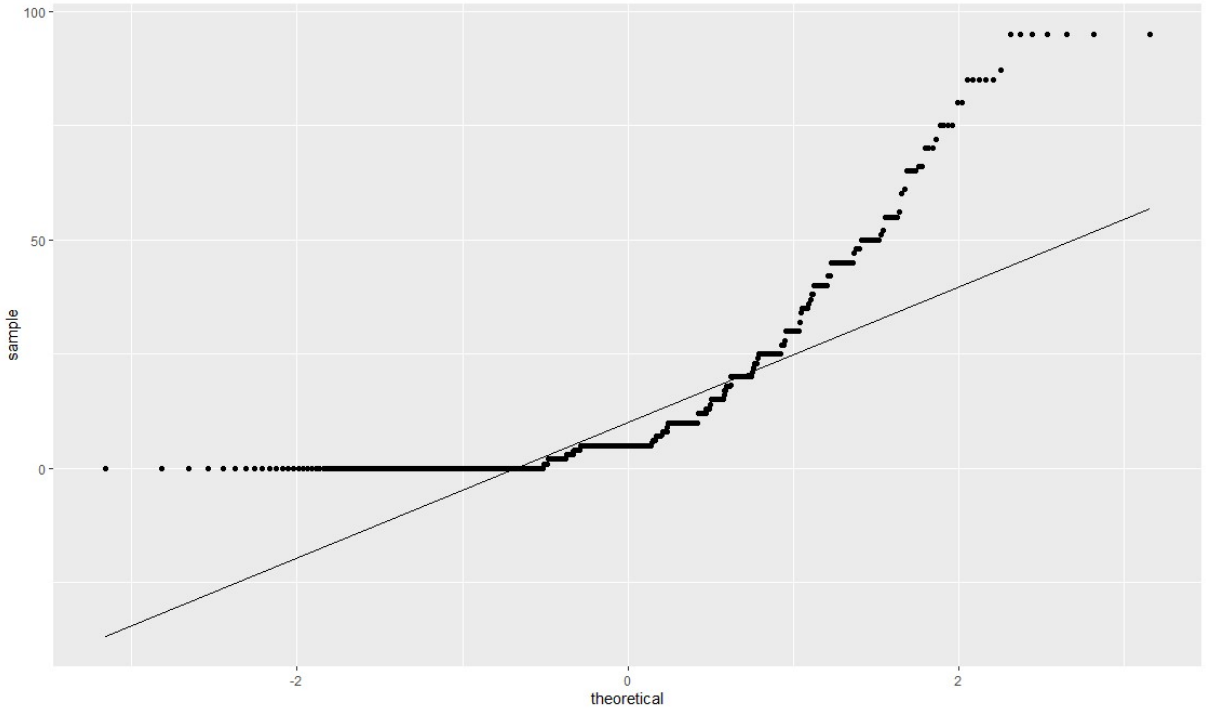

count
200
100
0
sl
l
sil
scl
cl
sicl
sic
c
NA
texcl

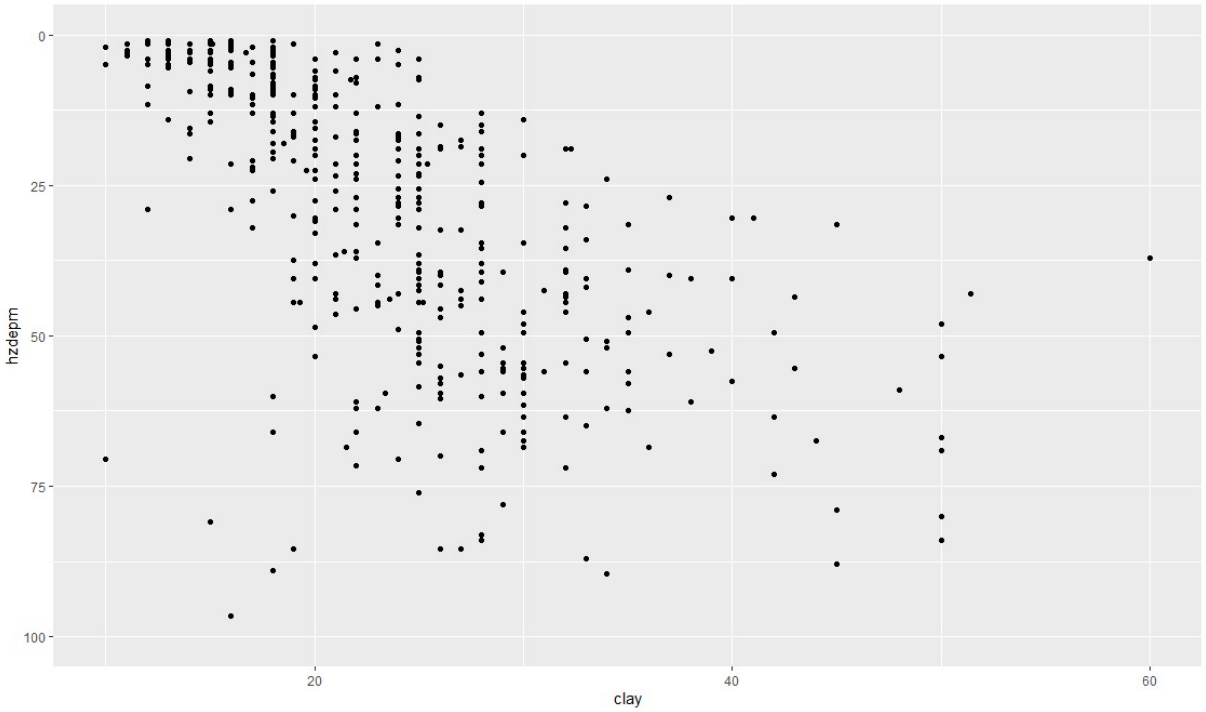

count
100
50
0
20
40
60
clay


density
0.04
0.02
0.00
20
40
60
clay


clay
60
40
20
A
BAt
Bt1
Bt2
Cr
R
not-used
genhz


60
40
20
0
sample
-2
0
2
theoretical


100
50
0
sample
-2
0
2
theoretical



Conclusion:

We have used Soil DB library in R for all the Graphs and EDA (Explanatory Data Analysis)

Experiment No: 8

Title: Basic and Advance graphics in R

Date of Performance:

Date of Submission:

Grade:

Signature:

Aim: Basic and Advance graphics in R

Theory and Program:

```
City = read.csv("http://course1.winona.edu/bdeppa/DSCI%20415/Data/City77.csv")
row.names(City) = City$City
City = City[,-1]
names(City)
hist(City$medinc,prob=T,nclass=20,col="blue",main="Histogram of Median Income",xlab="Median Income")
lines(density(City$medinc),lty=2,col="red",lwd=2)
require(ggplot2)
a = ggplot(City,aes(medinc))
a + geom_histogram(binwidth=1000,aes(y=..density..),fill="blue",color="black") +
  geom_density(linetype=2) +
  xlab("Median Income") +
  ggtitle("Median Income for Top 77 Cities")
plot(City$poverty,City$infmort,xlab="Percent Below Poverty Level",ylab="Infant Mortality Rate")
lines(lowess(City$poverty,City$infmort),lty=2,col="blue",lwd=2)
abline(lm(infmort~poverty,data=City),lwd=2)
title(main="Infant Morality vs. Poverty Level")
a = ggplot(City,aes(poverty,infmort))
a + geom_point() + geom_smooth() +
  geom_text(aes(label=row.names(City)),size=2) +
  xlab("Poverty") + ylab("Infant Mortality") +
  ggtitle("Infant Mortality vs. Poverty Rate")
Olives = read.csv("http://course1.winona.edu/bdeppa/DSCI%20415/Data/Olives.txt")
```

```
names(Olives)
par(mfrow=c(3,2))
boxplot(split(Olives$Oleic,Olives$Area.Name),xlab="Area",ylab="Oleic
Acid")
boxplot(split(Olives$Linoleic,Olives$Area.Name),xlab="Area",ylab="Lino
leic Acid")
boxplot(split(Olives$Strearic,Olives$Area.Name),xlab="Area",ylab="Stre
aric Acid")
boxplot(split(Olives$Linolenic,Olives$Area.Name),xlab="Area",ylab="Lin
olenic Acid")
boxplot(split(Olives$Eicosanoic,Olives$Area.Name),xlab="Area",ylab="E
icosanoic Acid")
boxplot(split(Olives$Eicosenoic,Olives$Area.Name),xlab="Area",ylab="E
icosenoic Acid")
require(s20x)
boxqq(Oleic~Region.Name,data=Olives)
install.packages("s20x")
a = ggplot(Olives,aes(Area.Name,Oleic))
a + geom_boxplot(fill="lightblue") + xlab("Area Name") +
  ylab("Oleic Acid") + ggtitle("Oleic Acid vs. Growing Area")
install.packages("violinmplot")
require(violinmplot)
violinmplot(Area.Name~Oleic,xlab="Oleic                    Acid",ylab="Area
Name",data=Olives,horizontal=T)
library(ggplot2)
a = ggplot(Olives,aes(Area.Name,Oleic))
a + geom_violin(fill="lightblue") + xlab("Area Name") +
  ylab("Oleic Acid") + ggtitle("Oleic Acid vs. Growing Area")
Statplot(Olives$Oleic,xname="Oleic")
install.packages("Statplot")
data("UCBAdmissions")
UCBAdmissions
temp = data.frame(UCBAdmissions)
```

```
temp
DeptCount = margin.table(UCBAdmissions,3)
DeptCount
par(mfrow=c(1,2))
pie(DeptCount)
barplot(DeptCount,xlab="Department",ylab="Frequency")
par(mfrow=c(1,2))
DeptAdmit = margin.table(UCBAdmissions,c(1,3))
DeptAdmit
barplot(DeptAdmit,xlab="Department",ylab="Frequency")
barplot(DeptAdmit,xlab="Department",ylab="Frequency",beside=TRUE
)
GenderAdmit = margin.table(UCBAdmissions,c(1,2))
GenderAdmit
pie(GenderAdmit[,1],main="Males")
pie(GenderAdmit[,2],main="Females")
barplot(GenderAdmit,xlab="Gender",ylab="Frequency")
barplot(GenderAdmit,xlab="Gender",ylab="Frequency",beside=T)
par(mfrow=c(1,2))
mosaicplot(~Gender+Admit,data=UCBAdmissions,color=T)
mosaicplot(~Admit+Gender,data=UCBAdmissions,color=T)
par(mfrow=c(1,2))
mosaicplot(~Dept+Admit,data=UCBAdmissions,color=T)
mosaicplot(~Admit+Dept,data=UCBAdmissions,color=T)
mosaicplot(~Dept+Gender+Admit,data=UCBAdmissions,color=T)
mosaicplot(~Dept+Admit+Gender,data=UCBAdmissions,color=T)
mosaicplot(~Gender+Dept+Admit,data=UCBAdmissions,color=T)
names(Olives)
olive.mat = Olives[,c(7,6,5,3)]
pairs(olive.mat)
panel.cor = function (x, y, digits = 2, prefix = "", cex.cor)
{
  usr <- par("usr")
```

```
  on.exit(par(usr))
  par(usr = c(0, 1, 0, 1))
  r <- abs(cor(x, y))
  txt <- format(c(r, 0.123456789), digits = digits)[1]
  txt <- paste(prefix, txt, sep = "")
  if (missing(cex.cor))
    cex <- 0.8/strwidth(txt)
  text(0.5, 0.5, txt, cex = cex * r)
}

panel.trendsd = function (x, y, f = 0.5)
{
  par(err = -1)
  xs <- sort(x, index = T)
  x <- xs$x
  ix <- xs$ix
  y <- y[ix]
  trend <- lowess(x, y, f)
  e2 <- (y - trend$y)^2
  scatter <- lowess(x, e2)
  uplim <- trend$y + sqrt(abs(scatter$y))
  lowlim <- trend$y - sqrt(abs(scatter$y))
  points(x, y, pch = 1)
  lines(trend, col = "Blue")
  lines(scatter$x, uplim, lty = 2, col = "Red")
  lines(scatter$x, lowlim, lty = 2, col = "Red")
  invisible()
}

panel.hist = function (x, ...)
{
  usr <- par("usr")
  on.exit(par(usr))
```

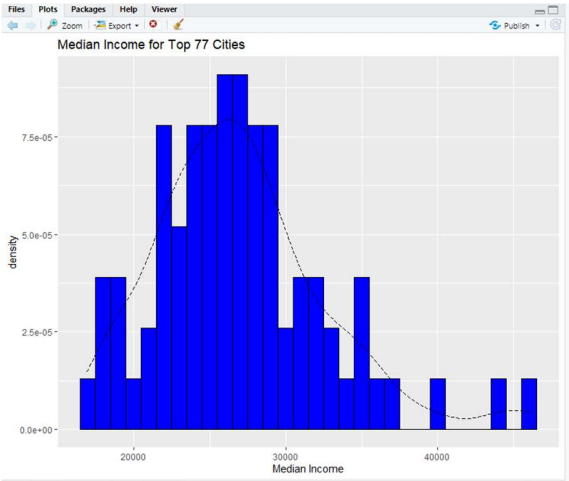
```
  par(usr = c(usr[1:2], 0, 1.5))
  h <- hist(x, plot = FALSE)
  breaks <- h$breaks
  nB <- length(breaks)
  y <- h$counts
  y <- y/max(y)
  rect(breaks[-nB], 0, breaks[-1], y, col = "cyan", ...)
}

pairs.trendsd = function(data,...) {
  pairs(data,lower.panel=panel.cor,upper.panel=panel.trendsd,
       diag.panel=panel.hist,...)}
require(lattice)
pairs.trendsd(olive.mat)
Boston                                                                    =
read.csv("http://course1.winona.edu/bdeppa/DSCI%20415/Data/Bosto
n.txt")
names(Boston)
install.packages("corrgram")
require(corrgram)
corrgram(Boston[,-
c(1:4)],lower.panel=panel.pts,upper.panel=panel.pie)
corrgram(Boston[,-
c(1:4)],lower.panel=panel.shade,upper.panel=panel.pie)
require(corrplot)
Boston.corr = cor(Boston[,-c(1:4)])   # find pairwise correlations between
all variables.
options(digits=2)  # reduce the number of significant digits shown.
Boston.corr
corrplot(Boston.corr)
corrplot(Boston.corr,method="ellipse")
data(iris)
names(iris)
```

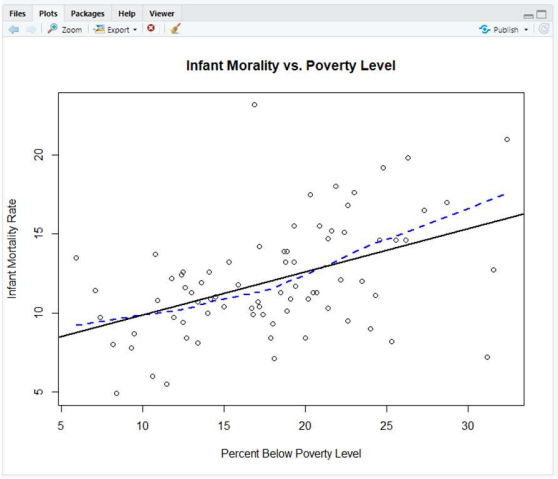
```
library(lattice)
xyplot(Sepal.Length~Petal.Length|Species,data=iris)
xyplot(Sepal.Length~Petal.Length|Species,layout=c(2,2),data=iris)
SepWid = equal.count(iris$Sepal.Width)
plot(SepWid)
print(SepWid)
xyplot(Sepal.Length~Petal.Length|SepWid,data=iris)
xyplot(Sepal.Length~Petal.Length|SepWid,groups=Species,layout=c(3,2
),
    auto.key=list(columns=3),main="Sepal Length * Petal Length | Sepal
 Width",data=iris)
splom(~iris[,1:4],groups=Species,auto.key=list(columns=3),data=iris)
bwplot(Species~Sepal.Width,data=iris,xlab="Sepal Width
(mm)",main="Boxplots of Iris Sepal Width")
stripplot(Species ~ jitter(Sepal.Width), data = iris, xlab="Sepal Width
(mm)",main="Sepal Width Across Species")
stripplot(Species ~ jitter(Sepal.Width), data = iris, aspect = 1,
      jitter=T,xlab="Sepal Width (mm)",main="Sepal Width Across
Species")
coplot(Sepal.Width~Sepal.Length|Species,data=iris)
coplot(Sepal.Width~Sepal.Length|Petal.Width,number=4,overlap=.2,da
ta=iris)
Ozdata =
read.csv("http://course1.winona.edu/bdeppa/DSCI%20415/Data/Ozdat
a.csv")
names(Ozdata)
coplot(upoz~safb|inbh*v500,number=c(4,4),panel=panel.smooth,data=
Ozdata)
coplot(upoz~safb|inbh*v500,number=c(4,4),overlap=.25,panel=functio
n(x,y,...) panel.smooth(x,y,span=.6,...),data=Ozdata)
install.packages("ash")
library(ash)
```

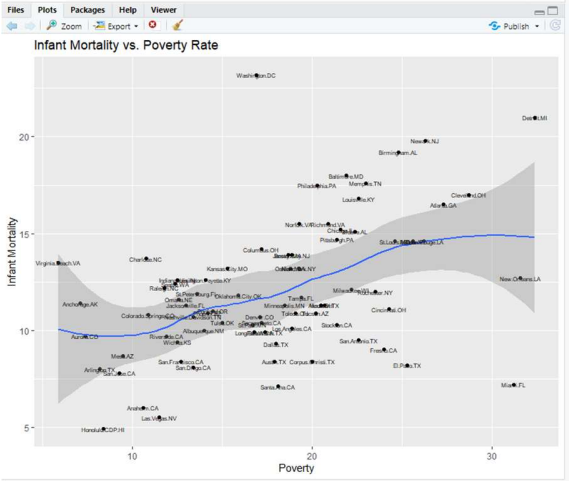
```
Swiss =
read.csv("http://course1.winona.edu/bdeppa/DSCI%20415/Data/Swiss.
csv")
names(Swiss)
d1 = ash1(bin1(Swiss$diagon,nbin=50),3)
hist(Swiss$diagon,nclass=20,prob=T,col="blue",main="Histogram of Bill
Diagonal (mm)")
lines(d1)
Statplot(Swiss$diagon)
diagrt.bin <- bin2(cbind(Swiss$diagon,Swiss$right),nbin=c(50,50))
diagrt.1 <- ash2(diagrt.bin,m=c(5,5))
persp(diagrt.1,xlab="Diagonal Length",ylab="Right
Height",zlab="",cex=.5,theta=-45,phi=30,shade=1,col="royal blue")
image(diagrt.1,xlab="Bill Diagonal",ylab="Right Height")
contour(diagrt.1,xlab="Bill Diagonal",ylab="Right Height",add=T)
points(Swiss$diagon,Swiss$right,pch=as.character(Swiss$genu),cex=.4)
Genuine <- Swiss$genu
Genuine[Genuine==0] <- "Forged"
Genuine[Genuine==1] <- "Real"
xyplot(Swiss$diagon~Swiss$bottom|Genuine,groups=Genuine)
splom(~Swiss[,1:6],groups=Genuine,auto.key=list(columns=2))
pairs.image <- function(x) {
  pairs(x,panel=function(x,y) {
    foo <- bin2(cbind(x,y),nbin=c(75,75))
    foo <- ash2(foo,m=c(6,6))
    image(foo,add=T,xlab="",ylab="",col=topo.colors(1000))
    points(x,y,pch=".")
  })
}
pairs.image(Swiss[,1:6])
pairs.persp <- function(x) {
  par(bg="sky blue")
  pairs(x,panel=function(x,y) {
```

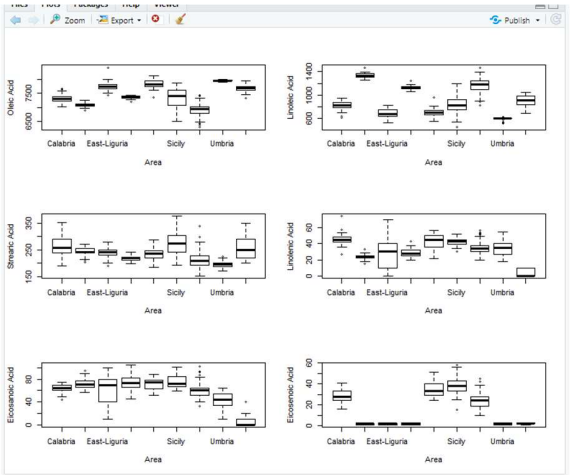
```
    foo <- bin2(cbind(x,y),nbin=c(75,75))
    foo <- ash2(foo,m=c(8,8))
    par(new=T)
    persp(foo,xlab="",ylab="",theta=-45,phi=35,col="royal blue",
        shade=.75,box=F,scale=F,border=NA,expand=.9)
  })
  par(new=F,bg="white")
}
pairs.persp(Swiss[,1:3])
```

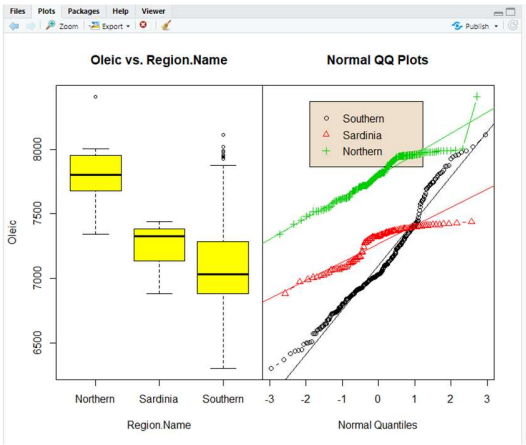
Outputs:

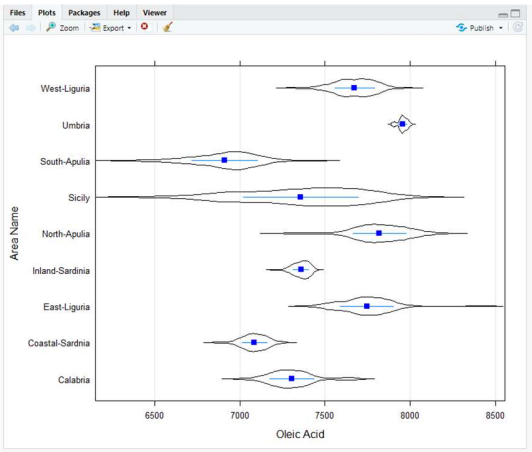


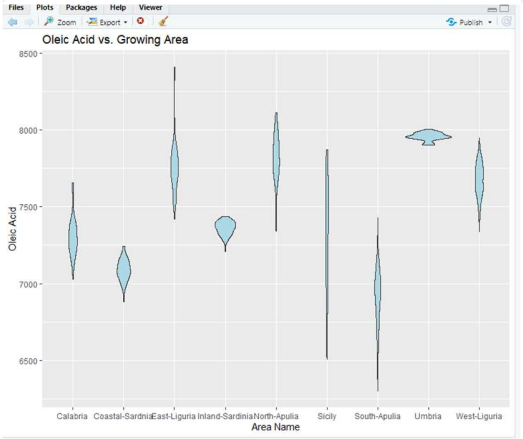




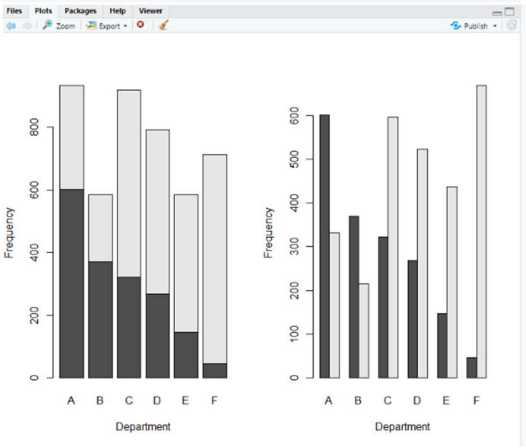


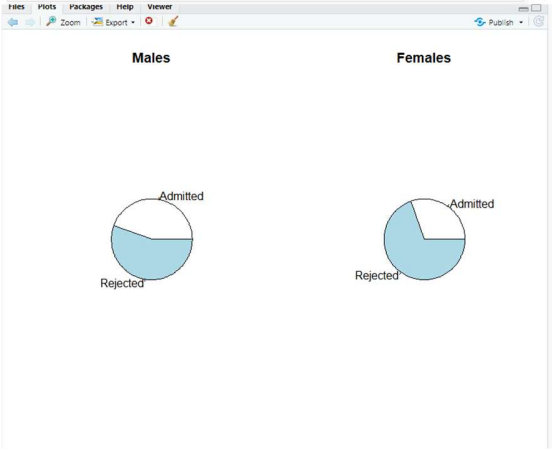

Oleic vs. Region.Name
Normal QQ Plots
Southern
Sardinia
Northern
Oleic
Region.Name
Normal Quantiles

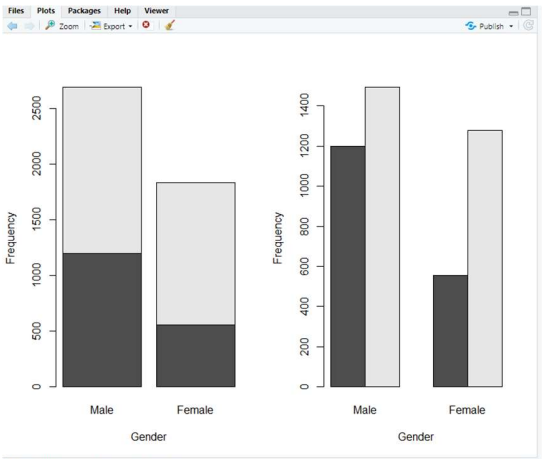

West-Liguria
Umbria
South-Apulia
Sicily
North-Apulia
Inland-Sardinia
East-Liguria
Coastal-Sardnia
Calabria
Area Name
Oleic Acid

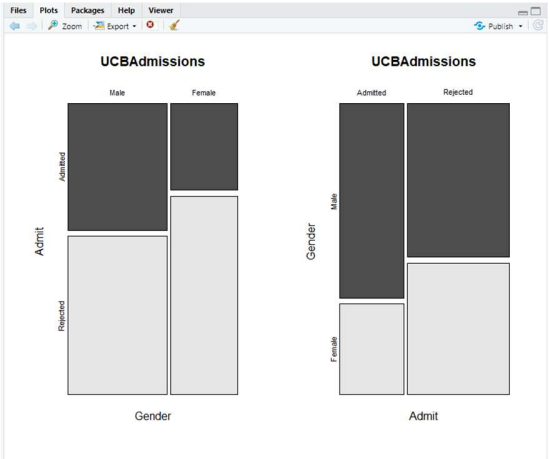

Oleic Acid vs. Growing Area
Oleic Acid
Area Name

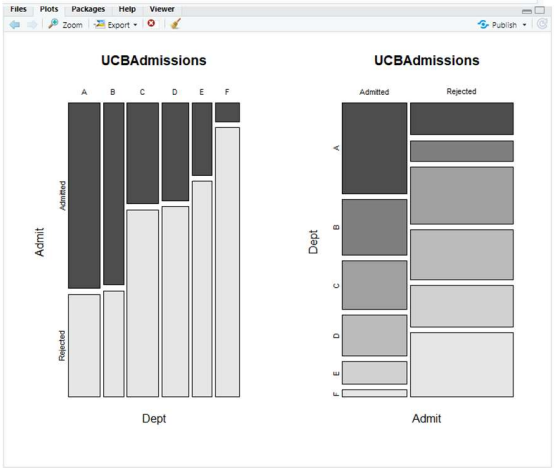

Frequency
Department


Frequency
Department

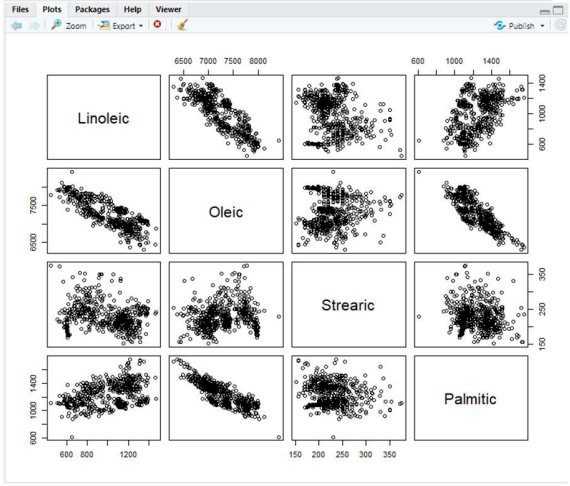

Males
Females
Admitted
Rejected

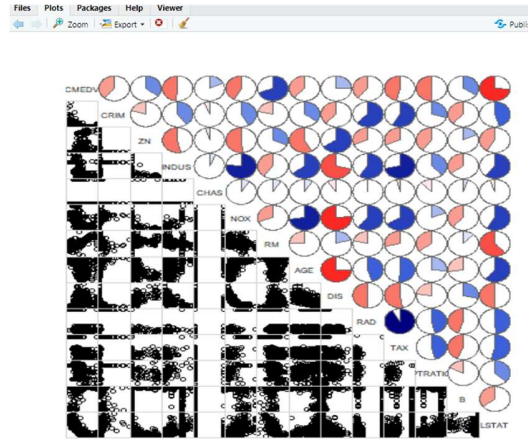

Frequency
Male
Female
Gender
Frequency
Male
Female
Gender

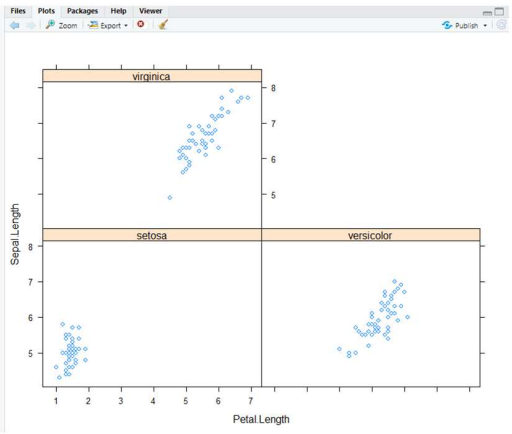

UCBAdmissions
Male
Female
Admitted
Rejected
Admit
Gender
UCBAdmissions
Admitted
Rejected
Male
Female
Gender
Admit

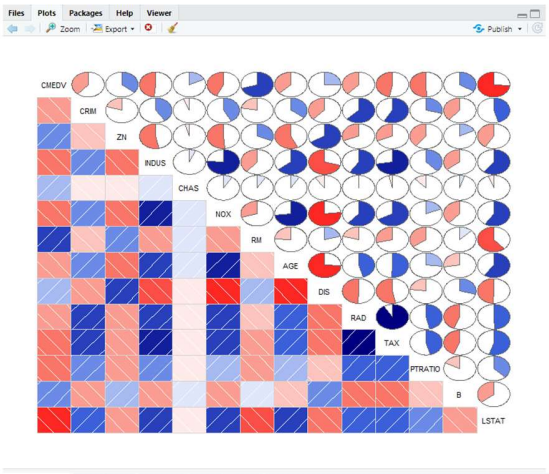

UCBAdmissions
A
B
C
D
E
F
Admitted
Rejected
Admit
Dept
UCBAdmissions
Admitted
Rejected
A
B
C
D
E
F
Dept
Admit

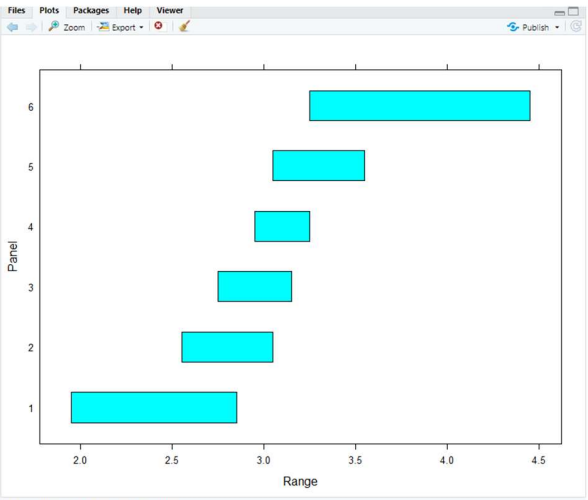

UCBAdmissions
Male
Female
Gender
Dept
UCBAdmissions
Admitted
Rejected
Admit
Dept

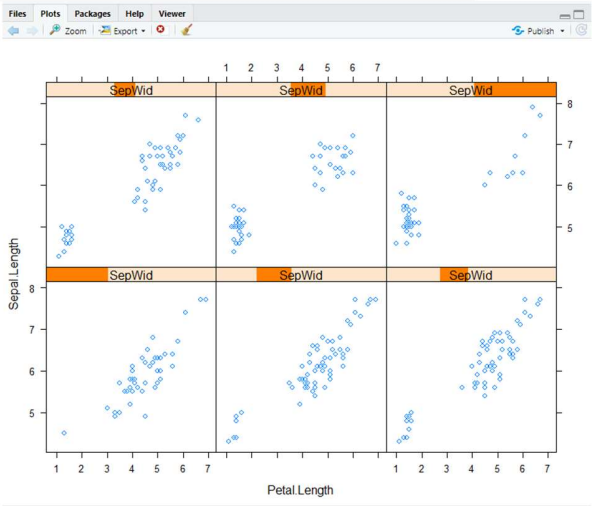

Linoleic
Oleic
Strearic
Palmitic

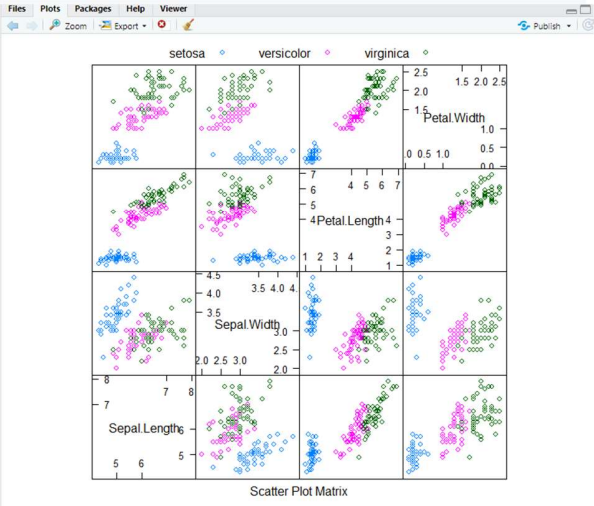

CMEDV
CRIM
ZN
INDUS
CHAS
NOX
RM
AGE
DIS
RAD
TAX
PTRATIO
B
LSTAT

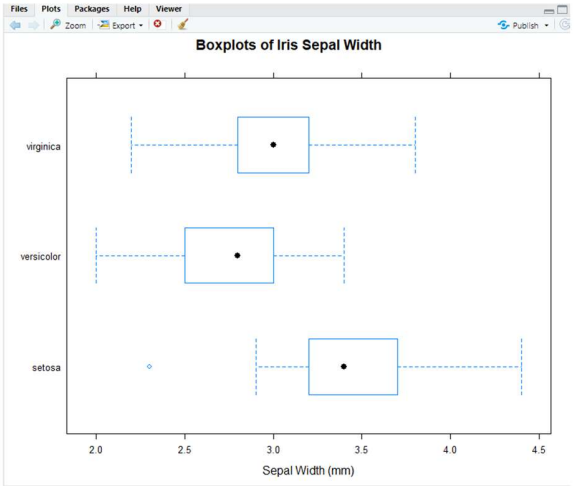

virginica
setosa
versicolor
Sepal.Length
Petal.Length

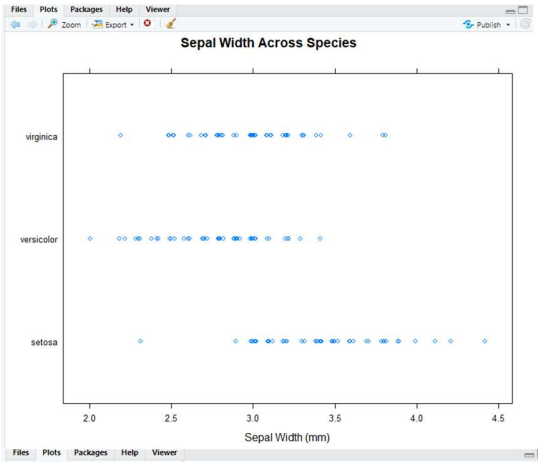

CMEDV
CRIM
ZN
INDUS
CHAS
NOX
RM
AGE
DIS
RAD
TAX
PTRATIO
B
LSTAT

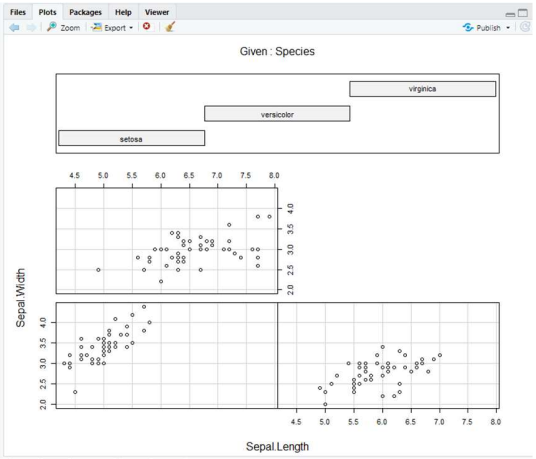

Panel
Range

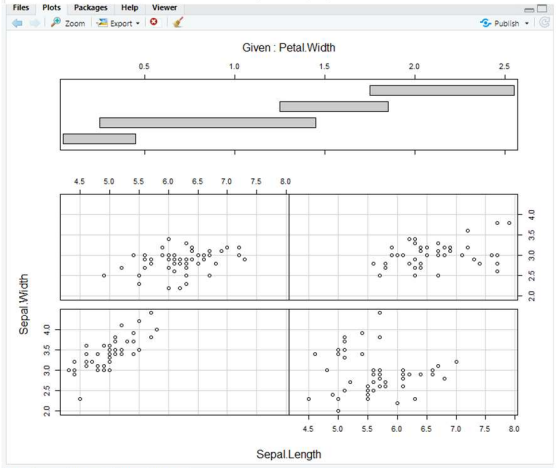

SepWid
Sepal.Length
Petal.Length

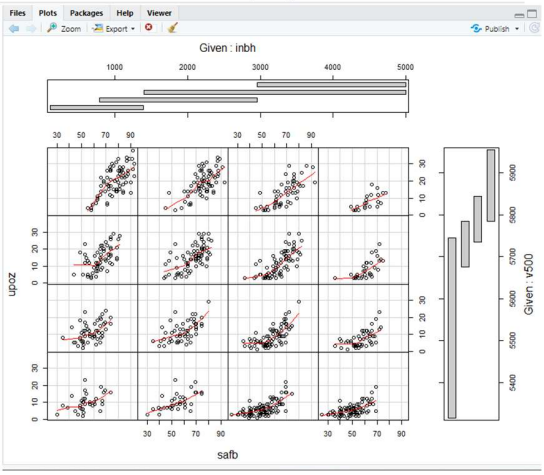

setosa
versicolor
virginica
Petal.Width
Petal.Length
Sepal.Width
Sepal.Length
Scatter Plot Matrix

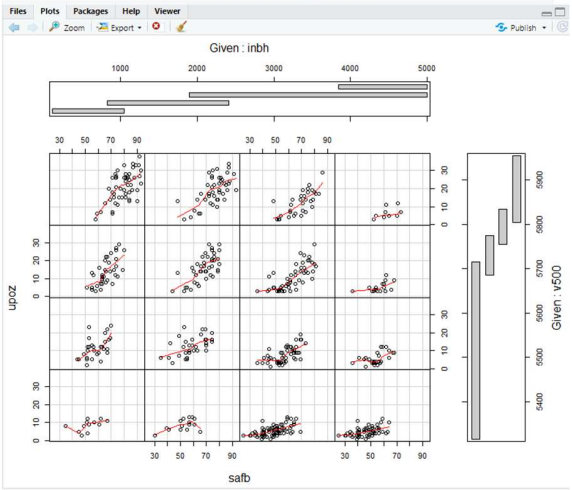

Boxplots of Iris Sepal Width
virginica
versicolor
setosa
Sepal Width (mm)

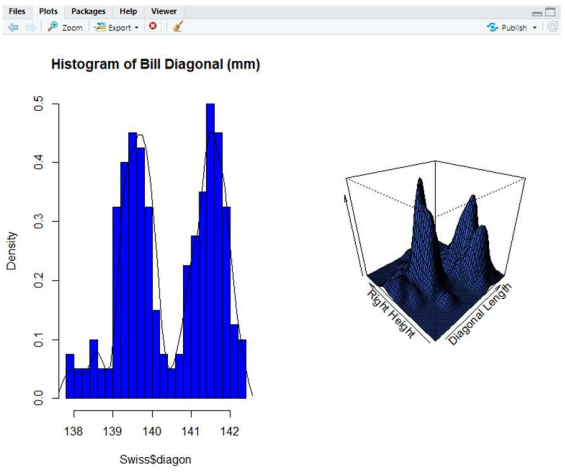

Files Plots Packages Help Viewer
Zoom Export Publish
Sepal Width Across Species
virginica
versicolor
setosa
2.0 2.5 3.0 3.5 4.0 4.5
Sepal Width (mm)

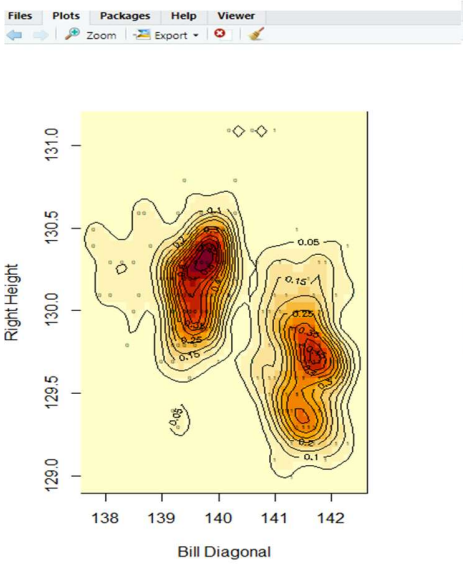

Files Plots Packages Help Viewer
Zoom Export Publish
Given : Species
virginica
versicolor
setosa
Sepal.Width
Sepal.Length

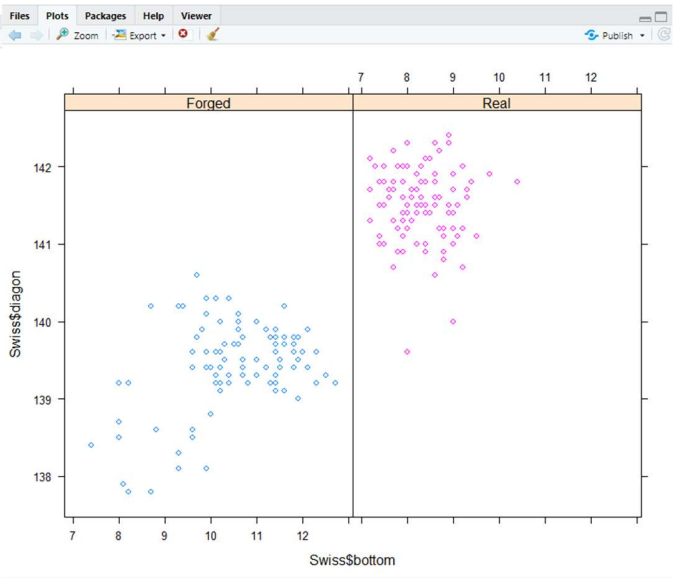

Files Plots Packages Help Viewer
Zoom Export Publish
Given : Petal.Width
Sepal.Width
Sepal.Length

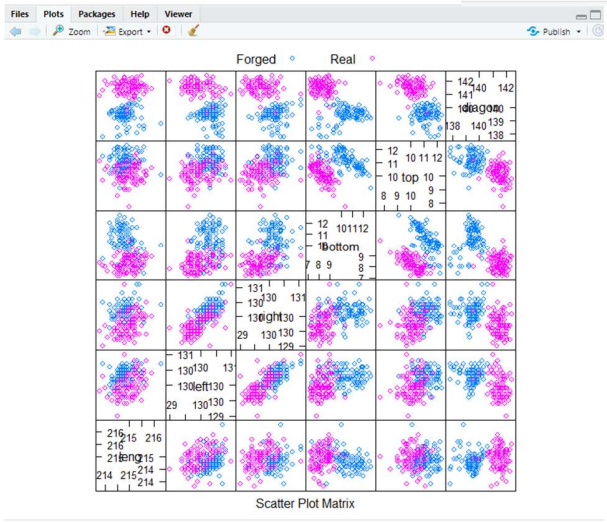

Files Plots Packages Help Viewer
Zoom Export Publish
Given : inbh
up2z
safb
Given : v500

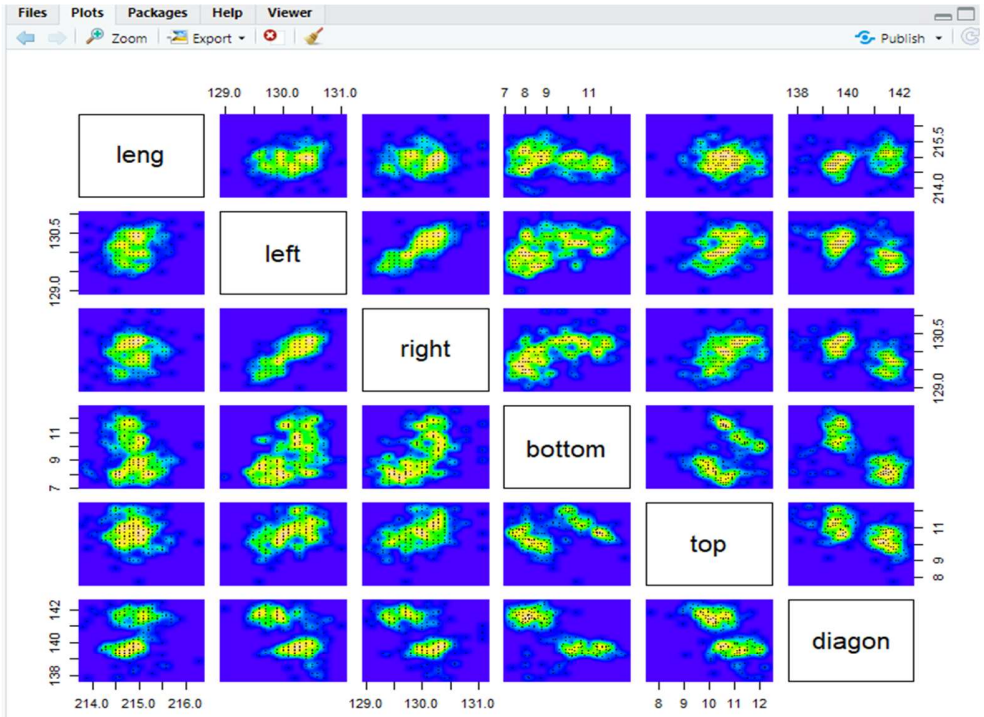

Files Plots Packages Help Viewer
Zoom Export Publish
Given : inbh
up2z
safb
Given : v500

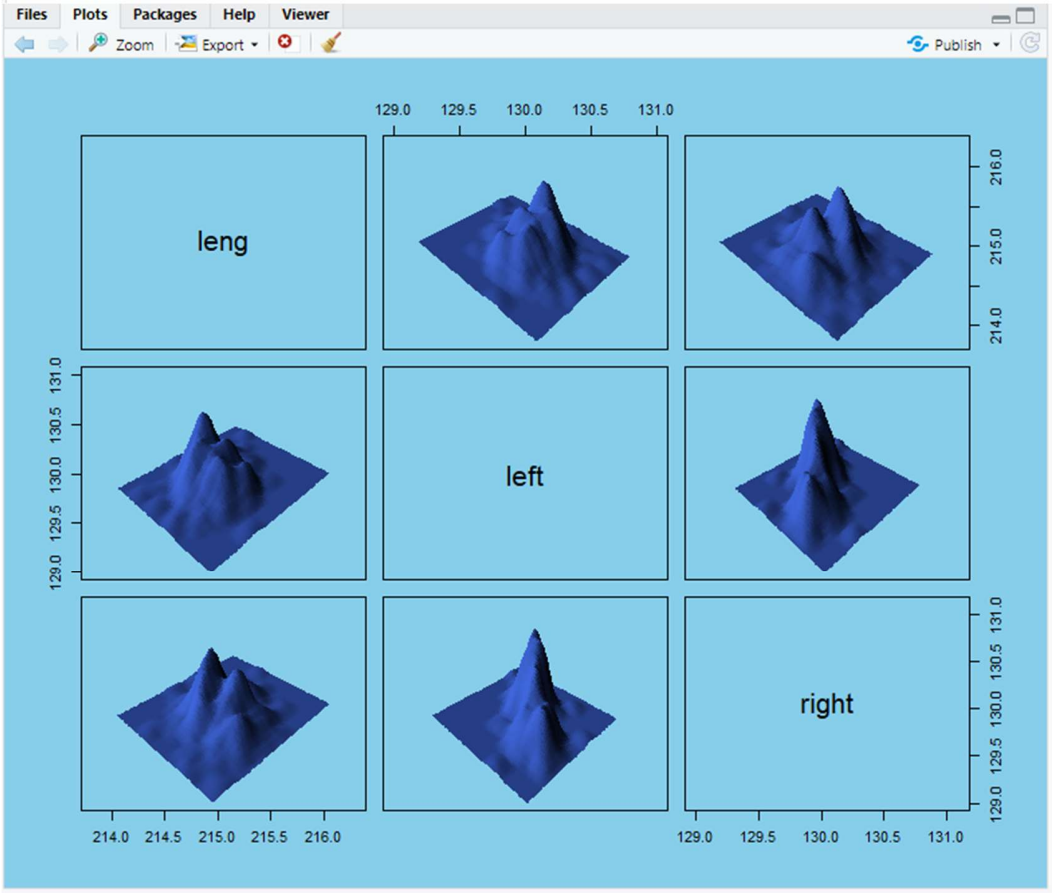

Files Plots Packages Help Viewer
Zoom Export Publish
Histogram of Bill Diagonal (mm)
Density
138 139 140 141 142
Swiss$diagon
Right Height
Diagonal Length


Files Plots Packages Help Viewer
Zoom Export
Right Height
Bill Diagonal
129.0 129.5 130.0 130.5 131.0
138 139 140 141 142


Files Plots Packages Help Viewer
Zoom Export Publish
Forged
Real
Swiss$diagon
Swiss$bottom
138 139 140 141 142
7 8 9 10 11 12


Files Plots Packages Help Viewer
Zoom Export Publish
Forged
Real
Scatter Plot Matrix


Files
Plots
Packages
Help
Viewer
Zoom
Export
Publish
leng
left
right
bottom
top
diagon


Files
Plots
Packages
Help
Viewer
Zoom
Export
Publish
leng
left
right

Conclusion:

Hence we have executed the Basic graphics and Advance graphics using dataset in R

Experiment No: 9

Title: Basics of correlation simple and multiple regression in R.

Date of Performance:

Date of Submission:

Grade:

Signature:

Aim: Basics of correlation simple and multiple Regression in R.

Theory:

The overall goal is to give you a very quick introduction to conducting correlation and regression analyses in R.

Correlation:

The Pearson product moment correlation seeks to measure the linear association between two variables, xx and yy on a standardized scale ranging from r=−1−−1r=−1−−1.

The correlation of x and y is a covariance that has been standardized by the standard deviations of xx and yy. This yields a scale-insensitive measure of the linear association of xx and yy.

Regression:

We can easily extend to larger regression models by adding terms to the right side of the formula. For example, in the mtcars dataset (car performance statistics from 1974 Motor Trend), we could examine the extent to which the gas mileage (mpg) is a function of both gross horsepower (hp) and transmission (am, where 0 is 'automatic' and 1 is 'manual').

Program and Outputs:

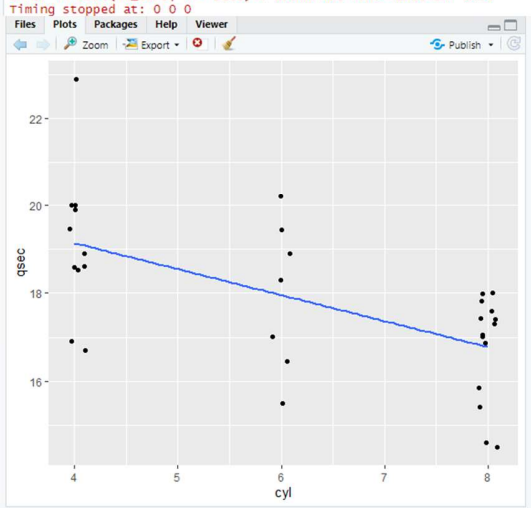
```
~/
        C:\Users\SAYALI\AppData\Local\Temp\RtmpOsOoyH\downloaded_packages
> library(dplyr)

Attaching package: 'dplyr'

The following objects are masked from 'package:raster':

    intersect, select, union

The following objects are masked from 'package:aqp':

    slice, union

The following objects are masked from 'package:stats':

    filter, lag

The following objects are masked from 'package:base':

    intersect, setdiff, setequal, union

> to_correlate <- mtcars %>% dplyr::select(qsec, cyl, disp, hp)
> cor(to_correlate)
           qsec        cyl       disp         hp
qsec  1.0000000 -0.5912421 -0.4336979 -0.7082234
cyl  -0.5912421  1.0000000  0.9020329  0.8324475
disp -0.4336979  0.9020329  1.0000000  0.7909486
hp   -0.7082234  0.8324475  0.7909486  1.0000000
> cor.test(mtcars$qsec, mtcars$cyl)

        Pearson's product-moment correlation

data:  mtcars$qsec and mtcars$cyl
t = -4.0154, df = 30, p-value = 0.0003661
alternative hypothesis: true correlation is not equal to 0
95 percent confidence interval:
 -0.7792781 -0.3055388
sample estimates:
       cor
-0.5912421

> cor.test(mtcars$qsec, mtcars$cyl, conf.level = 0.9)

        Pearson's product-moment correlation

data:  mtcars$qsec and mtcars$cyl
t = -4.0154, df = 30, p-value = 0.0003661
alternative hypothesis: true correlation is not equal to 0
90 percent confidence interval:
 -0.7552287 -0.3576005
sample estimates:
       cor
-0.5912421

> library(ggplot2)
> ggplot(to_correlate, aes(x=cyl, y=qsec)) + geom_jitter(width=0.1) + stat_smooth(method="lm", se=FALSE)
`geom_smooth()` using formula 'y ~ x'
```

```
~/
`geom_smooth()` using formula 'y ~ x'
> Hmisc::rcorr(to_correlate %>% as.matrix())
     qsec   cyl  disp    hp
qsec  1.00 -0.59 -0.43 -0.71
cyl  -0.59  1.00  0.90  0.83
disp -0.43  0.90  1.00  0.79
hp   -0.71  0.83  0.79  1.00

n= 32


P
     qsec   cyl    disp   hp
qsec        0.0004 0.0131 0.0000
cyl  0.0004        0.0000 0.0000
disp 0.0131 0.0000        0.0000
hp   0.0000 0.0000 0.0000
> stargazer(cor(to_correlate), type = "html")
Error in stargazer(cor(to_correlate), type = "html") :
  could not find function "stargazer"
> install.packages("stargazer")
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/SAYALI/Documents/R/win-library/3.6'
(as 'lib' is unspecified)
trying URL 'https://cran.rstudio.com/bin/windows/contrib/3.6/stargazer_5.2.2.zip'
Content type 'application/zip' length 622660 bytes (608 KB)
downloaded 608 KB

package 'stargazer' successfully unpacked and MD5 sums checked

The downloaded binary packages are in
        C:\Users\SAYALI\AppData\Local\Temp\RtmpOsOoyH\downloaded_packages
> library(stargazer)

Please cite as:

 Hlavac, Marek (2018). stargazer: Well-Formatted Regression and Summary Statistics Tables.
 R package version 5.2.2. https://CRAN.R-project.org/package=stargazer

> stargazer(cor(to_correlate), type = "html")

<table style="text-align:center"><tr><td colspan="5" style="border-bottom: 1px solid black"></td></tr><tr><td style="text-align:left"></td><td>qsec</td><td>cyl
</td><td>disp</td><td>hp</td></tr>
<tr><td colspan="5" style="border-bottom: 1px solid black"></td></tr><tr><td style="text-align:left">qsec</td><td>1</td><td>-0.591</td><td>-0.434</td><td>-0.70
8</td></tr>
<tr><td style="text-align:left">cyl</td><td>-0.591</td><td>1</td><td>0.902</td><td>0.832</td></tr>
<tr><td style="text-align:left">disp</td><td>-0.434</td><td>0.902</td><td>1</td><td>0.791</td></tr>
<tr><td style="text-align:left">hp</td><td>-0.708</td><td>0.832</td><td>0.791</td><td>1</td></tr>
<tr><td colspan="5" style="border-bottom: 1px solid black"></td></tr></table>
> apaTables::apa.cor.table(to_correlate)
Error in loadNamespace(name) : there is no package called 'apaTables'
> install.packages(apaTables)
Error in install.packages : object 'apaTables' not found
> install.packages("apaTables")
WARNING: Rtools is required to build R packages but is not currently installed. Please download and install the appropriate version of Rtools before proceeding
```

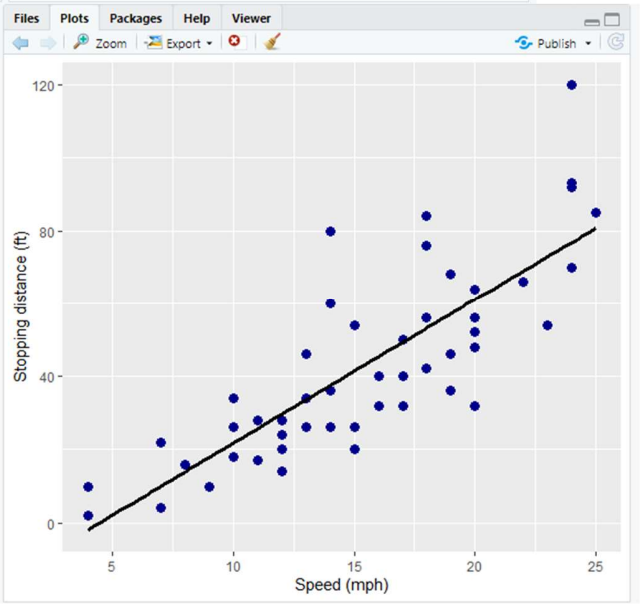
```
> apaTables::apa.cor.table(to_correlate)


Means, standard deviations, and correlations with confidence intervals


  Variable M      SD     1            2          3
  1. qsec  17.85  1.79
  2. cyl   6.19   1.79   -.59**
                         [-.78, -.31]
  3. disp  230.72 123.94 -.43*        .90**
                         [-.68, -.10] [.81, .95]
  4. hp    146.69 68.56  -.71**       .83**      .79**
                         [-.85, -.48] [.68, .92] [.61, .89]


Note. M and SD are used to represent mean and standard deviation, respectively.
Values in square brackets indicate the 95% confidence interval.
The confidence interval is a plausible range of population correlations
that could have caused the sample correlation (Cumming, 2014).
* indicates p < .05. ** indicates p < .01.


> ctest <- cor.test(mtcars$qsec, mtcars$cyl)
> str(ctest)
List of 9
 $ statistic  : Named num -4.02
  ..- attr(*, "names")= chr "t"
 $ parameter  : Named int 30
  ..- attr(*, "names")= chr "df"
 $ p.value    : num 0.000366
 $ estimate   : Named num -0.591
  ..- attr(*, "names")= chr "cor"
 $ null.value : Named num 0
  ..- attr(*, "names")= chr "correlation"
 $ alternative: chr "two.sided"
 $ method     : chr "Pearson's product-moment correlation"
 $ data.name  : chr "mtcars$qsec and mtcars$cyl"
 $ conf.int   : num [1:2] -0.779 -0.306
  ..- attr(*, "conf.level")= num 0.95
 - attr(*, "class")= chr "htest"
> ctest$p.value
[1] 0.0003660533
> ctest$estimate
       cor
-0.5912421
> broom::glance(ctest)
# A tibble: 1 x 8
  estimate statistic  p.value parameter conf.low conf.high method                  alternative
     <dbl>     <dbl>    <dbl>     <int>    <dbl>     <dbl> <chr>                   <chr>
1   -0.591     -4.02 0.000366        30   -0.779    -0.306 Pearson's product-moment~ two.sided
> cor.test(mtcars$qsec, mtcars$cyl, method = "spearman")

        Spearman's rank correlation rho
```

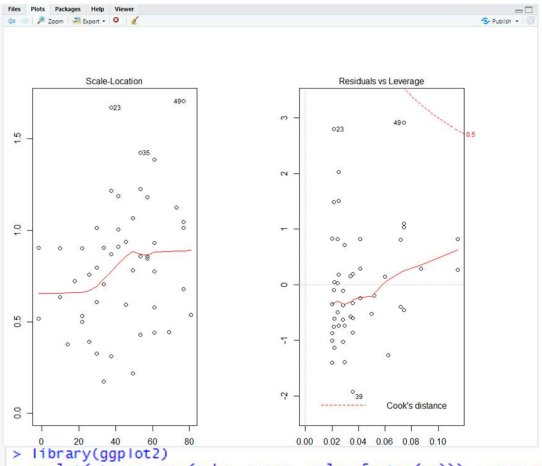
```
> cor(to_correlate, method = "spearman")
           qsec       cyl       disp         hp
qsec  1.0000000 -0.5723509 -0.4597818 -0.6666060
cyl  -0.5723509  1.0000000  0.9276516  0.9017909
disp -0.4597818  0.9276516  1.0000000  0.8510426
hp   -0.6666060  0.9017909  0.8510426  1.0000000
> to_correlate_miss <- to_correlate
> to_correlate_miss$qsec[c(1, 5)] <- NA
> cor(to_correlate_miss)
     qsec       cyl       disp         hp
qsec    1        NA         NA         NA
cyl    NA 1.0000000 0.9020329 0.8324475
disp   NA 0.9020329 1.0000000 0.7909486
hp     NA 0.8324475 0.7909486 1.0000000
> cor(to_correlate_miss, use="pairwise.complete.obs")
           qsec       cyl       disp         hp
qsec  1.0000000 -0.5961232 -0.4483569 -0.7313008
cyl  -0.5961232  1.0000000  0.9020329  0.8324475
disp -0.4483569  0.9020329  1.0000000  0.7909486
hp   -0.7313008  0.8324475  0.7909486  1.0000000
> ggplot(cars, aes(x=speed, y=dist)) +
+     geom_point(color='darkblue', size = 3) +
+     geom_smooth(method=lm, se=FALSE, fullrange=TRUE, color='black', size=1.2) +
+     labs(x="Speed (mph)", y="Stopping distance (ft)")
`geom_smooth()` using formula 'y ~ x'
> lm_cars <- lm(dist ~ speed, data=cars)
> summary(lm_cars)

Call:
lm(formula = dist ~ speed, 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) -17.5791     6.7584  -2.601   0.0123 *
speed         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
```

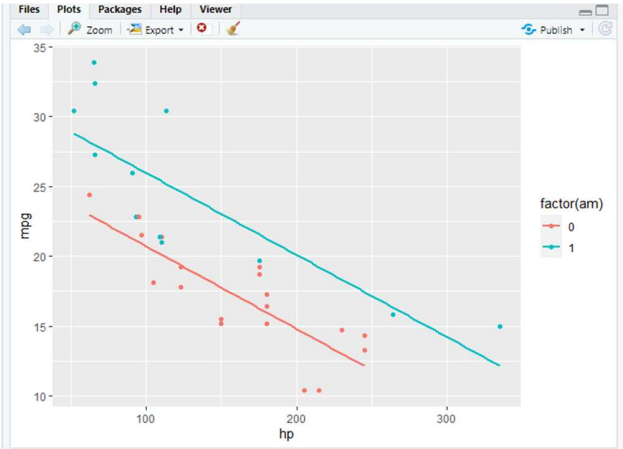
```
> plot(lm_cars)
Hit <Return> to see next plot:
Hit <Return> to see next plot: return
> autoplot(lm_cars)
Error: Objects of type lm not supported by autoplot.
Run `rlang::last_error()` to see where the error occurred.
> system.time(lm_cars.boot <- Boot(lm_cars, R=2000))
Error in Boot(lm_cars, R = 2000) : could not find function "Boot"
Timing stopped at: 0 0 0
> install.packages("boot")
WARNING: Rtools is required to build R packages but is not currently installed

https://cran.rstudio.com/bin/windows/Rtools/
Installing package into 'C:/Users/SAYALI/Documents/R/win-library/3.6'
(as 'lib' is unspecified)
trying URL 'https://cran.rstudio.com/bin/windows/contrib/3.6/boot_1.3-24.zip'
Content type 'application/zip' length 639491 bytes (624 KB)
downloaded 624 KB

package 'boot' successfully unpacked and MD5 sums checked

The downloaded binary packages are in
        C:\Users\SAYALI\AppData\Local\Temp\RtmpOsOoyH\downloaded_packages
> system.time(lm_cars.boot <- Boot(lm_cars, R=2000))
Error in Boot(lm_cars, R = 2000) : could not find function "Boot"
Timing stopped at: 0 0 0
> library(boot)
> system.time(lm_cars.boot <- Boot(lm_cars, R=2000))
Error in Boot(lm_cars, R = 2000) : could not find function "Boot"
Timing stopped at: 0 0 0
```




```
> library(boot)
> system.time(lm_cars.boot <- Boot(lm_cars, R=2000))
Error in Boot(lm_cars, R = 2000) : could not find function "Boot"
Timing stopped at: 0 0 0
> summary(lm_cars.boot, high.moments=TRUE)
Error in summary(lm_cars.boot, high.moments = TRUE) :
  object 'lm_cars.boot' not found
> mpg_model <- lm(mpg ~ hp + am, mtcars)
> summary(mpg_model)

Call:
lm(formula = mpg ~ hp + am, data = mtcars)

Residuals:
    Min      1Q  Median      3Q     Max
-4.3843 -2.2642  0.1366  1.6968  5.8657

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) 26.584914   1.425094  18.655  < 2e-16 ***
hp          -0.058888   0.007857  -7.495 2.92e-08 ***
am           5.277085   1.079541   4.888 3.46e-05 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 2.909 on 29 degrees of freedom
Multiple R-squared:  0.782,     Adjusted R-squared:  0.767
F-statistic: 52.02 on 2 and 29 DF,  p-value: 2.55e-10

> library(ggplot2)
```



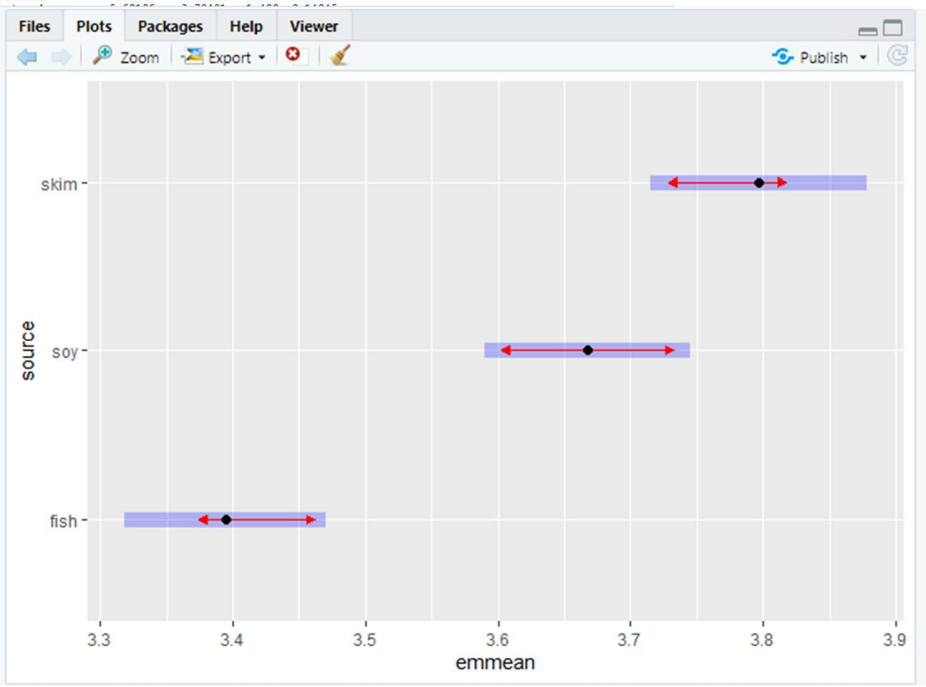





```
> library(ggplot2)
> ggplot(mtcars, aes(x=hp, y=mpg, color=factor(am))) + geom_point() + stat_smooth(method=lm, se=FALSE)
`geom_smooth()` using formula 'y ~ x'
>
> view(mpg_model)
> broom::glance(mpg_model)
# A tibble: 1 x 11
  r.squared adj.r.squared sigma statistic  p.value    df logLik   AIC   BIC deviance
      <dbl>         <dbl> <dbl>     <dbl>    <dbl> <int>  <dbl> <dbl> <dbl>    <dbl>
1     0.782         0.767  2.91      52.0 2.55e-10     3  -78.0  164.  170.      245.
# ... with 1 more variable: df.residual <int>
> broom::tidy(mpg_model)
# A tibble: 3 x 5
  term        estimate std.error statistic  p.value
  <chr>          <dbl>     <dbl>     <dbl>    <dbl>
1 (Intercept)  26.6      1.43        18.7  1.07e-17
2 hp           -0.0589   0.00786     -7.50 2.92e- 8
3 am            5.28     1.08         4.89 3.46e- 5
> int_model <- lm(mpg ~ hp*wt + am, mtcars)
> summary(int_model)

Call:
lm(formula = mpg ~ hp * wt + am, data = mtcars)

Residuals:
    Min      1Q  Median      3Q     Max
-2.9845 -1.6580 -0.7407  1.4362  4.5266

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) 49.452241   5.280731   9.365 5.69e-10 ***
hp          -0.119303   0.026550  -4.494 0.000119 ***
wt          -8.100558   1.789325  -4.527 0.000108 ***
am           0.125107   1.333431   0.094 0.925942
hp:wt        0.027488   0.008473   3.244 0.003130 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 2.192 on 27 degrees of freedom
Multiple R-squared:  0.8848,    Adjusted R-squared:  0.8677
F-statistic: 51.84 on 4 and 27 DF,  p-value: 2.765e-12

> interact_plot(int_model, pred = "hp", modx = "wt")
Error in interact_plot(int_model, pred = "hp", modx = "wt") :
  could not find function "interact_plot"
>
> install.packages("interact_plot")
WARNING: Rtools is required to build R packages but is not currently installed. Please download and in

https://cran.rstudio.com/bin/windows/Rtools/
Installing package into 'C:/Users/SAYALI/Documents/R/win-library/3.6'
(as 'lib' is unspecified)
Warning in install.packages :
  package 'interact_plot' is not available (for R version 3.6.3)
> interact_plot(int_model, pred = "hp", modx = "wt")
Error in interact_plot(int_model, pred = "hp", modx = "wt") :
```

```
> library(emmeans)
> data(pigs, package="emmeans")
> pigs <- pigs %>% mutate(log_conc=log(conc), percent_fac=factor(percent))
> pigs.lm <- lm(log_conc ~ source + percent_fac, data = pigs)
> summary(pigs.lm)

Call:
lm(formula = log_conc ~ source + percent_fac, data = pigs)

Residuals:
     Min       1Q   Median       3Q      Max
-0.18858 -0.05980 -0.02158  0.08120  0.24601

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)    3.22029    0.05361  60.067  < 2e-16 ***
sourcesoy      0.27277    0.05293   5.153 3.19e-05 ***
sourceskim     0.40228    0.05416   7.428 1.50e-07 ***
percent_fac12  0.17955    0.05608   3.202 0.003960 **
percent_fac15  0.21740    0.05973   3.640 0.001370 **
percent_fac18  0.29985    0.06762   4.434 0.000191 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.1151 on 23 degrees of freedom
Multiple R-squared:  0.7566,    Adjusted R-squared:  0.7037
F-statistic:  14.3 on 5 and 23 DF,  p-value: 2.054e-06

> pigs.emm.s <- emmeans(pigs.lm, "source")
> print(pigs.emm.s)
 source emmean     SE df lower.CL upper.CL
 fish     3.39 0.0367 23     3.32     3.47
 soy      3.67 0.0374 23     3.59     3.74
 skim     3.80 0.0394 23     3.72     3.88

Results are averaged over the levels of: percent_fac
Confidence level used: 0.95
> pigs.emm.p <- emmeans(pigs.lm, "percent_fac")
> print(pigs.emm.p)
 percent_fac emmean     SE df lower.CL upper.CL
 9             3.45 0.0409 23     3.36     3.53
 12            3.62 0.0384 23     3.55     3.70
 15            3.66 0.0437 23     3.57     3.75
 18            3.75 0.0530 23     3.64     3.85

Results are averaged over the levels of: source
Confidence level used: 0.95
> print(emmeans(pigs.lm, ~source*percent_fac))
 source percent_fac emmean     SE df lower.CL upper.CL
 fish   9             3.22 0.0536 23     3.11     3.33
 soy    9             3.49 0.0498 23     3.39     3.60
 skim   9             3.62 0.0501 23     3.52     3.73
 fish   12            3.40 0.0493 23     3.30     3.50
 soy    12            3.67 0.0489 23     3.57     3.77
 skim   12            3.80 0.0494 23     3.70     3.90
 fish   15            3.44 0.0548 23     3.32     3.55
 soy    15            3.71 0.0507 23     3.61     3.82

> pig_pairs <- pairs(pigs.emm.s)
> print(pig_pairs)
 contrast   estimate     SE df t.ratio p.value
 fish - soy   -0.273 0.0529 23 -5.153  0.0001
 fish - skim  -0.402 0.0542 23 -7.428  <.0001
 soy - skim   -0.130 0.0530 23 -2.442  0.0570

Results are averaged over the levels of: percent_fac
P value adjustment: tukey method for comparing a family of 3 estimates
> pigs.emm.s@linfct
     (Intercept) sourcesoy sourceskim percent_fac12 percent_fac15 percent_fac18
[1,]           1         0          0          0.25          0.25          0.25
[2,]           1         1          0          0.25          0.25          0.25
[3,]           1         0          1          0.25          0.25          0.25
> pig_pairs@linfct
     (Intercept) sourcesoy sourceskim percent_fac12 percent_fac15 percent_fac18
[1,]           0        -1          0             0             0             0
[2,]           0         0         -1             0             0             0
[3,]           0         1         -1             0             0             0
> plot(pigs.emm.s, comparisons = TRUE)
> fitiris <- lm(Petal.Length ~ Petal.Width * Species, data = iris)
>
> summary(fitiris)

Call:
lm(formula = Petal.Length ~ Petal.Width * Species, data = iris)

Residuals:
     Min       1Q   Median       3Q      Max
-0.84099 -0.19343 -0.03686  0.16314  1.17065

Coefficients:
                             Estimate Std. Error t value Pr(>|t|)
(Intercept)                    1.3276     0.1309  10.139  < 2e-16 ***
Petal.Width                    0.5465     0.4900   1.115   0.2666
Speciesversicolor              0.4537     0.3737   1.214   0.2267
Speciesvirginica               2.9131     0.4060   7.175 3.53e-11 ***
Petal.Width:Speciesversicolor  1.3228     0.5552   2.382   0.0185 *
Petal.Width:Speciesvirginica   0.1008     0.5248   0.192   0.8480
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.3615 on 144 degrees of freedom
Multiple R-squared:  0.9595,    Adjusted R-squared:  0.9581
F-statistic: 681.9 on 5 and 144 DF,  p-value: < 2.2e-16

> car::Anova(fitiris, type="III")
Anova Table (Type III tests)

Response: Petal.Length
                  Sum Sq  Df  F value    Pr(>F)
(Intercept)        13.4329   1 102.8050 < 2.2e-16 ***
Petal.Width         0.1625   1   1.2438 0.2665889
Species             6.7474   2  25.8196 2.614e-10 ***
Petal.Width:Species  2.0178   2   7.7213 0.0006525 ***
Residuals          18.8156 144
---
```

```
18          18   3   1   1
> org.int <- lm(cbind(sales1, sales2) ~ price1 * price2 + day + store, data = oranges)
> summary(org.int)
Response sales1 :

Call:
lm(formula = sales1 ~ price1 * price2 + day + store, data = oranges)

Residuals:
    Min      1Q  Median      3Q     Max
-7.4646 -2.8230  0.4627  1.5425  6.8493

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)   67.91880   30.97512   2.193  0.03920 *
price1        -1.41364    0.60170  -2.349  0.02819 *
price2        -0.52938    0.68462  -0.773  0.44760
day2           0.19984    2.57232   0.078  0.93878
day3           7.57474    2.52069   3.005  0.00652 **
day4           2.90480    2.52012   1.153  0.26143
day5           9.68284    2.56659   3.773  0.00105 **
day6           5.51804    2.53842   2.174  0.04076 *
store2         1.58480    2.91591   0.544  0.59225
store3        -0.61385    3.01569  -0.204  0.84057
store4         2.54945    3.03622   0.840  0.41012
store5         1.83524    2.86046   0.642  0.52777
store6         7.01866    2.76358   2.540  0.01867 *
price1:price2  0.01369    0.01367   1.001  0.32755
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 4.213 on 22 degrees of freedom
Multiple R-squared:  0.761,     Adjusted R-squared:  0.6197
F-statistic: 5.387 on 13 and 22 DF,  p-value: 0.0002804


Response sales2 :

Call:
lm(formula = sales2 ~ price1 * price2 + day + store, data = oranges)

Residuals:
    Min      1Q  Median      3Q     Max
-8.5004 -1.8640 -0.1674  2.7834  9.9762

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)  -47.88395   38.70597  -1.237  0.22909
price1         1.69081    0.75187   2.249  0.03487 *
price2         0.89372    0.85549   1.045  0.30751
day2          -2.03219    3.21433  -0.632  0.53375
day3          10.10986    3.14981   3.210  0.00404 **
day4           3.96694    3.14910   1.260  0.22097
day5           7.72972    3.20716   2.410  0.02475 *
day6           5.06292    3.17196   1.596  0.12472
store2         6.73894    3.64367   1.849  0.07787 .
store3         2.05515    3.76835   0.545  0.59099
```



Conclusion:

We have seen correlation and Regression examples

Experiment No: 10

Title: Clustering in R K-means.

Date of Performance:

Date of Submission:

Grade:

Signature:

Aim: Clustering in R- K means.

Theory:

What is clustering?
Clustering is a technique of data segmentation that partitions the data into several groups based on their similarity.
Basically, we group the data through a statistical operation. These smaller groups that are formed from the bigger data are known as clusters. These cluster exhibit the following properties:

- They are discovered while carrying out the operation and the knowledge of their number is not known in advance.
- Clusters are the aggregation of similar objects that share common characteristics.

Applications of R clustering are as follows:

- **Marketing –** In the area of marketing, we use clustering to explore and select customers that are potential buyers of the product. This differentiates the most likeable customers from the ones who possess the least tendency to purchase the product. After the clusters have been developed, *businesses can keep a track of their customers and make necessary decisions to retain them in that cluster.*
- **Retail –** Retail industries make use of clustering to group customers based on their preferences, style, choice of wear as well as store preferences. This allows them to manage their stores in a much more efficient manner.
- **Medical Science –** Medicine and health industries make use of clustering algorithms to *facilitate efficient diagnosis and treatment of their patients as well as the discovery of new medicines.* Based on the age, group, genetic coding of the patients, these

organizations are better capable to understand diagnosis through robust clustering.

- **Sociology –** Clustering is used in Data Mining operations to *divide people based on their demographics, lifestyle, socioeconomic status,* etc. This can help the law enforcement agencies to group potential criminals and even identify them with an efficient implementation of the clustering algorithm.

K-Means Clustering in R

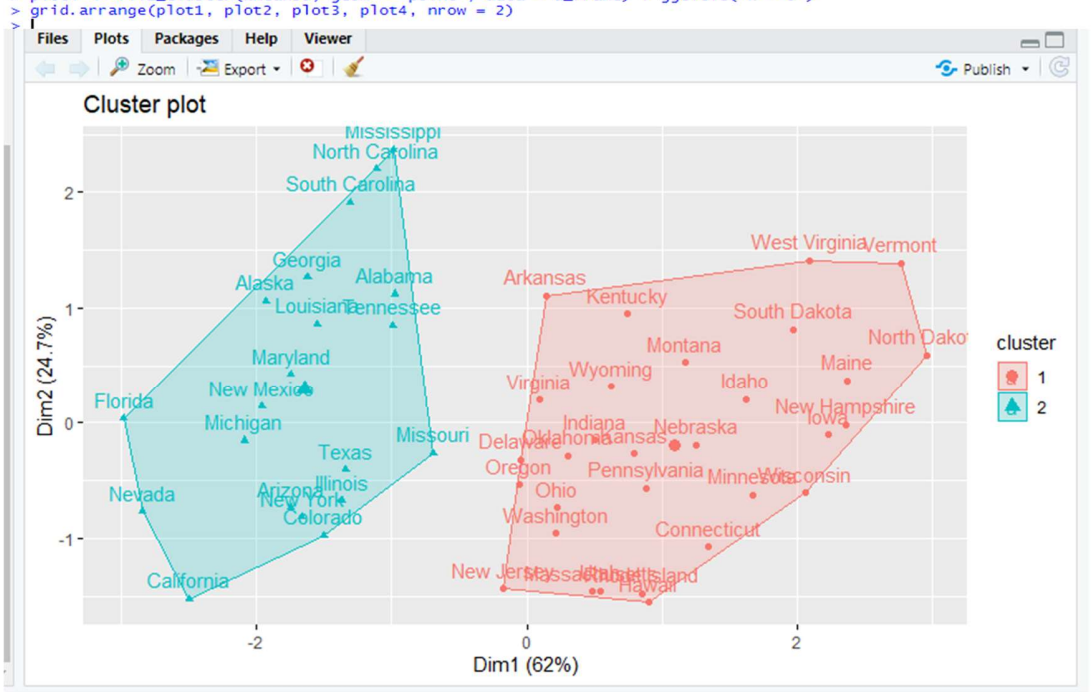
One of the most popular partitioning algorithms in clustering is the K-means cluster analysis in R. It is an unsupervised learning algorithm. It tries to cluster data based on their similarity. Also, we have specified the number of clusters and we want that the data must be grouped into the same clusters. The algorithm assigns each observation to a cluster and also finds the centroid of each cluster.

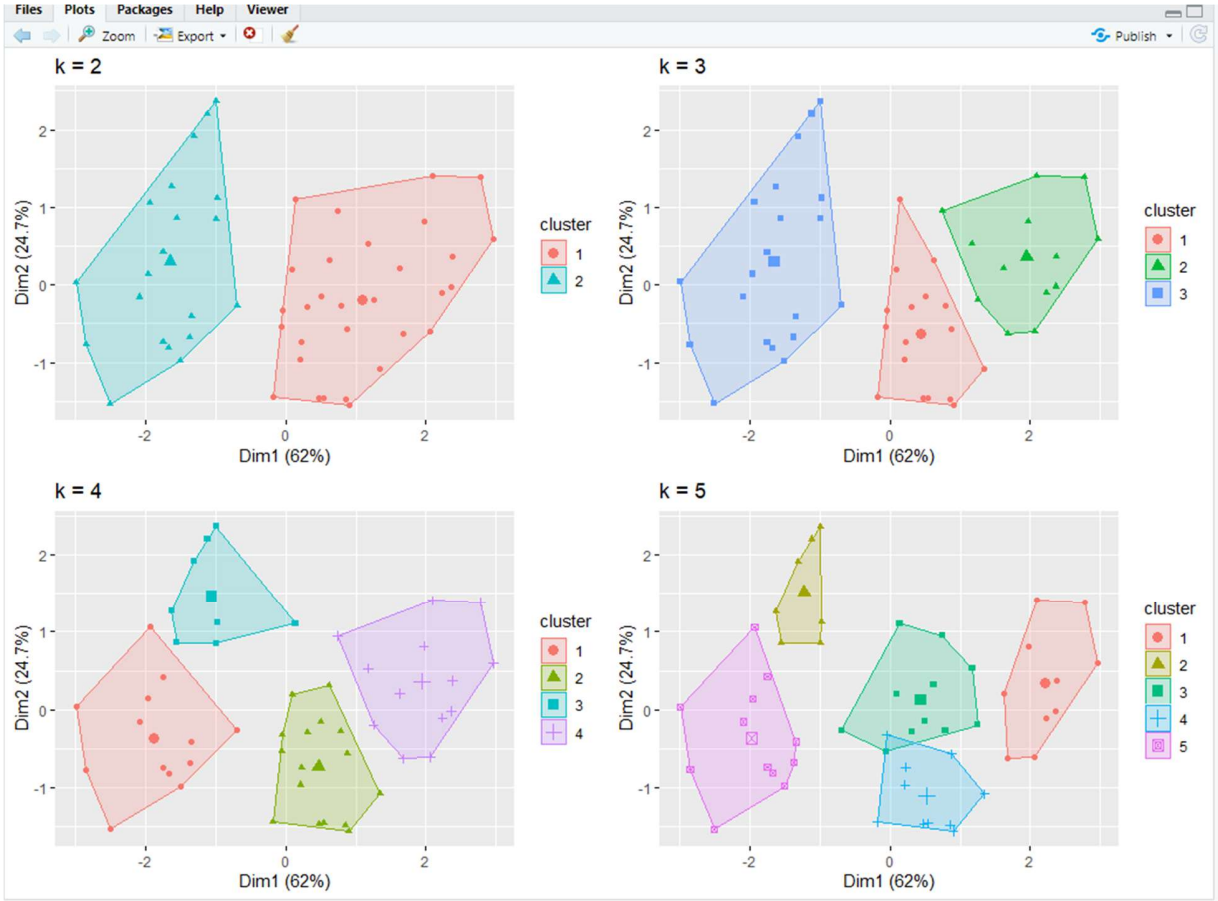
The K-means Algorithm:

- Selects K centroids (K rows chosen at random).
- Then, we have to assign each data point to its closest centroid.
- Moreover, it recalculates the centroids as the average of all data points in a cluster.
- Assigns data points to their closest centroids.
- Moreover, we have to continue steps 3 and 4 until the observations are not reassigned.

Program with Output:

```
v tibble  2.1.3     v dplyr   0.8.5
v tidyr   1.0.2     v stringr 1.4.0
v readr   1.3.1     v forcats 0.5.0
-- Conflicts ------------------------------------------ tidyverse_conflicts() --
x dplyr::filter() masks stats::filter()
x dplyr::lag()    masks stats::lag()
> library(cluster)
> library(factoextra)
Welcome! Want to learn more? See two factoextra-related books at https://goo.gl/ve3WBa
> library(gridExtra)

Attaching package: 'gridExtra'

The following object is masked from 'package:dplyr':

    combine

> data('USArrests')
> d_frame <- USArrests
> d_frame <- na.omit(d_frame)
> d_frame <- scale(d_frame)
> head(d_frame)
              Murder   Assault   UrbanPop         Rape
Alabama    1.24256408 0.7828393 -0.5209066 -0.003416473
Alaska     0.50786248 1.1068225 -1.2117642  2.484202941
Arizona    0.07163341 1.4788032  0.9989801  1.042878388
Arkansas   0.23234938 0.2308680 -1.0735927 -0.184916602
California 0.27826823 1.2628144  1.7589234  2.067820292
Colorado   0.02571456 0.3988593  0.8608085  1.864967207
> kmeans2 <- kmeans(d_frame, centers = 2, nstart = 25)
> str(kmeans2)
List of 9
 $ cluster     : Named int [1:50] 2 2 2 1 2 2 1 1 2 2 ...
  ..- attr(*, "names")= chr [1:50] "Alabama" "Alaska" "Arizona" "Arkansas" ...
 $ centers     : num [1:2, 1:4] -0.67 1.005 -0.676 1.014 -0.132 ...
  ..- attr(*, "dimnames")=List of 2
  .. ..$ : chr [1:2] "1" "2"
  .. ..$ : chr [1:4] "Murder" "Assault" "UrbanPop" "Rape"
 $ totss       : num 196
 $ withinss    : num [1:2] 56.1 46.7
 $ tot.withinss: num 103
 $ betweenss   : num 93.1
 $ size        : int [1:2] 30 20
 $ iter        : int 1
 $ ifault      : int 0
 - attr(*, "class")= chr "kmeans"
> fviz_cluster(kmeans2, data = d_frame
+
+ )
> kmeans3 <- kmeans(d_frame, centers = 3, nstart = 25)
> kmeans4 <- kmeans(d_frame, centers = 4, nstart = 25)
> kmeans5 <- kmeans(d_frame, centers = 5, nstart = 25)
> plot1 <- fviz_cluster(kmeans2, geom = "point", data = d_frame) + ggtitle("k = 2")
> plot2 <- fviz_cluster(kmeans3, geom = "point", data = d_frame) + ggtitle("k = 3")
> plot3 <- fviz_cluster(kmeans4, geom = "point", data = d_frame) + ggtitle("k = 4")
> plot4 <- fviz_cluster(kmeans5, geom = "point", data = d_frame) + ggtitle("k = 5")
> grid.arrange(plot1, plot2, plot3, plot4, nrow = 2)
>
```





Conclusion:

In this experiment we executed the k- means clustering, and studied how the algorithm works.

Experiment No: 11

Title: Classification on large & noisy dataset with R- Logistic regression, Naïve bayes.

Date of Performance:

Date of Submission:

Grade:

Signature:

Aim: Classification on large & noisy dataset with R- Logistic regression, Naïve bayes.

Theory:

What is the new difficulty we come across with this dataset? Let me summarize it:

1. Target variable is categorical binomial, with a very high class imbalance.
2. Size of the data set is fairly large. Training set is 60,000 x 171 and test set is 16,000 x 171.
3. Huge missing values issue.
4. Potential presence of outliers and multicollinearity.
5. There is specific cost associated to type 1 errors and type 2 errors, which requires that we minimize type 2 errors.

Cost-metric of miss-classification:

Predicted class	True class	
	pos	neg
pos	-	Cost_1
neg	Cost_2	-

Cost_1 = 10 and cost_2 = 500

In this case Cost_1 refers to the cost that an unnecessary check needs to be done by an mechanic at an workshop, while Cost_2 refer to the cost of missing a faulty truck, which may cause a breakdown.

```
Total_cost = Cost_1 * No_Instances + Cost_2 * No_Instances
```

Program and Outputs:

Data	
d_frame	num [1:50, 1:4] 1.2426 0.5079 0.0716 0.2323 0.…
dt	10 obs. of 2 variables
full_dataset	12996374 obs. of 2 variables
full_imputed	12996374 obs. of 2 variables
imputed_full	Large mids (21 elements, 1.1 Gb)
issue_columns	1 obs. of 5 variables
kmeans2	List of 9
kmeans3	List of 9
kmeans4	List of 9
kmeans5	List of 9
na_count_full_im…	2 obs. of 1 variable
plot1	List of 9
plot2	List of 9
plot3	List of 9
plot4	List of 9
test_data	2736187 obs. of 1 variable
test_data_bind	2736187 obs. of 2 variables
training_data	10260187 obs. of 1 variable
training_data_bi…	10260187 obs. of 2 variables
USArrests	50 obs. of 4 variables
Values	
full_imputed_fil…	Large factor (12996374 elements, 126.7 Mb)
issue_columns_na…	"set"

```
Error:  \U  used without hex digits in character string starting   C:\U
> training_data <- read.csv("C:/Users/SAYALI/Desktop/aps_failure_training_set.csv")
> test_data <- read.csv("C:/Users/SAYALI/Downloads/aps_failure_test_set.csv")
> dim(training_data)
[1] 10260187        1
> dim(test_data)
[1] 2736187       1
> library(dplyr)
set
> issue_columns <- subset(imputed_full$loggedEvents,
+                         meth == "constant" | meth == "collinear")
> print(issue_columns)
  it im dep     meth out
1  0  0     constant set
> issue_columns_names <- as.character(issue_columns[, "out"])
> issue_columns_names <- issue_columns_names[-2]
> print(issue_columns_names)
[1] "set"
> full_imputed_filtered <- full_imputed[ , !(names(full_imputed) %in%
+                                              issue_columns_names)]
> dim(full_imputed_filtered)
NULL
> training_data_imp <- subset(full_imputed_filtered, set == "TRAIN")
```

```
> library(broom)
> dt %>%
+     map(., summary) %>%
+     map_df(broom::tidy)
# A tibble: 2 x 7
  minimum      q1 median    mean    q3 maximum    na
    <dbl>   <dbl>  <dbl>   <dbl> <dbl>   <dbl> <dbl>
1   -1.19  0.0620  0.465  0.564  0.941    2.20    NA
2   -2.22 -0.454   0.444 -0.0799 1.05     1.15     1
> summary(dt)
       X1                X2
 Min.   :-1.19   Min.   :-2.22
 1st Qu.: 0.06   1st Qu.:-0.45
 Median : 0.47   Median : 0.44
 Mean   : 0.56   Mean   :-0.08
 3rd Qu.: 0.94   3rd Qu.: 1.05
 Max.   : 2.20   Max.   : 1.15
                 NA's   :1
> summary_df_t_2 %>% summarise(Min = mean(Min.),
+                              first_Q = mean(`1st Qu.`),
+                              Median = median(Median),
+                              Mean = mean(Mean),
+                              third_Q = mean(`3rd Qu.`),
+                              Max = max(Max.),
+                              mean_MV = mean(Missing_values),
+                              obs = mean(obs),
+                              mean_MV_perc = mean_MV / obs)
Error in eval(lhs, parent, parent) : object 'summary_df_t_2' not found
> dt %>% summarise(Min = mean(Min.),
+                              first_Q = mean(`1st Qu.`),
+                              Median = median(Median),
+                              Mean = mean(Mean),
+                              third_Q = mean(`3rd Qu.`),
+                              Max = max(Max.),
+                              mean_MV = mean(Missing_values),
+                              obs = mean(obs),
+                              mean_MV_perc = mean_MV / obs)
> training_data_bind <- training_data
> test_data_bind <- test_data
> training_data_bind$set <- "TRAIN"
> test_data_bind$set <- "TEST"
> full_dataset <- rbind(training_data_bind, test_data_bind)
> dim(full_dataset)
[1] 12996374        2
> set.seed(123)
> imputed_full <- mice(full_dataset,
+                      m=1,
+                      maxit = 5,
+                      method = "mean",
+                      seed = 500)

 iter imp variable
  1   1
  2   1
  3   1
  4   1
  5   1
Warning message:
Number of logged events: 1
> )
Error: unexpected ')' in ")"
> imputed_full <- mice(full_dataset,
+                      m=1,
+                      maxit = 5,
+                      method = "mean",
+                      seed = 500)

 iter imp variable
  1   1
  2   1
  3   1
  4   1
  5   1
Warning message:
Number of logged events: 1
> full_imputed <- complete(imputed_full, 1)
> (na_count_full_imputed <-data.frame(sapply(full_imputed, function(y) sum(length(which(is.na(y)))))))
                                                                                    sapply.full_imputed..function.y..sum.length.which.is.na.y.....
This.file.is.part.of.APS.Failure.and.Operational.Data.for.Scania.Trucks.                                                                          0
set                                                                                                                                              0
> issue_columns <- subset(imputed_full$loggedEvents,
+                         meth == "constant" | meth == "collinear")
> print(issue_columns)
  it im dep     meth out
1  0  0     constant set
> issue_columns_names <- as.character(issue_columns[, "out"])
> issue_columns_names <- issue_columns_names[-2]
```

Conclusion:

We have performed classification on a large and noisy dataset with R.