Experiment No. 1

DATTA MEGHE COLLEGE OF ENGINEERING Page No. :
Date :

EXPERIMENT 1

AIM : Installation of R and RStudio in Ubuntu.

THEORY :

R-STUDIO

R-Studio is an integrated development environment for R, a programming language for statistical computing and graphics, it is available in two formats : RStudio Desktop is a regular desktop application while RStudio server runs on a remote server and allows accessing RStudio using a web server.

INSTALLING RSTUDIO IN UBUNTU

There are two ways to install R in ubuntu, one is through the terminal, and the other is through the Ubuntu Software center.

Through Terminal

1. Press Ctrl+Alt+T to open terminal
2. Then execute sudo apt-get update

$ sudo apt-get update

3. After that, sudo apt-get install r-base

$ sudo apt-get install r-base

To run R statistical package, execute R in the terminal.

$R

Through Ubuntu Software center

1. Open Ubuntu Software center
2. Search for r-base.
3. Click install
4. Then run R by executing R in terminal

$R

To install RStudio IDE, do the following:

1. Go to RStudio IDE download page
2. Click Download RStudio Desktop.
3. Then click for the download link recommended for your system.
4. Run the downloaded file to start the set up wizard.
5. Click "Next" until "Finish".

To open RStudio IDE, search for RStudio in the Ubuntu dash.

To install RKWard KDE, do the following:

1. Open Ubuntu Software Centre
2. Search for RKWard
3. Then click Install.

CONSOLE	EDITOR
HELP / PLOTS HELP / PIGS	ENVIRONMENT

CONCLUSION: We succesfully installed R-Studio in Ubuntu.

Output:

Conclusion : Thus we have installed R and Rstudio.

Experiment No. 2

DATTA MEGHE COLLEGE OF ENGINEERING Page No. : Date :

EXPERIMENT 2

AIM: Basic functionality of R-variables, basic arithmatics, help commands.

THEORY:

VARIABLES IN R

A variable provides us with named storage that our programs can manipulat 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 of underline characters

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

```
var 1 = c(0, 1, 2, 3)
var 2 <- c("learn", "R")
c(TRUE, 1) -> var 3
```

Data Type of R
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.

Finding Variables
To know all the variables currently available in workspace we use the ls() function.

```
print (ls())
```

Deleting variables
Variables can be deleted by rm() function.

```
rm (var, 3)
```

BASIC OPERATORS IN RSTUDIO

R Arithmatic Operators

+	Addition
-	Subtraction
*	Multiplication
/	Division
^	Exponent
%%	Modulus
%/%	Integer Division

R Relational Operators

<	Less than
>	Greater than
<=	Less than or equal to
>=	Greater than or equal to
==	Equal to
!=	Not Equal to

R Logical operators

!	logical NOT
&	Element-wise Logical AND
&&	Logical AND
|	Element wise logical OR
||	Logical OR

BASIC COMMANDS IN R

help() - Obtain documentation for a given R command

help(mean)

example() - view some examples on the use of a command.

example(Histogram)

c(), scan() - Enter data Manually to vector R

K = c(8, 6, 7, 4, 5)

seq() - Make arithmatic progression vector

y = seq(7, 41, 1.5)

view() - View data in spreadsheet format.

CONCLUSION: We learned, understood and practised with various variables, operators and basic commands in R.

Output :

Code :

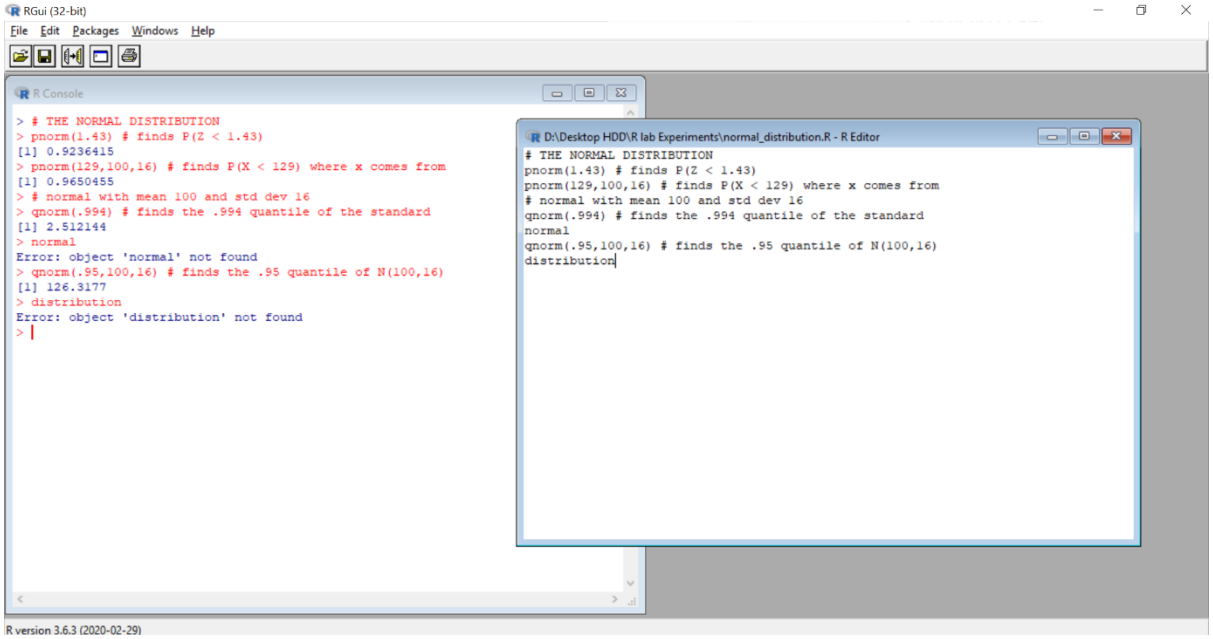
```
# BASIC ARITHMETIC OPERATORS
2-5 # subtraction
6/3 # division
3+2*5 # note order of operations exists
(3+2)*5 # if you need, force operations using
# redundant parentheses
4^3 # raise to a power is ^
exp(4) # e^4 = 54.598 (give or take)
log(2.742) # natural log of 2.74
log10(1000) # common log of 1000
pi # 3.14159...



# VARIABLE NAMES
x <- 5
x # 1 way to print out contents of a variable
var1 <- 7/2
print(var1) # another way to print contents
valid.variable.name <- 18.6
# you can even have long variable names
# if you are of that kind of weirdness
valid.variable.name



a <- 3 # This stores the value 3 into variable a
a
sqrt(a) # square root of a (which is 3 at this moment)
a^4 # a raised to the fourth power
log(a) # natural log of a
log10(a) # common log of a
exp(a) # e to the power a
tan(a) # tangent of a
b <- pi # pie, the number approx 3.14159
b



# VECTORS
x <- c(2,3,5,1,4,4) # create a row vector with those 6 numbers
# and then call the vector x
x
sum(x) # sums the elements of vector x
mean(x) # finds the mean of vector x
```

```
sd(x) # finds standard deviation of vector x
median(x) # finds the median of x
sqrt(x) # finds square root of every element of x
x^2 # finds the square of every element of x
seq(1,10) # makes a vector 10 long, from 1 to 10
seq(1,10, 2) # makes a vector from 1 to 10, starting with 1, then skipping 1 ending up with only odds from 1 to 10
seq(1:10) # same as seq(1,10)
seq(1,10,by=2) # same as seq(1,10,2)
y <- c(1:7) # create a row vector with elements 1 through 6
y
z <- 1:7 # same as y--use when you increment by 1
z
w <- c(1:12,0,-6) # use the c( when you increment by 1 followed by more numbers not in sequence
w


# SOME OPERATIONS WITH NUMERICAL VECTORS AND LOGICAL VECTORS
x <- c(-5,0,7,-6,14,27) # data vector x
x[1] # prints first element of x
x[length(x)] # prints last element of x
i<-3
x[i] # the ith entry if 1<=i<=n, NA if i>n, all but the ith if -n<=i<=-1, an error if i < -n, and an empty vector if i=0
x[c(2,3)] # the second and 3rd entries
x[-c(2,3)] # all but the second and third entries
x[i] <- 5 # assign a value of 5 to the first entry; also can use x[i] = 5
x[c(1,4)] <- c(2,3) # assign values to the first and fourth entries
y
x[indices] <- y # assign to those indices indicated by the values
# of INDICES: if y is not long enough, values
# are recycled; if y is too long, just its initial
# values are used and a warning is issued
x
x < 3 # vector with length n of TRUE or FALSE # depending if x[i] < 3 which(x<3) which indices correspond to the TRUE values of x<3
x[x<3] # the x values when x<3 is TRUE -- same as x[which(x<3)]


# INTRO TO GRAPHING
# input a vector x with data
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)
# input vector y of consecutive numbers
# 1 through 29
y <- c(1:29)
# make a histogram of x
hist(x)
# make a scatter plot of y vs x
```

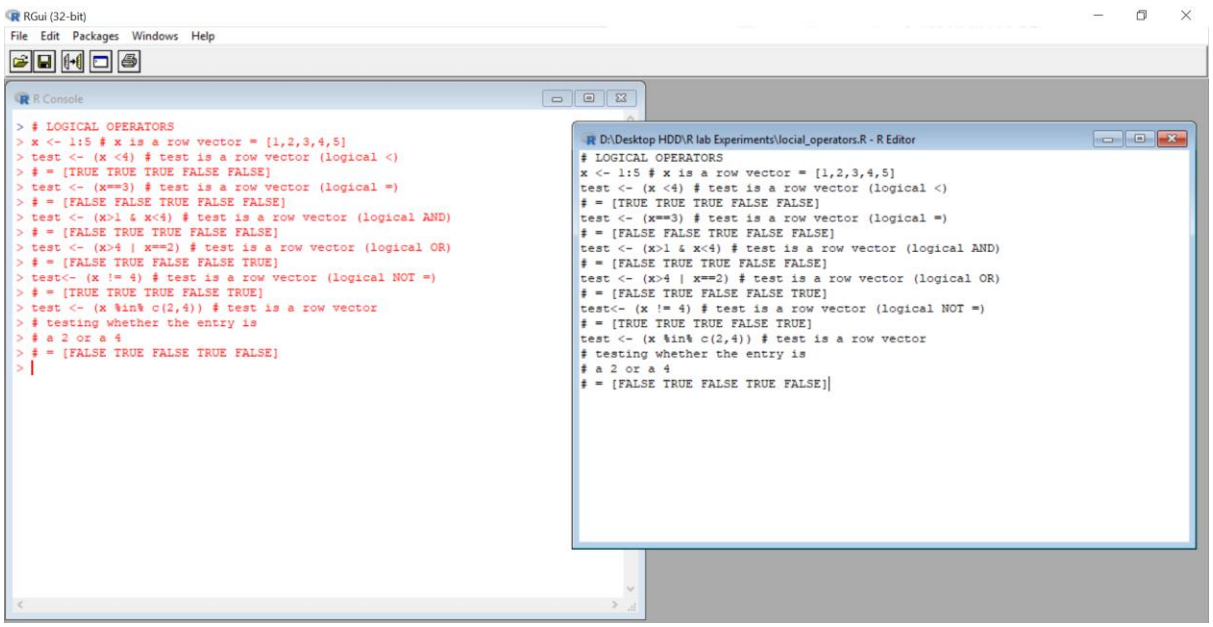
```
plot(x,y)
# make a time plot of x, connecting dots
plot(x, type="b")
# make a histogram of a sampling of 20 from
# the binomial distribution B(12,.4)
y <- rbinom(20, 12, .4)
hist(y)
# make another binomial histogram, but using
a
# sample of 200
y <- rbinom(200, 12, .4)
hist(y)
# make a boxplot of x data
boxplot(x)
# make side by side boxplots of x and y data
boxplot(x,y)



#LOGICAL OPERATORS
x <- 1:5 # x is a row vector = [1,2,3,4,5]
test <- (x <4) # test is a row vector (logical <)
# = [TRUE TRUE TRUE FALSE FALSE]
test <- (x==3) # test is a row vector (logical =)
# = [FALSE FALSE TRUE FALSE FALSE]
test <- (x>1 & x<4) # test is a row vector (logical AND)
# = [FALSE TRUE TRUE FALSE FALSE]
test <- (x>4 | x==2) # test is a row vector (logical OR)
# = [FALSE TRUE FALSE FALSE TRUE]
test<- (x != 4) # test is a row vector (logical NOT =)
# = [TRUE TRUE TRUE FALSE TRUE]
test <- (x %in% c(2,4)) # test is a row vector
# testing whether the entry is
# a 2 or a 4
# = [FALSE TRUE FALSE TRUE FALSE]



# THE NORMAL DISTRIBUTION
pnorm(1.43) # finds P(Z < 1.43)
pnorm(129,100,16) # finds P(X < 129) where x comes from
# normal with mean 100 and std dev 16
qnorm(.994) # finds the .994 quantile of the standard normal
qnorm(.95,100,16) # finds the .95 quantile of N(100,16) distribution



# THINGS TO KNOW
x <- runif(30,0,1) # creates a vector 30 long
x # of random values from U(0,1)
y <- c() # before you can use a vector in aloop
```

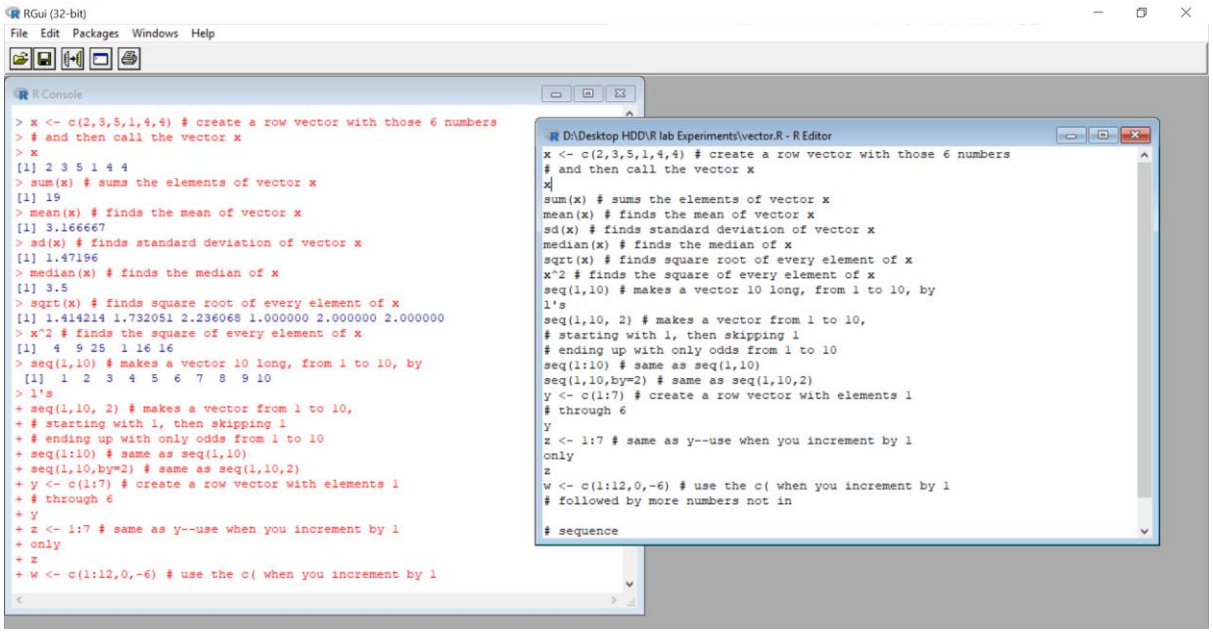
```
for (i in 1:10) { # you must "save space" for it by
  y[i] <- rbinom(1,3,.2) # typing the y <- c() command
}
y
s <- c(1:5) # for R you can use <- or =
s # for a variable assignment
t = c(2:6)
t
help(mean)
```

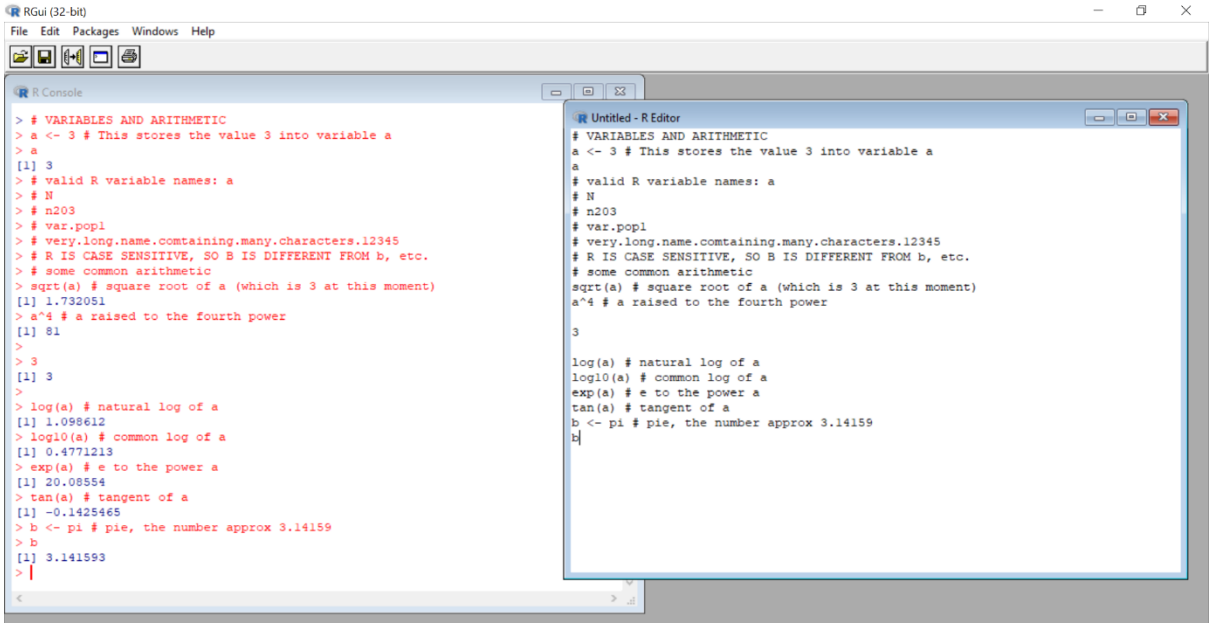
Screenshots :



R version 3.6.3 (2020-02-29)

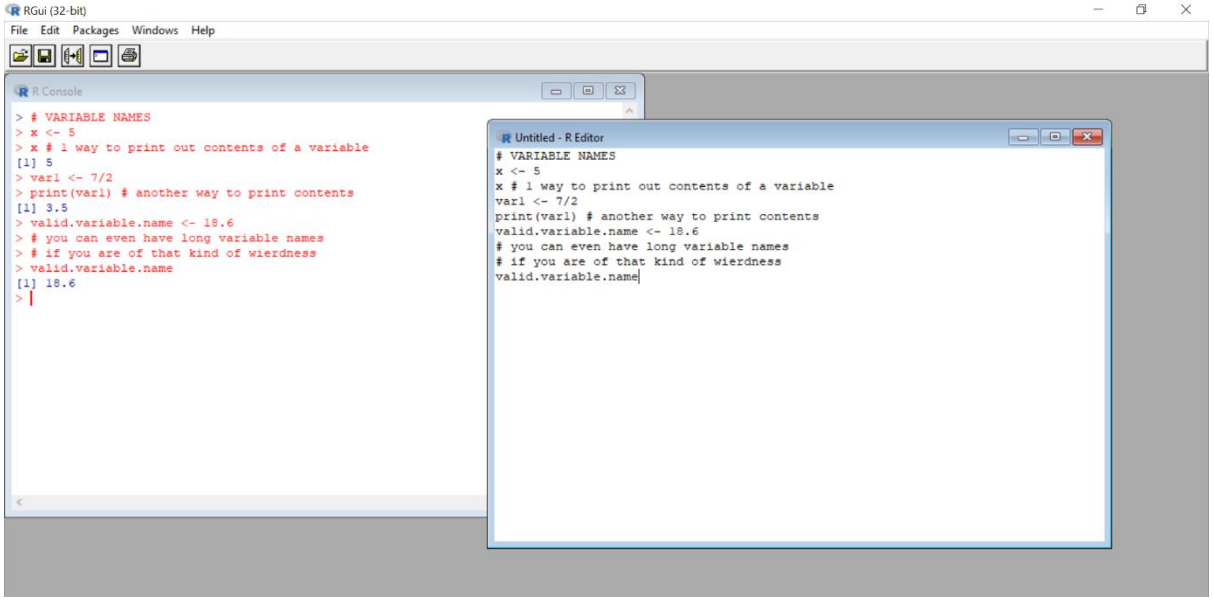


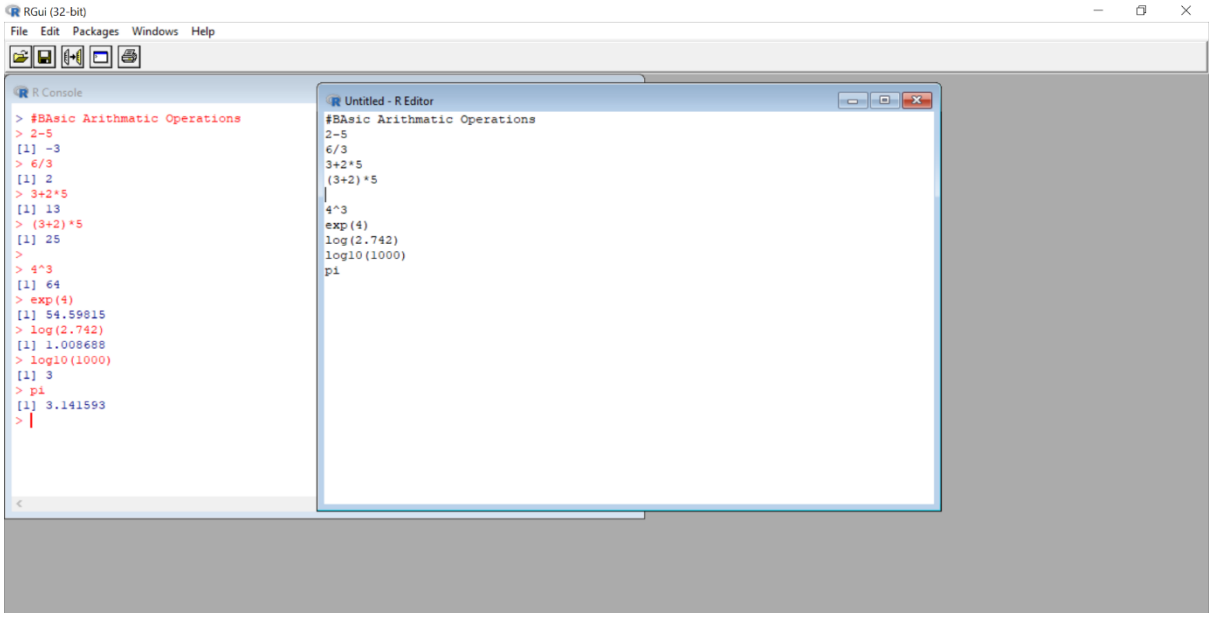

RGui (32-bit)
File Edit Packages Windows Help
R Console
> x <- c(2,3,5,1,4,4) # create a row vector with those 6 numbers
> # and then call the vector x
> x
[1] 2 3 5 1 4 4
> sum(x) # sums the elements of vector x
[1] 19
> mean(x) # finds the mean of vector x
[1] 3.166667
> sd(x) # finds standard deviation of vector x
[1] 1.47196
> median(x) # finds the median of x
[1] 3.5
> sqrt(x) # finds square root of every element of x
[1] 1.414214 1.732051 2.236068 1.000000 2.000000 2.000000
> x^2 # finds the square of every element of x
[1] 4 9 25 1 16 16
> seq(1,10) # makes a vector 10 long, from 1 to 10, by
[1] 1 2 3 4 5 6 7 8 9 10
> 1's
+ seq(1,10, 2) # makes a vector from 1 to 10,
+ # starting with 1, then skipping 1
+ # ending up with only odds from 1 to 10
+ seq(1:10) # same as seq(1,10)
+ seq(1,10,by=2) # same as seq(1,10,2)
+ y <- c(1:7) # create a row vector with elements 1
+ # through 6
+ y
+ z <- 1:7 # same as y--use when you increment by 1
+ only
+ z
+ w <- c(1:12,0,-6) # use the c(when you increment by 1
D:\Desktop HDD\R lab Experiments\vector.R - R Editor
x <- c(2,3,5,1,4,4) # create a row vector with those 6 numbers
# and then call the vector x
x
sum(x) # sums the elements of vector x
mean(x) # finds the mean of vector x
sd(x) # finds standard deviation of vector x
median(x) # finds the median of x
sqrt(x) # finds square root of every element of x
x^2 # finds the square of every element of x
seq(1,10) # makes a vector 10 long, from 1 to 10, by
1's
seq(1,10, 2) # makes a vector from 1 to 10,
# starting with 1, then skipping 1
# ending up with only odds from 1 to 10
seq(1:10) # same as seq(1,10)
seq(1,10,by=2) # same as seq(1,10,2)
y <- c(1:7) # create a row vector with elements 1
# through 6
y
z <- 1:7 # same as y--use when you increment by 1
only
z
w <- c(1:12,0,-6) # use the c(when you increment by 1
# followed by more numbers not in
# sequence

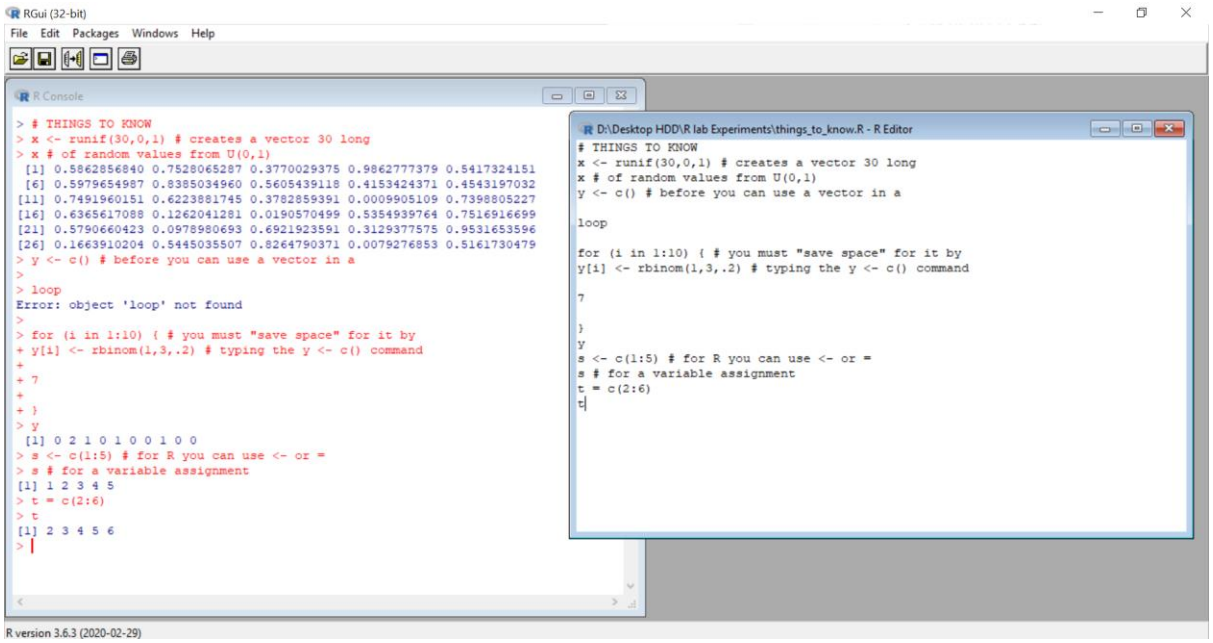

RGui (32-bit)
File Edit Packages Windows Help
R Console
> # VARIABLES AND ARITHMETIC
> a <- 3 # This stores the value 3 into variable a
> a
[1] 3
> # valid R variable names: a
> # N
> # n203
> # var.pop1
> # very.long.name.comtaining.many.characters.12345
> # R IS CASE SENSITIVE, SO B IS DIFFERENT FROM b, etc.
> # some common arithmetic
> sqrt(a) # square root of a (which is 3 at this moment)
[1] 1.732051
> a^4 # a raised to the fourth power
[1] 81
>
> 3
[1] 3
>
> log(a) # natural log of a
[1] 1.098612
> log10(a) # common log of a
[1] 0.4771213
> exp(a) # e to the power a
[1] 20.08554
> tan(a) # tangent of a
[1] -0.1425465
> b <- pi # pie, the number approx 3.14159
> b
[1] 3.141593
>
Untitled - R Editor
# VARIABLES AND ARITHMETIC
a <- 3 # This stores the value 3 into variable a
a
# valid R variable names: a
# N
# n203
# var.pop1
# very.long.name.comtaining.many.characters.12345
# R IS CASE SENSITIVE, SO B IS DIFFERENT FROM b, etc.
# some common arithmetic
sqrt(a) # square root of a (which is 3 at this moment)
a^4 # a raised to the fourth power

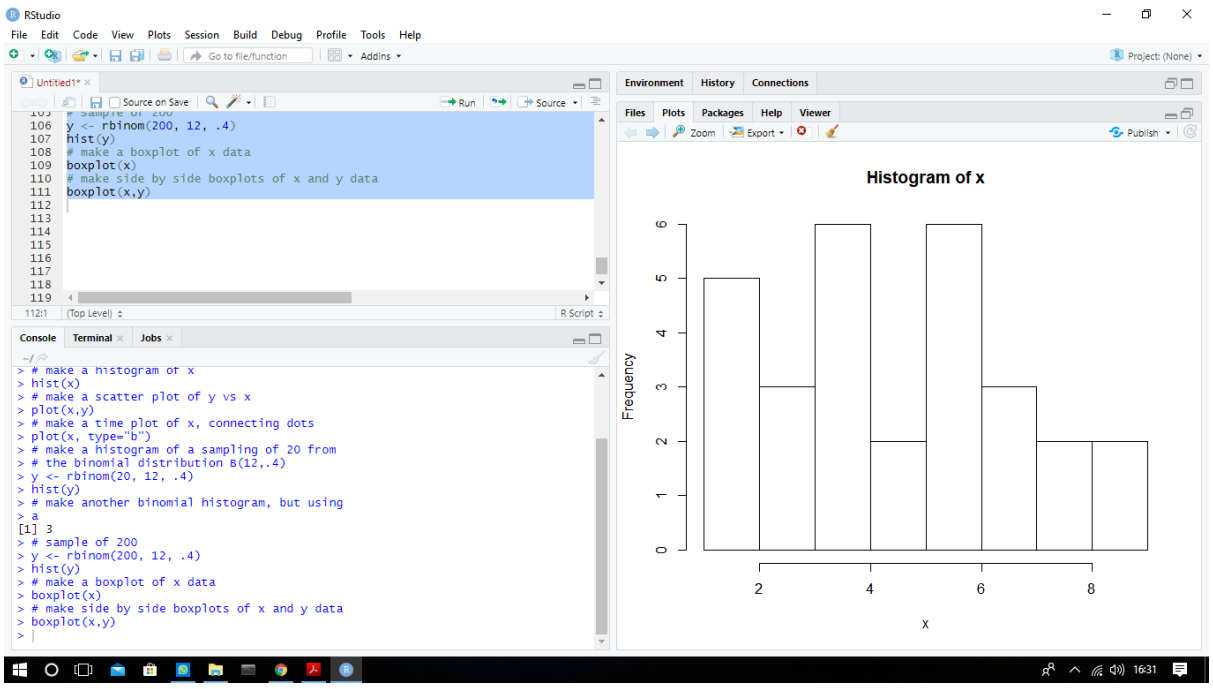
3

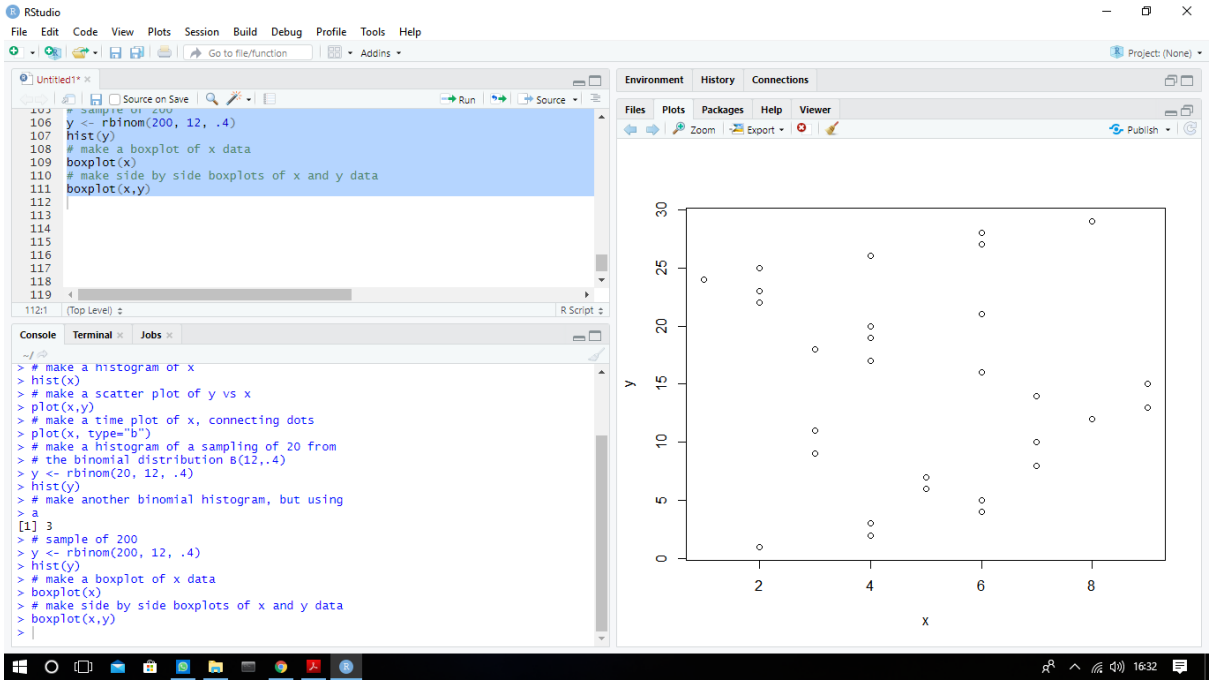
log(a) # natural log of a
log10(a) # common log of a
exp(a) # e to the power a
tan(a) # tangent of a
b <- pi # pie, the number approx 3.14159
b

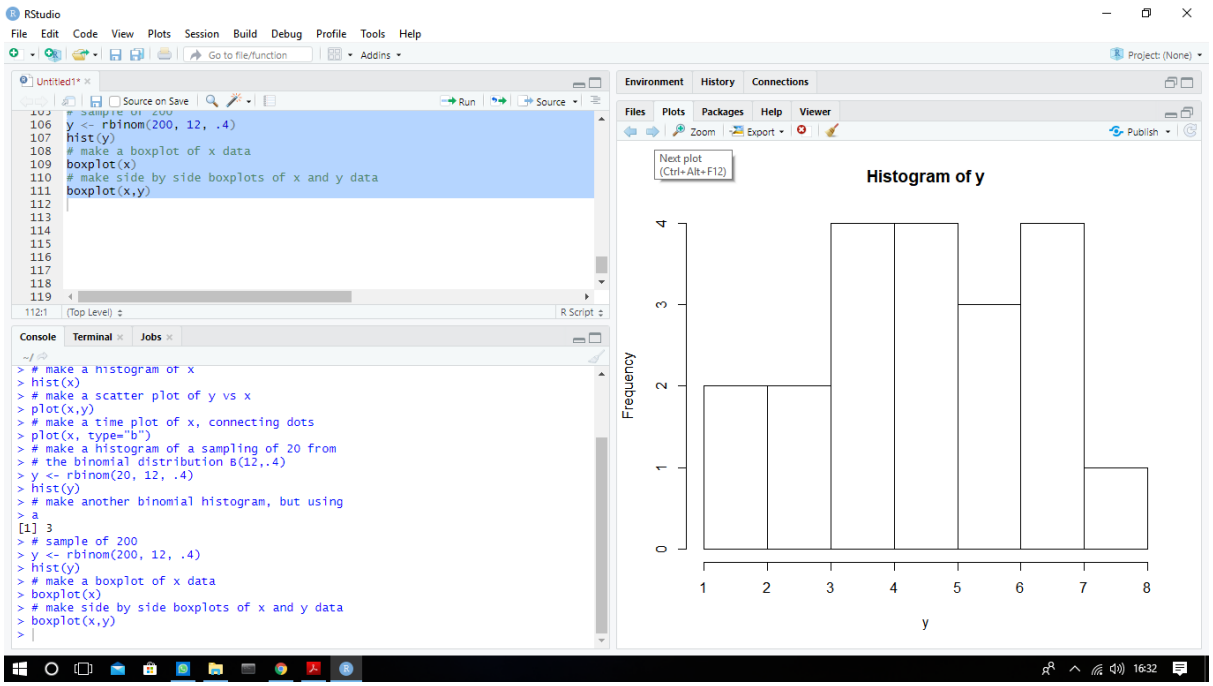

RGui (32-bit)
File Edit Packages Windows Help
R Console
> # VARIABLE NAMES
> x <- 5
> x # 1 way to print out contents of a variable
[1] 5
> var1 <- 7/2
> print(var1) # another way to print contents
[1] 3.5
> valid.variable.name <- 18.6
> # you can even have long variable names
> # if you are of that kind of wierdness
> valid.variable.name
[1] 18.6
>
Untitled - R Editor
# VARIABLE NAMES
x <- 5
x # 1 way to print out contents of a variable
var1 <- 7/2
print(var1) # another way to print contents
valid.variable.name <- 18.6
# you can even have long variable names
# if you are of that kind of wierdness
valid.variable.name

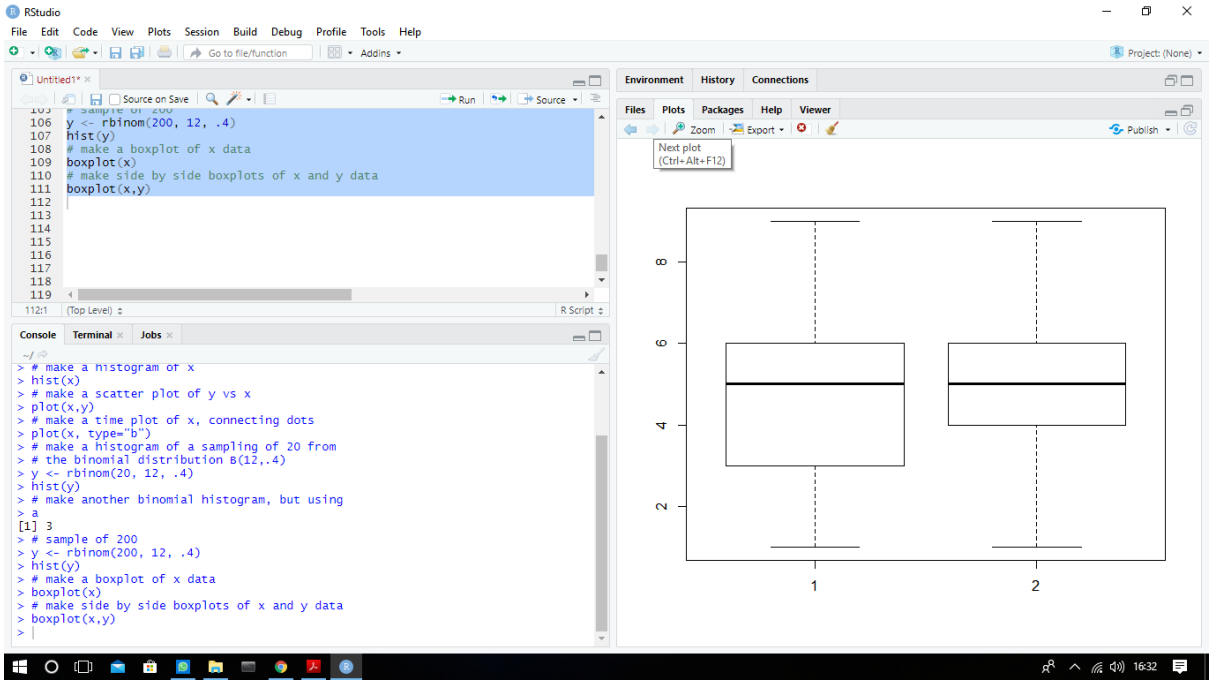

RGui (32-bit)
File Edit Packages Windows Help
R Console
> #BAsic Arithmatic Operations
> 2-5
[1] -3
> 6/3
[1] 2
> 3+2*5
[1] 13
> (3+2)*5
[1] 25
>
> 4^3
[1] 64
> exp(4)
[1] 54.59815
> log(2.742)
[1] 1.008688
> log10(1000)
[1] 3
> pi
[1] 3.141593
>
Untitled - R Editor
#BAsic Arithmatic Operations
2-5
6/3
3+2*5
(3+2)*5
4^3
exp(4)
log(2.742)
log10(1000)
pi

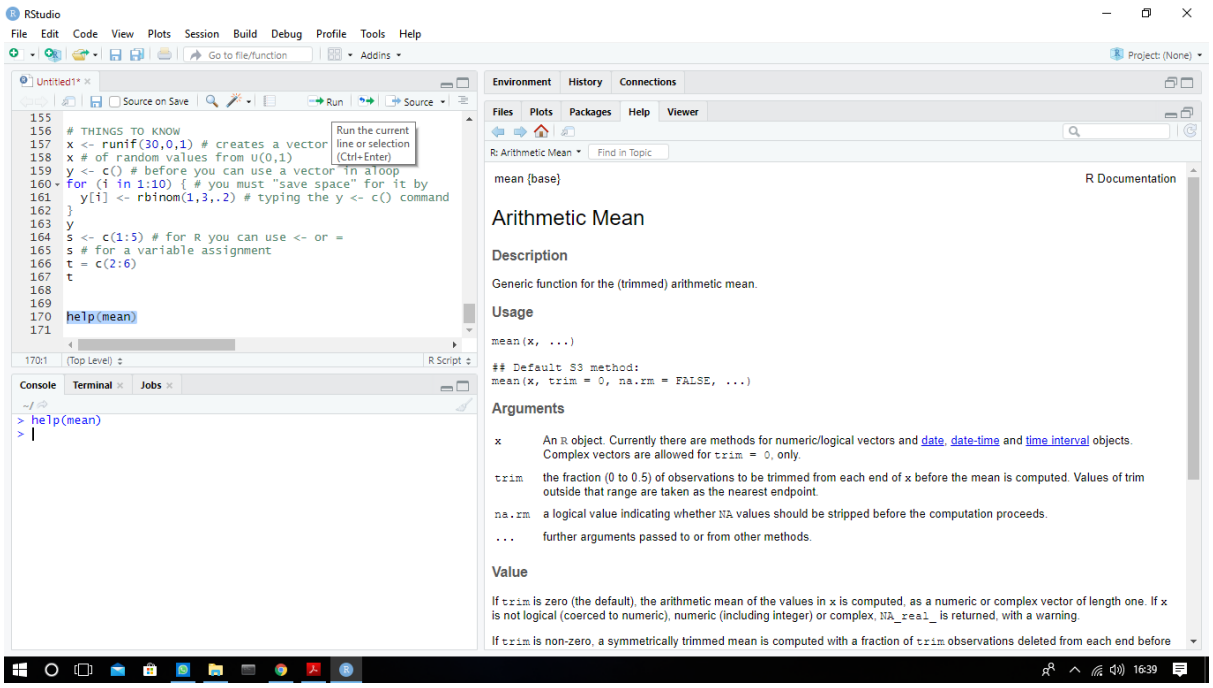

RGui (32-bit)
File Edit Packages Windows Help
R Console
> # THINGS TO KNOW
> x <- runif(30,0,1) # creates a vector 30 long
> x # of random values from U(0,1)
[1] 0.5862856840 0.7528065287 0.3770029375 0.9862777379 0.5417324151
[6] 0.5979654987 0.8385034960 0.5605439118 0.4153424371 0.4543197032
[11] 0.7491960151 0.6223881745 0.3782859391 0.0009905109 0.7398805227
[16] 0.6365617088 0.1262041281 0.0190570499 0.5354939764 0.7516916699
[21] 0.5790660423 0.0978980693 0.6921923591 0.3129377575 0.9531653596
[26] 0.1663910204 0.5445035507 0.8264790371 0.0079276853 0.5161730479
> y <- c() # before you can use a vector in a
>
> loop
Error: object 'loop' not found
>
> for (i in 1:10) { # you must "save space" for it by
+ y[i] <- rbinom(1,3,.2) # typing the y <- c() command
+
+ 7
+
+ }
> y
[1] 0 2 1 0 1 0 0 1 0 0
> s <- c(1:5) # for R you can use <- or =
> s # for a variable assignment
[1] 1 2 3 4 5
> t = c(2:6)
> t
[1] 2 3 4 5 6
>
D:\Desktop HDD\R lab Experiments\things_to_know.R - R Editor
# THINGS TO KNOW
x <- runif(30,0,1) # creates a vector 30 long
x # of random values from U(0,1)
y <- c() # before you can use a vector in a
loop
for (i in 1:10) { # you must "save space" for it by
y[i] <- rbinom(1,3,.2) # typing the y <- c() command
7
}
y
s <- c(1:5) # for R you can use <- or =
s # for a variable assignment
t = c(2:6)
t
R version 3.6.3 (2020-02-29)


RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
Untitled1*
106 y <- rbinom(200, 12, .4)
107 hist(y)
108 # make a boxplot of x data
109 boxplot(x)
110 # make side by side boxplots of x and y data
111 boxplot(x,y)
Console Terminal Jobs
> # make a histogram of x
> hist(x)
> # make a scatter plot of y vs x
> plot(x,y)
> # make a time plot of x, connecting dots
> plot(x, type="b")
> # make a histogram of a sampling of 20 from
> # the binomial distribution B(12,.4)
> y <- rbinom(20, 12, .4)
> hist(y)
> # make another binomial histogram, but using
> a
[1] 3
> # sample of 200
> y <- rbinom(200, 12, .4)
> hist(y)
> # make a boxplot of x data
> boxplot(x)
> # make side by side boxplots of x and y data
> boxplot(x,y)
Environment History Connections
Files Plots Packages Help Viewer
Histogram of x
Frequency
x
0 1 2 3 4 5 6
2 4 6 8


RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
Untitled1*
106 y <- rbinom(200, 12, .4)
107 hist(y)
108 # make a boxplot of x data
109 boxplot(x)
110 # make side by side boxplots of x and y data
111 boxplot(x,y)
Console Terminal Jobs
> # make a histogram of x
> hist(x)
> # make a scatter plot of y vs x
> plot(x,y)
> # make a time plot of x, connecting dots
> plot(x, type="b")
> # make a histogram of a sampling of 20 from
> # the binomial distribution B(12,.4)
> y <- rbinom(20, 12, .4)
> hist(y)
> # make another binomial histogram, but using
> a
[1] 3
> # sample of 200
> y <- rbinom(200, 12, .4)
> hist(y)
> # make a boxplot of x data
> boxplot(x)
> # make side by side boxplots of x and y data
> boxplot(x,y)
Environment History Connections
Files Plots Packages Help Viewer
y
x
16:32


RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
Untitled1*
106 y <- rbinom(200, 12, .4)
107 hist(y)
108 # make a boxplot of x data
109 boxplot(x)
110 # make side by side boxplots of x and y data
111 boxplot(x,y)
Console Terminal Jobs
> # make a histogram of x
> hist(x)
> # make a scatter plot of y vs x
> plot(x,y)
> # make a time plot of x, connecting dots
> plot(x, type="b")
> # make a histogram of a sampling of 20 from
> # the binomial distribution B(12,.4)
> y <- rbinom(20, 12, .4)
> hist(y)
> # make another binomial histogram, but using
> a
[1] 3
> # sample of 200
> y <- rbinom(200, 12, .4)
> hist(y)
> # make a boxplot of x data
> boxplot(x)
> # make side by side boxplots of x and y data
> boxplot(x,y)
Environment History Connections
Files Plots Packages Help Viewer
Next plot
(Ctrl+Alt+F12)
Histogram of y
Frequency
y
16:32





Conclusion : Thus we have studied the basic functionality of R.

Experiment No. 3

DATTA MEGHE COLLEGE OF ENGINEERING Page No. : Date :

EXPERIMENT 3

AIM : Basic data types and data structures in R-vector, metrics, list, dataframe.

THEORY :

BASIC DATA TYPES AND DATA STRUCTURES

Everything in R is an object

R has 6 basic data type

- Character
- numeric
- integer
- logical
- complex

R provides may functions to examine features of vectors and other objects

- class() - what kind of object it is?
- type of() - What is the object's data type
- length() - How long is it?
- attributes - does it have any metadata?

R has many data structures

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

VECTORS

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

THE DIFFERENT VECTOR MODES :

A vector is a collection of elements that are mostly commonly of mode character, logical, integer or numeric

You can create empty vector with vector(). It is more common to use direct constructors such as character(), numeric(), etc.

EXAMINING VECTOR OUTPUT

The functions typeof(), length(), class() and str() provide useful information about your vectors and R objects in general.

The function c() can also be used to add elements to a vector.

```
typeof(z)
length(z)
```

```
class(z)
str(z)

z ← c(z, "Annette")
```

OTHER SPECIAL VALUES

Inf is infinity. You can have either positive or negative infinity.

```
1/0
[1] Inf
```

NaN means not a number. It's an undefined value.

```
0/0
[1] NaN
```

CONCLUSION : We studied about data structures, and vectors and executed commands regarding the same. Outputs were observed carefully.

Output :

Code :

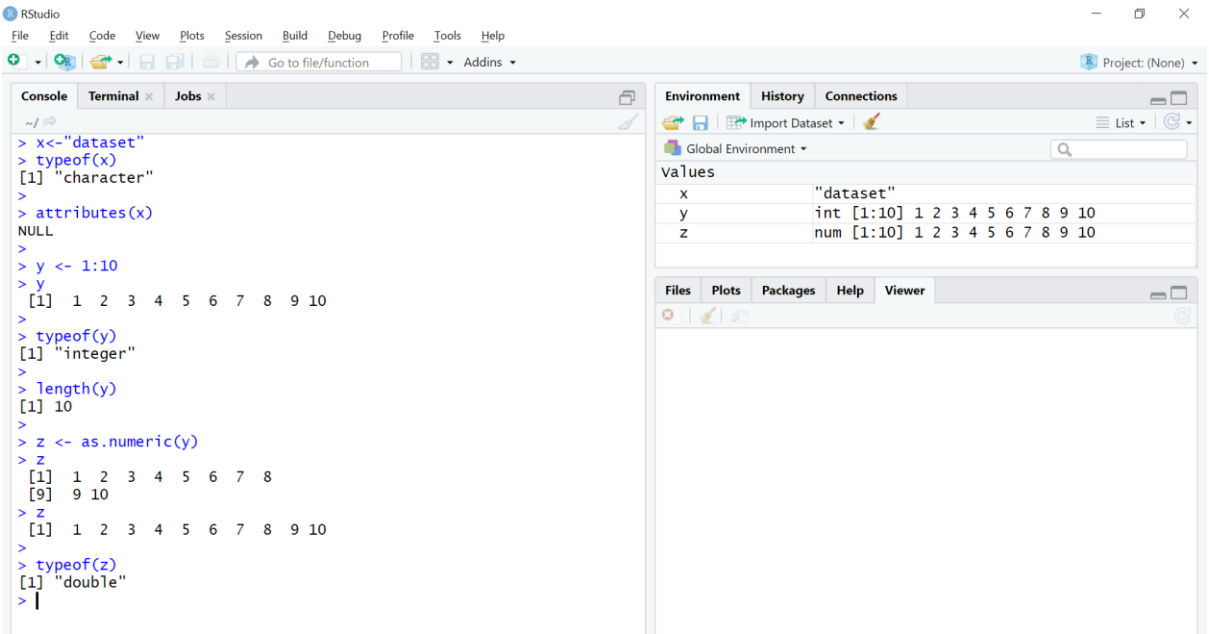
```
x <- "dataset"
typeof(x)
attributes(x)
y <- 1:10
y
typeof(y)
length(y)
z <- as.numeric(y)
z
typeof(z)
vector()
vector("character", length = 5) # a vector of mode 'character' with 5 elements
character(5) # the same thing, but using the constructor directly
numeric(5)   # a numeric vector with 5 elements
logical(5)   # a logical vector with 5 elements
x <- c(1, 2, 3)
x1 <- c(1L, 2L, 3L)
y <- c(TRUE, TRUE, FALSE, FALSE)
z <- c("Sarah", "Tracy", "Jon")
typeof(z)
length(z)
class(z)
str(z)
z <- c(z, "Annette")
z
z <- c("Greg", z)
z
series <- 1:10
seq(10)
seq(from = 1, to = 10, by = 0.1)
x <- c(0.5, NA, 0.7)
x
x <- c(TRUE, FALSE, NA)
x
x <- c("a", NA, "c", "d", "e")
x
x <- c(1+5i, 2-3i, NA)
x
x <- c("a", NA, "c", "d", NA)
y <- c("a", "b", "c", "d", "e")
```

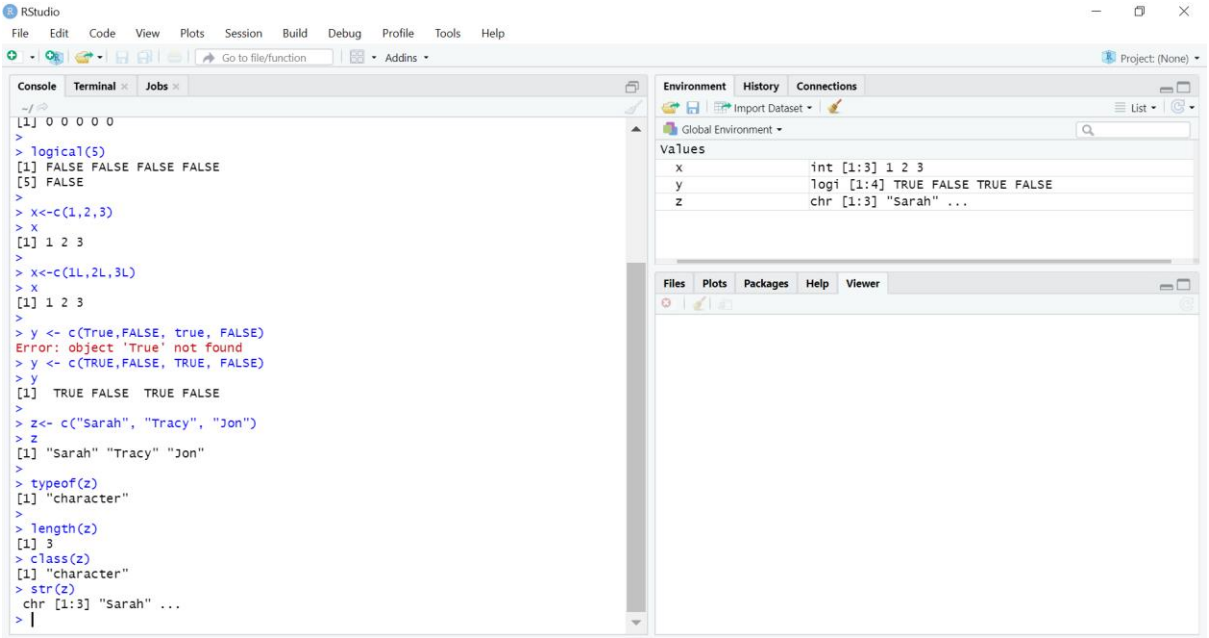
```
is.na(x)
is.na(y)
anyNA(x)
anyNA(y)
1/0
0/0
xx <- c(1.7, "a")
xx <- c(TRUE, 2)
xx <- c("a", TRUE)
as.numeric("1") #You can also control how vectors are coerced explicitly using the as.<class_name>() functions:
as.character(1:2)
length(1:10) #to get length of list
nchar("Software Carpentry") #to get length of a character string
m <- matrix(nrow = 2, ncol = 2)
m
dim(m)
m <- matrix(c(1:3))
class(m)
typeof(m)
FOURS <- matrix(
  c(4, 4, 4, 4),
  nrow = 2,
  ncol = 2)
typeof(FOURS[1])
typeof(FOURS)
m <- matrix(1:6, nrow = 2, ncol = 3)
m      <- 1:10 #another way to make a matrix
dim(m) <- c(2, 5)
x <- 1:3
y <- 10:12
cbind(x, y)
rbind(x, y)
mdat <- matrix(c(1, 2, 3, 11, 12, 13),
               nrow = 2,
               ncol = 3,
               byrow = TRUE)
mdat
mdat[2, 3]
x <- list(1, "a", TRUE, 1+4i)
x
x <- vector("list", length = 5) # empty list
length(x)
x[[1]]
```

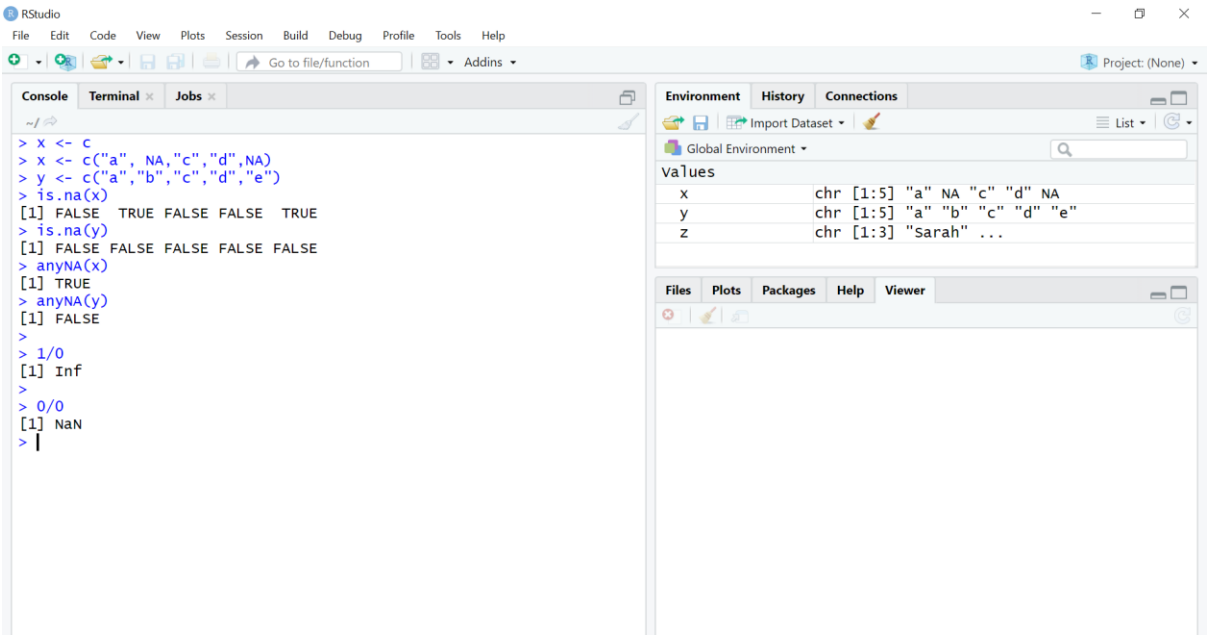
#Vectors can be coerced to lists as follows:

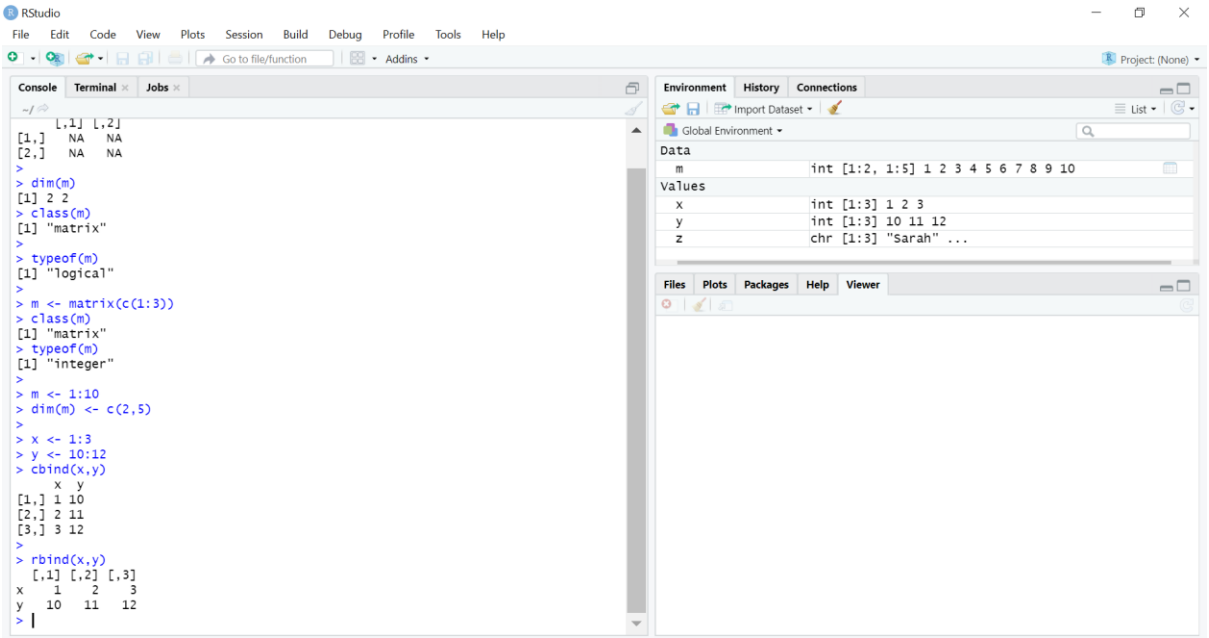
```
x <- 1:10
x <- as.list(x)
length(x)
xlist <- list(a = "Karthik Ram", b = 1:10, data = head(iris)) #Elements of a list can be named (i.e. lists can have the names attribute)
xlist
names(xlist)

dat <- data.frame(id = letters[1:10], x = 1:10, y = 11:20) #To create data frames by hand
dat
is.list(dat) #data frame is a special list
class(dat)
dat[1, 3]
#As data frames are also lists, it is possible to refer to columns (which are elements of such list) using the list notation, i.e. either double square brackets or a $
dat[["y"]]
dat$y
```

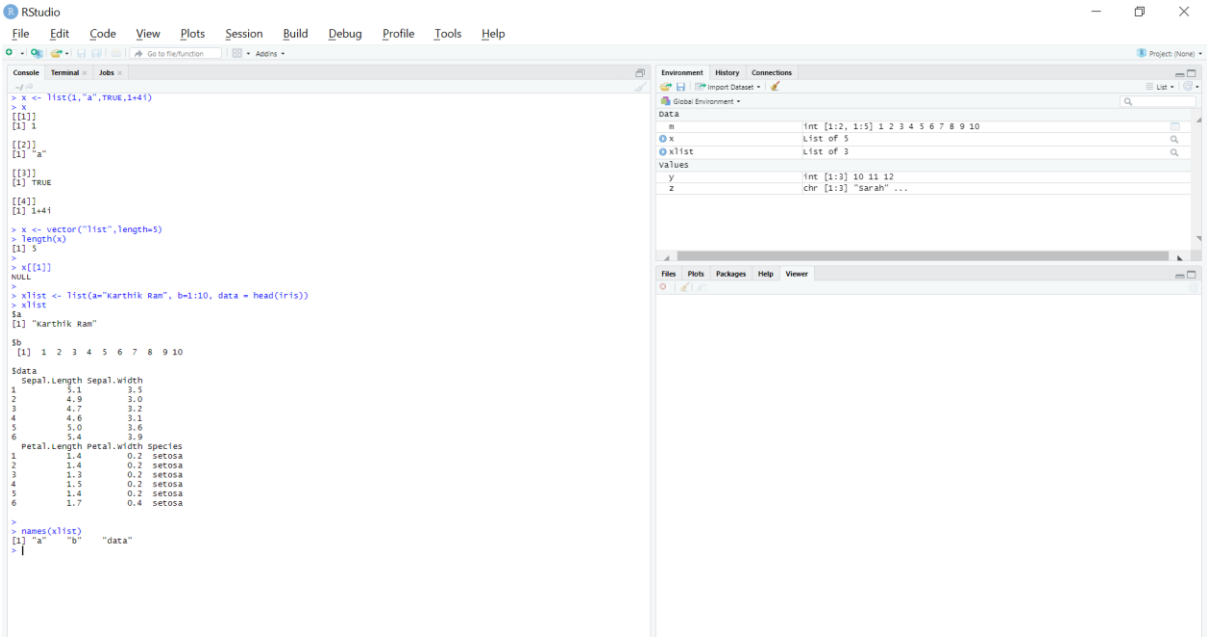
Screenshots :

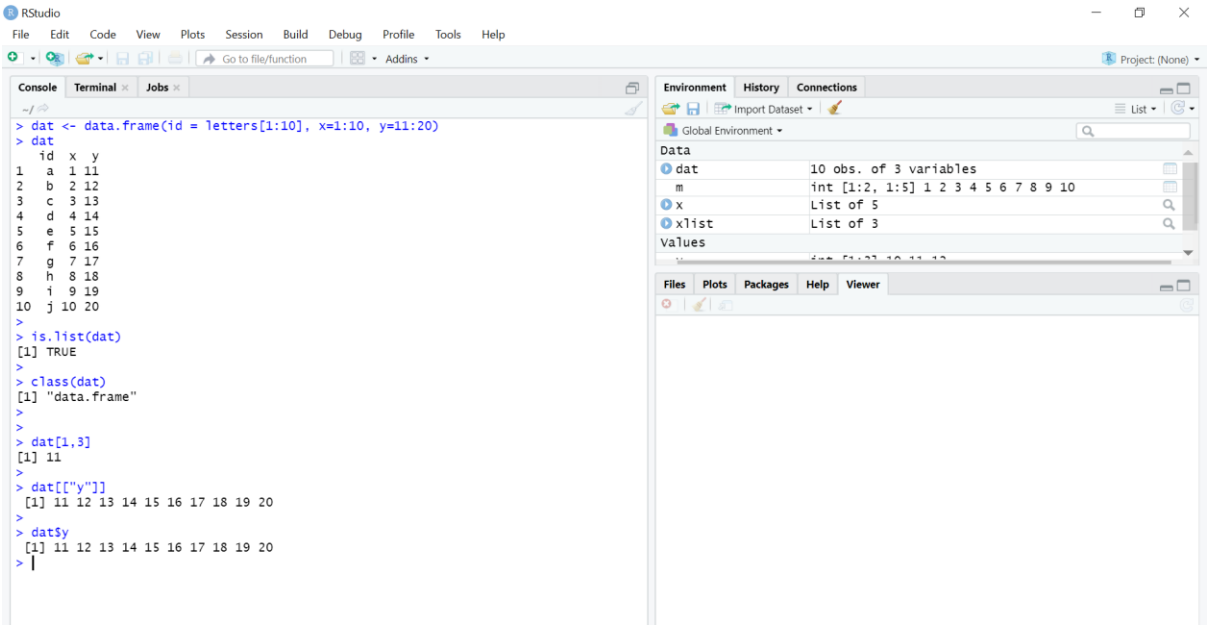



RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
Go to file/function
Addins
Project: (None)
Console Terminal Jobs
[1] 0 0 0 0 0
> logical(5)
[1] FALSE FALSE FALSE FALSE
[5] FALSE
> x<-c(1,2,3)
> x
[1] 1 2 3
> x<-c(1L,2L,3L)
> x
[1] 1 2 3
> y <- c(True,FALSE, true, FALSE)
Error: object 'True' not found
> y <- c(TRUE,FALSE, TRUE, FALSE)
> y
[1] TRUE FALSE TRUE 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" ...
Environment History Connections
Import Dataset
List
Global Environment
Values
x int [1:3] 1 2 3
y logi [1:4] TRUE FALSE TRUE FALSE
z chr [1:3] "Sarah" ...
Files Plots Packages Help Viewer


RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
Go to file/function
Addins
Project: (None)
Console Terminal Jobs
> x <- c
> 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
> 1/0
[1] Inf
> 0/0
[1] NaN
Environment History Connections
Import Dataset
List
Global Environment
Values
x chr [1:5] "a" NA "c" "d" NA
y chr [1:5] "a" "b" "c" "d" "e"
z chr [1:3] "Sarah" ...
Files Plots Packages Help Viewer


RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
Console Terminal Jobs
[,1] [,2]
[1,] NA NA
[2,] NA NA
> dim(m)
[1] 2 2
> class(m)
[1] "matrix"
> typeof(m)
[1] "logical"
> m <- matrix(c(1:3))
> class(m)
[1] "matrix"
> typeof(m)
[1] "integer"
> m <- 1:10
> dim(m) <- c(2,5)
> 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
Environment History Connections
Import Dataset
Global Environment
Data
m int [1:2, 1:5] 1 2 3 4 5 6 7 8 9 10
Values
x int [1:3] 1 2 3
y int [1:3] 10 11 12
z chr [1:3] "Sarah" ...
Files Plots Packages Help Viewer


RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
> 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
> xlist <- list(a="Karthik Ram", b=1:10, data = head(iris))
> xlist
$a
[1] "Karthik Ram"
$b
[1] 1 2 3 4 5 6 7 8 9 10
$data
Sepal.Length Sepal.Width
1 5.1 3.5
2 4.9 3.0
3 4.7 3.2
4 4.6 3.1
5 5.0 3.6
6 5.4 3.9
Petal.Length Petal.Width Species
1 1.4 0.2 setosa
2 1.4 0.2 setosa
3 1.3 0.2 setosa
4 1.5 0.2 setosa
5 1.4 0.2 setosa
6 1.7 0.4 setosa
> names(xlist)
[1] "a" "b" "data"
Environment History Connections
Import Dataset
Global Environment
Data
m int [1:2, 1:5] 1 2 3 4 5 6 7 8 9 10
x List of 5
xlist List of 3
Values
y int [1:3] 10 11 12
z chr [1:3] "Sarah" ...
Files Plots Packages Help Viewer



Conclusion : Thus, we have studied basic Data types and data structures in R - vector ,matrices,list and dataframes.

Experiment No. 4

DATTA MEGHE COLLEGE OF ENGINEERING Page No. : Date :

EXPERIMENT 4

AIM: Programming constructs in R- loops, conditional executions.

THEORY :

FOR LOOP IN R

Syntax of for loop in R-

```
for (i in 1:n)
{
  statement
}
```

Syntax of Nested loop in R

```
for (i in 1:n)
{
  for (j in 1:n)
  {
    statement
  }
}
```

Break statement in R for loops
A break statement is used inside a loop to stop the iterations and the flow control outside the loop.

```
x <- 1:5
for (i in x) {
```

```
    if (i==3) {
        break
    }
    print (i)
}
```

The use of next in R for loop:
"next" discontinues a particular iteration and jumps to next cycle.

IF ELSE CONDITION IN R

If Else conditional statements are important part of any programming.
R if statement syntax:

```
if (expression)
  {
    statement
  }
```

If the expression is true statement gets executed. If false, noting happens

If else statement syntax in R:

```
if (expression)
{
    statement 1
}
else
  {
    statement 2
  }
```

The else part is evaluated only when expression is false.

REPEAT FUNCTION IN R

The Repeat function (loop) in R executes a same block of code iteratively until a stop condition is met.

Syntax of Repeat :

```
repeat :
  if (condition) {
     break
  }
}
```

REPLICATE FUNCTION IN R

Rep() is the function in R that replicates the values.

CONCLUSION : We learned and understood various conditions, loops and functions. We implemented them and observed the outputs.

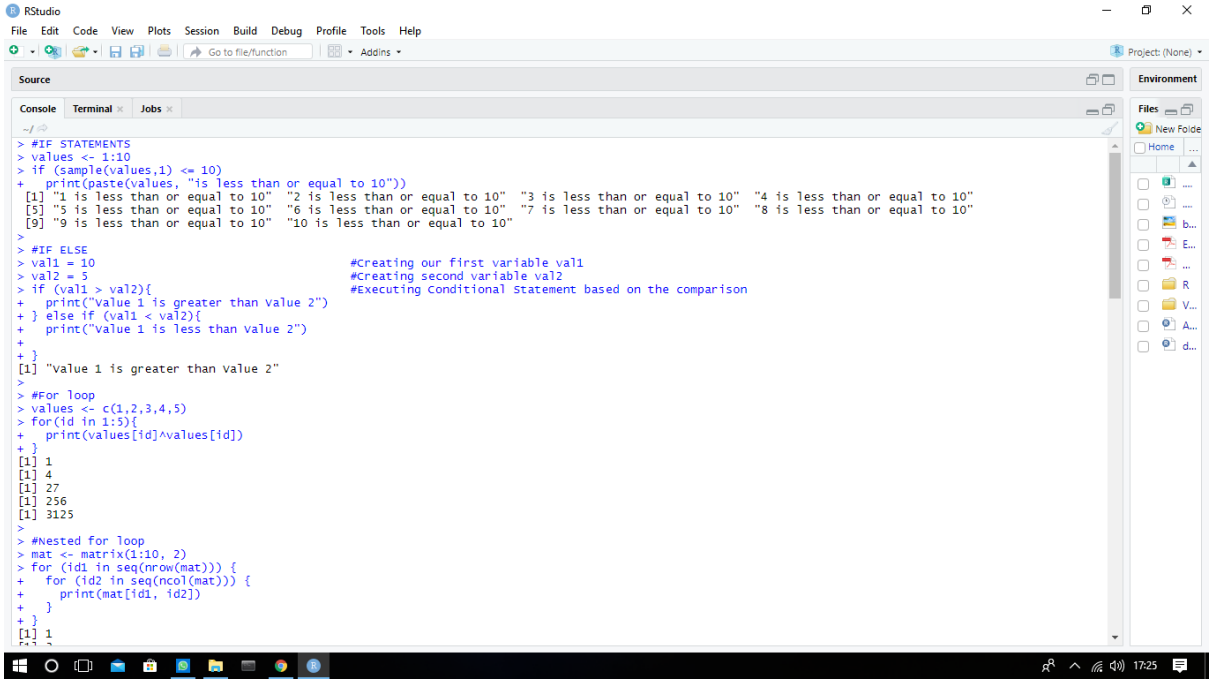
Output :

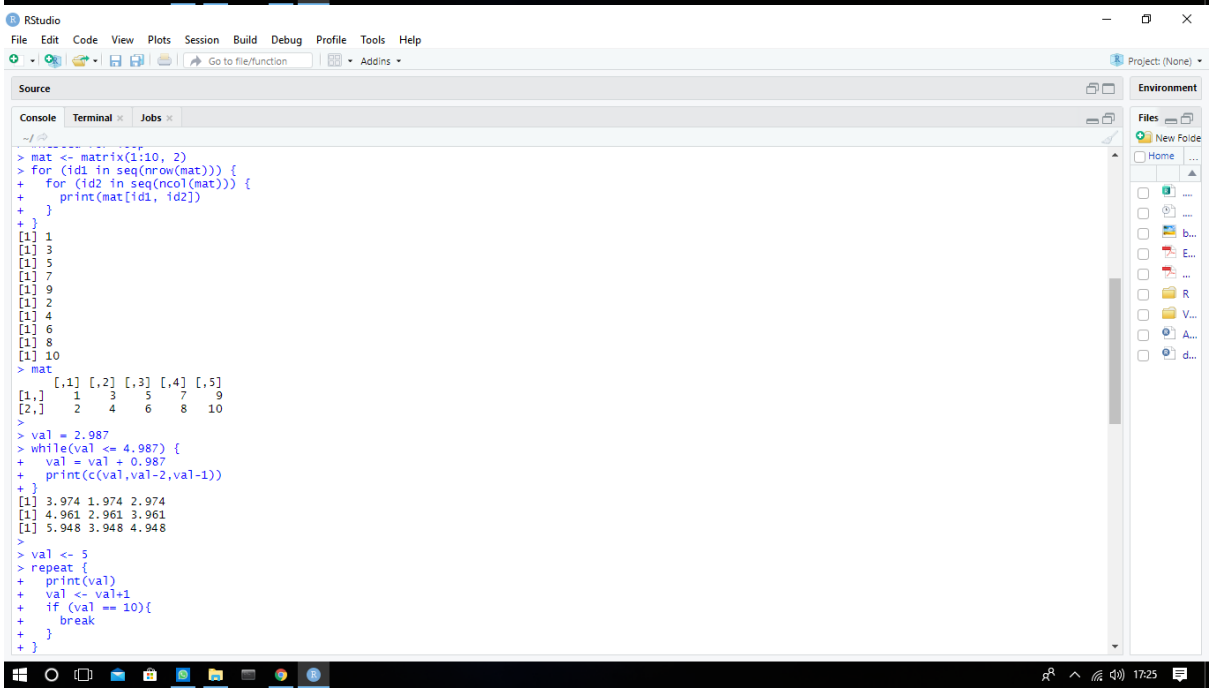
Code :

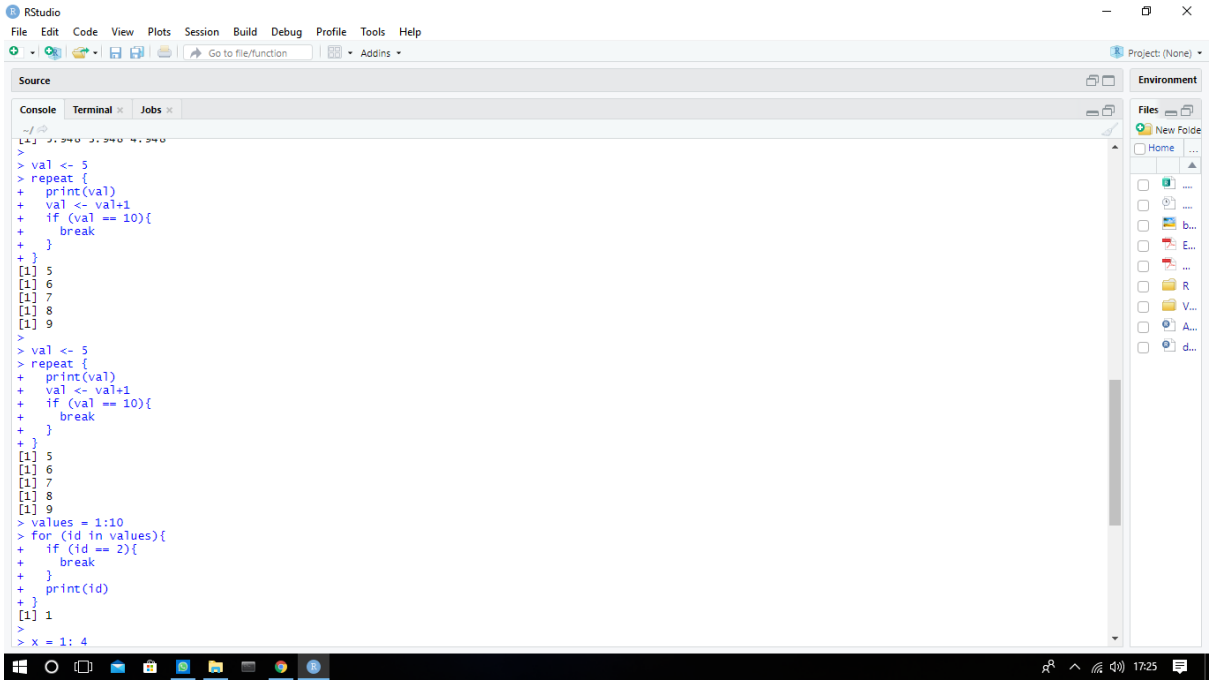
```
#IF STATEMENTS
values <- 1:10
if (sample(values,1) <= 10)
  print(paste(values, "is less than or equal to 10"))

#if else
val1 = 10                          #Creating our first variable val1
val2 = 5                           #Creating second variable val2
if (val1 > val2){                     #Executing Conditional Statement based on the comparison
  print("Value 1 is greater than Value 2")
} else if (val1 < val2){
  print("Value 1 is less than Value 2")

}

#For loop
values <- c(1,2,3,4,5)
for(id in 1:5){
  print(values[id]^values[id])
}

#Nested for loop
mat <- matrix(1:10, 2)
for (id1 in seq(nrow(mat))) {
  for (id2 in seq(ncol(mat))) {
    print(mat[id1, id2])
  }
}
mat

val = 2.987
while(val <= 4.987) {
  val = val + 0.987
  print(c(val,val-2,val-1))
}

val <- 5
repeat {
  print(val)
  val <- val+1
```

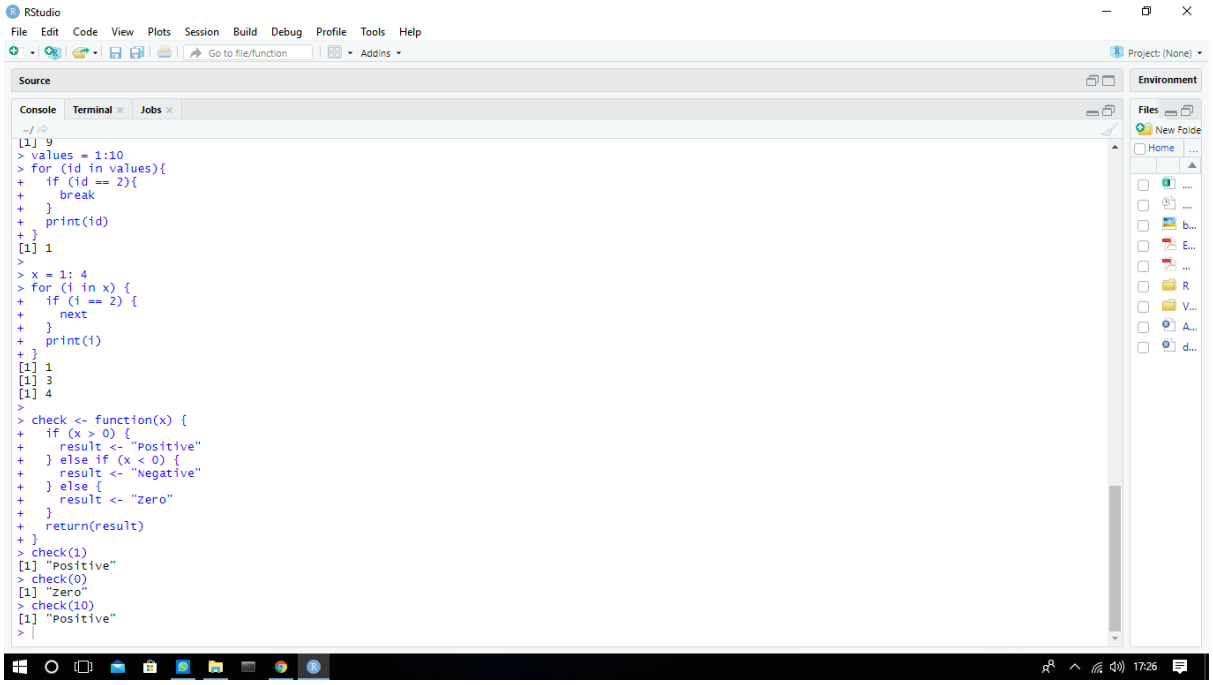
```
  if (val == 10){
    break
  }
}

val <- 5
repeat {
  print(val)
  val <- val+1
  if (val == 10){
    break
  }
}
values = 1:10
for (id in values){
  if (id == 2){
    break
  }
  print(id)
}
x = 1: 4
for (i in x) {
  if (i == 2) {
    next
  }
  print(i)
}

check <- function(x) {
  if (x > 0) {
    result <- "Positive"
  } else if (x < 0) {
    result <- "Negative"
  } else {
    result <- "Zero"
  }
  return(result)
}
check(1)
check(0)
check(10)
```

Screenshots :









Conclusion : Thus we have studied the programming constructs in R.

Experiment No. 5

DATTA MEGHE COLLEGE OF ENGINEERING Page No. : _____ Date : _____

EXPERIMENT NO: 5

AIM: Tables with labels in R, multiple response variables, Excel function transition.

THEORY :

EXPSS

expss computes and displays tables with support for 'SPSS' - style labels, multiple/ nested banners, weights, multiple - response variables and significance testing. There are facilities for nice outputs of tables in 'knitr', R notebooks, 'Shiny' and 'Jupyter' notebooks. Proper methods for labelled variables and values labels to support to base R functionalities and to some functions from other packages.

Additionally, the package - offers useful functions for data processing in marketing research / social surveys - popular data transformation functions from SPSS statistics and 'EXCEL'.

expss is on CRAN, so for installation you can print in console

```
> install.packages("expss")
```

For quick cross tabulation there are Fre and cro family of function. For simplicity we demonstrate cro-cpct which calculates column percent. Other functions include :

cro_cases - for count
cro-rpct - for row count
cro_tpct - table percent
cro-fun - for custom summary functions.

> cro(mtcars$am, mtcars$vs)

> cro-cpct (mtcars$cyl, list(total(), mtcars$am, mtcars$vs))

> mtcars %>% calc-prob_cpct(cy1, list(total(), am, vs()))

CONCLUSION : We learned about tables in R and how to convert excel functions to R for functionalities. We executed the commands and output for same was carefully observed.

Output:

Code:

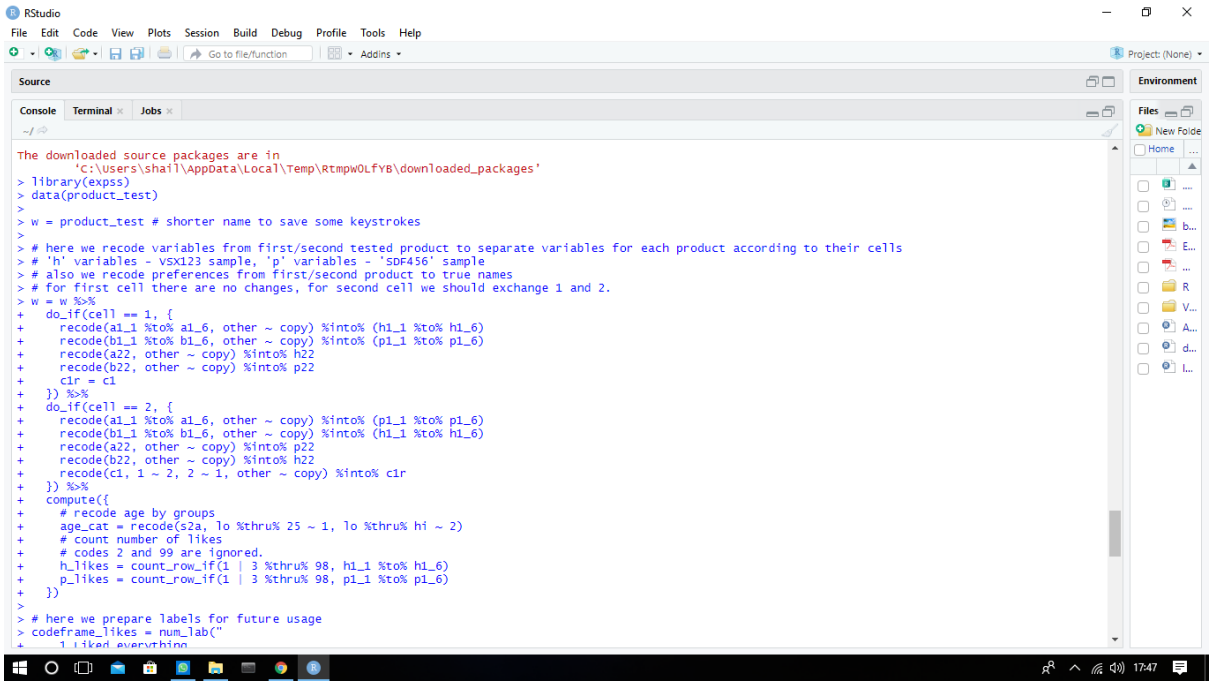
```
install.packages("expss")
library(expss)
data(product_test)

w = product_test # shorter name to save some keystrokes

# here we recode variables from first/second tested product to separate variables for each product according to their cells
# 'h' variables - VSX123 sample, 'p' variables - 'SDF456' sample
# also we recode preferences from first/second product to true names
# for first cell there are no changes, for the second cell we should exchange 1 and 2.
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)
  })

# here we prepare labels for future usage
codeframe_likes = num_lab("
    1 Liked everything
    2 Disliked everything
    3 Chocolate
    4 Appearance
    5 Taste
    6 Stuffing
    7 Nuts
    8 Consistency
    98 Other
```

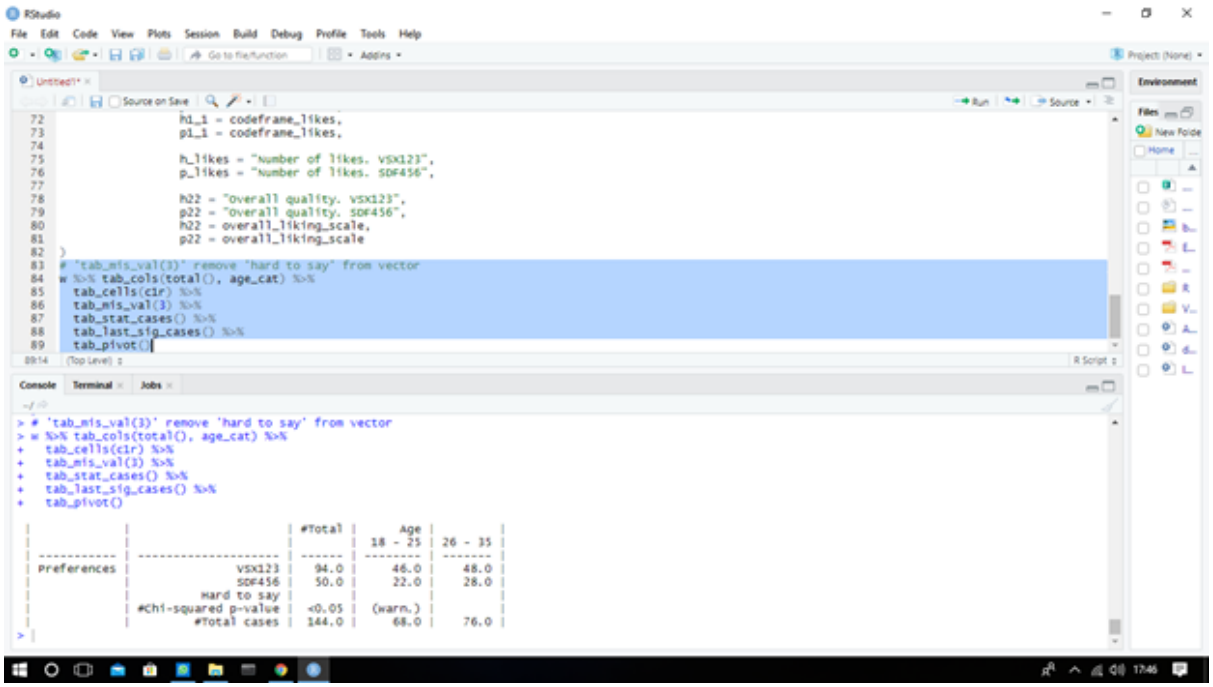
```
   99 Hard to answer
")

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
)
# 'tab_mis_val(3)' remove 'hard to say' from vector
w %>% tab_cols(total(), age_cat) %>%
  tab_cells(c1r) %>%
  tab_mis_val(3) %>%
  tab_stat_cases() %>%
  tab_last_sig_cases() %>%
  tab_pivot()

# lets specify repeated parts of table creation chains
banner = w %>% tab_cols(total(), age_cat, c1r)
# column percent with significance
tab_cpct_sig = . %>% tab_stat_cpct() %>%
  tab_last_sig_cpct(sig_labels = paste0("<b>",LETTERS, "</b>"))
```

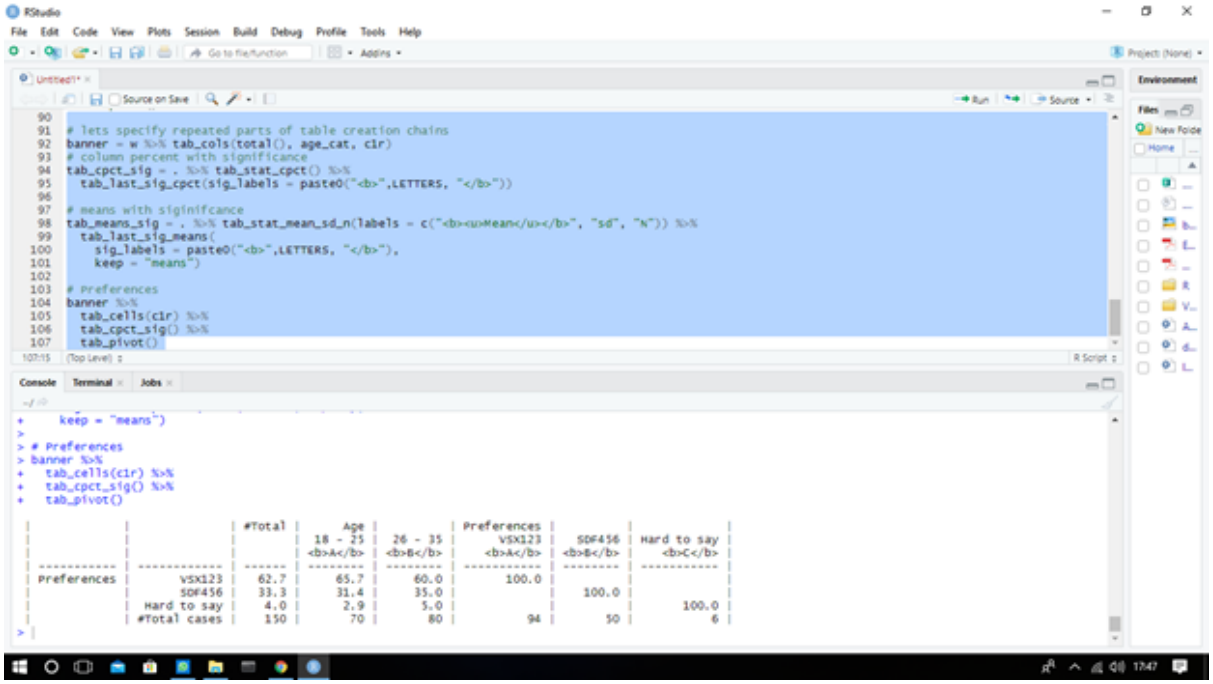
```
# means with significance
tab_means_sig = . %>% tab_stat_mean_sd_n(labels = c("<b><u>Mean</u></b>", "sd", "N"))
%>%
  tab_last_sig_means(
    sig_labels = paste0("<b>",LETTERS, "</b>"),
    keep = "means")

# Preferences
banner %>%
  tab_cells(c1r) %>%
  tab_cpct_sig() %>%
  tab_pivot()

# Overall liking
banner %>%
  tab_cells(h22) %>%
  tab_means_sig() %>%
  tab_cpct_sig() %>%
  tab_cells(p22) %>%
  tab_means_sig() %>%
  tab_cpct_sig() %>%
  tab_pivot()

# Likes
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()

# below more complicated table where we compare likes side by side
# Likes - side by side comparison
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")
write_labelled_csv(w, file = "product_test.csv")

library(expss)
w = text_to_columns("
       a  b  c
```

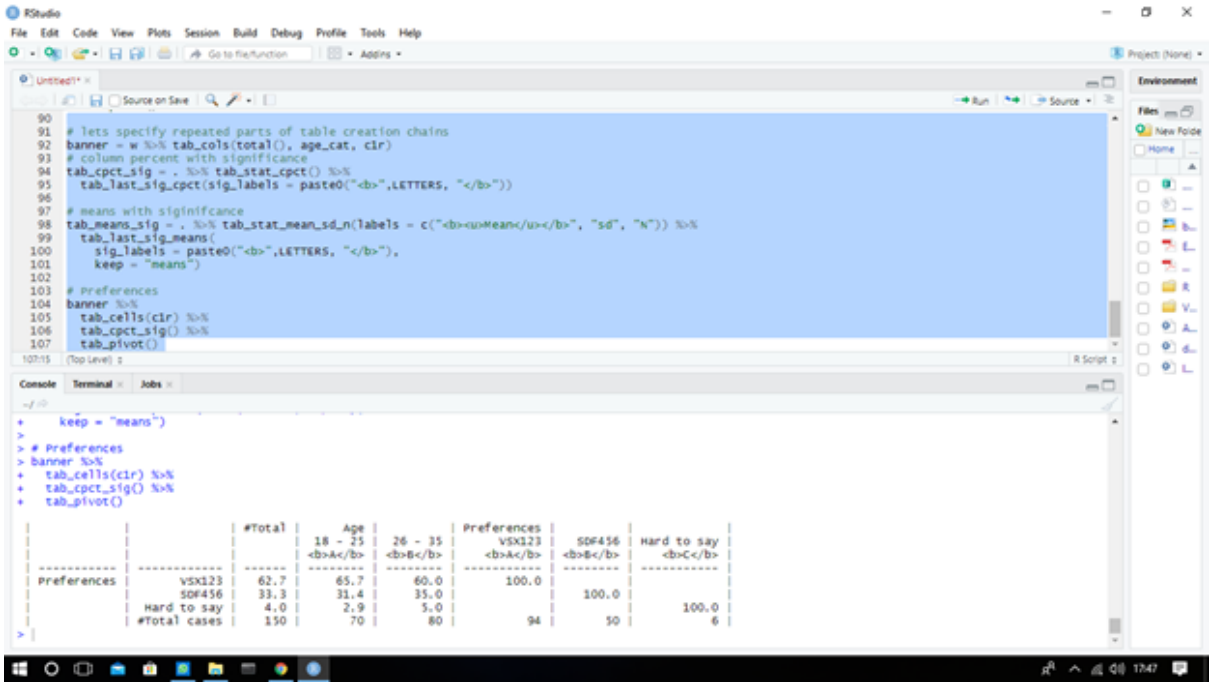
```
        2 15 50
        1 70 80
        3 30 40
        2 30 40
")
w$d = ifelse(w$b>60, 1, 0)
w = compute(w, {
  d = ifelse(b>60, 1, 0)
  e = 42
  abc_sum = sum_row(a, b, c)
  abc_mean = mean_row(a, b, c)
})
count_if(1, w)
calculate(w, count_if(1, a, b, c))

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)
sum(w, na.rm = TRUE)
w$d = mean_row(w)
#or
w = compute(w, {
  d = mean_row(a, b, c)
})
sum_col(w$a)
sum_if(gt(40), w)
#or
calculate(w, sum_if(gt(40), a, b, c))
w$d = sum_row_if(lt(40), w)
#or
w = compute(w, {
  d = sum_row_if(lt(40), a, b, c)
})
mean_col_if(lt(3), w$a, data = w$b)
#or
calculate(w, mean_col_if(lt(3), a, data = sheet(b, c)))

dict = text_to_columns("
    x  y
    1  apples
    2  oranges
    3  peaches
")
w$d = vlookup(w$a, dict, 2)
#or
w$d = vlookup(w$a, dict, "y")
```

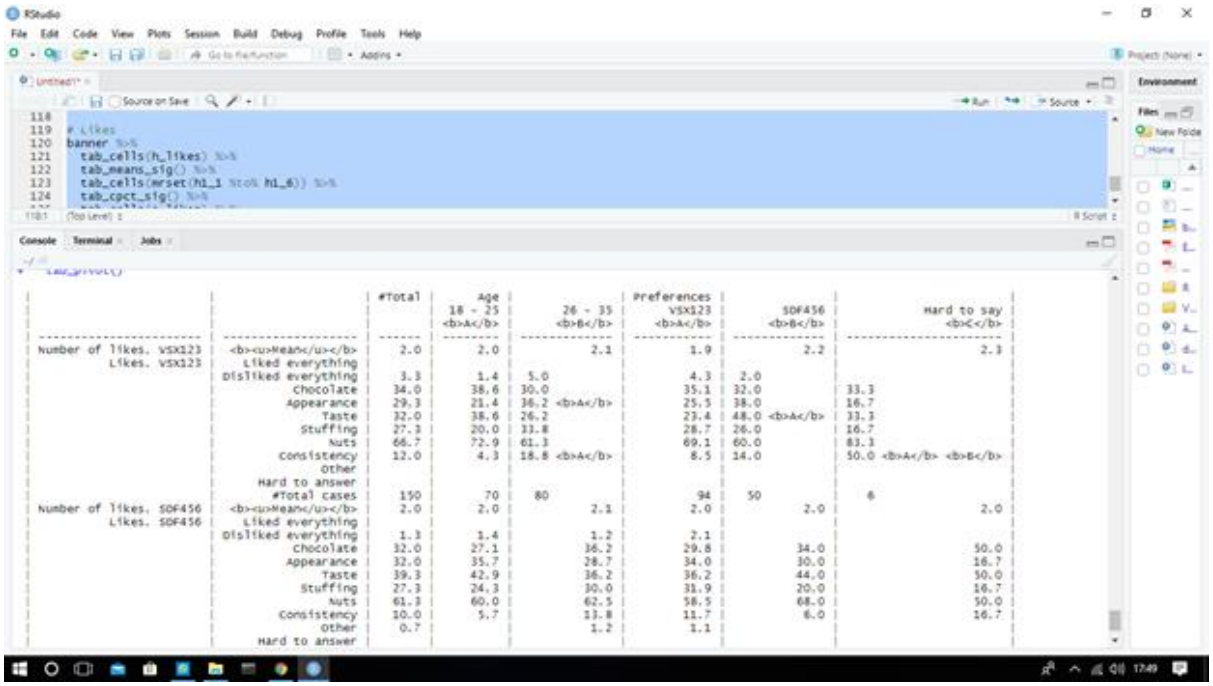
Screenshots:





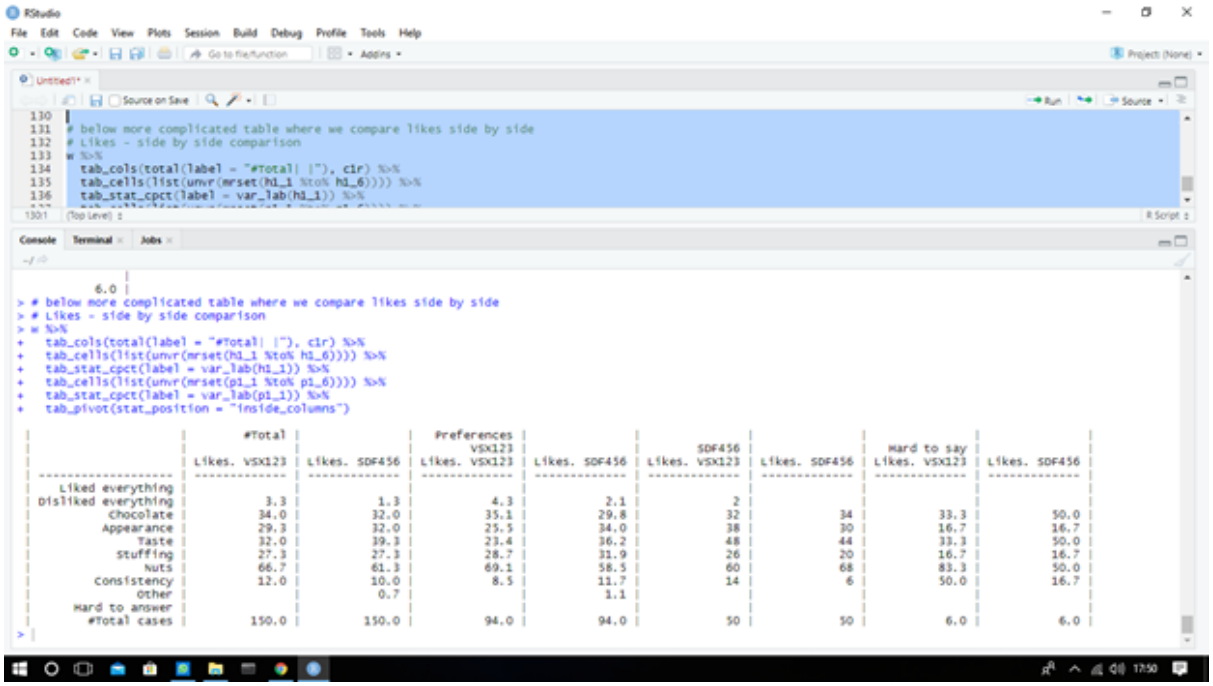

RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
Untitled1*
Source on Save
Run
Source
Environment
Files
New Folder
Home
# lets specify repeated parts of table creation chains
banner = w %>% tab_cols(total(), age_cat, c1r)
# column percent with significance
tab_cpct_sig = . %>% tab_stat_cpct() %>%
tab_last_sig_cpct(sig_labels = paste0("",LETTERS, ""))
# means with siginifcance
tab_means_sig = . %>% tab_stat_mean_sd_n(labels = c("<u>Mean</u>", "sd", "N")) %>%
tab_last_sig_means(
sig_labels = paste0("",LETTERS, ""),
keep = "means")
# Preferences
banner %>%
tab_cells(c1r) %>%
tab_cpct_sig() %>%
tab_pivot()
107:15 (Top Level)
R Script
Console Terminal Jobs
keep = "means")
> # Preferences
> banner %>%
+ tab_cells(c1r) %>%
+ tab_cpct_sig() %>%
+ 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
17:47


RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
Untitled1*
Source on Save
Run
Source
Environment
Files
New Folder
Home
# lets specify repeated parts of table creation chains
banner = w %>% tab_cols(total(), age_cat, c1r)
# column percent with significance
tab_cpct_sig = . %>% tab_stat_cpct() %>%
tab_last_sig_cpct(sig_labels = paste0("",LETTERS, ""))
# means with siginifcance
tab_means_sig = . %>% tab_stat_mean_sd_n(labels = c("<u>Mean</u>", "sd", "N")) %>%
tab_last_sig_means(
sig_labels = paste0("",LETTERS, ""),
keep = "means")
# Preferences
banner %>%
tab_cells(c1r) %>%
tab_cpct_sig() %>%
tab_pivot()
107:15 (Top Level)
R Script
Console Terminal Jobs
keep = "means")
> # Preferences
> banner %>%
+ tab_cells(c1r) %>%
+ tab_cpct_sig() %>%
+ 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
17:47



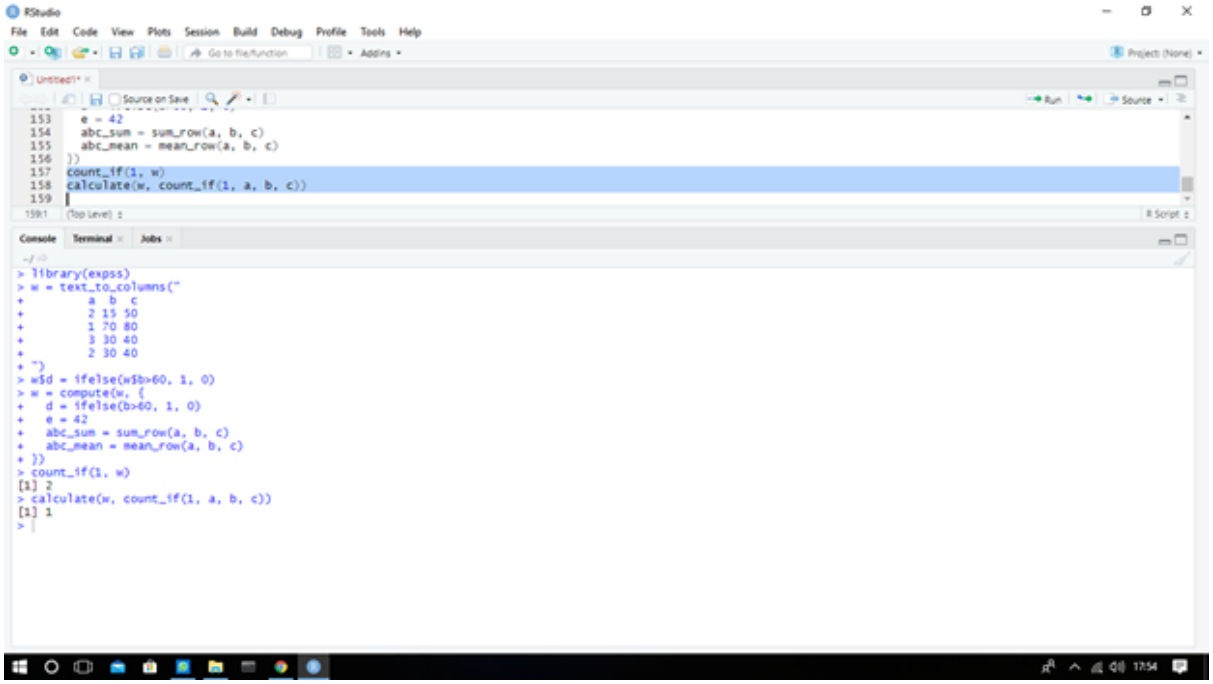
```
# Likes
banner %>%
  tab_cells(h_likes) %>%
  tab_means_sig() %>%
  tab_cells(mrset(h1_1 %to% h1_6)) %>%
  tab_cpct_sig() %>%
```

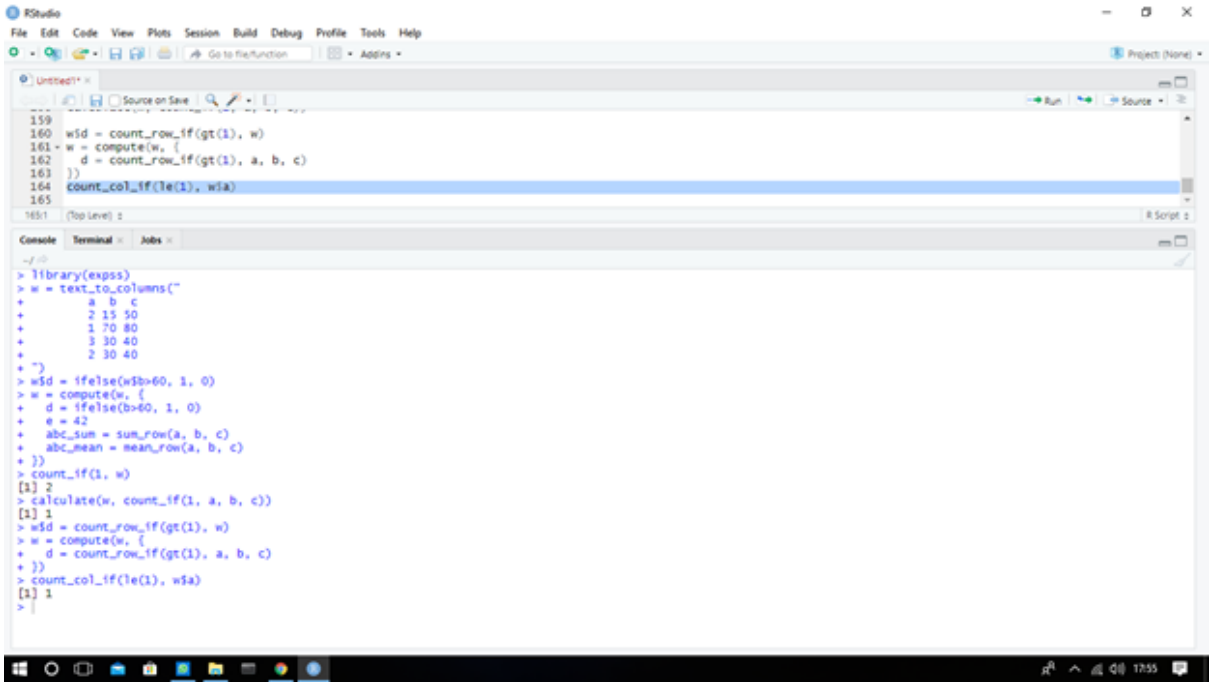
		#Total	Age		Preferences		
			18 - 25	26 - 35	VSX123	SDF456	Hard to say
			A	B	A	B	C
Number of likes. VSX123	<u>Mean</u>	2.0	2.0	2.1	1.9	2.2	2.3
Likes. VSX123	Liked everything						
	Disliked everything	3.3	1.4	5.0	4.3	2.0	
	Chocolate	34.0	38.6	30.0	35.1	32.0	33.3
	Appearance	29.3	21.4	36.2 A	25.5	38.0	16.7
	Taste	32.0	38.6	26.2	23.4	48.0 A	33.3
	Stuffing	27.3	20.0	33.8	28.7	26.0	16.7
	Nuts	66.7	72.9	61.3	69.1	60.0	83.3
	Consistency	12.0	4.3	18.8 A	8.5	14.0	50.0 A B
	Other						
	Hard to answer						
	#Total cases	150	70	80	94	50	6
Number of likes. SDF456	<u>Mean</u>	2.0	2.0	2.1	2.0	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	50.0
	Appearance	32.0	35.7	28.7	34.0	30.0	16.7
	Taste	39.3	42.9	36.2	36.2	44.0	50.0
	Stuffing	27.3	24.3	30.0	31.9	20.0	16.7
	Nuts	61.3	60.0	62.5	58.5	68.0	50.0
	Consistency	10.0	5.7	13.8	11.7	6.0	16.7
	Other	0.7		1.2	1.1		
	Hard to answer						

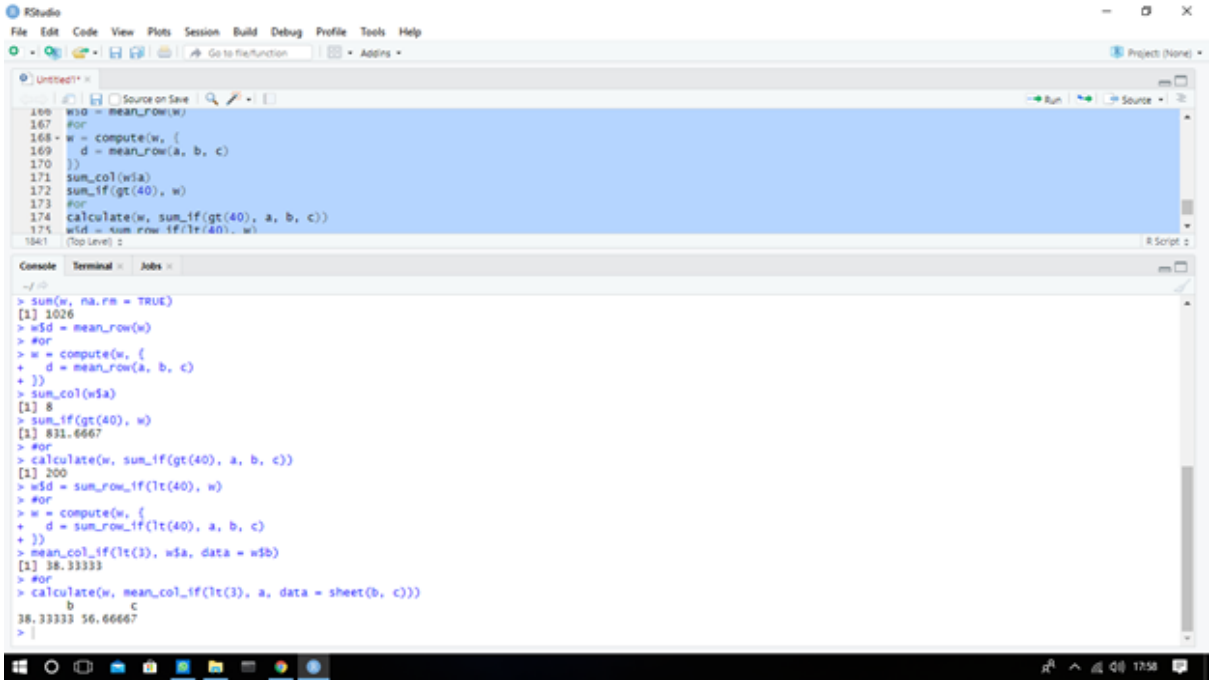


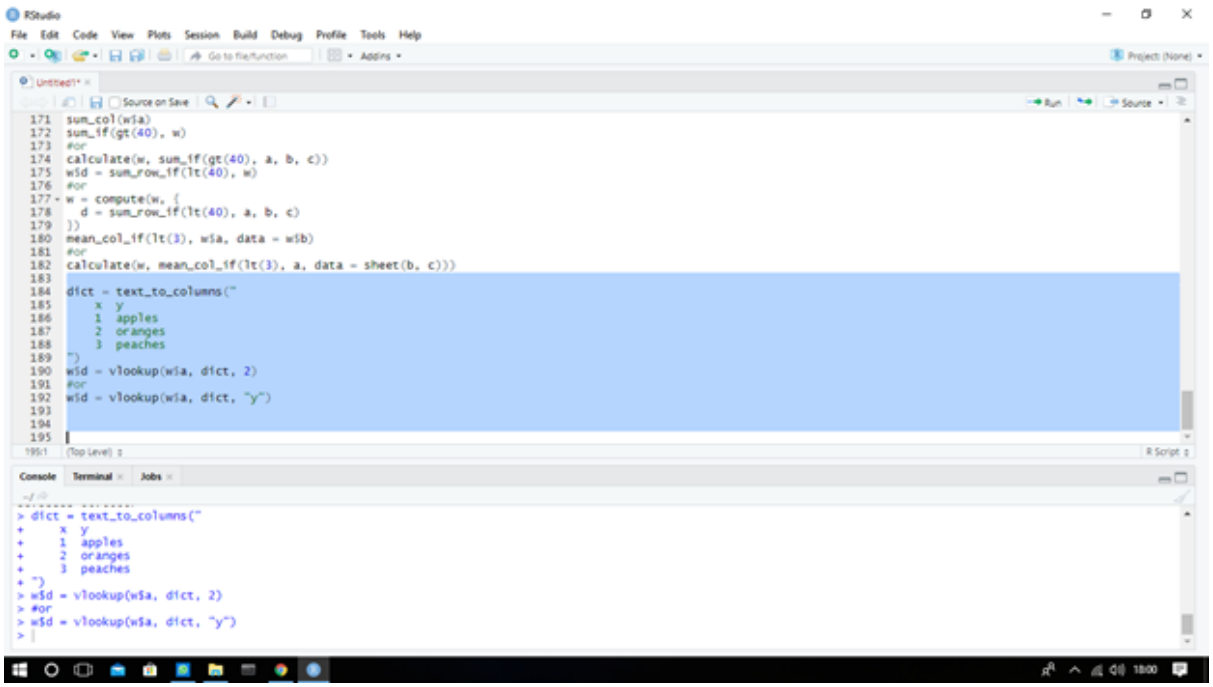
```
# below more complicated table where we compare likes side by side
# Likes - side by side comparison
w %>%
  tab_cols(total(label = "#Total| |"), cir) %>%
  tab_cells(list(unvr(mrset(h1_1 %to% h1_6)))) %>%
  tab_stat_cpct(label = var_lab(h1_1)) %>%
  tab_cells(list(unvr(mrset(p1_1 %to% p1_6)))) %>%
  tab_stat_cpct(label = var_lab(p1_1)) %>%
  tab_pivot(stat_position = "inside_columns")
```

	#Total		Preferences				Hard to say	
			VSX123		SDF456			
	Likes. VSX123	Likes. SDF456	Likes. VSX123	Likes. SDF456	Likes. VSX123	Likes. SDF456	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


RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
Untitled1*
Source on Save
Run
Source
153 e = 42
154 abc_sum = sum_row(a, b, c)
155 abc_mean = mean_row(a, b, c)
156 })
157 count_if(1, w)
158 calculate(w, count_if(1, a, b, c))
159
Console Terminal Jobs
> 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
17:54


RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
Untitled1*
Source on Save
Run
Source
160 w$d = count_row_if(gt(1), w)
161 w = compute(w, {
162 d = count_row_if(gt(1), a, b, c)
163 })
164 count_col_if(le(1), w$a)
165
Console Terminal Jobs
> 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
17:55





Conclusion: Thus, we have studied tables with labels in R.

Experiment No. 6

DATTA MEGHE COLLEGE OF ENGINEERING Page No. : Date :

EXPERIMENT NO: 6

AIM : Working with large datasets, with dplyr and data.table.

THEORY :

Wrangling Big Data is one of the best features of the R programming language, which boasts a Big Data Ecosystem that contains fast-in memory tools and distributed computational tools. With the NEW dtplyr package, data scientists with dplyr experience gains the benefits of data.table backend.

The new dtplyr package, which is an interface to the high performance data.table library.

Pros of dplyr :

(i) A 3x speed boost on the data joining and wrangling operations on dataset. The data operations were performed in 18 secs.

(ii) performs inplace operations which vastly accelerates big data computations.

(iii) Shows the data.table translation

Con of dplyr:

(i) For pure speed, you will need to learn all data.table's features including managing keys for first lookups.

(ii) In most cases, data.tables will be faster than dplyr because of overhead in the dplyr translation process. However we saw the difference to be very minimal.

DPLYR

dplyr is a grammar of data manipulation, providing a consistent set of verbs that help you solve the most common data manipulation challenges:

- mutate() adds new variables that are functions of existing variables.
- select() pick variables based on their names.
- filter() pick cases on their values.
- summarise() reduces multiple values down to a single memory summary.
- arrange() changes the ordering of the rows

They all combine naturally with group_by() which allows you to perform any operations "by group".

dplyr is designed to abstract over how the data is stored. That means as well as working with local data frames, you can also work with remote database tables, using exactly the same R code. Install the dplyr package then read vignette("databases", package="dplyr")

Data.table

data.table is one of the 15,000 add-on packages for the programming language R which is popular in these fields. It provides a high performance version of base R's data.frame with syntax and feature enhancements for ease of use, convenience and programming speed.

CONCLUSION: We learned and understood how large datasets are handled by R. We understood various functionalities of dplyr and data.table. We observed outputs for the same.

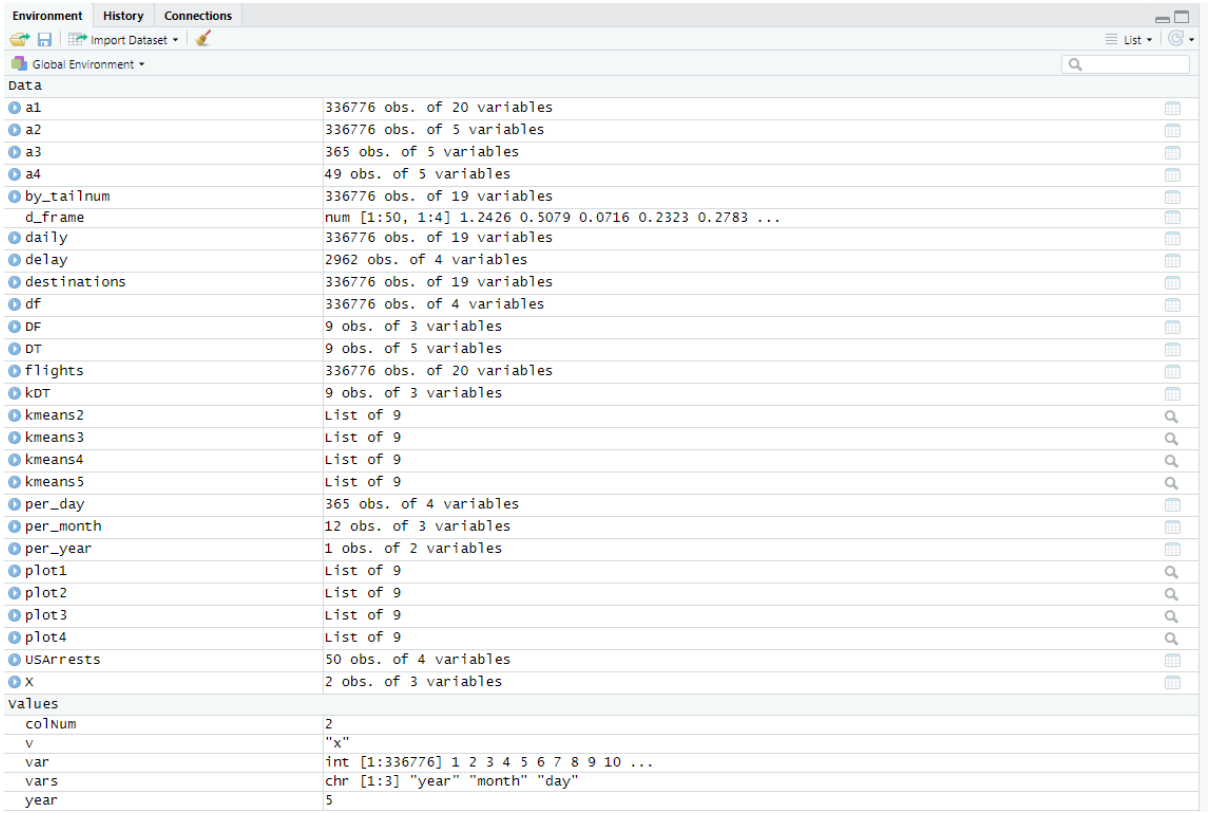
Output :

Code :

```
install.packages("nycflights13")
library(nycflights13)
dim(flights)
flights
library(dplyr)
filter(flights, month ==1, day==1)
arrange(flights, year, month, day)
arrange(flights, desc(arr_delay))
select(flights, year, month, day)
select(flights, year:day)
select(flights, -(year:day))
select(flights, tail_num = tailnum)
rename(flights, tail_num = tailnum)
mutate(flights, gain = arr_delay - dep_delay, speed = distance/air_time * 60)
mutate(flights, gain = arr_delay - dep_delay, gain_per_hour = gain/(air_time/60))
transmute(flights, gain = arr_delay - dep_delay, gain_per_hour = gain/(air_time/60))
summarise(flights, delay= mean(dep_delay, na.rm = TRUE))
sample_n(flights, 10)
sample_frac(flights,0.01)
by_tailnum<- group_by(flights, tailnum)
delay <- summarise(by_tailnum, count=n(), dist = mean(distance, na.rm = TRUE), delay =
mean(arr_delay, na.rm = TRUE))
delay <- filter(delay, count >20, dist <2000)
install.packages(ggplot)
destinations <- group_by(flights,dest)
summarise(destinations, planes = n_distinct(tailnum), flights= n())
daily <- group_by(flights, year, month, day)
(per_day   <- summarise(daily, flights = n()))
(per_month <- summarise(per_day, flights = sum(flights)))
(per_year  <- summarise(per_month, flights = sum(flights)))
select(flights, year)
select(flights, 1)
year <- 5
select(flights, year)
year <- "dep"
select(flights, starts_with(year))
year <- 5
select(flights, year, identity(year))
vars <- c("year", "month")
select(flights, vars, "day")
flights$vars <- flights$year
vars <- c("year", "month", "day")
select(flights, !! vars)
df <- select(flights, year:dep_time)
mutate(df, "year", 2)
```

```
mutate(df, year + 10)
var <- seq(1, nrow(df))
mutate(df, new = var)
group_by(df, month)
group_by(df, month = as.factor(month))
group_by(df, day_binned = cut(day, 3))
group_by(df, "month")
group_by_at(df, vars(year:day))
a1 <- group_by(flights, year, month, day)
a2 <- select(a1, arr_delay, dep_delay)
a3 <- summarise(a2,
                arr = mean(arr_delay, na.rm = TRUE),
                dep = mean(dep_delay, na.rm = TRUE))
a4 <- filter(a3, arr > 30 | dep > 30)
filter(
  summarise(
    select(
      group_by(flights, year, month, day),
      arr_delay, dep_delay
    ),
    arr = mean(arr_delay, na.rm = TRUE),
    dep = mean(dep_delay, na.rm = TRUE)
  ),
  arr > 30 | dep > 30
)
flights %>%
  group_by(year, month, day) %>%
  select(arr_delay, dep_delay) %>%
  summarise(
    arr = mean(arr_delay, na.rm = TRUE),
    dep = mean(dep_delay, na.rm = TRUE)
  ) %>%
  filter(arr > 30 | dep > 30)
require(data.table)
install.packages("data.table")
library(data.table)
require(data.table)
example(data.table)
```

Screenshots:



Environment | History | Connections

Import Dataset | List

Global Environment

Data	
a1	336776 obs. of 20 variables
a2	336776 obs. of 5 variables
a3	365 obs. of 5 variables
a4	49 obs. of 5 variables
by_tailnum	336776 obs. of 19 variables
d_frame	num [1:50, 1:4] 1.2426 0.5079 0.0716 0.2323 0.2783 ...
daily	336776 obs. of 19 variables
delay	2962 obs. of 4 variables
destinations	336776 obs. of 19 variables
df	336776 obs. of 4 variables
DF	9 obs. of 3 variables
DT	9 obs. of 5 variables
flights	336776 obs. of 20 variables
kDT	9 obs. of 3 variables
kmeans2	List of 9
kmeans3	List of 9
kmeans4	List of 9
kmeans5	List of 9
per_day	365 obs. of 4 variables
per_month	12 obs. of 3 variables
per_year	1 obs. of 2 variables
plot1	List of 9
plot2	List of 9
plot3	List of 9
plot4	List of 9
USArrests	50 obs. of 4 variables
x	2 obs. of 3 variables
values	
colNum	2
v	"x"
var	int [1:336776] 1 2 3 4 5 6 7 8 9 10 ...
vars	chr [1:3] "year" "month" "day"
year	5

```
> library(nycflights13)

Attaching package: 'nycflights13'

The following object is masked _by_ '.GlobalEnv':

    flights

> dim(flights)
[1] 336776     20
> flights
# A tibble: 336,776 x 20
    year month   day dep_time sched_dep_time dep_delay arr_time sched_arr_time arr_delay carrier flight tailnum origin dest  air_time distance  hour minute
   <int> <int> <int>    <int>          <int>     <dbl>    <int>          <int>     <dbl> <chr>    <int> <chr>   <chr>  <chr>    <dbl>    <dbl> <dbl>  <dbl>
 1  2013     1     1      517            515         2      830            819        11 UA        1545 N14228  EWR    IAH        227     1400     5     15
 2  2013     1     1      533            529         4      850            830        20 UA        1714 N24211  LGA    IAH        227     1416     5     29
 3  2013     1     1      542            540         2      923            850        33 AA        1141 N619AA  JFK    MIA        160     1089     5     40
 4  2013     1     1      544            545        -1     1004           1022       -18 B6          725 N804JB  JFK    BQN        183     1576     5     45
 5  2013     1     1      554            600        -6      812            837       -25 DL          461 N668DN  LGA    ATL        116      762     6      0
 6  2013     1     1      554            558        -4      740            728        12 UA        1696 N39463  EWR    ORD        150      719     5     58
 7  2013     1     1      555            600        -5      913            854        19 B6          507 N516JB  EWR    FLL        158     1065     6      0
 8  2013     1     1      557            600        -3      709            723       -14 EV        5708 N829AS  LGA    IAD         53      229     6      0
 9  2013     1     1      557            600        -3      838            846        -8 B6           79 N593JB  JFK    MCO        140      944     6      0
10  2013     1     1      558            600        -2      753            745         8 AA          301 N3ALAA  LGA    ORD        138      733     6      0
# ... with 336,766 more rows, and 2 more variables: time_hour <dttm>, vars <int>
> library(dplyr)
> filter(flights, month ==1, day==1)
# A tibble: 842 x 20
    year month   day dep_time sched_dep_time dep_delay arr_time sched_arr_time arr_delay carrier flight tailnum origin dest  air_time distance  hour minute
   <int> <int> <int>    <int>          <int>     <dbl>    <int>          <int>     <dbl> <chr>    <int> <chr>   <chr>  <chr>    <dbl>    <dbl> <dbl>  <dbl>
 1  2013     1     1      517            515         2      830            819        11 UA        1545 N14228  EWR    IAH        227     1400     5     15
 2  2013     1     1      533            529         4      850            830        20 UA        1714 N24211  LGA    IAH        227     1416     5     29
 3  2013     1     1      542            540         2      923            850        33 AA        1141 N619AA  JFK    MIA        160     1089     5     40
 4  2013     1     1      544            545        -1     1004           1022       -18 B6          725 N804JB  JFK    BQN        183     1576     5     45
 5  2013     1     1      554            600        -6      812            837       -25 DL          461 N668DN  LGA    ATL        116      762     6      0
 6  2013     1     1      554            558        -4      740            728        12 UA        1696 N39463  EWR    ORD        150      719     5     58
 7  2013     1     1      555            600        -5      913            854        19 B6          507 N516JB  EWR    FLL        158     1065     6      0
 8  2013     1     1      557            600        -3      709            723       -14 EV        5708 N829AS  LGA    IAD         53      229     6      0
 9  2013     1     1      557            600        -3      838            846        -8 B6           79 N593JB  JFK    MCO        140      944     6      0
10  2013     1     1      558            600        -2      753            745         8 AA          301 N3ALAA  LGA    ORD        138      733     6      0
# ... with 832 more rows, and 2 more variables: time_hour <dttm>, vars <int>
```

```
> arrange(flights, year, month, day)
# A tibble: 336,776 x 20
    year month   day dep_time sched_dep_time dep_delay arr_time sched_arr_time arr_delay carrier flight tailnum origin dest  air_time distance  hour minute
   <int> <int> <int>    <int>          <int>     <dbl>    <int>          <int>     <dbl> <chr>    <int> <chr>   <chr>  <chr>    <dbl>    <dbl> <dbl>  <dbl>
 1  2013     1     1      517            515         2      830            819        11 UA        1545 N14228  EWR    IAH        227     1400     5     15
 2  2013     1     1      533            529         4      850            830        20 UA        1714 N24211  LGA    IAH        227     1416     5     29
 3  2013     1     1      542            540         2      923            850        33 AA        1141 N619AA  JFK    MIA        160     1089     5     40
 4  2013     1     1      544            545        -1     1004           1022       -18 B6         725 N804JB  JFK    BQN        183     1576     5     45
 5  2013     1     1      554            600        -6      812            837       -25 DL         461 N668DN  LGA    ATL        116      762     6      0
 6  2013     1     1      554            558        -4      740            728        12 UA        1696 N39463  EWR    ORD        150      719     5     58
 7  2013     1     1      555            600        -5      913            854        19 B6         507 N516JB  EWR    FLL        158     1065     6      0
 8  2013     1     1      557            600        -3      709            723       -14 EV        5708 N829AS  LGA    IAD         53      229     6      0
 9  2013     1     1      557            600        -3      838            846        -8 B6          79 N593JB  JFK    MCO        140      944     6      0
10  2013     1     1      558            600        -2      753            745         8 AA         301 N3ALAA  LGA    ORD        138      733     6      0
# ... with 336,766 more rows, and 2 more variables: time_hour <dttm>, vars <int>
> arrange(flights, desc(arr_delay))
# A tibble: 336,776 x 20
    year month   day dep_time sched_dep_time dep_delay arr_time sched_arr_time arr_delay carrier flight tailnum origin dest  air_time distance  hour minute
   <int> <int> <int>    <int>          <int>     <dbl>    <int>          <int>     <dbl> <chr>    <int> <chr>   <chr>  <chr>    <dbl>    <dbl> <dbl>  <dbl>
 1  2013     6    15     1432           1935      1137     1607           2120      1127 MQ        3535 N504MQ  JFK    CMH         74      483    19     35
 2  2013     1    10     1121           1635      1126     1239           1810      1109 MQ        3695 N517MQ  EWR    ORD        111      719    16     35
 3  2013     9    20     1139           1845      1014     1457           2210      1007 AA         177 N338AA  JFK    SFO        354     2586    18     45
 4  2013     7    22      845           1600      1005     1044           1815       989 MQ        3075 N665MQ  JFK    CVG         96      589    16      0
 5  2013     4    10     1100           1900       960     1342           2211       931 DL        2391 N959DL  JFK    TPA        139     1005    19      0
 6  2013     3    17     2321            810       911      135           1020       915 DL        2119 N927DA  LGA    MSP        167     1020     8     10
 7  2013     7    22     2257            759       898      121           1026       895 DL        2047 N6716C  LGA    ATL        109      762     7     59
 8  2013    12     5      756           1700       896     1058           2020       878 AA         172 N5DMAA  EWR    MIA        149     1085    17      0
 9  2013     5     3     1133           2055       878     1250           2215       875 MQ        3744 N523MQ  EWR    ORD        112      719    20     55
10  2013     3    17     2321            810       911      135           1020       915 DL        2119 N927DA  LGA    MSP        167     1020     8     10
# ... with 336,766 more rows, and 2 more variables: time_hour <dttm>, vars <int>
> select(flights, year, month, day)
# A tibble: 336,776 x 3
    year month   day
   <int> <int> <int>
 1  2013     1     1
 2  2013     1     1
 3  2013     1     1
 4  2013     1     1
 5  2013     1     1
 6  2013     1     1
 7  2013     1     1
```

```
> select(flights, year, month, day)
# A tibble: 336,776 x 3
    year month   day
   <int> <int> <int>
 1  2013     1     1
 2  2013     1     1
 3  2013     1     1
 4  2013     1     1
 5  2013     1     1
 6  2013     1     1
 7  2013     1     1
 8  2013     1     1
 9  2013     1     1
10  2013     1     1
# ... with 336,766 more rows
> select(flights, year:day)
# A tibble: 336,776 x 3
    year month   day
   <int> <int> <int>
 1  2013     1     1
 2  2013     1     1
 3  2013     1     1
 4  2013     1     1
 5  2013     1     1
 6  2013     1     1
 7  2013     1     1
 8  2013     1     1
 9  2013     1     1
10  2013     1     1
# ... with 336,766 more rows
> select(flights, -(year:day))
# A tibble: 336,776 x 17
   dep_time sched_dep_time dep_delay arr_time sched_arr_time arr_delay carrier flight tailnum origin dest  air_time distance  hour minute time_hour            vars
      <int>          <int>     <dbl>    <int>          <int>     <dbl> <chr>    <int> <chr>   <chr>  <chr>    <dbl>    <dbl> <dbl>  <dbl> <dttm>              <int>
 1      517            515         2      830            819        11 UA        1545 N14228  EWR    IAH        227     1400     5     15 2013-01-01 05:00:00  2013
 2      533            529         4      850            830        20 UA        1714 N24211  LGA    IAH        227     1416     5     29 2013-01-01 05:00:00  2013
 3      542            540         2      923            850        33 AA        1141 N619AA  JFK    MIA        160     1089     5     40 2013-01-01 05:00:00  2013
 4      544            545        -1     1004           1022       -18 B6         725 N804JB  JFK    BQN        183     1576     5     45 2013-01-01 05:00:00  2013
 5      554            600        -6      812            837       -25 DL         461 N668DN  LGA    ATL        116      762     6      0 2013-01-01 06:00:00  2013
 6      554            558        -4      740            728        12 UA        1696 N39463  EWR    ORD        150      719     5     58 2013-01-01 05:00:00  2013
 7      555            600        -5      913            854        19 B6         507 N516JB  EWR    FLL        158     1065     6      0 2013-01-01 06:00:00  2013
```

```
> select(flights, tail_num = tailnum)
# A tibble: 336,776 x 1
   tail_num
   <chr>
 1 N14228
 2 N24211
 3 N619AA
 4 N804JB
 5 N668DN
 6 N39463
 7 N516JB
 8 N829AS
 9 N593JB
10 N3ALAA
# ... with 336,766 more rows
> rename(flights, tail_num = tailnum)
# A tibble: 336,776 x 20
    year month   day dep_time sched_dep_time dep_delay arr_time sched_arr_time arr_delay carrier flight tail_num origin dest  air_time distance  hour minute
   <int> <int> <int>    <int>          <int>     <dbl>    <int>          <int>     <dbl> <chr>    <int> <chr>    <chr>  <chr>    <dbl>    <dbl> <dbl>  <dbl>
 1  2013     1     1      517            515         2      830            819        11 UA        1545 N14228   EWR    IAH        227     1400     5     15
 2  2013     1     1      533            529         4      850            830        20 UA        1714 N24211   LGA    IAH        227     1416     5     29
 3  2013     1     1      542            540         2      923            850        33 AA        1141 N619AA   JFK    MIA        160     1089     5     40
 4  2013     1     1      544            545        -1     1004           1022       -18 B6         725 N804JB   JFK    BQN        183     1576     5     45
 5  2013     1     1      554            600        -6      812            837       -25 DL         461 N668DN   LGA    ATL        116      762     6      0
 6  2013     1     1      554            558        -4      740            728        12 UA        1696 N39463   EWR    ORD        150      719     5     58
 7  2013     1     1      555            600        -5      913            854        19 B6         507 N516JB   EWR    FLL        158     1065     6      0
 8  2013     1     1      557            600        -3      709            723       -14 EV        5708 N829AS   LGA    IAD         53      229     6      0
 9  2013     1     1      557            600        -3      838            846        -8 B6          79 N593JB   JFK    MCO        140      944     6      0
10  2013     1     1      558            600        -2      753            745         8 AA         301 N3ALAA   LGA    ORD        138      733     6      0
# ... with 336,766 more rows, and 2 more variables: time_hour <dttm>, vars <int>
> mutate(flights, gain = arr_delay - dep_delay, speed = distance/air_time * 60)
# A tibble: 336,776 x 22
    year month   day dep_time sched_dep_time dep_delay arr_time sched_arr_time arr_delay carrier flight tailnum origin dest  air_time distance  hour minute
   <int> <int> <int>    <int>          <int>     <dbl>    <int>          <int>     <dbl> <chr>    <int> <chr>   <chr>  <chr>    <dbl>    <dbl> <dbl>  <dbl>
 1  2013     1     1      517            515         2      830            819        11 UA        1545 N14228  EWR    IAH        227     1400     5     15
 2  2013     1     1      533            529         4      850            830        20 UA        1714 N24211  LGA    IAH        227     1416     5     29
 3  2013     1     1      542            540         2      923            850        33 AA        1141 N619AA  JFK    MIA        160     1089     5     40
 4  2013     1     1      544            545        -1     1004           1022       -18 B6         725 N804JB  JFK    BQN        183     1576     5     45
 5  2013     1     1      554            600        -6      812            837       -25 DL         461 N668DN  LGA    ATL        116      762     6      0
 6  2013     1     1      554            558        -4      740            728        12 UA        1696 N39463  EWR    ORD        150      719     5     58
 7  2013     1     1      555            600        -5      913            854        19 B6         507 N516JB  EWR    FLL        158     1065     6      0
```

```
> mutate(flights, gain = arr_delay - dep_delay, gain_per_hour = gain/(air_time/60))
# A tibble: 336,776 x 22
    year month   day dep_time sched_dep_time dep_delay arr_time sched_arr_time arr_delay carrier flight tailnum origin dest  air_time distance  hour minute
   <int> <int> <int>    <int>          <int>     <dbl>    <int>          <int>     <dbl> <chr>    <int> <chr>   <chr>  <chr>    <dbl>    <dbl> <dbl>  <dbl>
 1  2013     1     1      517            515         2      830            819        11 UA        1545 N14228  EWR    IAH        227     1400     5     15
 2  2013     1     1      533            529         4      850            830        20 UA        1714 N24211  LGA    IAH        227     1416     5     29
 3  2013     1     1      542            540         2      923            850        33 AA        1141 N619AA  JFK    MIA        160     1089     5     40
 4  2013     1     1      544            545        -1     1004           1022       -18 B6          725 N804JB  JFK    BQN        183     1576     5     45
 5  2013     1     1      554            600        -6      812            837       -25 DL          461 N668DN  LGA    ATL        116      762     6      0
 6  2013     1     1      554            558        -4      740            728        12 UA        1696 N39463  EWR    ORD        150      719     5     58
 7  2013     1     1      555            600        -5      913            854        19 B6          507 N516JB  EWR    FLL        158     1065     6      0
 8  2013     1     1      557            600        -3      709            723       -14 EV        5708 N829AS  LGA    IAD         53      229     6      0
 9  2013     1     1      557            600        -3      838            846        -8 B6           79 N593JB  JFK    MCO        140      944     6      0
10  2013     1     1      558            600        -2      753            745         8 AA         301 N3ALAA  LGA    ORD        138      733     6      0
# ... with 336,766 more rows, and 4 more variables: time_hour <dttm>, vars <int>, gain <dbl>, gain_per_hour <dbl>
> transmute(flights, gain = arr_delay - dep_delay, gain_per_hour = gain/(air_time/60))
# A tibble: 336,776 x 2
    gain gain_per_hour
   <dbl>         <dbl>
 1     9          2.38
 2    16          4.23
 3    31         11.6
 4   -17         -5.57
 5   -19         -9.83
 6    16          6.4
 7    24          9.11
 8   -11        -12.5
 9    -5         -2.14
10    10          4.35
# ... with 336,766 more rows
> summarise(flights, delay= mean(dep_delay, na.rm = TRUE))
# A tibble: 1 x 1
  delay
  <dbl>
1  12.6
> sample_n(flights, 10)
# A tibble: 10 x 20
    year month   day dep_time sched_dep_time dep_delay arr_time sched_arr_time arr_delay carrier flight tailnum origin dest  air_time distance  hour minute
   <int> <int> <int>    <int>          <int>     <dbl>    <int>          <int>     <dbl> <chr>    <int> <chr>   <chr>  <chr>    <dbl>    <dbl> <dbl>  <dbl>
 1  2013     9    28      741            738         3      823            835       -12 EV        3847 N15555  EWR    BDL         23      116     7     38
 2  2013     8     1     1852           1755        57     2226           2114        72 UA         212 N561UA  EWR    SFO        348     2565    17     55
```

```
> by_tailnum<- group_by(flights, tailnum)
> delay <- summarise(by_tailnum, count=n(), dist = mean(distance, na.rm = TRUE), delay = mean(arr_delay, na.rm = TRUE))
> delay <- filter(delay, count >20, dist <2000)
> destinations <- group_by(flights,dest)
> summarise(destinations, planes = n_distinct(tailnum), flights= n())
# A tibble: 105 x 3
   dest  planes flights
   <chr>  <int>   <int>
 1 ABQ      108     254
 2 ACK       58     265
 3 ALB      172     439
 4 ANC        6       8
 5 ATL     1180   17215
 6 AUS      993    2439
 7 AVL      159     275
 8 BDL      186     443
 9 BGR       46     375
10 BHM       45     297
# ... with 95 more rows
> daily <- group_by(flights, year, month, day)
> (per_day   <- summarise(daily, flights = n()))
# A tibble: 365 x 4
# Groups:   year, month [12]
    year month   day flights
   <int> <int> <int>   <int>
 1  2013     1     1     842
 2  2013     1     2     943
 3  2013     1     3     914
 4  2013     1     4     915
 5  2013     1     5     720
 6  2013     1     6     832
 7  2013     1     7     933
 8  2013     1     8     899
 9  2013     1     9     902
10  2013     1    10     932
# ... with 355 more rows
```

```
> (per_month <- summarise(per_day, flights = sum(flights)))
# A tibble: 12 x 3
# Groups:   year [1]
    year month flights
   <int> <int>   <int>
 1  2013     1   27004
 2  2013     2   24951
 3  2013     3   28834
 4  2013     4   28330
 5  2013     5   28796
 6  2013     6   28243
 7  2013     7   29425
 8  2013     8   29327
 9  2013     9   27574
10  2013    10   28889
11  2013    11   27268
12  2013    12   28135
> (per_year  <- summarise(per_month, flights = sum(flights)))
# A tibble: 1 x 2
   year flights
  <int>   <int>
1  2013  336776
```

```
> select(flights, year)
# A tibble: 336,776 x 1
    year
   <int>
 1  2013
 2  2013
 3  2013
 4  2013
 5  2013
 6  2013
 7  2013
 8  2013
 9  2013
10  2013
# ... with 336,766 more rows
> select(flights, 1)
# A tibble: 336,776 x 1
    year
   <int>
 1  2013
 2  2013
 3  2013
 4  2013
 5  2013
 6  2013
 7  2013
 8  2013
 9  2013
10  2013
# ... with 336,766 more rows
> year <- 5
> select(flights, year)
# A tibble: 336,776 x 1
    year
   <int>
 1  2013
 2  2013
 3  2013
 4  2013
 5  2013
 6  2013
```

```
> year <-  dep
> select(flights, starts_with(year))
# A tibble: 336,776 x 2
   dep_time dep_delay
      <int>     <dbl>
 1      517         2
 2      533         4
 3      542         2
 4      544        -1
 5      554        -6
 6      554        -4
 7      555        -5
 8      557        -3
 9      557        -3
10      558        -2
# ... with 336,766 more rows
> year <- 5
> select(flights, year, identity(year))
# A tibble: 336,776 x 2
    year sched_dep_time
   <int>          <int>
 1  2013            515
 2  2013            529
 3  2013            540
 4  2013            545
 5  2013            600
 6  2013            558
 7  2013            600
 8  2013            600
 9  2013            600
10  2013            600
# ... with 336,766 more rows
> vars <- c("year", "month")
> select(flights, vars, "day")
# A tibble: 336,776 x 2
    vars   day
   <int> <int>
 1  2013     1
 2  2013     1
 3  2013     1
 4  2013     1
```

```
> select(flights, vars, "day")
# A tibble: 336,776 x 2
    vars   day
   <int> <int>
 1  2013     1
 2  2013     1
 3  2013     1
 4  2013     1
 5  2013     1
 6  2013     1
 7  2013     1
 8  2013     1
 9  2013     1
10  2013     1
# ... with 336,766 more rows
> flights$vars <- flights$year
> vars <- c("year", "month", "day")
> select(flights, !! vars)
# A tibble: 336,776 x 3
    year month   day
   <int> <int> <int>
 1  2013     1     1
 2  2013     1     1
 3  2013     1     1
 4  2013     1     1
 5  2013     1     1
 6  2013     1     1
 7  2013     1     1
 8  2013     1     1
 9  2013     1     1
10  2013     1     1
# ... with 336,766 more rows
> df <- select(flights, year:dep_time)
> mutate(df, "year", 2)
# A tibble: 336,776 x 6
    year month   day dep_time `"year"`   `2`
   <int> <int> <int>    <int> <chr>    <dbl>
 1  2013     1     1      517 year         2
 2  2013     1     1      533 year         2
 3  2013     1     1      542 year         2
```

```
> mutate(df, year + 10)
# A tibble: 336,776 x 5
    year month   day dep_time `year + 10`
   <int> <int> <int>    <int>       <dbl>
 1  2013     1     1      517        2023
 2  2013     1     1      533        2023
 3  2013     1     1      542        2023
 4  2013     1     1      544        2023
 5  2013     1     1      554        2023
 6  2013     1     1      554        2023
 7  2013     1     1      555        2023
 8  2013     1     1      557        2023
 9  2013     1     1      557        2023
10  2013     1     1      558        2023
# ... with 336,766 more rows
> var <- seq(1, nrow(df))
> mutate(df, new = var)
# A tibble: 336,776 x 5
    year month   day dep_time   new
   <int> <int> <int>    <int> <int>
 1  2013     1     1      517     1
 2  2013     1     1      533     2
 3  2013     1     1      542     3
 4  2013     1     1      544     4
 5  2013     1     1      554     5
 6  2013     1     1      554     6
 7  2013     1     1      555     7
 8  2013     1     1      557     8
 9  2013     1     1      557     9
10  2013     1     1      558    10
# ... with 336,766 more rows
> group_by(df, month)
# A tibble: 336,776 x 4
# Groups:   month [12]
    year month   day dep_time
   <int> <int> <int>    <int>
 1  2013     1     1      517
 2  2013     1     1      533
 3  2013     1     1      542
 4  2013     1     1      544
 5  2013     1     1      554
```

```
> group_by(df, month = as.factor(month))
# A tibble: 336,776 x 4
# Groups:   month [12]
    year month   day dep_time
   <int> <fct> <int>    <int>
 1  2013 1         1      517
 2  2013 1         1      533
 3  2013 1         1      542
 4  2013 1         1      544
 5  2013 1         1      554
 6  2013 1         1      554
 7  2013 1         1      555
 8  2013 1         1      557
 9  2013 1         1      557
10  2013 1         1      558
# ... with 336,766 more rows
> group_by(df, day_binned = cut(day, 3))
# A tibble: 336,776 x 5
# Groups:   day_binned [3]
    year month   day dep_time day_binned
   <int> <int> <int>    <int> <fct>
 1  2013     1     1      517 (0.97,11]
 2  2013     1     1      533 (0.97,11]
 3  2013     1     1      542 (0.97,11]
 4  2013     1     1      544 (0.97,11]
 5  2013     1     1      554 (0.97,11]
 6  2013     1     1      554 (0.97,11]
 7  2013     1     1      555 (0.97,11]
 8  2013     1     1      557 (0.97,11]
 9  2013     1     1      557 (0.97,11]
10  2013     1     1      558 (0.97,11]
# ... with 336,766 more rows
> group_by(df, "month")
# A tibble: 336,776 x 5
# Groups:   "month" [1]
    year month   day dep_time `"month"`
   <int> <int> <int>    <int> <chr>
 1  2013     1     1      517 month
```

```
> group_by_at(df, vars(year:day))
# A tibble: 336,776 x 4
# Groups:   year, month, day [365]
    year month   day dep_time
   <int> <int> <int>    <int>
 1  2013     1     1      517
 2  2013     1     1      533
 3  2013     1     1      542
 4  2013     1     1      544
 5  2013     1     1      554
 6  2013     1     1      554
 7  2013     1     1      555
 8  2013     1     1      557
 9  2013     1     1      557
10  2013     1     1      558
# ... with 336,766 more rows
> a1 <- group_by(flights, year, month, day)
> a2 <- select(a1, arr_delay, dep_delay)
Adding missing grouping variables: `year`, `month`, `day`
> a4 <- filter(a3, arr > 30 | dep > 30)
> filter(
+   summarise(
+     select(
+       group_by(flights, year, month, day),
+       arr_delay, dep_delay
+     ),
+     arr = mean(arr_delay, na.rm = TRUE),
+     dep = mean(dep_delay, na.rm = TRUE)
+   ),
+   arr > 30 | dep > 30
+ )
Adding missing grouping variables: `year`, `month`, `day`
# A tibble: 49 x 5
# Groups:   year, month [11]
    year month   day   arr   dep
   <int> <int> <int> <dbl> <dbl>
 1  2013     1    16  34.2  24.6
 2  2013     1    31  32.6  28.7
```

```
# A tibble: 49 x 5
# Groups:   year, month [11]
    year month   day   arr   dep
   <int> <int> <int> <dbl> <dbl>
 1  2013     1    16  34.2  24.6
 2  2013     1    31  32.6  28.7
 3  2013     2    11  36.3  39.1
 4  2013     2    27  31.3  37.8
 5  2013     3     8  85.9  83.5
 6  2013     3    18  41.3  30.1
 7  2013     4    10  38.4  33.0
 8  2013     4    12  36.0  34.8
 9  2013     4    18  36.0  34.9
10  2013     4    19  47.9  46.1
# ... with 39 more rows
> flights %>%
+   group_by(year, month, day) %>%
+   select(arr_delay, dep_delay) %>%
+   summarise(
+     arr = mean(arr_delay, na.rm = TRUE),
+     dep = mean(dep_delay, na.rm = TRUE)
+   ) %>%
+   filter(arr > 30 | dep > 30)
Adding missing grouping variables: `year`, `month`, `day`
# A tibble: 49 x 5
# Groups:   year, month [11]
    year month   day   arr   dep
   <int> <int> <int> <dbl> <dbl>
 1  2013     1    16  34.2  24.6
 2  2013     1    31  32.6  28.7
 3  2013     2    11  36.3  39.1
 4  2013     2    27  31.3  37.8
 5  2013     3     8  85.9  83.5
 6  2013     3    18  41.3  30.1
 7  2013     4    10  38.4  33.0
 8  2013     4    12  36.0  34.8
 9  2013     4    18  36.0  34.9
10  2013     4    19  47.9  46.1
# ... with 39 more rows
> |
```

```
> require(data.table)
> example(data.table)

dt.tbl> ## Not run:
dt.tbl> ##D example(data.table)  # to run these examples yourself
dt.tbl> ## End(Not run)
dt.tbl> DF = data.frame(x=rep(c("b","a","c"),each=3), y=c(1,3,6), v=1:9)

dt.tbl> DT = data.table(x=rep(c("b","a","c"),each=3), y=c(1,3,6), v=1:9)

dt.tbl> DF
  x y v
1 b 1 1
2 b 3 2
3 b 6 3
4 a 1 4
5 a 3 5
6 a 6 6
7 c 1 7
8 c 3 8
9 c 6 9

dt.tbl> DT
   x y v
1: b 1 1
2: b 3 2
3: b 6 3
4: a 1 4
5: a 3 5
6: a 6 6
7: c 1 7
8: c 3 8
9: c 6 9

dt.tbl> identical(dim(DT), dim(DF))    # TRUE
[1] TRUE

dt.tbl> identical(DF$a, DT$a)          # TRUE
[1] TRUE

dt.tbl> is.list(DF)                    # TRUE
[1] TRUE

dt.tbl> is.list(DT)                    # TRUE
[1] TRUE

dt.tbl> is.data.frame(DT)              # TRUE
[1] TRUE

dt.tbl> tables()
   NAME NROW NCOL MB    COLS KEY
1:   DT    9    3  0   x,y,v
2:  kDT    9    3  0   x,y,v x,y
3:    X    2    3  0 x,v,foo
Total: 0MB

dt.tbl> # basic row subset operations
```

Conclusion : Thus, we have studied working with large datasets with dplyr and data.table.

Experiment No : 7

Aim : EDA with R - Range, Summary, Mean, Variance, Median, Standard Deviation, Histogram, Box plot, Scatter Graph.

Theory:

Exploratory data analysis or "EDA" is a critical first step in analyzing the data from an experiment. Here are the main reasons we use EDA:

- detection of mistakes
- checking of assumptions
- preliminary selection of appropriate models
- determining relationships among the explanatory variables, and
- assessing the direction and rough size of relationships between explanatory and outcome variables.

Loosely speaking, any method of looking at data that does not include formal statistical modeling and inference falls under the term exploratory data analysis.

Types of EDA:

Exploratory data analysis is generally cross-classified in two ways. First, each method is either non-graphical or graphical. And second, each method is either univariate or multivariate (usually just bivariate).
Non-graphical methods generally involve calculation of summary statistics, while graphical methods obviously summarize the data in a diagrammatic or pictorial way. Univariate methods look at one variable (data column) at a time, while multivariate methods look at two or more variables at a time to explore relationships. Usually our multivariate EDA will be bivariate (looking at exactly two variables), but occasionally it will involve three or more variables.

Univariate non-graphical EDA:

The data that come from making a particular measurement on all of the subjects in a sample represent our observations for a single characteristic such as age, gender, speed at a task, or response to a stimulus. We should think of these measurements as representing a "sample distribution" of the variable, which in turn more or less represents the "population distribution" of the variable. The usual goal of univariate non-graphical EDA is to better appreciate the "sample distribution" and also to make some tentative conclusions about what population distribution(s) is/are compatible with the sample distribution. Outlier detection is also a part of this analysis.

1. Categorical data
2. Quantitative data

Several statistics are commonly used as a measure of the spread of a distribution, including variance, standard deviation, and interquartile range. Spread is an indicator of how far away from the center we are still likely to find data values. The variance is a standard measure of spread. It is calculated for a list of numbers, e.g., the n observations of a particular measurement labeled x1 through xn, based on the n sample deviations (or just "deviations").A third measure of spread is the interquartile range. To define IQR, we first need to define the concepts of quartiles. The quartiles of a population or a sample are the three values which divide the distribution or observed data into even fourths.

Skewness is a measure of asymmetry. Kurtosis is a more subtle measure of peakedness compared to a Gaussian distribution.

Univariate graphical EDA:
If we are focusing on data from observation of a single variable on n subjects, i.e., a sample of size n, then in addition to looking at the various sample statistics discussed in the previous section, we also need to look graphically at the distribution of the sample. Non-graphical and graphical methods complement each other.

1. Histogram
2. Stem and leaf graph - A stem and leaf plot shows all data values and the shape of the distribution.
3. Boxplots - Boxplots show robust measures of location and spread as well as providing information about symmetry and outliers.
4. Quantile-normal plots - Quantile-Normal plots allow detection of non-normality and diagnosis of skewness and kurtosis.

Multivariate non-graphical EDA:

Multivariate non-graphical EDA techniques generally show the relationship between two or more variables in the form of either cross-tabulation or statistics.

Multivariate graphical EDA:

There are few useful techniques for graphical EDA of two categorical random variables. The only one used commonly is a grouped barplot with each group representing one level of one of the variables and each bar within a group representing the levels of the other variable.

Output:

Code:

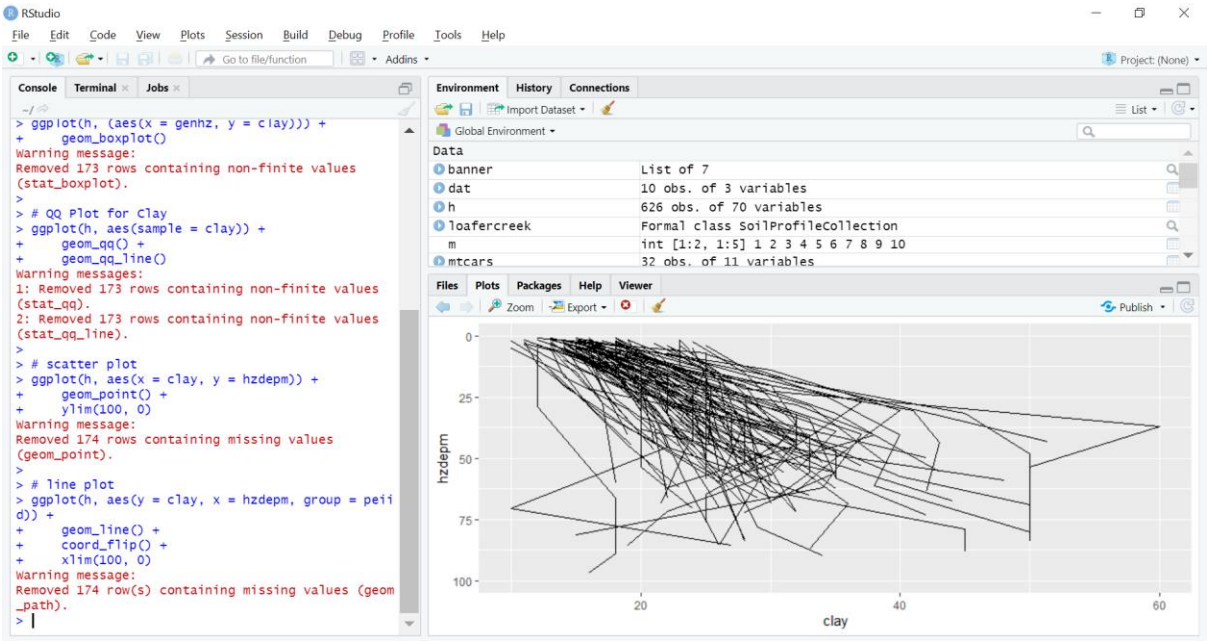
```
install.packages('aqp', dep=TRUE)
install.packages("soilDB")
```

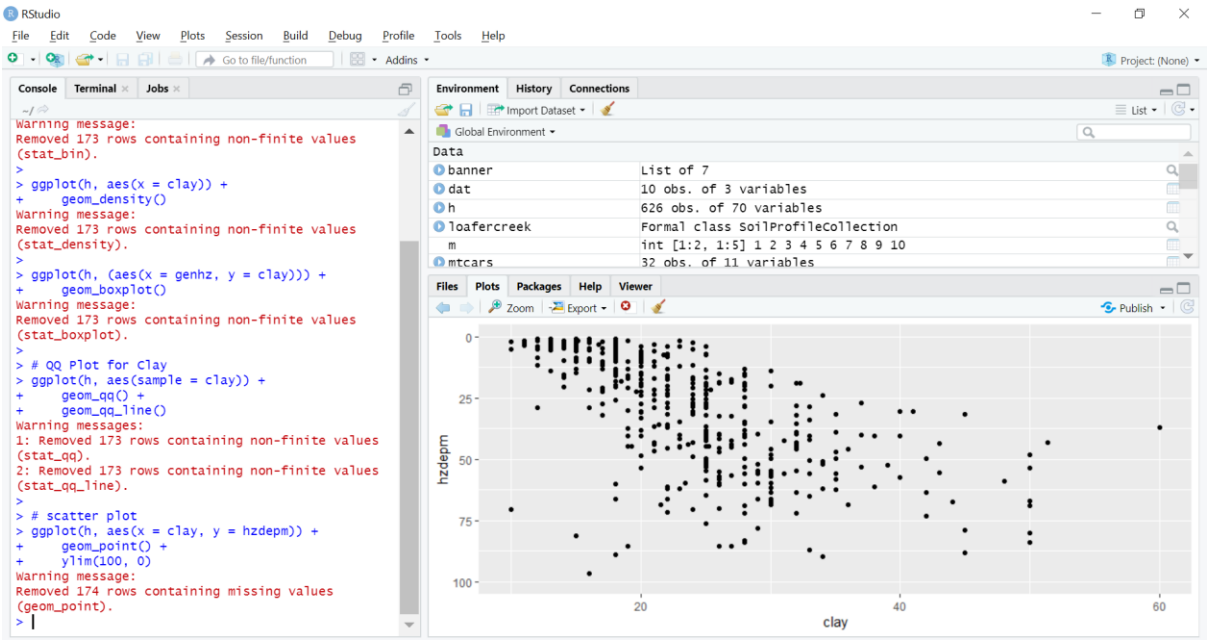
```
library(aqp)
library(soilDB)

# Load from the the loakercreek dataset
data("loafercreek")

# Construct generalized horizon designations
n <- c("A",
       "BAt",
       "Bt1",
       "Bt2",
       "Cr",
       "R")
# REGEX rules
p <- c("A",
       "BA|AB",
       "Bt|Bw",
       "Bt3|Bt4|2B|C",
       "Cr",
       "R")

# Compute genhz labels and add to loafercreek dataset
loafercreek$genhz <- generalize.hz(loafercreek$hzname, n, p)

# Extract the horizon table
h <- horizons(loafercreek)

# Examine the matching of pairing of the genhz label to the hznames
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)

# first remove missing values and create a new vector
```

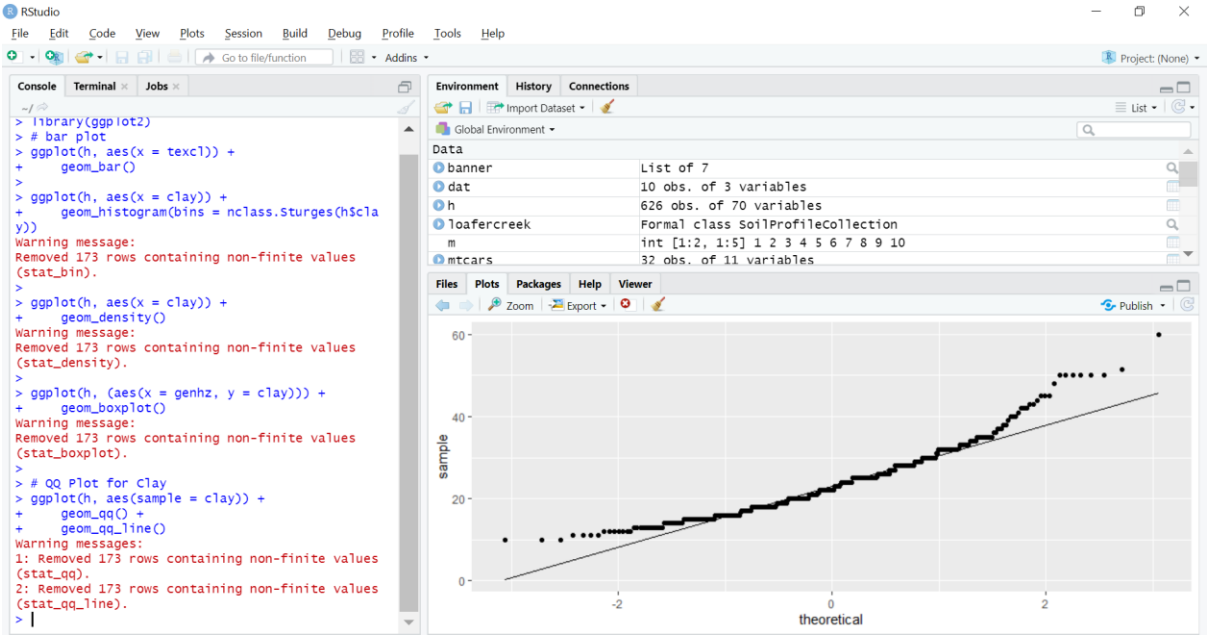
```
clay <- na.exclude(h$clay)
mean(clay)

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)))

aggregate(clay ~ genhz, data = h, mean)
aggregate(clay ~ genhz, data = h, median)
# or we could use the summary() function to get both the mean and median
aggregate(clay ~ genhz, data = h, summary)

var(h$clay, na.rm=TRUE)
sd(h$clay, na.rm = TRUE)
cv <- sd(clay) / mean(clay) * 100
cv

quantile(clay)
range(clay)
diff(range(clay))
IQR(clay)

install.packages("ggplot2")
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 = genhz, y = clay))) +
  geom_boxplot()

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)
```

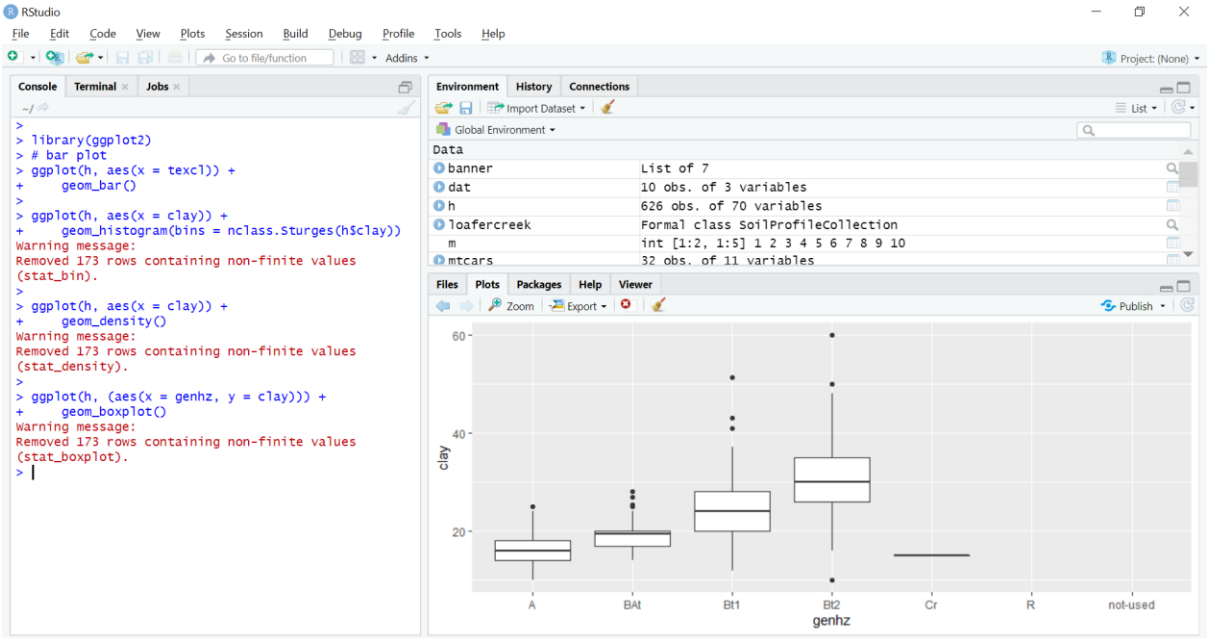
```
# scatter plot
ggplot(h, aes(x = clay, y = hzdepm)) +
  geom_point() +
  ylim(100, 0)
```

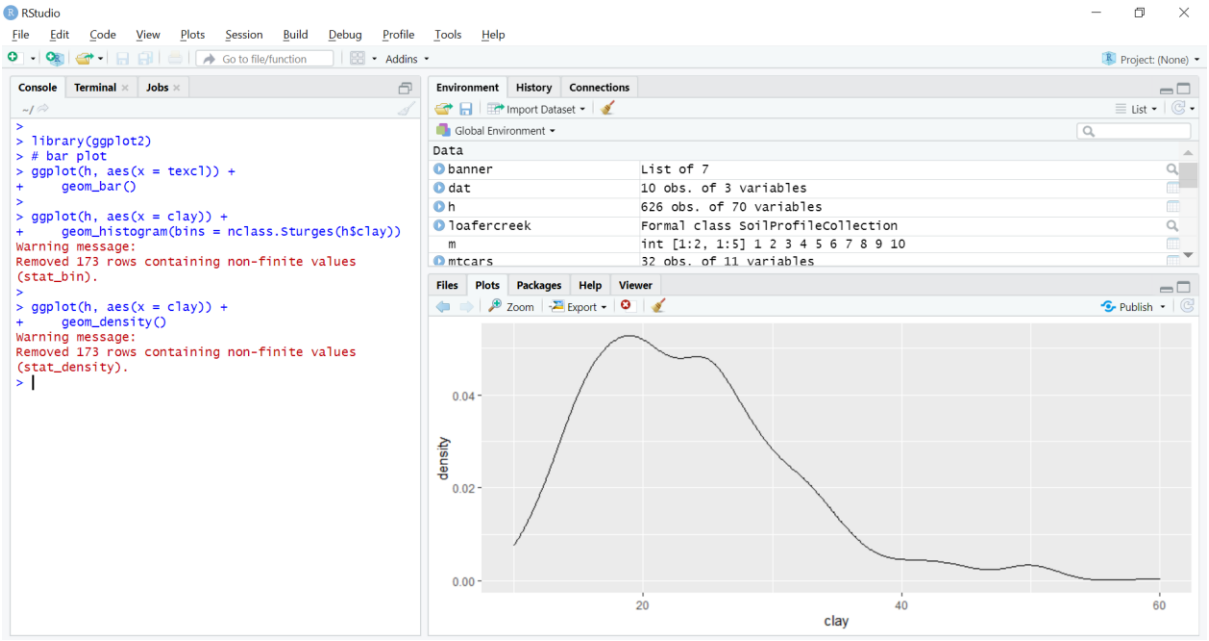
Screenshots:

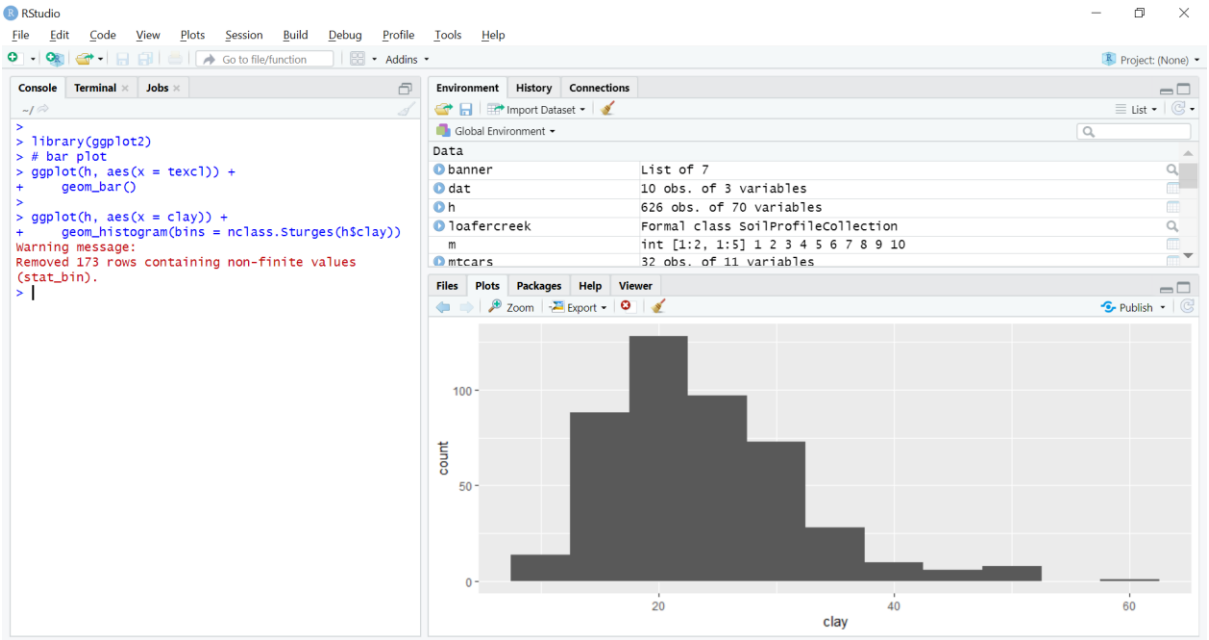





RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
Console Terminal Jobs
> library(ggplot2)
> # bar plot
> ggplot(h, aes(x = texcl)) +
+ geom_bar()
>
> ggplot(h, aes(x = clay)) +
+ geom_histogram(bins = nclass.Sturges(h$clay))
Warning message:
Removed 173 rows containing non-finite values (stat_bin).
>
> ggplot(h, aes(x = clay)) +
+ geom_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).
Environment History Connections
Import Dataset
Global Environment
Data
banner List of 7
dat 10 obs. of 3 variables
h 626 obs. of 70 variables
loafercreek Formal class SoilProfileCollection
m int [1:2, 1:5] 1 2 3 4 5 6 7 8 9 10
mtcars 32 obs. of 11 variables
Files Plots Packages Help Viewer
Zoom Export Publish
sample
theoretical
60
40
20
0
-2
0
2


RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
Console Terminal Jobs
> 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_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).
Environment History Connections
Import Dataset
Global Environment
Data
banner List of 7
dat 10 obs. of 3 variables
h 626 obs. of 70 variables
loafercreek Formal class SoilProfileCollection
m int [1:2, 1:5] 1 2 3 4 5 6 7 8 9 10
mtcars 32 obs. of 11 variables
Files Plots Packages Help Viewer
Zoom Export Publish
clay
genhz
60
40
20
A
BAt
Bt1
Bt2
Cr
R
not-used


RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
Console Terminal Jobs
> 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_density()
Warning message:
Removed 173 rows containing non-finite values
(stat_density).
Environment History Connections
Import Dataset
Global Environment
Data
banner List of 7
dat 10 obs. of 3 variables
h 626 obs. of 70 variables
loafercreek Formal class SoilProfileCollection
m int [1:2, 1:5] 1 2 3 4 5 6 7 8 9 10
mtcars 32 obs. of 11 variables
Files Plots Packages Help Viewer
Zoom Export Publish
density
0.04
0.02
0.00
20
40
60
clay


RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
Console Terminal Jobs
> 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).
Environment History Connections
Import Dataset
Global Environment
Data
banner List of 7
dat 10 obs. of 3 variables
h 626 obs. of 70 variables
loafercreek Formal class SoilProfileCollection
m int [1:2, 1:5] 1 2 3 4 5 6 7 8 9 10
mtcars 32 obs. of 11 variables
Files Plots Packages Help Viewer
Zoom Export Publish
count
100
50
0
20
40
60
clay

RStudio

File Edit Code View Plots Session Build Debug Profile Tools Help

Go to file/function | Addins | Project: (None)

Console Terminal Jobs

```
>
> library(ggplot2)
> # bar plot
> ggplot(h, aes(x = texcl)) +
+     geom_bar()
>
```

Environment History Connections

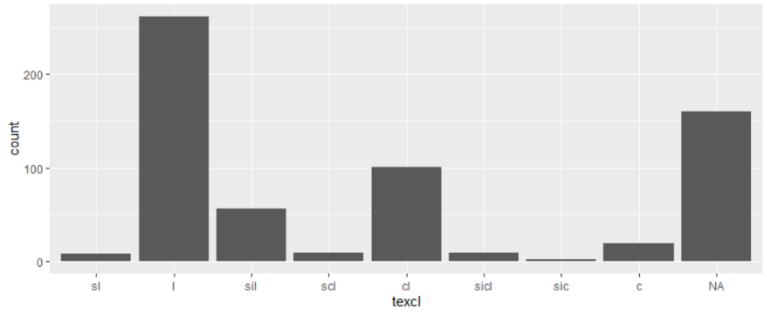
Import Dataset | List | Global Environment

Data

banner	List of 7
dat	10 obs. of 3 variables
h	626 obs. of 70 variables
loafercreek	Formal class SoilProfileCollection
m	int [1:2, 1:5] 1 2 3 4 5 6 7 8 9 10
mtcars	32 obs. of 11 variables

Files Plots Packages Help Viewer

Zoom | Export | Publish



RStudio

File Edit Code View Plots Session Build Debug Profile Tools Help

Console Terminal Jobs

```
> # 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, which 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 sicl  sc
  A         0  0   0   0    0   0   0    0    0   6   0    0  78  27   0   0   0    0   0
  BAt       0  0   0   0    0   0   0    0    0   1   0    0  31   4   0   0   2    1   0
  Bt1       0  0   0   0    0   0   0    0    0   1   0    0 125  20   0   4  46    5   0
  Bt2       0  0   0   0    0   0   0    0    0   0   0    0  28   5   0   5  52    3   0
  Cr        0  0   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    0   0
  not-used  0  0   0   0    0   0   0    0    0   0   0    0   0   0   0   0   1    0   0

          sic   c
  A         0   0
  BAt       0   0
  Bt1       1   2
  Bt2       1  16
  Cr        0   1
  R         0   0
  not-used  0   0
> 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   0 111
  BAt       0  0   0   0    0   0   0    0    0   1   0    0  31   4   0   0   2    1   0   0  39
  Bt1       0  0   0   0    0   0   0    0    0   1   0    0 125  20   0   4  46    5   0   1 204
  Bt2       0  0   0   0    0   0   0    0    0   0   0    0  28   5   0   5  52    3   0   1 110
  Cr        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
  not-used  0  0   0   0    0   0   0    0    0   0   0    0   0   0   0   0   1    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|  cl| chl| cl| gr|  l| pg| scl| sic| sicl| sil| sl| spm|
```

Environment History Connections

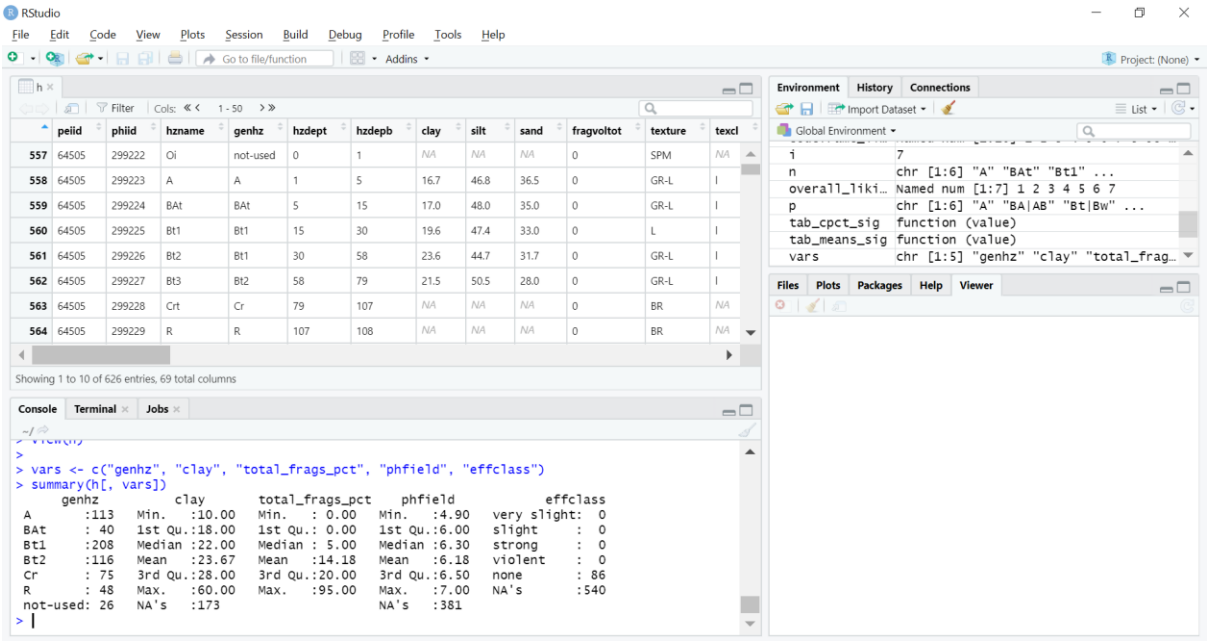
Import Dataset | List | Global Environment

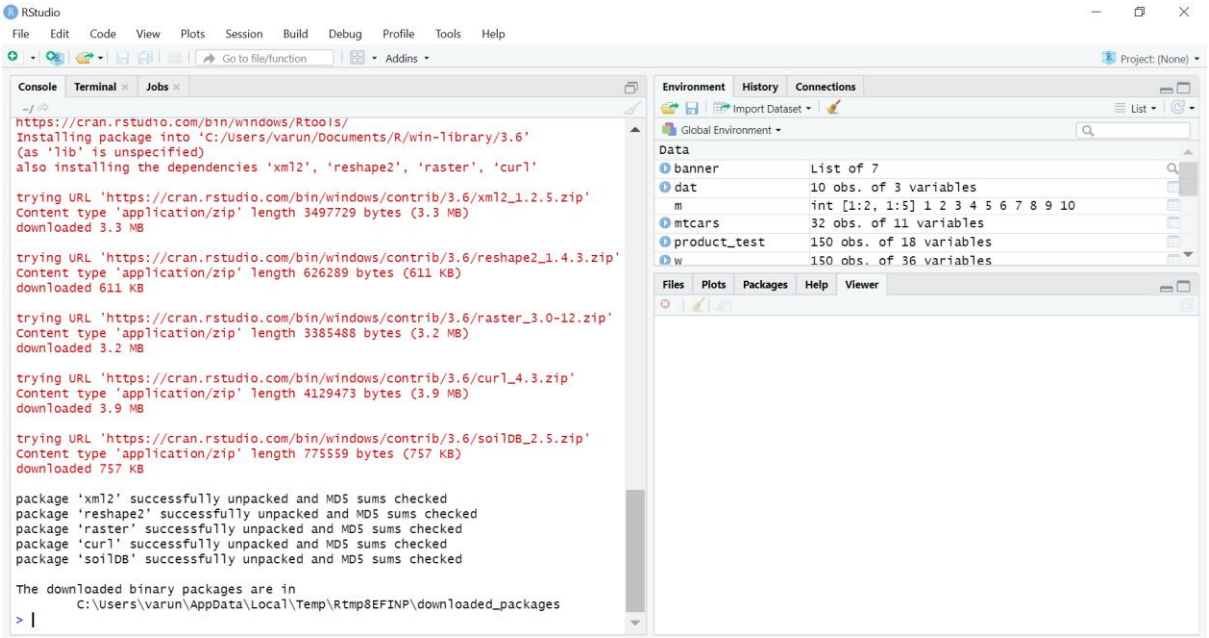
xlist	List of 3

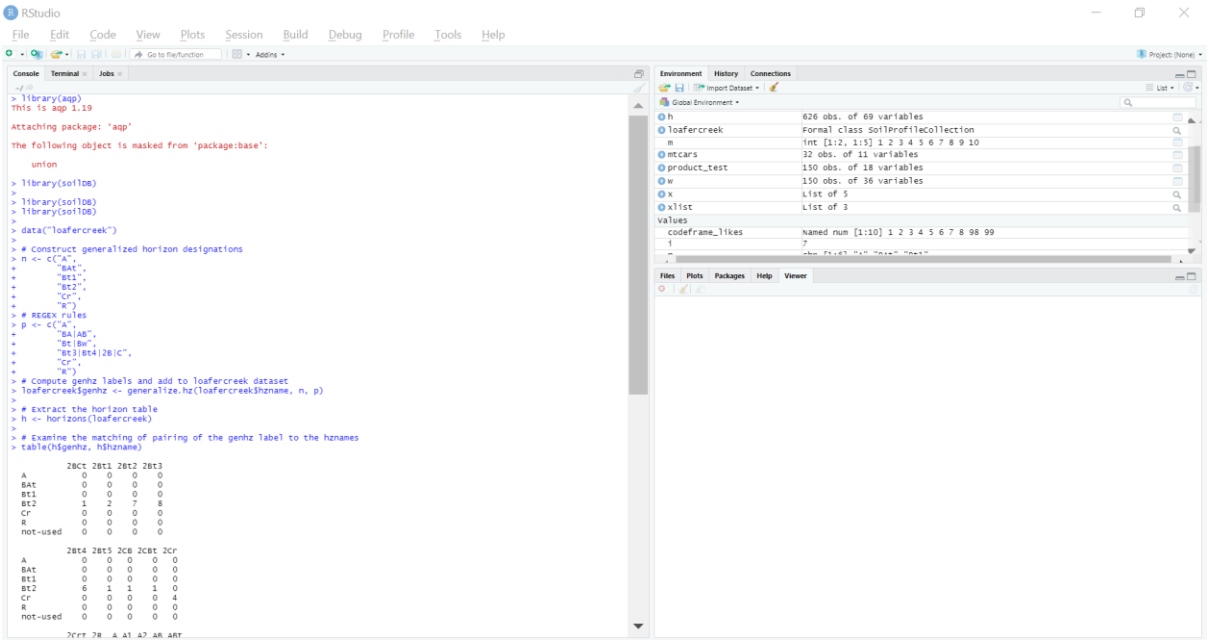
values

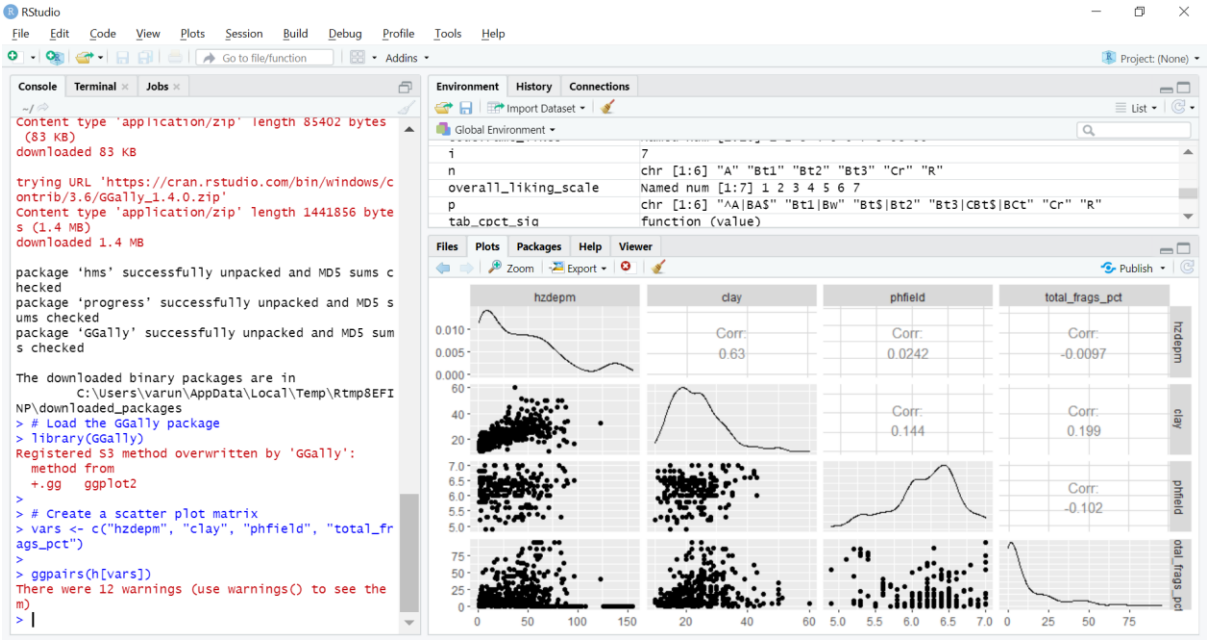
clay	num [1:453] 16.7 17 19.6 23.6 21.5 ...
codeframe_likes	Named num [1:10] 1 2 3 4 5 6 7 8 98 99
i	7
n	chr [1:6] "A" "Bt1" "Bt2" "Bt3" "Cr" "R"
overall_liking_scale	Named num [1:7] 1 2 3 4 5 6 7
p	chr [1:6] "^A|BA$" "Bt1\|Bw" "Bt$|Bt2" "Bt3|CBt$|BCt" "Cr"...
tab_cpct_sig	function (value)
tab_means_sig	function (value)
vars	chr [1:5] "genhz" "clay" "total_frags_pct" "phfield" "ef"...
y	int [1:3] 10 11 12

Files Plots Packages Help Viewer


RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
Go to file/function
Addins
Project: (None)
h
Filter
Cols: 1 - 50
peiid phiid hzname genhz hzdept hzdepb clay silt sand fragvoltot texture texcl
557 64505 299222 Oi not-used 0 1 NA NA NA 0 SPM NA
558 64505 299223 A A 1 5 16.7 46.8 36.5 0 GR-L l
559 64505 299224 BAt BAt 5 15 17.0 48.0 35.0 0 GR-L l
560 64505 299225 Bt1 Bt1 15 30 19.6 47.4 33.0 0 L l
561 64505 299226 Bt2 Bt1 30 58 23.6 44.7 31.7 0 GR-L l
562 64505 299227 Bt3 Bt2 58 79 21.5 50.5 28.0 0 GR-L l
563 64505 299228 Crt Cr 79 107 NA NA NA 0 BR NA
564 64505 299229 R R 107 108 NA NA NA 0 BR NA
Showing 1 to 10 of 626 entries, 69 total columns
Environment History Connections
Import Dataset
List
Global Environment
i 7
n chr [1:6] "A" "BAt" "Bt1" ...
overall_liki... Named num [1:7] 1 2 3 4 5 6 7
p chr [1:6] "A" "BA|AB" "Bt|Bw" ...
tab_cpct_sig function (value)
tab_means_sig function (value)
vars chr [1:5] "genhz" "clay" "total_frag...
Files Plots Packages Help Viewer
Console Terminal Jobs
~/
> 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
>


RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
Go to file/function
Addins
Project: (None)
Console Terminal Jobs
~/
https://cran.rstudio.com/bin/windows/Rtools/
Installing package into 'C:/Users/varun/Documents/R/win-library/3.6'
(as 'lib' is unspecified)
also installing the dependencies 'xml2', 'reshape2', 'raster', 'curl'
trying URL 'https://cran.rstudio.com/bin/windows/contrib/3.6/xml2_1.2.5.zip'
Content type 'application/zip' length 3497729 bytes (3.3 MB)
downloaded 3.3 MB
trying URL 'https://cran.rstudio.com/bin/windows/contrib/3.6/reshape2_1.4.3.zip'
Content type 'application/zip' length 626289 bytes (611 KB)
downloaded 611 KB
trying URL 'https://cran.rstudio.com/bin/windows/contrib/3.6/raster_3.0-12.zip'
Content type 'application/zip' length 3385488 bytes (3.2 MB)
downloaded 3.2 MB
trying URL 'https://cran.rstudio.com/bin/windows/contrib/3.6/curl_4.3.zip'
Content type 'application/zip' length 4129473 bytes (3.9 MB)
downloaded 3.9 MB
trying URL 'https://cran.rstudio.com/bin/windows/contrib/3.6/soilDB_2.5.zip'
Content type 'application/zip' length 775559 bytes (757 KB)
downloaded 757 KB
package 'xml2' successfully unpacked and MD5 sums checked
package 'reshape2' successfully unpacked and MD5 sums checked
package 'raster' successfully unpacked and MD5 sums checked
package 'curl' successfully unpacked and MD5 sums checked
package 'soilDB' successfully unpacked and MD5 sums checked
The downloaded binary packages are in
C:\Users\varun\AppData\Local\Temp\Rtmp8EFINP\downloaded_packages
>
Environment History Connections
Import Dataset
List
Global Environment
Data
banner List of 7
dat 10 obs. of 3 variables
m int [1:2, 1:5] 1 2 3 4 5 6 7 8 9 10
mtcars 32 obs. of 11 variables
product_test 150 obs. of 18 variables
w 150 obs. of 36 variables
Files Plots Packages Help Viewer





Conclusion: Thus, we have studied EDA in R.

Experiment No. 8

Aim : Basic and Advanced graphics in R

Theory :

One of the main reasons data analysts turn to R is for its strong graphic capabilities. These include density plots (histograms and kernel density plots), dot plots, bar charts (simple, stacked, grouped), line charts, pie charts (simple, annotated, 3D), box plots (simple, notched, violin plots, bagplots) and Scatter Plots (simple, with fit lines, scatterplot matrices, high density plots, and 3D plots).

The R base function plot() can be used to create graphs. Some of them are :

- A scatter plot can be created using the function plot(x, y). The function lm() will be used to fit linear models between y and x. A regression line will be added on the plot using the function abline(), which takes the output of lm() as an argument. You can also add a smoothing line using the function loess().
- Boxplots are a measure of how well distributed is the data in a data set. It divides the data set into three quartiles. This graph represents the minimum, maximum, median, first quartile and third quartile in the data set. It is also useful in comparing the distribution of data across data sets by drawing boxplots for each of them. Boxplots are created in R by using the boxplot() function.
- A histogram represents the frequencies of values of a variable bucketed into ranges. Histogram is similar to a bar chart but the difference is it groups the values into continuous ranges. Each bar in histogram represents the height of the number of values present in that range. R creates a histogram using hist() function. This function takes a vector as an input and uses some more parameters to plot histograms.
- A line chart is a graph that connects a series of points by drawing line segments between them. These points are ordered in one of their coordinate (usually the x-coordinate) values. Line charts are usually used in identifying the trends in data.The plot() function in R is used to create the line graph.
- A bar chart represents data in rectangular bars with length of the bar proportional to the value of the variable. R uses the function barplot() to create bar charts. R can draw both vertical and Horizontal bars in the bar chart. In the bar chart each of the bars can be given different colors.
- A pie-chart is a representation of values as slices of a circle with different colors. The slices are labeled and the numbers corresponding to each slice is also represented in the chart. In R the pie chart is created using the pie() function which takes positive numbers as a vector input.

Output :

Code :

```
City = read.csv("http://course1.winona.edu/bdeppa/DSCI%20415/Data/City77.csv")
# Copy city names to row names, i.e. observation labels
row.names(City) = City$City
# Remove city name column from the data frame
City = City[,-1]
names(City)

hist(City$medinc,prob=T,nclass=20,col="pink",main="Histogram of Median
Income",xlab="Median Income")
lines(density(City$medinc),lty=2,col="red",lwd=2)

require(ggplot2)

a = ggplot(City,aes(medinc))
a + geom_histogram(binwidth=1000,aes(y=..density..),fill="pink",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="pink",lwd=2)
abline(lm(infmort~poverty,data=City),lwd=2)
# identify(City$poverty,City$infmort,labels=row.names(City))
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)) # sets a plotting region for with 3 rows and 2 columns
boxplot(split(Olives$Oleic,Olives$Area.Name),xlab="Area",ylab="Oleic Acid")
boxplot(split(Olives$Linoleic,Olives$Area.Name),xlab="Area",ylab="Linoleic Acid")
boxplot(split(Olives$Strearic,Olives$Area.Name),xlab="Area",ylab="Strearic Acid")
boxplot(split(Olives$Linolenic,Olives$Area.Name),xlab="Area",ylab="Linolenic Acid")
boxplot(split(Olives$Eicosanoic,Olives$Area.Name),xlab="Area",ylab="Eicosanoic Acid")
boxplot(split(Olives$Eicosenoic,Olives$Area.Name),xlab="Area",ylab="Eicosenoic Acid")

require(s20x)

boxqq(Oleic~Region.Name,data=Olives)

a = ggplot(Olives,aes(Area.Name,Oleic))
a + geom_boxplot(fill="lightblue") + xlab("Area Name") +
  ylab("Oleic Acid") + ggtitle("Oleic Acid vs. Growing Area")
```

```
require(violinmplot)

violinmplot(Area.Name~Oleic,xlab="Oleic Acid",ylab="Area
Name",data=Olives,horizontal=T)

a = ggplot(Olives,aes(Area.Name,Oleic))
a + geom_violin(fill="lightblue") + xlab("Area Name") +
  ylab("Oleic Acid") + ggtitle("Oleic Acid vs. Growing Area")

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)
```

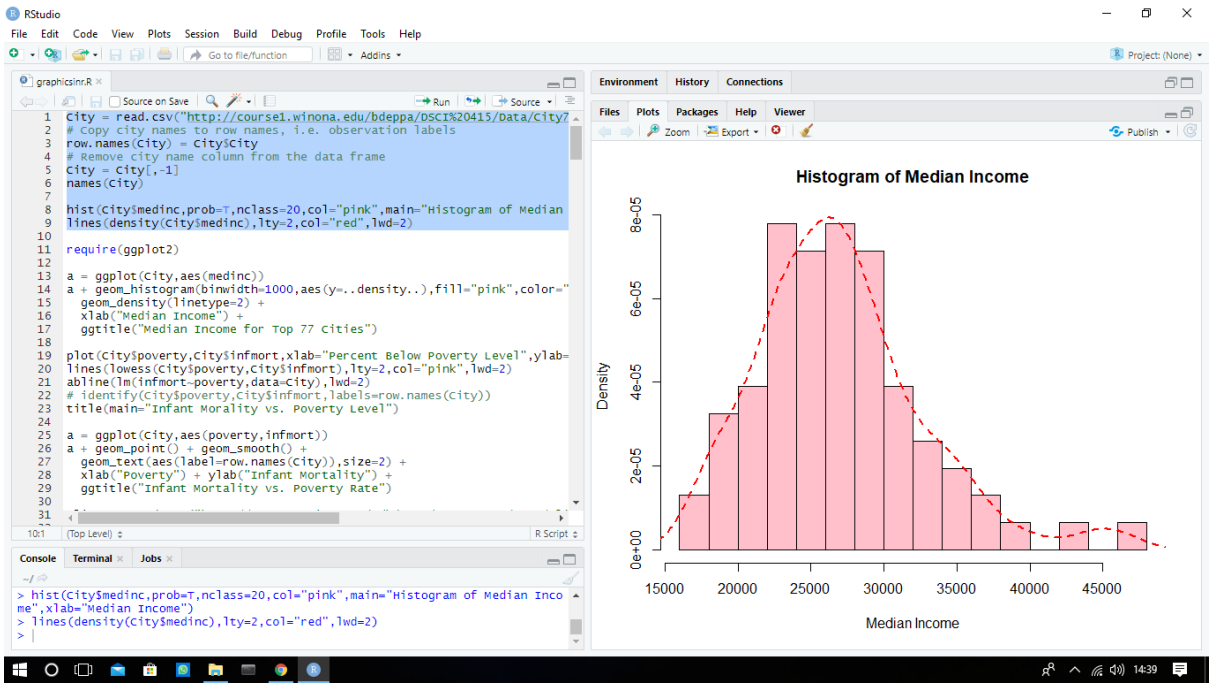
```
mosaicplot(~Dept+Admit+Gender,data=UCBAdmissions,color=T)

mosaicplot(~Gender+Dept+Admit,data=UCBAdmissions,color=T)

mosaicplot(~Admit+Dept+Gender,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/Boston.txt")
names(Boston)

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")

corrplot(Boston.corr,order="FPC")

corrplot(Boston.corr,order="hclust")

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,data=iris)

coplot(Sepal.Width~Sepal.Length|Petal.Width*Petal.Length,number=c(3,3),
      overlap=.5,col=as.numeric(iris$Species),pch=as.numeric(iris$Species)+1,data=iris)

Ozdata = read.csv("http://course1.winona.edu/bdeppa/DSCI%20415/Data/Ozdata.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=function(x,y,...)
panel.smooth(x,y,span=.6,...),data=Ozdata)


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="pink",main="Histogram of Bill Diagonal (mm)")
lines(d1)

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="pink")


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="red",
          shade=.75,box=F,scale=F,border=NA,expand=.9)
  })
  par(new=F,bg="white")
}
pairs.persp(Swiss[,1:3])

NHL = read.csv("http://course1.winona.edu/bdeppa/DSCI%20415/Data/NHL.csv")
names(NHL)

plot(NHL$GF,NHL$GA,xlab="Goals For",ylab="Goals Against",main="Plot of GA vs. GF
with Symbols = Wins")
symbols(NHL$GF,NHL$GA,circles=NHL$W,add=T,inches=.25)  # circles are too big if
inches is not specified!
symbols(NHL$GF,NHL$GA,circles=NHL$W,add=T,bg="lightblue",inches=.25)  # bg
option colors circles light blue
text(NHL$GF,NHL$GA,labels=NHL$TEAM,cex=.6)

kodiak.crab = read.csv("http://course1.winona.edu/bdeppa/DSCI%20415/Data/kodiak.csv")
survey.crab = read.csv("http://course1.winona.edu/bdeppa/DSCI%20415/Data/survey.csv")
attach(kodiak.crab)  # This file contains latitude and longitude of the island coastline.
attach(survey.crab)  # This file contains crab counts by type, year, and location (lat, long)

names(survey.crab)
```

```
table(year)

crab.plot <- function(lat,long,y,year,yname=deparse(substitute(y))){

symbols(long[year==year],lat[year==year],circles=y[year==year],inches=.1,bg="blue",xlab=
"Longitude",ylab="Latitude",
        main=paste("Circles Proportional to",yname))
  title(sub= paste("Year =",year))
  points(kodiak.crab$longit,kodiak.crab$latit,pch=".")
}
nr.pp = nrec/npots
crab.plot(lat,long,y=nr.pp,year=83,yname="Number of Recruits per Pot")

detach(survey.crab)
detach(kodiak.crab)  # if you attach, be sure to detach when finished!

attach(City)        # This command readies the data frame City for analysis.

names(City)          # The names command lists the names of the variables in a data set.

View(City)

city.mat <-cbind(popdens,pctblack,pcthisp,medinc,pct.pa,poverty,infmort,unempl)
stars(city.mat, key.loc = c(15,1.25),main = "Starplots for U.S.
Cities",label=row.names(City),cex=.7)

palette(rainbow(12, s = 0.6, v = 0.75))
stars(city.mat,len=.80,key.loc = c(15,1.25),main = "Starplots for U.S. Cities",draw.segments
= TRUE,label=row.names(City),cex=.7)

palette("default")

library(MASS)

parcoord(city.mat)
names(Olives)
olive.mat = Olives[,3:9]
palette(rainbow(12,s=0.6,v=0.75))
parcoord(olive.mat,col=as.numeric(Olives$Area.Name))
```
Screenshots:


RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
graphicsinr.R
City = read.csv("http://course1.winona.edu/bdeppa/DSCI%20415/Data/City7
# Copy city names to row names, i.e. observation labels
row.names(City) = City$City
# Remove city name column from the data frame
City = City[,-1]
names(City)
hist(City$medinc,prob=T,nclass=20,col="pink",main="Histogram of Median
lines(density(City$medinc),lty=2,col="red",lwd=2)
require(ggplot2)
a = ggplot(City,aes(medinc))
a + geom_histogram(binwidth=1000,aes(y=..density..),fill="pink",color="
geom_density(linetype=2) +
xlab("Median Income") +
ggtitle("Median Income for Top 77 Cities")
plot(City$poverty,City$infmort,xlab="Percent Below Poverty Level",ylab=
lines(lowess(City$poverty,City$infmort),lty=2,col="pink",lwd=2)
abline(lm(infmort~poverty,data=City),lwd=2)
# identify(City$poverty,City$infmort,labels=row.names(City))
title(main="Infant Morality vs. Poverty Level")
a = ggplot(City,aes(poverty,infmort))
a + geom_point() + geom_smooth() +
geom_text(aes(label=row.names(City)),size=2) +
xlab("Poverty") + ylab("Infant Mortality") +
ggtitle("Infant Mortality vs. Poverty Rate")
Console Terminal Jobs
> hist(City$medinc,prob=T,nclass=20,col="pink",main="Histogram of Median Inco
me",xlab="Median Income")
> lines(density(City$medinc),lty=2,col="red",lwd=2)
Environment History Connections
Files Plots Packages Help Viewer
Histogram of Median Income
Density
0e+00
2e-05
4e-05
6e-05
8e-05
15000
20000
25000
30000
35000
40000
45000
Median Income

RStudio

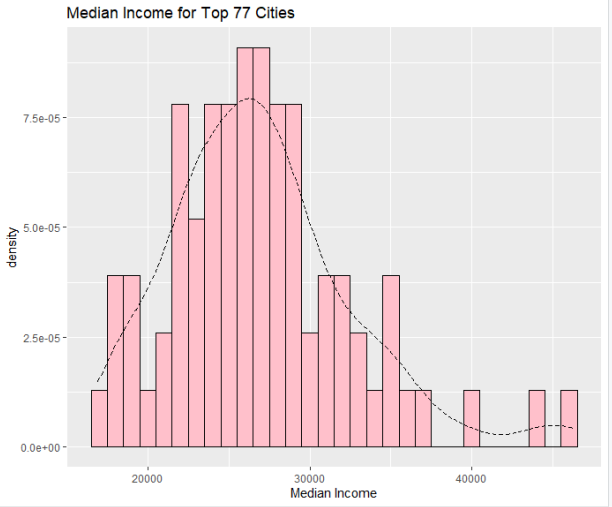
File Edit Code View Plots Session Build Debug Profile Tools Help

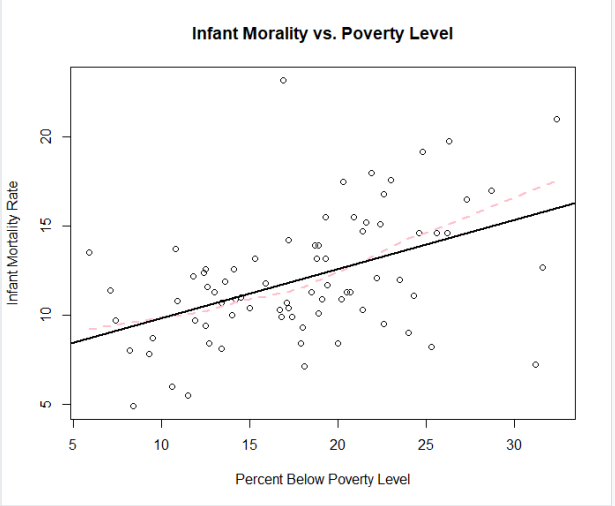
graphicsinr.R

```
City = read.csv("http://course1.winona.edu/bdeppa/DSCI%20415/Data/City7
# Copy city names to row names, i.e. observation labels
row.names(City) = City$City
# Remove city name column from the data frame
City = City[,-1]
names(City)

hist(City$medinc,prob=T,nclass=20,col="pink",main="Histogram of Median
lines(density(City$medinc),lty=2,col="red",lwd=2)

require(ggplot2)

a = ggplot(City,aes(medinc))
a + geom_histogram(binwidth=1000,aes(y=..density..),fill="pink",color="
  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=
lines(lowess(City$poverty,City$infmort),lty=2,col="pink",lwd=2)
abline(lm(infmort~poverty,data=City),lwd=2)
# identify(City$poverty,City$infmort,labels=row.names(City))
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")
```

Console Terminal Jobs

```
> abline(lm(infmort~poverty,data=City),lwd=2)
> # identify(City$poverty,City$infmort,labels=row.names(City))
> title(main="Infant Morality vs. Poverty Level")
>
```





RStudio

File Edit Code View Plots Session Build Debug Profile Tools Help

graphicsinr.R

```
23 title(main="Infant Morality vs. Poverty Level")
24
25 a = ggplot(City,aes(poverty,infmort))
26 a + geom_point() + geom_smooth() +
27   geom_text(aes(label=row.names(City)),size=2) +
28   xlab("Poverty") + ylab("Infant Mortality") +
29   ggtitle("Infant Mortality vs. Poverty Rate")
30
31 olives = read.csv("http://course1.winona.edu/bdeppa/DSCI%20415/Data/Oli
32 names(olives)
33
34 par(mfrow=c(3,2)) # sets a plotting region for with 3 rows and 2 column
35 boxplot(split(olives$Oleic,olives$Area.Name),xlab="Area",ylab="Oleic Ac
36 boxplot(split(olives$Linoleic,olives$Area.Name),xlab="Area",ylab="Linol
37 boxplot(split(olives$Strearic,olives$Area.Name),xlab="Area",ylab="Strea
38 boxplot(split(olives$Linolenic,olives$Area.Name),xlab="Area",ylab="Lino
39 boxplot(split(olives$Eicosanoic,olives$Area.Name),xlab="Area",ylab="Eic
40 boxplot(split(olives$Eicosenoic,olives$Area.Name),xlab="Area",ylab="Eic
41
42 require(s20x)
43
44 boxqq(Oleic~Region.Name,data=Olives)
45
46 a = ggplot(Olives,aes(Area.Name,Oleic))
47 a + geom_boxplot(fill="lightblue") + xlab("Area Name") +
48   ylab("Oleic Acid") + ggtitle("Oleic Acid vs. Growing Area")
49
50 require(violinmplot)
51
52 violinmplot(Area.Name~Oleic,xlab="Oleic Acid",ylab="Area Name",data=Oli
53
```

41:1 (Top Level) R Script

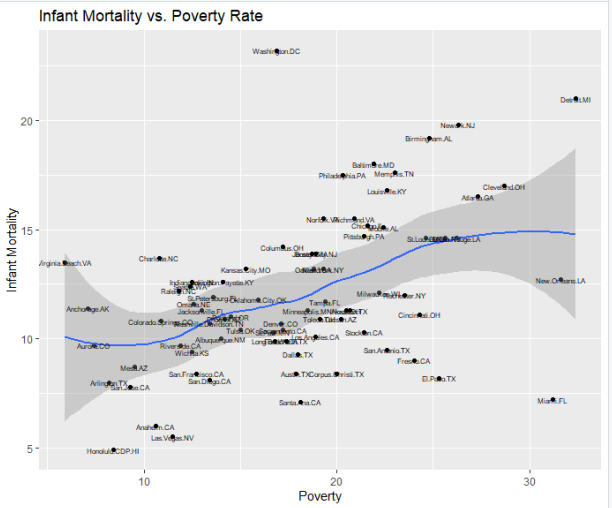
Console Terminal Jobs

```
oic Acid")
> boxplot(split(Olives$Eicosenoic,Olives$Area.Name),xlab="Area",ylab="Eicosen
oic Acid")
>
```

Environment History Connections

Files Plots Packages Help Viewer

Zoom Export Publish



14:42

RStudio

File Edit Code View Plots Session Build Debug Profile Tools Help

graphicsinr.R

```
23 title(main="Infant Morality vs. Poverty Level")
24
25 a = ggplot(City,aes(poverty,infmort))
26 a + geom_point() + geom_smooth() +
27   geom_text(aes(label=row.names(City)),size=2) +
28   xlab("Poverty") + ylab("Infant Mortality") +
29   ggtitle("Infant Mortality vs. Poverty Rate")
30
31 olives = read.csv("http://course1.winona.edu/bdeppa/DSCI%20415/Data/Oli
32 names(olives)
33
34 par(mfrow=c(3,2)) # sets a plotting region for with 3 rows and 2 column
35 boxplot(split(olives$Oleic,olives$Area.Name),xlab="Area",ylab="Oleic Ac
36 boxplot(split(olives$Linoleic,olives$Area.Name),xlab="Area",ylab="Linol
37 boxplot(split(olives$Strearic,olives$Area.Name),xlab="Area",ylab="Strea
38 boxplot(split(olives$Linolenic,olives$Area.Name),xlab="Area",ylab="Lino
39 boxplot(split(olives$Eicosanoic,olives$Area.Name),xlab="Area",ylab="Eic
40 boxplot(split(olives$Eicosenoic,olives$Area.Name),xlab="Area",ylab="Eic
41
42 require(s20x)
43
44 boxqq(Oleic~Region.Name,data=Olives)
45
46 a = ggplot(Olives,aes(Area.Name,Oleic))
47 a + geom_boxplot(fill="lightblue") + xlab("Area Name") +
48   ylab("Oleic Acid") + ggtitle("Oleic Acid vs. Growing Area")
49
50 require(violinmplot)
51
52 violinmplot(Area.Name~Oleic,xlab="Oleic Acid",ylab="Area Name",data=Oli
53
```

41:1 (Top Level) R Script

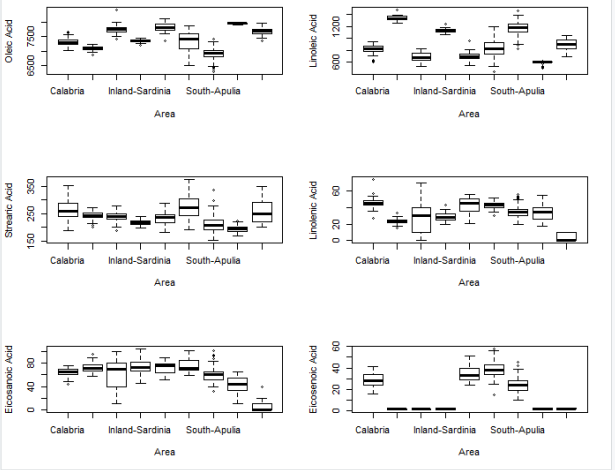
Console Terminal Jobs

```
oic Acid")
> boxplot(split(Olives$Eicosenoic,Olives$Area.Name),xlab="Area",ylab="Eicosen
oic Acid")
>
```

Environment History Connections

Files Plots Packages Help Viewer

Zoom Export Publish



14:43

RStudio

File Edit Code View Plots Session Build Debug Profile Tools Help

graphicsnr.R

```
23  title(main="Infant Morality vs. Poverty Level")
24
25  a = ggplot(City,aes(poverty,infmort))
26  a + geom_point() + geom_smooth() +
27    geom_text(aes(label=row.names(City)),size=2) +
28    xlab("Poverty") + ylab("Infant Mortality") +
29    ggtitle("Infant Mortality vs. Poverty Rate")
30
31  olives = read.csv("http://course1.winona.edu/bdeppa/DSCI%20415/Data/Oli
32  names(olives)
33
34  par(mfrow=c(3,2)) # sets a plotting region for with 3 rows and 2 column
35  boxplot(split(olives$Oleic,olives$Area.Name),xlab="Area",ylab="Oleic Ac
36  boxplot(split(olives$Linoleic,olives$Area.Name),xlab="Area",ylab="Linol
37  boxplot(split(olives$Strearic,olives$Area.Name),xlab="Area",ylab="Strea
38  boxplot(split(olives$Linolenic,olives$Area.Name),xlab="Area",ylab="Lino
39  boxplot(split(olives$Eicosanoic,olives$Area.Name),xlab="Area",ylab="Eic
40  boxplot(split(olives$Eicosenoic,olives$Area.Name),xlab="Area",ylab="Eic
41
42  require(s20x)
43
44  boxqq(Oleic~Region.Name,data=olives)
45
46  a = ggplot(olives,aes(Area.Name,Oleic))
47  a + geom_boxplot(fill="lightblue") + xlab("Area Name") +
48    ylab("Oleic Acid") + ggtitle("Oleic Acid vs. Growing Area")
49
50  require(violinmplot)
51
52  violinmplot(Area.Name~Oleic,xlab="Oleic Acid",ylab="Area Name",data=Oli
53
```

48:62 (Top Level) R Script

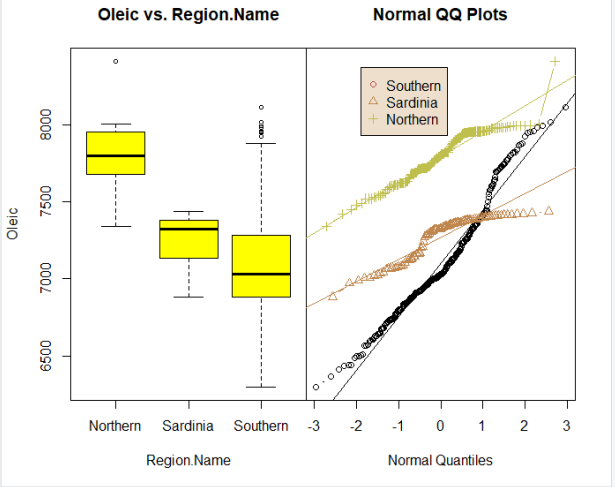
Console Terminal Jobs

```
> a = ggplot(olives,aes(Area.Name,Oleic))
> a + geom_boxplot(fill="lightblue") + xlab("Area Name") +
+   ylab("Oleic Acid") + ggtitle("Oleic Acid vs. Growing Area")
>
```

Environment History Connections

Files Plots Packages Help Viewer

Zoom Export Publish



14:43

RStudio

File Edit Code View Plots Session Build Debug Profile Tools Help

graphicsnr.R

```
23  title(main="Infant Morality vs. Poverty Level")
24
25  a = ggplot(City,aes(poverty,infmort))
26  a + geom_point() + geom_smooth() +
27    geom_text(aes(label=row.names(City)),size=2) +
28    xlab("Poverty") + ylab("Infant Mortality") +
29    ggtitle("Infant Mortality vs. Poverty Rate")
30
31  olives = read.csv("http://course1.winona.edu/bdeppa/DSCI%20415/Data/Oli
32  names(olives)
33
34  par(mfrow=c(3,2)) # sets a plotting region for with 3 rows and 2 column
35  boxplot(split(olives$Oleic,olives$Area.Name),xlab="Area",ylab="Oleic Ac
36  boxplot(split(olives$Linoleic,olives$Area.Name),xlab="Area",ylab="Linol
37  boxplot(split(olives$Strearic,olives$Area.Name),xlab="Area",ylab="Strea
38  boxplot(split(olives$Linolenic,olives$Area.Name),xlab="Area",ylab="Lino
39  boxplot(split(olives$Eicosanoic,olives$Area.Name),xlab="Area",ylab="Eic
40  boxplot(split(olives$Eicosenoic,olives$Area.Name),xlab="Area",ylab="Eic
41
42  require(s20x)
43
44  boxqq(Oleic~Region.Name,data=olives)
45
46  a = ggplot(olives,aes(Area.Name,Oleic))
47  a + geom_boxplot(fill="lightblue") + xlab("Area Name") +
48    ylab("Oleic Acid") + ggtitle("Oleic Acid vs. Growing Area")
49
50  require(violinmplot)
51
52  violinmplot(Area.Name~Oleic,xlab="Oleic Acid",ylab="Area Name",data=Oli
53
```

48:62 (Top Level) R Script

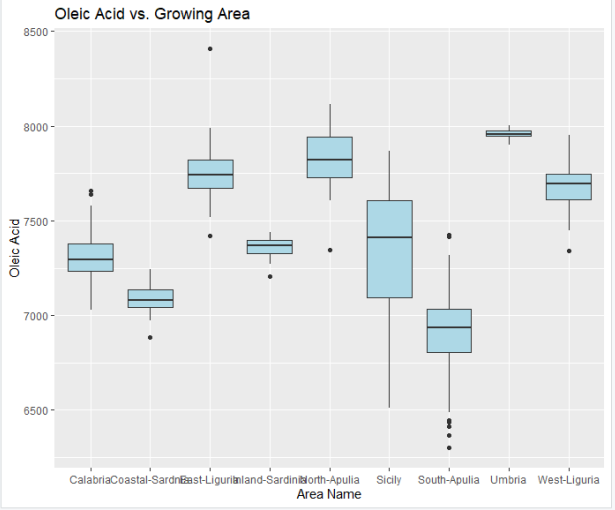
Console Terminal Jobs

```
> a = ggplot(olives,aes(Area.Name,Oleic))
> a + geom_boxplot(fill="lightblue") + xlab("Area Name") +
+   ylab("Oleic Acid") + ggtitle("Oleic Acid vs. Growing Area")
>
```

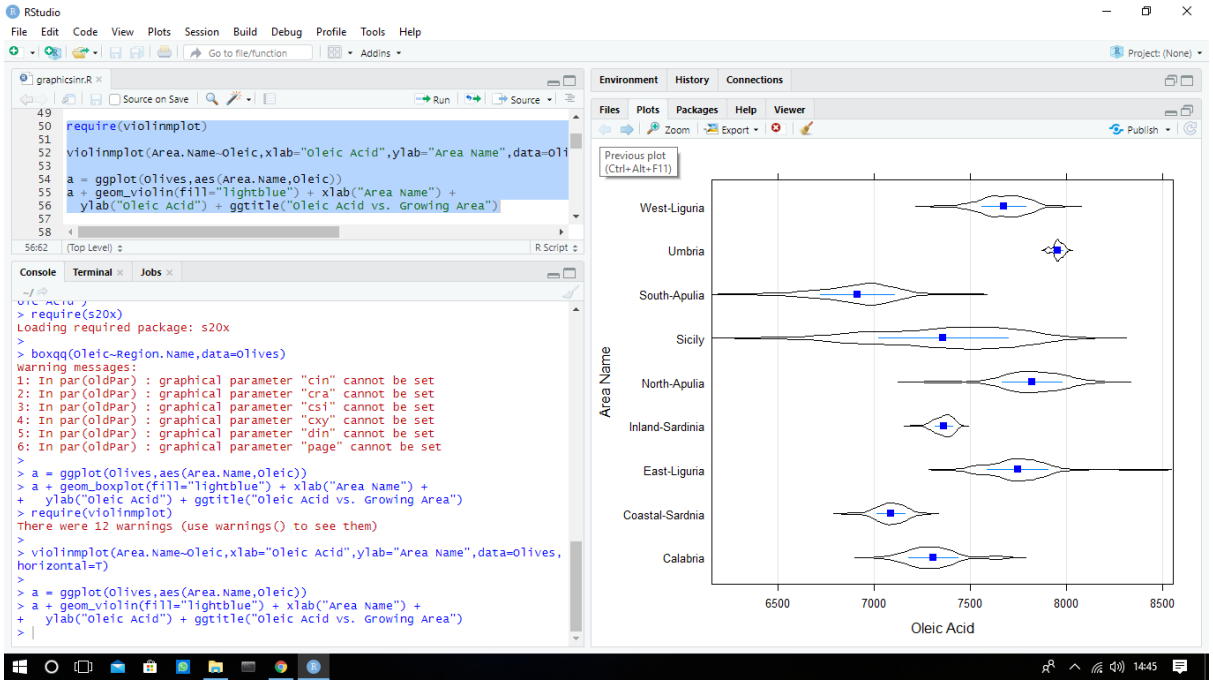
Environment History Connections

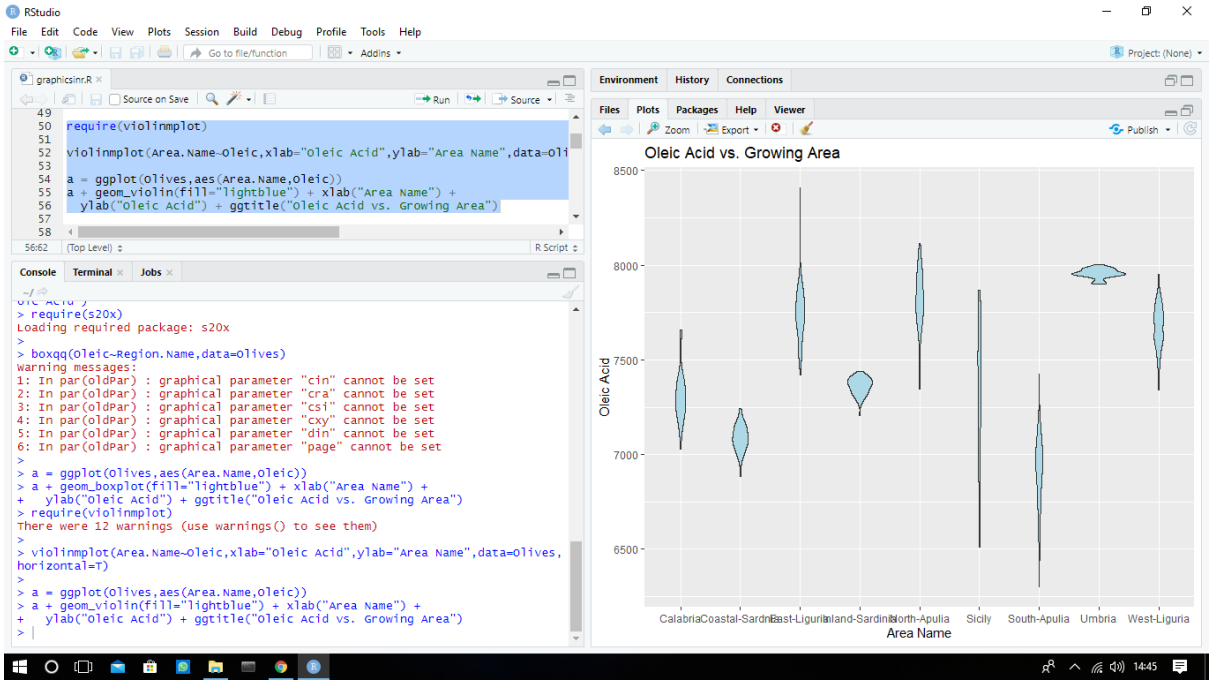
Files Plots Packages Help Viewer

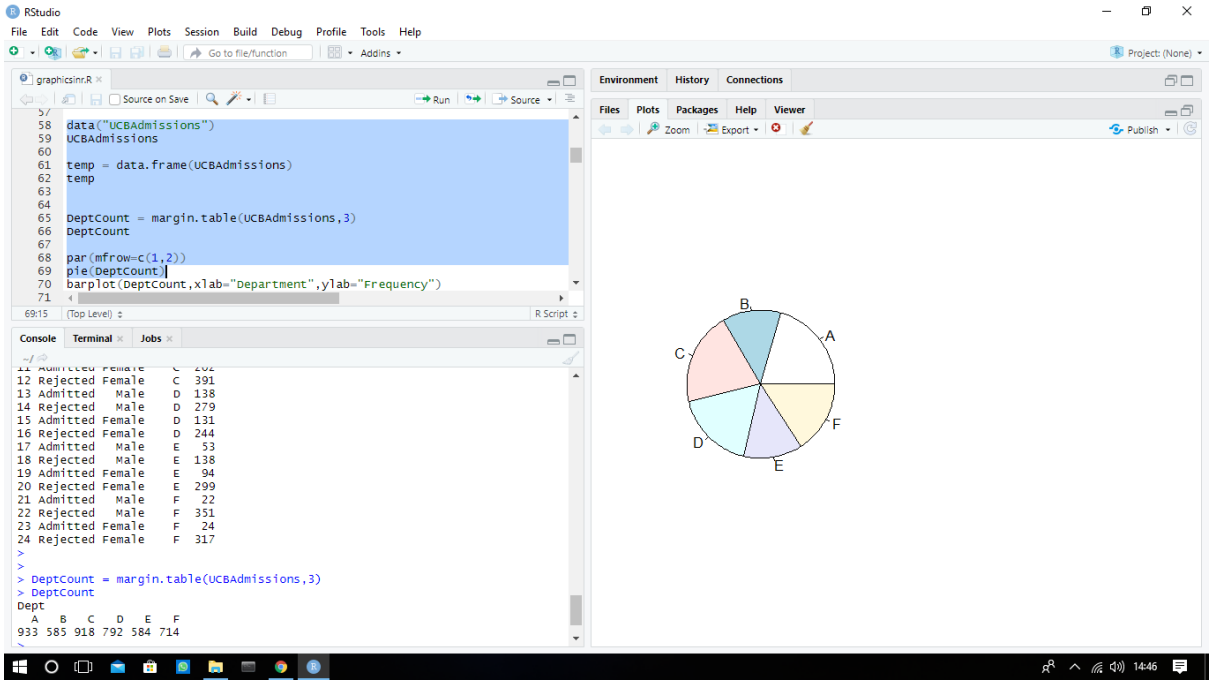
Zoom Export Publish

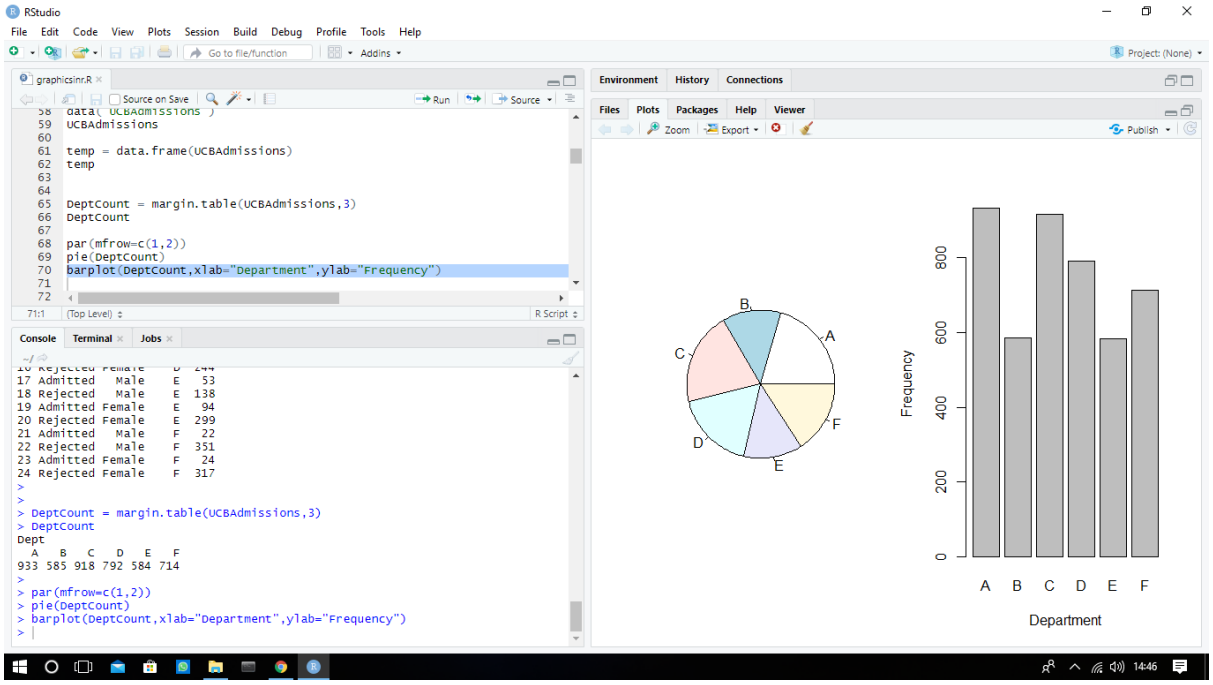


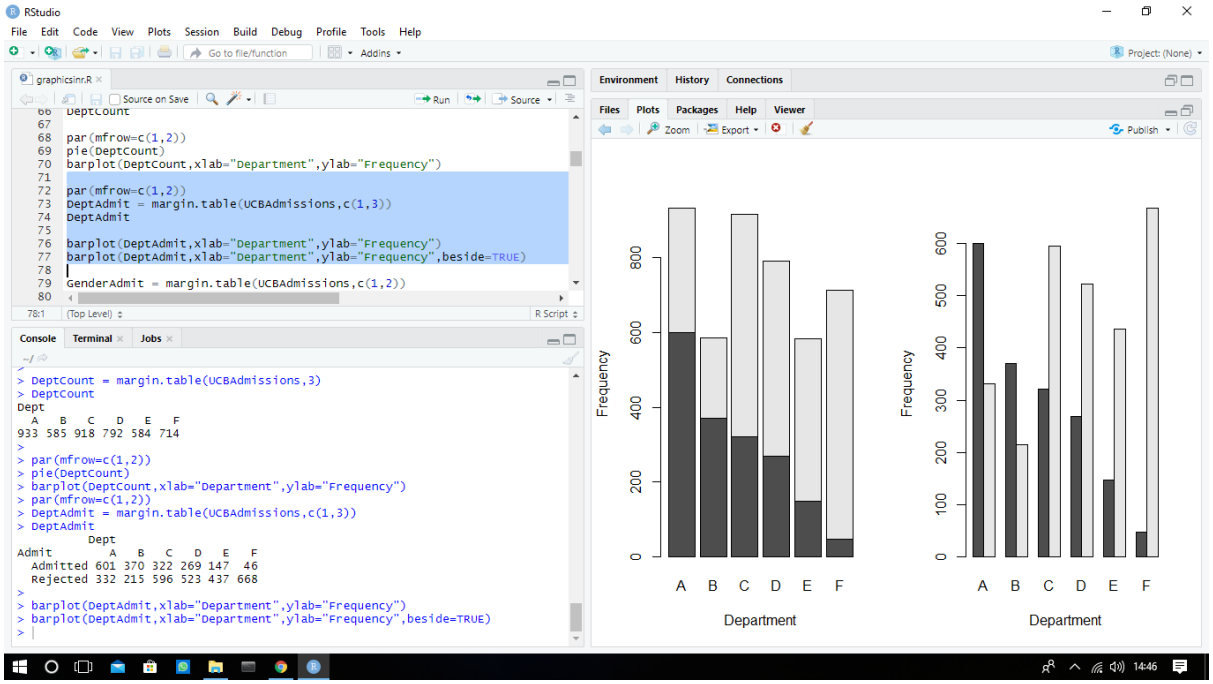
14:43

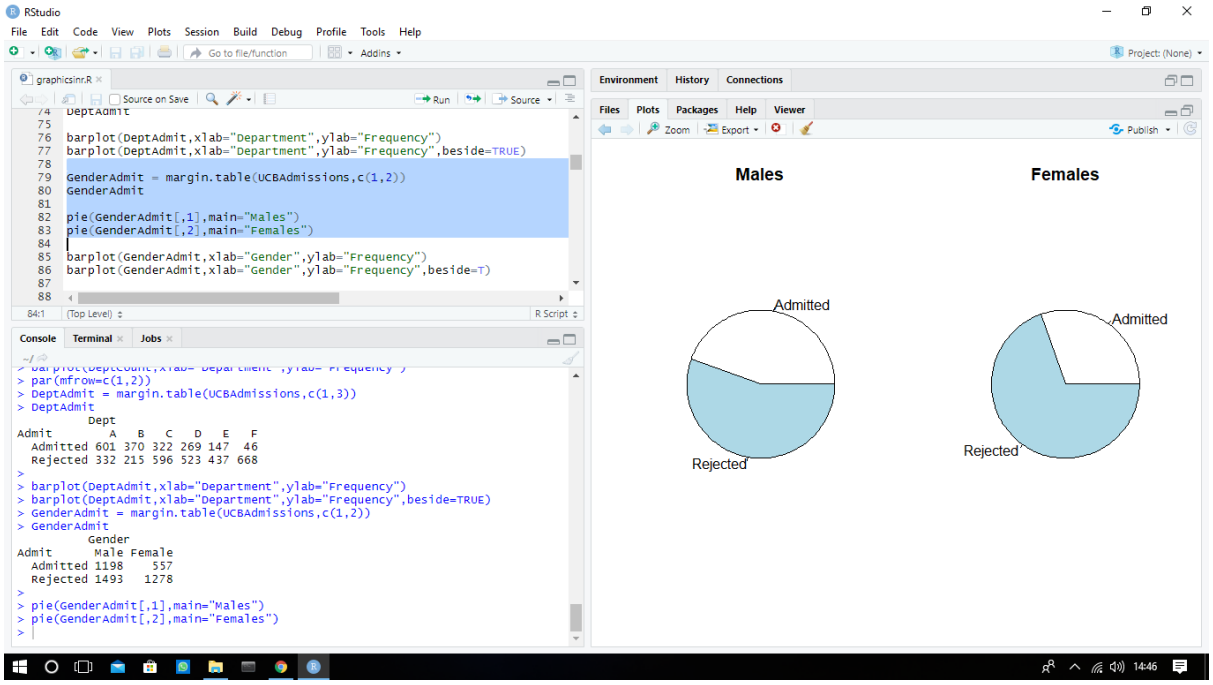

RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
graphicsinr.R
require(violinmplot)
violinmplot(Area.Name~Oleic,xlab="Oleic Acid",ylab="Area Name",data=Oli
a = ggplot(Olives,aes(Area.Name,Oleic))
a + geom_violin(fill="lightblue") + xlab("Area Name") +
ylab("Oleic Acid") + ggtitle("Oleic Acid vs. Growing Area")
Console Terminal Jobs
> require(s20x)
Loading required package: s20x
> boxqq(Oleic~Region.Name,data=Olives)
Warning messages:
1: In par(oldPar) : graphical parameter "cin" cannot be set
2: In par(oldPar) : graphical parameter "cra" cannot be set
3: In par(oldPar) : graphical parameter "csi" cannot be set
4: In par(oldPar) : graphical parameter "cxy" cannot be set
5: In par(oldPar) : graphical parameter "din" cannot be set
6: In par(oldPar) : graphical parameter "page" cannot be set
> a = ggplot(Olives,aes(Area.Name,Oleic))
> a + geom_boxplot(fill="lightblue") + xlab("Area Name") +
+ ylab("Oleic Acid") + ggtitle("Oleic Acid vs. Growing Area")
> require(violinmplot)
There were 12 warnings (use warnings() to see them)
> violinmplot(Area.Name~Oleic,xlab="Oleic Acid",ylab="Area Name",data=Olives,
horizontal=T)
> a = ggplot(Olives,aes(Area.Name,Oleic))
> a + geom_violin(fill="lightblue") + xlab("Area Name") +
+ ylab("Oleic Acid") + ggtitle("Oleic Acid vs. Growing Area")
Environment History Connections
Files Plots Packages Help Viewer
Previous plot (Ctrl+Alt+F11)
West-Liguria
Umbria
South-Apulia
Sicily
North-Apulia
Inland-Sardinia
East-Liguria
Coastal-Sardnia
Calabria
Area Name
6500 7000 7500 8000 8500
Oleic Acid


RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
graphicsinr.R
require(violinmplot)
violinmplot(Area.Name~Oleic,xlab="Oleic Acid",ylab="Area Name",data=Oli
a = ggplot(Olives,aes(Area.Name,Oleic))
a + geom_violin(fill="lightblue") + xlab("Area Name") +
ylab("Oleic Acid") + ggtitle("Oleic Acid vs. Growing Area")
Console Terminal Jobs
> require(s20x)
Loading required package: s20x
> boxqq(Oleic~Region.Name,data=Olives)
Warning messages:
1: In par(oldPar) : graphical parameter "cin" cannot be set
2: In par(oldPar) : graphical parameter "cra" cannot be set
3: In par(oldPar) : graphical parameter "csi" cannot be set
4: In par(oldPar) : graphical parameter "cxy" cannot be set
5: In par(oldPar) : graphical parameter "din" cannot be set
6: In par(oldPar) : graphical parameter "page" cannot be set
> a = ggplot(Olives,aes(Area.Name,Oleic))
> a + geom_boxplot(fill="lightblue") + xlab("Area Name") +
+ ylab("Oleic Acid") + ggtitle("Oleic Acid vs. Growing Area")
> require(violinmplot)
There were 12 warnings (use warnings() to see them)
> violinmplot(Area.Name~Oleic,xlab="Oleic Acid",ylab="Area Name",data=Olives,
horizontal=T)
> a = ggplot(Olives,aes(Area.Name,Oleic))
> a + geom_violin(fill="lightblue") + xlab("Area Name") +
+ ylab("Oleic Acid") + ggtitle("Oleic Acid vs. Growing Area")
Environment History Connections
Files Plots Packages Help Viewer
Oleic Acid vs. Growing Area
8500
8000
7500
7000
6500
Oleic Acid
Calabria Coastal-Sardnia East-Liguria Inland-Sardinia North-Apulia Sicily South-Apulia Umbria West-Liguria
Area Name


RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
graphicsinr.R
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")
Console Terminal Jobs
12 Rejected Female C 391
13 Admitted Male D 138
14 Rejected Male D 279
15 Admitted Female D 131
16 Rejected Female D 244
17 Admitted Male E 53
18 Rejected Male E 138
19 Admitted Female E 94
20 Rejected Female E 299
21 Admitted Male F 22
22 Rejected Male F 351
23 Admitted Female F 24
24 Rejected Female F 317
> DeptCount = margin.table(UCBAdmissions,3)
> DeptCount
Dept
A B C D E F
933 585 918 792 584 714
Environment History Connections
Files Plots Packages Help Viewer
A
B
C
D
E
F
14:46


RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
graphicsinr.R
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")
Console Terminal Jobs
17 Admitted Male E 53
18 Rejected Male E 138
19 Admitted Female E 94
20 Rejected Female E 299
21 Admitted Male F 22
22 Rejected Male F 351
23 Admitted Female F 24
24 Rejected Female F 317
> DeptCount = margin.table(UCBAdmissions,3)
> DeptCount
Dept
A B C D E F
933 585 918 792 584 714
> par(mfrow=c(1,2))
> pie(DeptCount)
> barplot(DeptCount,xlab="Department",ylab="Frequency")
Environment History Connections
Files Plots Packages Help Viewer
A
B
C
D
E
F
Frequency
0
200
400
600
800
A B C D E F
Department
14:46


RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
graphicsinr.R
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))
Console Terminal Jobs
> DeptCount = margin.table(UCBAdmissions,3)
> DeptCount
Dept
A B C D E F
933 585 918 792 584 714
> 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
Dept
Admit A B C D E F
Admitted 601 370 322 269 147 46
Rejected 332 215 596 523 437 668
> barplot(DeptAdmit,xlab="Department",ylab="Frequency")
> barplot(DeptAdmit,xlab="Department",ylab="Frequency",beside=TRUE)
Environment History Connections
Files Plots Packages Help Viewer
Frequency
Department
A B C D E F


RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
graphicsinr.R
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)
Console Terminal Jobs
> par(mfrow=c(1,2))
> DeptAdmit = margin.table(UCBAdmissions,c(1,3))
> DeptAdmit
Dept
Admit A B C D E F
Admitted 601 370 322 269 147 46
Rejected 332 215 596 523 437 668
> barplot(DeptAdmit,xlab="Department",ylab="Frequency")
> barplot(DeptAdmit,xlab="Department",ylab="Frequency",beside=TRUE)
> GenderAdmit = margin.table(UCBAdmissions,c(1,2))
> GenderAdmit
Gender
Admit Male Female
Admitted 1198 557
Rejected 1493 1278
> pie(GenderAdmit[,1],main="Males")
> pie(GenderAdmit[,2],main="Females")
Environment History Connections
Files Plots Packages Help Viewer
Males
Admitted
Rejected
Females
Admitted
Rejected

RStudio

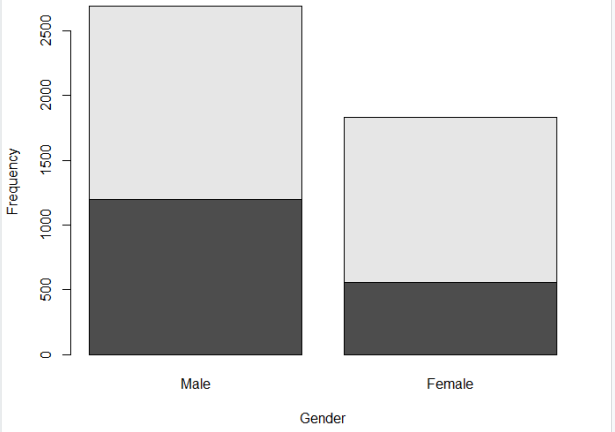
File Edit Code View Plots Session Build Debug Profile Tools Help

graphicsnr.R

```
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)
```

Console Terminal Jobs

```
> DeptAdmit
          Dept
Admit        A   B   C   D   E   F
  Admitted 601 370 322 269 147  46
  Rejected 332 215 596 523 437 668
>
> barplot(DeptAdmit,xlab="Department",ylab="Frequency")
> barplot(DeptAdmit,xlab="Department",ylab="Frequency",beside=TRUE)
> GenderAdmit = margin.table(UCBAdmissions,c(1,2))
> GenderAdmit
          Gender
Admit      Male Female
  Admitted 1198    557
  Rejected 1493   1278
>
> pie(GenderAdmit[,1],main="Males")
> pie(GenderAdmit[,2],main="Females")
> barplot(GenderAdmit,xlab="Gender",ylab="Frequency")
> barplot(GenderAdmit,xlab="Gender",ylab="Frequency",beside=T)
>
```



RStudio

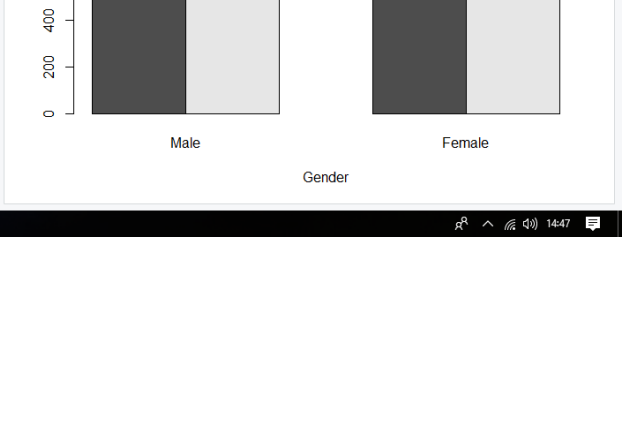
File Edit Code View Plots Session Build Debug Profile Tools Help

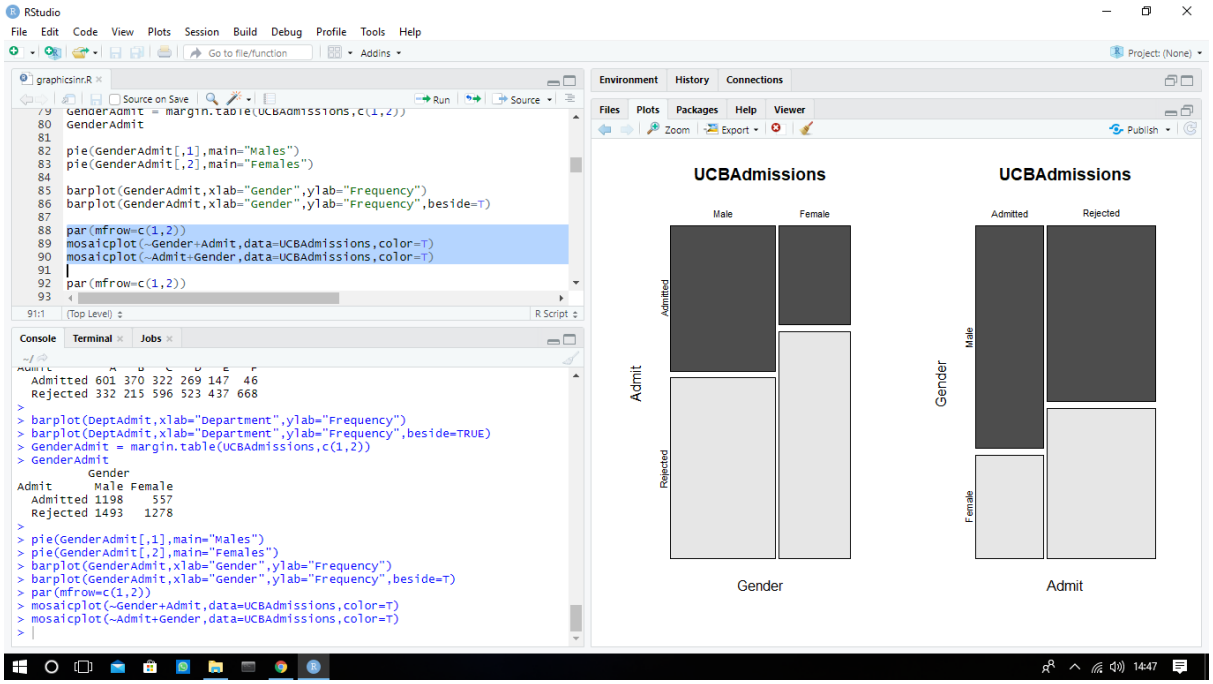
graphicsnr.R

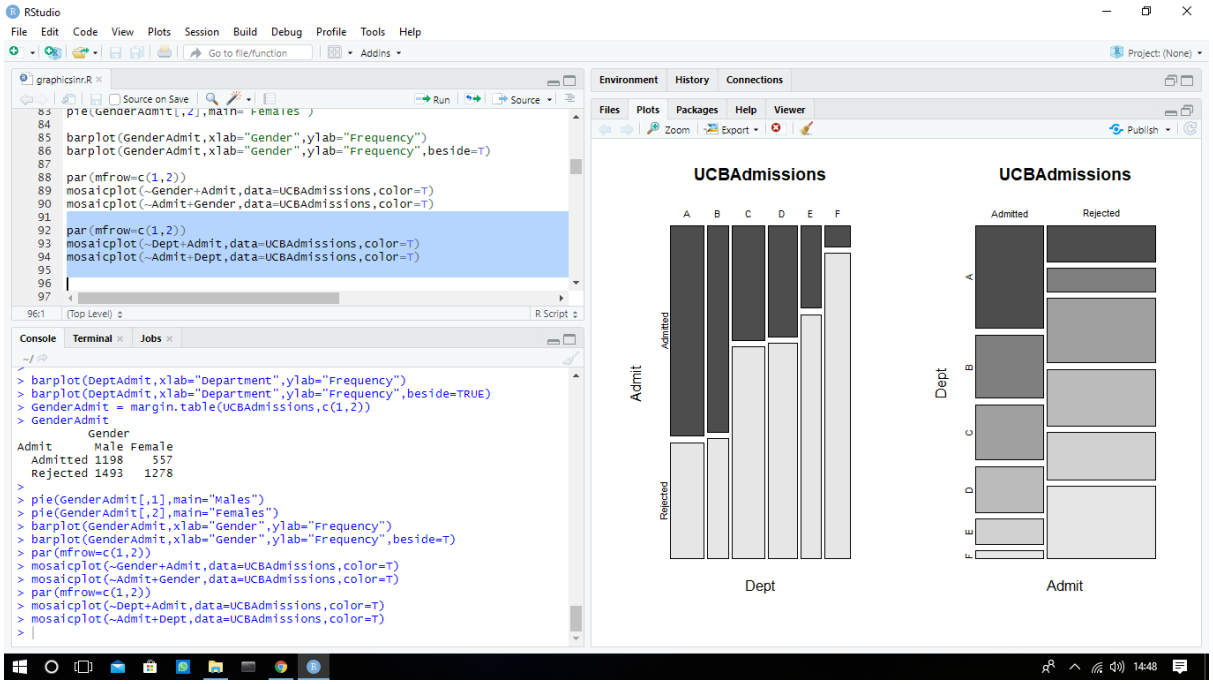
```
barplot(DeptAdmit,xlab="Department",ylab="Frequency",beside=TRUE)

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)
```

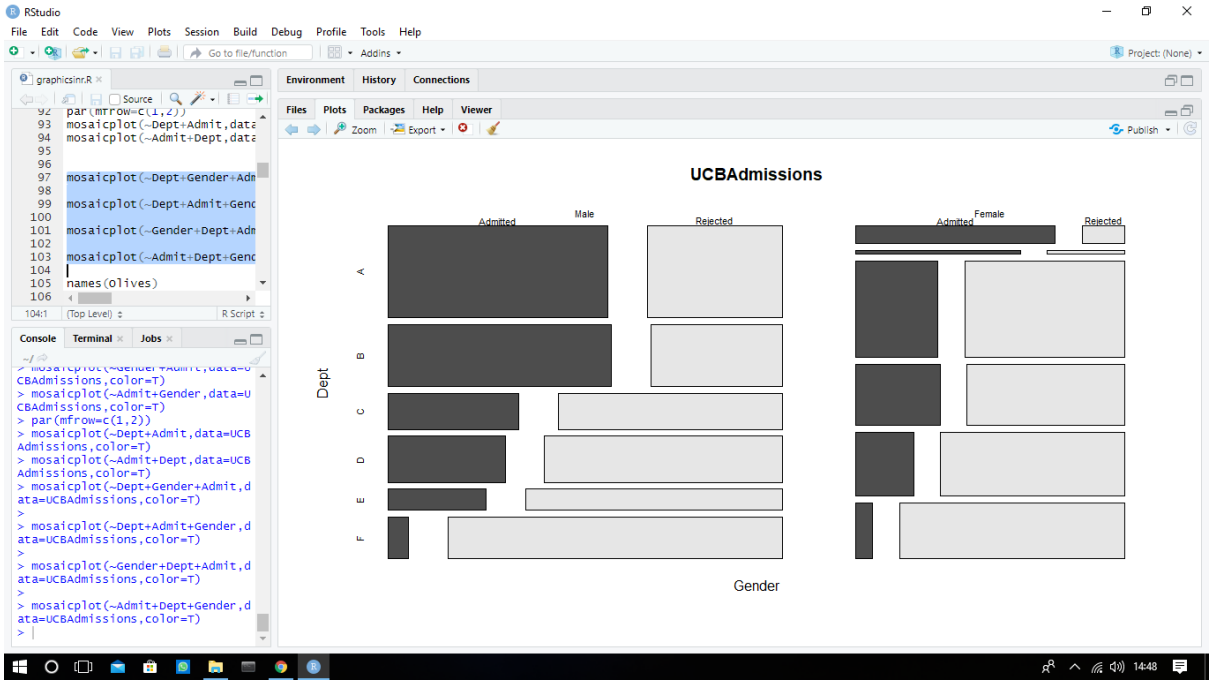
Console Terminal Jobs

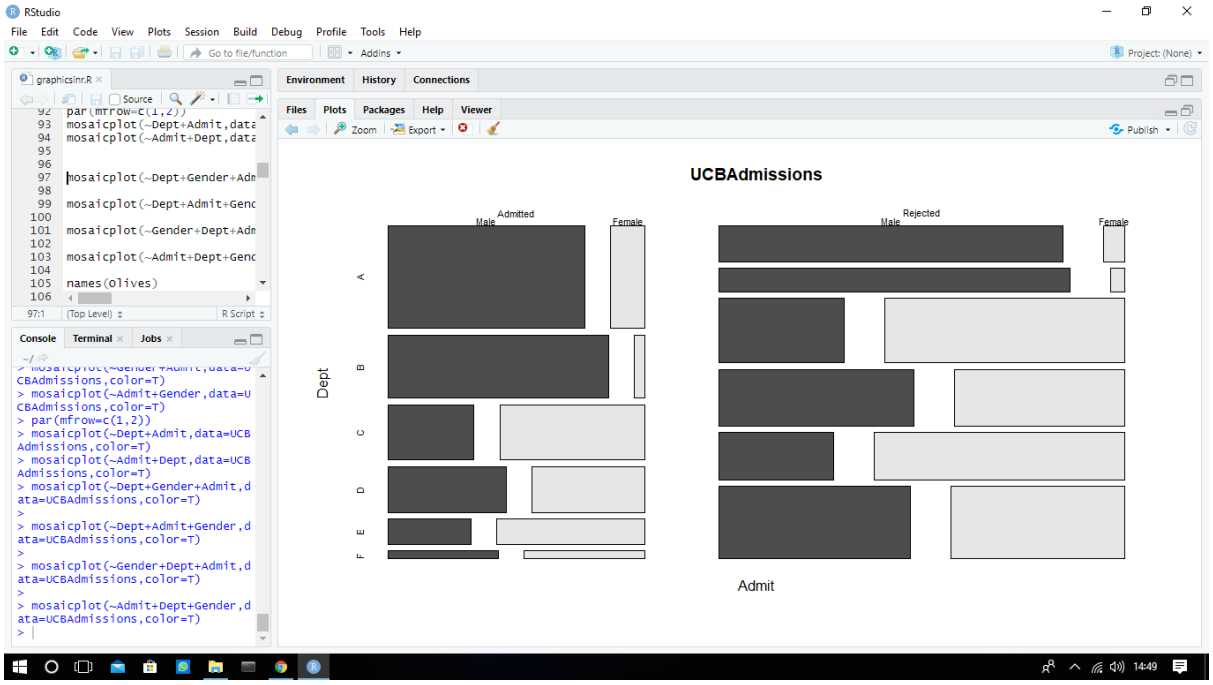
```
> DeptAdmit
          Dept
Admit        A   B   C   D   E   F
  Admitted 601 370 322 269 147  46
  Rejected 332 215 596 523 437 668
>
> barplot(DeptAdmit,xlab="Department",ylab="Frequency")
> barplot(DeptAdmit,xlab="Department",ylab="Frequency",beside=TRUE)
> GenderAdmit = margin.table(UCBAdmissions,c(1,2))
> GenderAdmit
          Gender
Admit      Male Female
  Admitted 1198    557
  Rejected 1493   1278
>
> pie(GenderAdmit[,1],main="Males")
> pie(GenderAdmit[,2],main="Females")
> barplot(GenderAdmit,xlab="Gender",ylab="Frequency")
> barplot(GenderAdmit,xlab="Gender",ylab="Frequency",beside=T)
>
```




RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
graphicsnr.R
Source on Save
Run
Source
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))
91:1 (Top Level) R Script
Console Terminal Jobs
Admitted 601 370 322 269 147 46
Rejected 332 215 596 523 437 668
> barplot(DeptAdmit,xlab="Department",ylab="Frequency")
> barplot(DeptAdmit,xlab="Department",ylab="Frequency",beside=TRUE)
> GenderAdmit = margin.table(UCBAdmissions,c(1,2))
> GenderAdmit
Gender
Admit Male Female
Admitted 1198 557
Rejected 1493 1278
> 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)
Environment History Connections
Files Plots Packages Help Viewer
Zoom Export Publish
UCBAdmissions
Male Female
Admitted Rejected
Admit
Gender
UCBAdmissions
Admitted Rejected
Male Female
Gender
Admit
14:47


RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
graphicsnr.R
Source on Save
Run
Source
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)
96:1 (Top Level) R Script
Console Terminal Jobs
> barplot(DeptAdmit,xlab="Department",ylab="Frequency")
> barplot(DeptAdmit,xlab="Department",ylab="Frequency",beside=TRUE)
> GenderAdmit = margin.table(UCBAdmissions,c(1,2))
> GenderAdmit
Gender
Admit Male Female
Admitted 1198 557
Rejected 1493 1278
> 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)
Environment History Connections
Files Plots Packages Help Viewer
Zoom Export Publish
UCBAdmissions
A B C D E F
Admitted Rejected
Admit
Dept
UCBAdmissions
Admitted Rejected
A B C D E F
Dept
Admit
14:48


RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
graphicsinr.R
Environment History Connections
Files Plots Packages Help Viewer
Console Terminal Jobs
UCBAdmissions
Male
Female
Admitted
Rejected
Dept
A
B
C
D
E
F
Gender

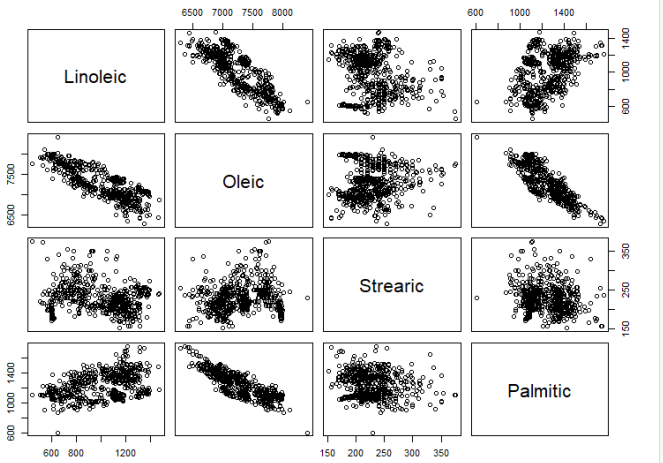

RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
graphicsinr.R
Environment History Connections
Files Plots Packages Help Viewer
Console Terminal Jobs
UCBAdmissions
Admitted
Rejected
Male
Female
Dept
A
B
C
D
E
F
Admit

RStudio

File Edit Code View Plots Session Build Debug Profile Tools Help

```
require(corrplot)

Boston.corr = cor(Boston[,-c(1:4)])   # find pairwise correla
options(digits=2)  # reduce the number of significant digits
Boston.corr

corrplot(Boston.corr)

corrplot(Boston.corr,method="ellipse")

corrplot(Boston.corr,order="FPC")

corrplot(Boston.corr,order="hclust")
```

```
          TAX PTRATIO      B  LSTAT
CMEDV  -0.472   -0.51  0.335 -0.741
CRIM    0.583    0.29 -0.385  0.456
ZN     -0.315   -0.39  0.176 -0.413
INDUS   0.721    0.38 -0.357  0.604
CHAS   -0.036   -0.12  0.049 -0.054
NOX     0.668    0.19 -0.380  0.591
RM     -0.292   -0.36  0.128 -0.614
AGE     0.506    0.26 -0.274  0.602
DIS    -0.534   -0.23  0.292 -0.497
RAD     0.910    0.46 -0.444  0.489
TAX     1.000    0.46 -0.442  0.544
PTRATIO 0.461    1.00 -0.177  0.374
B      -0.442   -0.18  1.000 -0.366
LSTAT   0.544    0.37 -0.366  1.000
> corrplot(Boston.corr)
> corrplot(Boston.corr,method="ellipse")
```



RStudio

File Edit Code View Plots Session Build Debug Profile Tools Help

```
require(corrplot)

Boston.corr = cor(Boston[,-c(1:4)])   # find pairwise correla
options(digits=2)  # reduce the number of significant digits
Boston.corr

corrplot(Boston.corr)

corrplot(Boston.corr,method="ellipse")

corrplot(Boston.corr,order="FPC")

corrplot(Boston.corr,order="hclust")
```

```
          TAX PTRATIO      B  LSTAT
CMEDV  -0.472   -0.51  0.335 -0.741
CRIM    0.583    0.29 -0.385  0.456
ZN     -0.315   -0.39  0.176 -0.413
INDUS   0.721    0.38 -0.357  0.604
CHAS   -0.036   -0.12  0.049 -0.054
NOX     0.668    0.19 -0.380  0.591
RM     -0.292   -0.36  0.128 -0.614
AGE     0.506    0.26 -0.274  0.602
DIS    -0.534   -0.23  0.292 -0.497
RAD     0.910    0.46 -0.444  0.489
TAX     1.000    0.46 -0.442  0.544
PTRATIO 0.461    1.00 -0.177  0.374
B      -0.442   -0.18  1.000 -0.366
LSTAT   0.544    0.37 -0.366  1.000
> corrplot(Boston.corr)
> corrplot(Boston.corr,method="ellipse")
```

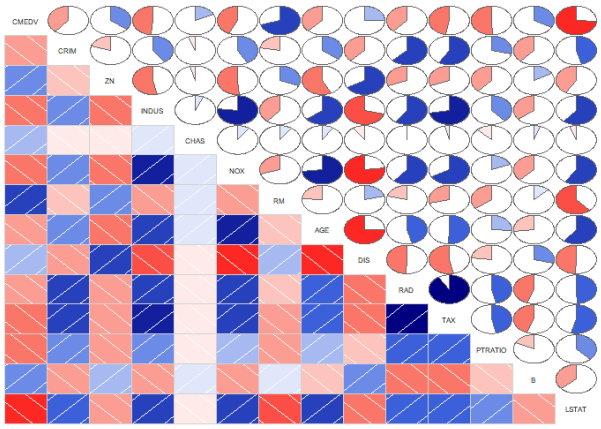
RStudio

File Edit Code View Plots Session Build Debug Profile Tools Help

```
172 require(corrplot)
173
174 Boston.corr = cor(Boston[,-c(1:4)])   # find pairwise correla
175 options(digits=2)  # reduce the number of significant digits
176 Boston.corr
177
178 corrplot(Boston.corr)
179
180 corrplot(Boston.corr,method="ellipse")
181
182 corrplot(Boston.corr,order="FPC")
183
184 corrplot(Boston.corr,order="hclust")
185
186
```

181:1 (Top Level) R Script

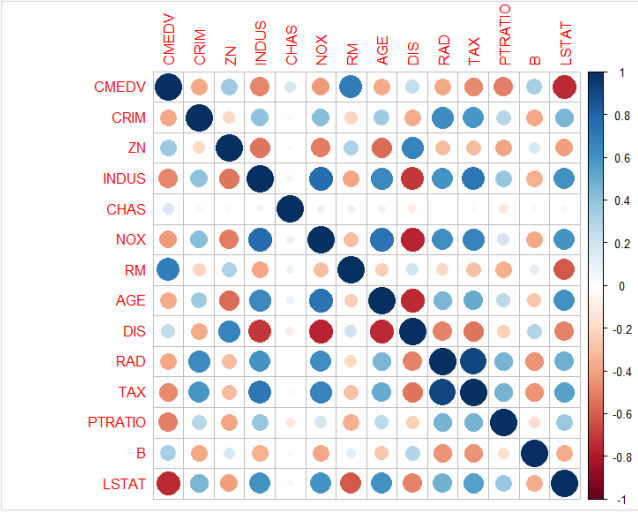
```
          TAX PTRATIO      B  LSTAT
CMEDV  -0.472   -0.51  0.335 -0.741
CRIM    0.583    0.29 -0.385  0.456
ZN     -0.315   -0.39  0.176 -0.413
INDUS   0.721    0.38 -0.357  0.604
CHAS   -0.036   -0.12  0.049 -0.054
NOX     0.668    0.19 -0.380  0.591
RM     -0.292   -0.36  0.128 -0.614
AGE     0.506    0.26 -0.274  0.602
DIS    -0.534   -0.23  0.292 -0.497
RAD     0.910    0.46 -0.444  0.489
TAX     1.000    0.46 -0.442  0.544
PTRATIO 0.461    1.00 -0.177  0.374
B      -0.442   -0.18  1.000 -0.366
LSTAT   0.544    0.37 -0.366  1.000
>
> corrplot(Boston.corr)
>
> corrplot(Boston.corr,method="ellipse")
>
```



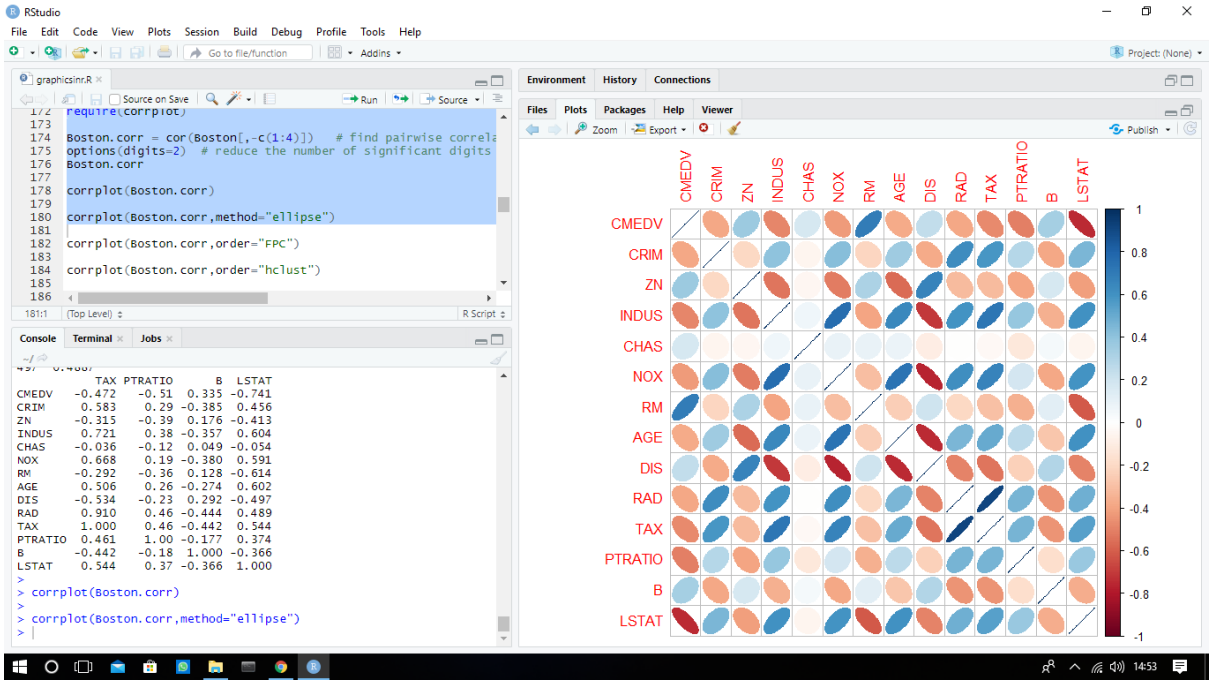
14:52

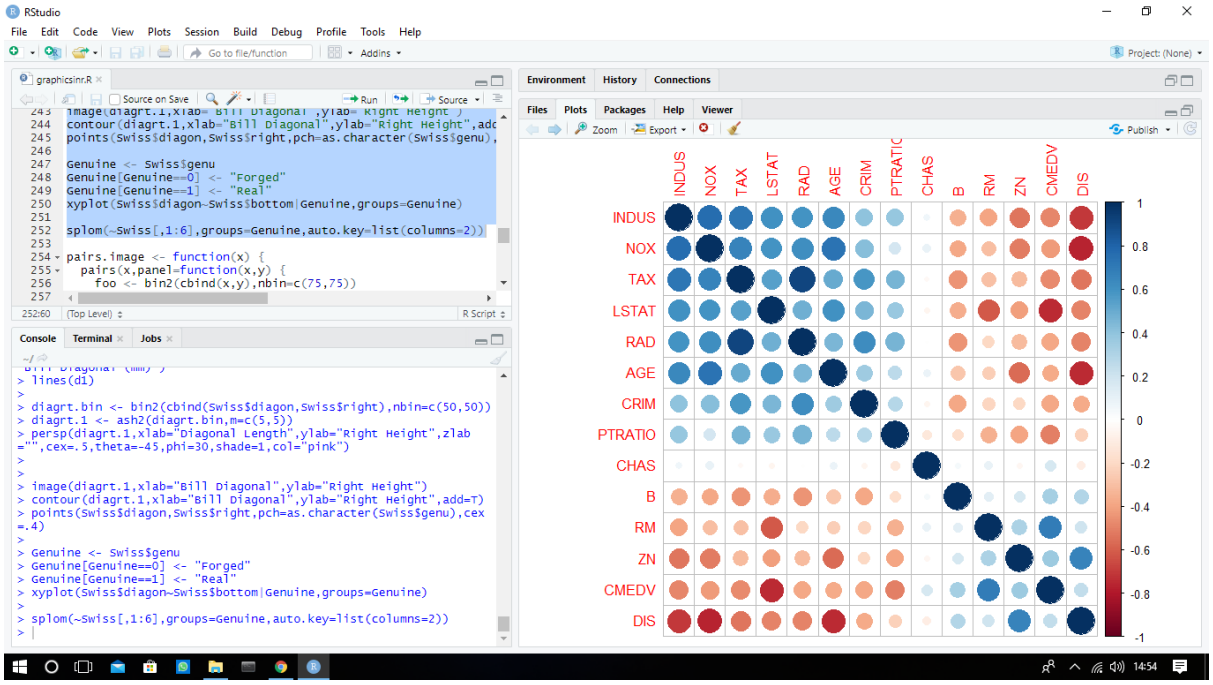
RStudio

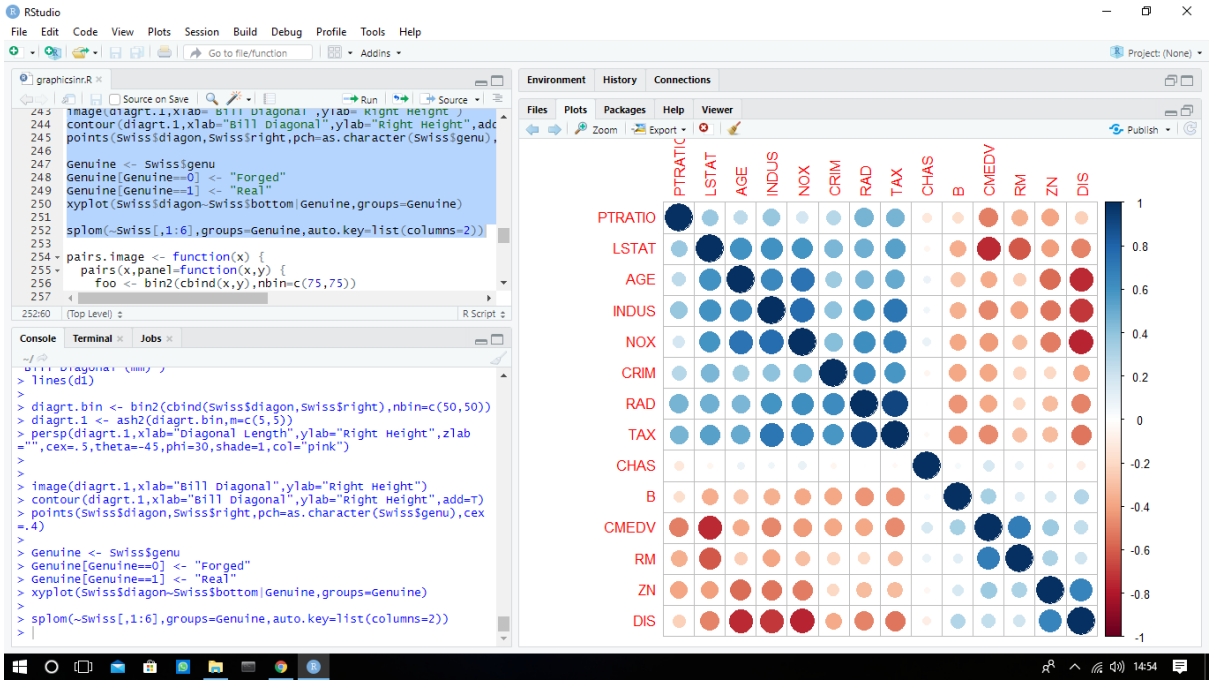
File Edit Code View Plots Session Build Debug Profile Tools Help

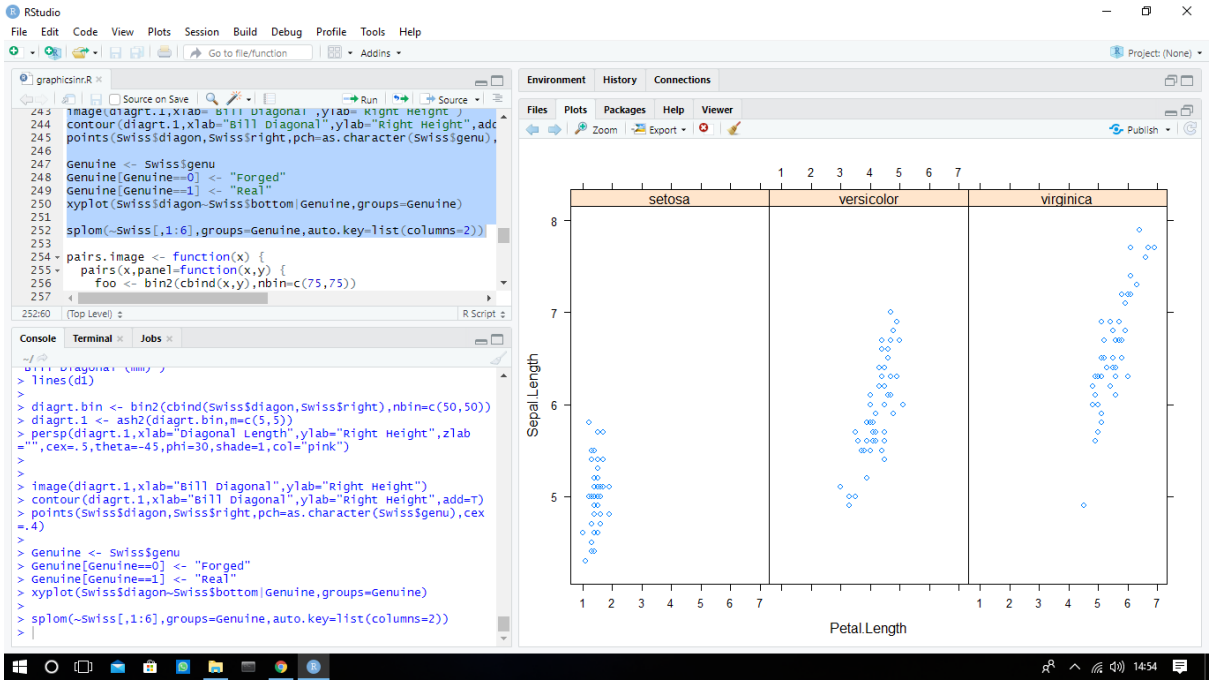


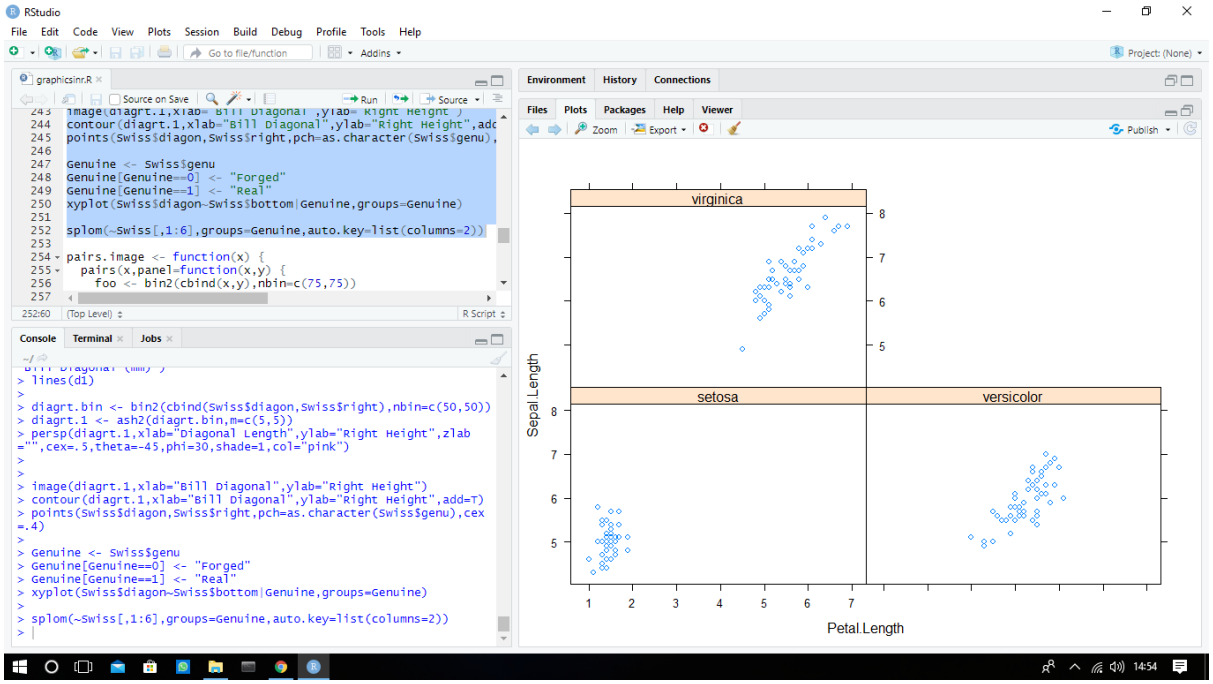
14:53


RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
graphicsinr.R
require(corrplot)
Boston.corr = cor(Boston[,-c(1:4)]) # find pairwise correla
options(digits=2) # reduce the number of significant digits
Boston.corr
corrplot(Boston.corr)
corrplot(Boston.corr,method="ellipse")
corrplot(Boston.corr,order="FPC")
corrplot(Boston.corr,order="hclust")
Console Terminal Jobs
TAX PTRATIO B LSTAT
CMEDV -0.472 -0.51 0.335 -0.741
CRIM 0.583 0.29 -0.385 0.456
ZN -0.315 -0.39 0.176 -0.413
INDUS 0.721 0.38 -0.357 0.604
CHAS -0.036 -0.12 0.049 -0.054
NOX 0.668 0.19 -0.380 0.591
RM -0.292 -0.36 0.128 -0.614
AGE 0.506 0.26 -0.274 0.602
DIS -0.534 -0.23 0.292 -0.497
RAD 0.910 0.46 -0.444 0.489
TAX 1.000 0.46 -0.442 0.544
PTRATIO 0.461 1.00 -0.177 0.374
B -0.442 -0.18 1.000 -0.366
LSTAT 0.544 0.37 -0.366 1.000
> corrplot(Boston.corr)
> corrplot(Boston.corr,method="ellipse")
Environment History Connections
Files Plots Packages Help Viewer
CMEDV CRIM ZN INDUS CHAS NOX RM AGE DIS RAD TAX PTRATIO B LSTAT
1 0.8 0.6 0.4 0.2 0 -0.2 -0.4 -0.6 -0.8 -1
14:53


RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
graphicsinr.R
contour(diagrt.1,xlab="Bill Diagonal",ylab="Right Height",add
points(Swiss$diagon,Swiss$right,pch=as.character(Swiss$genu),
Genuine <- Swiss$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))
Console Terminal Jobs
> lines(d1)
> 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="pink")
> 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))
Environment History Connections
Files Plots Packages Help Viewer
INDUS NOX TAX LSTAT RAD AGE CRIM PTRATIO CHAS B RM ZN CMEDV DIS
1 0.8 0.6 0.4 0.2 0 -0.2 -0.4 -0.6 -0.8 -1
14:54


RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
graphicsinr.R
Source on Save
Run
Source
243 image(diagrt.1,xlab="Bill Diagonal",ylab="Right Height")
244 contour(diagrt.1,xlab="Bill Diagonal",ylab="Right Height",add
245 points(Swiss$diagon,Swiss$right,pch=as.character(Swiss$genu),
246
247 Genuine <- Swiss$genu
248 Genuine[Genuine==0] <- "Forged"
249 Genuine[Genuine==1] <- "Real"
250 xyplot(Swiss$diagon~Swiss$bottom|Genuine,groups=Genuine)
251
252 splom(~Swiss[,1:6],groups=Genuine,auto.key=list(columns=2))
253
254 pairs.image <- function(x) {
255 pairs(x,panel=function(x,y) {
256 foo <- bin2(cbind(x,y),nbin=c(75,75))
257
252:60 (Top Level) R Script
Console Terminal Jobs
> lines(d1)
>
> 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="pink")
>
>
> 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))
>
Environment History Connections
Files Plots Packages Help Viewer
Zoom Export Publish
PTRATIO LSTAT AGE INDUS NOX CRIM RAD TAX CHAS B CMEDV RM ZN DIS
1 0.8 0.6 0.4 0.2 0 -0.2 -0.4 -0.6 -0.8 -1
14:54


RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
graphicsinr.R
Source on Save
Run
Source
243 image(diagrt.1,xlab="Bill Diagonal",ylab="Right Height")
244 contour(diagrt.1,xlab="Bill Diagonal",ylab="Right Height",add
245 points(Swiss$diagon,Swiss$right,pch=as.character(Swiss$genu),
246
247 Genuine <- Swiss$genu
248 Genuine[Genuine==0] <- "Forged"
249 Genuine[Genuine==1] <- "Real"
250 xyplot(Swiss$diagon~Swiss$bottom|Genuine,groups=Genuine)
251
252 splom(~Swiss[,1:6],groups=Genuine,auto.key=list(columns=2))
253
254 pairs.image <- function(x) {
255 pairs(x,panel=function(x,y) {
256 foo <- bin2(cbind(x,y),nbin=c(75,75))
257
252:60 (Top Level) R Script
Console Terminal Jobs
> lines(d1)
>
> 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="pink")
>
>
> 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))
>
Environment History Connections
Files Plots Packages Help Viewer
Zoom Export Publish
setosa versicolor virginica
Sepal.Length
Petal.Length
14:54

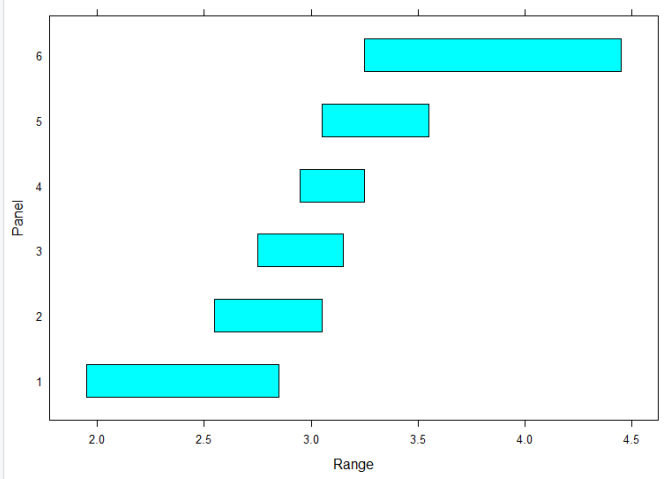

RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
graphicsnr.R
Source on Save
Run
Source
image(diagrt.1,xlab= Bill Diagonal ",ylab= Right Height ")
contour(diagrt.1,xlab="Bill Diagonal",ylab="Right Height",add
points(Swiss$diagon,Swiss$right,pch=as.character(Swiss$genu),
Genuine <- Swiss$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))
252:60
(Top Level)
R Script
Console
Terminal
Jobs
> lines(d1)
> 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="pink")
> 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))
Environment
History
Connections
Files
Plots
Packages
Help
Viewer
Zoom
Export
Publish
virginica
setosa
versicolor
Sepal.Length
Petal.Length
14:54

RStudio

File Edit Code View Plots Session Build Debug Profile Tools Help

```
244 contour(diagrt.1,xlab="Bill Diagonal",ylab="Right Height",add
245 points(Swiss$diagon,Swiss$right,pch=as.character(Swiss$genu),
246
247 Genuine <- Swiss$genu
248 Genuine[Genuine==0] <- "Forged"
249 Genuine[Genuine==1] <- "Real"
250 xyplot(Swiss$diagon~Swiss$bottom|Genuine,groups=Genuine)
251
252 splom(~Swiss[,1:6],groups=Genuine,auto.key=list(columns=2))
253
254 pairs.image <- function(x) {
255   pairs(x,panel=function(x,y) {
256     foo <- bin2(cbind(x,y),nbin=c(75,75))
257
```

```
> lines(d1)
>
> 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="pink")
>
>
> 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))
>
```

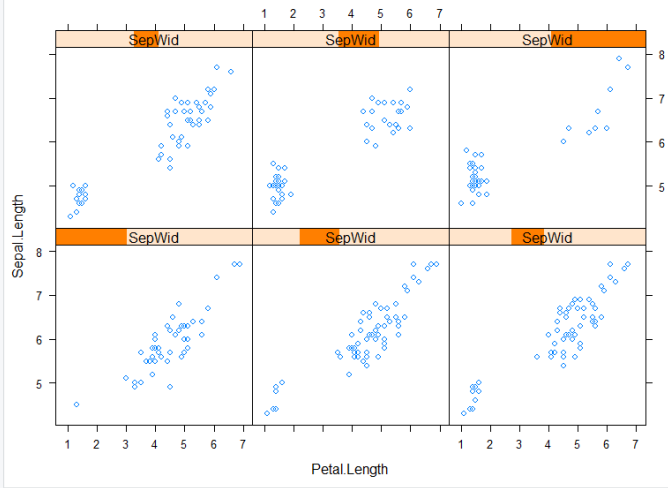


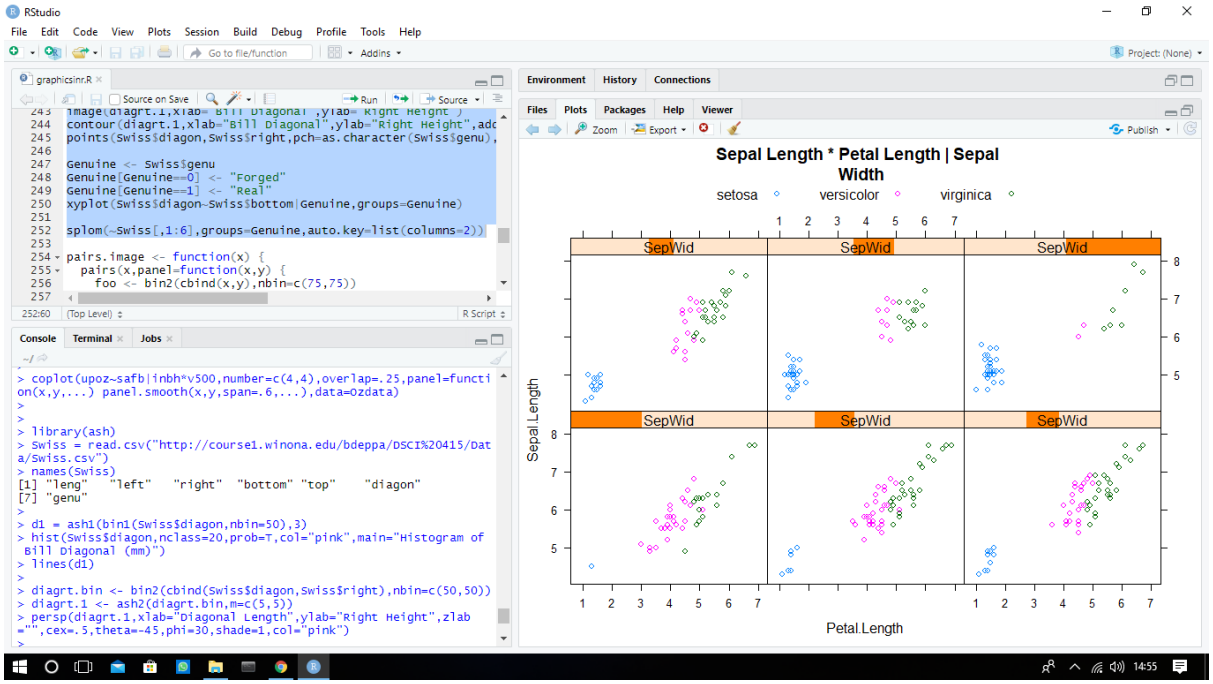
RStudio

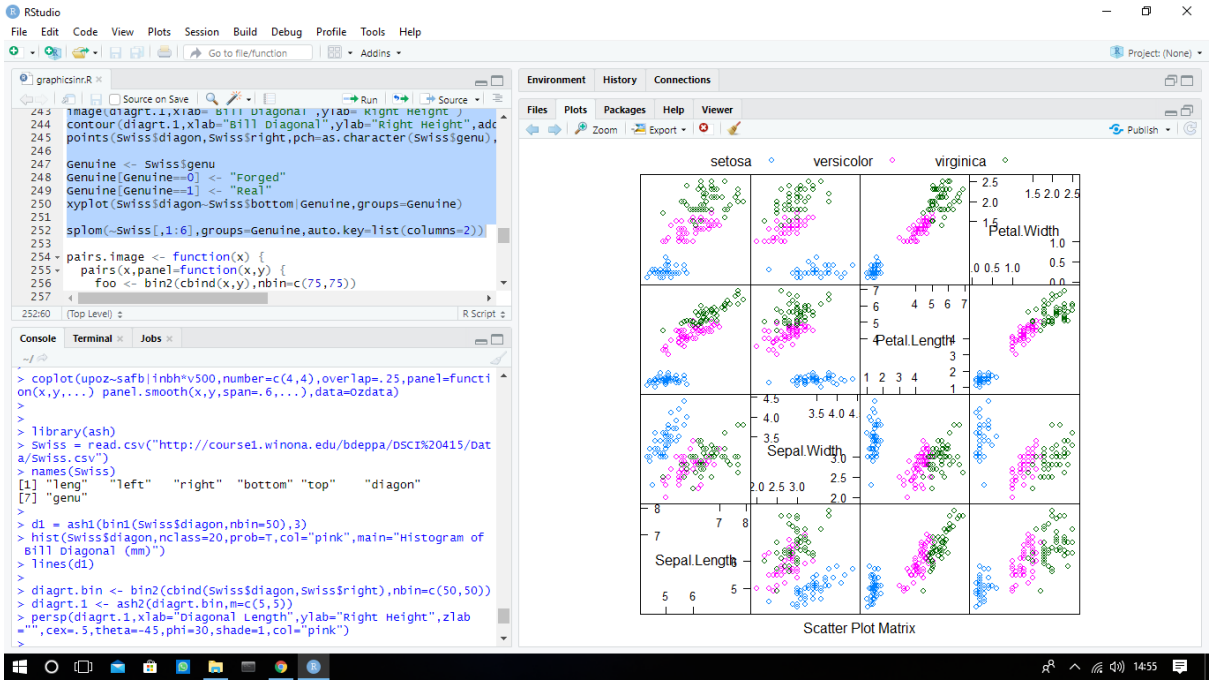
File Edit Code View Plots Session Build Debug Profile Tools Help

```
244 contour(diagrt.1,xlab="Bill Diagonal",ylab="Right Height",add
245 points(Swiss$diagon,Swiss$right,pch=as.character(Swiss$genu),
246
247 Genuine <- Swiss$genu
248 Genuine[Genuine==0] <- "Forged"
249 Genuine[Genuine==1] <- "Real"
250 xyplot(Swiss$diagon~Swiss$bottom|Genuine,groups=Genuine)
251
252 splom(~Swiss[,1:6],groups=Genuine,auto.key=list(columns=2))
253
254 pairs.image <- function(x) {
255   pairs(x,panel=function(x,y) {
256     foo <- bin2(cbind(x,y),nbin=c(75,75))
257
```

```
> coplot(upoz~safb|inbh*v500,number=c(4,4),overlap=.25,panel=functi
on(x,y,...) panel.smooth(x,y,span=.6,...),data=Ozdata)
>
>
> library(ash)
> Swiss = read.csv("http://course1.winona.edu/bdeppa/DSCI%20415/Dat
a/Swiss.csv")
> names(Swiss)
[1] "leng"   "left"   "right"  "bottom" "top"    "diagon"
[7] "genu"
>
> d1 = ash1(bin1(Swiss$diagon,nbin=50),3)
> hist(Swiss$diagon,nclass=20,prob=T,col="pink",main="Histogram of
 Bill Diagonal (mm)")
> lines(d1)
>
> 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="pink")
>
```




RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
graphicsnr.R
Source on Save
Run
Source
243 image(diagrt.1,xlab="Bill Diagonal",ylab="Right Height")
244 contour(diagrt.1,xlab="Bill Diagonal",ylab="Right Height",add
245 points(Swiss$diagon,Swiss$right,pch=as.character(Swiss$genu),
246
247 Genuine <- Swiss$genu
248 Genuine[Genuine==0] <- "Forged"
249 Genuine[Genuine==1] <- "Real"
250 xyplot(Swiss$diagon~Swiss$bottom|Genuine,groups=Genuine)
251
252 splom(~Swiss[,1:6],groups=Genuine,auto.key=list(columns=2))
253
254 pairs.image <- function(x) {
255 pairs(x,panel=function(x,y) {
256 foo <- bin2(cbind(x,y),nbin=c(75,75))
257
252:60 (Top Level) R Script
Console Terminal Jobs
> coplot(upoz~safb|inbh*v500,number=c(4,4),overlap=.25,panel=functi
on(x,y,...) panel.smooth(x,y,span=.6,...),data=Ozdata)
>
>
> library(ash)
> Swiss = read.csv("http://course1.winona.edu/bdeppa/DSCI%20415/Dat
a/Swiss.csv")
> names(Swiss)
[1] "leng" "left" "right" "bottom" "top" "diagon"
[7] "genu"
>
> d1 = ash1(bin1(Swiss$diagon,nbin=50),3)
> hist(Swiss$diagon,nclass=20,prob=T,col="pink",main="Histogram of
Bill Diagonal (mm)")
> lines(d1)
>
> 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="pink")
>
Environment History Connections
Files Plots Packages Help Viewer
Zoom Export Publish
Sepal Length * Petal Length | Sepal Width
setosa versicolor virginica
SepWid
Sepal.Length
Petal.Length
14:55

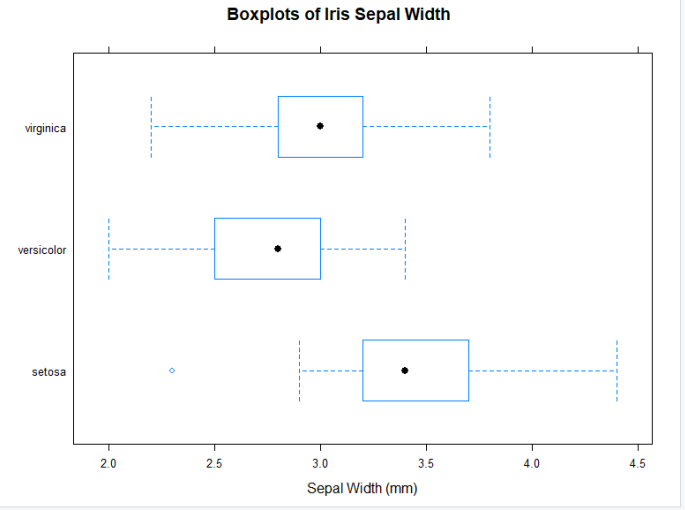

RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
graphicsnr.R
Source on Save
Run
Source
243 image(diagrt.1,xlab="Bill Diagonal",ylab="Right Height")
244 contour(diagrt.1,xlab="Bill Diagonal",ylab="Right Height",add
245 points(Swiss$diagon,Swiss$right,pch=as.character(Swiss$genu),
246
247 Genuine <- Swiss$genu
248 Genuine[Genuine==0] <- "Forged"
249 Genuine[Genuine==1] <- "Real"
250 xyplot(Swiss$diagon~Swiss$bottom|Genuine,groups=Genuine)
251
252 splom(~Swiss[,1:6],groups=Genuine,auto.key=list(columns=2))
253
254 pairs.image <- function(x) {
255 pairs(x,panel=function(x,y) {
256 foo <- bin2(cbind(x,y),nbin=c(75,75))
257
252:60 (Top Level) R Script
Console Terminal Jobs
> coplot(upoz~safb|inbh*v500,number=c(4,4),overlap=.25,panel=functi
on(x,y,...) panel.smooth(x,y,span=.6,...),data=Ozdata)
>
>
> library(ash)
> Swiss = read.csv("http://course1.winona.edu/bdeppa/DSCI%20415/Dat
a/Swiss.csv")
> names(Swiss)
[1] "leng" "left" "right" "bottom" "top" "diagon"
[7] "genu"
>
> d1 = ash1(bin1(Swiss$diagon,nbin=50),3)
> hist(Swiss$diagon,nclass=20,prob=T,col="pink",main="Histogram of
Bill Diagonal (mm)")
> lines(d1)
>
> 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="pink")
>
Environment History Connections
Files Plots Packages Help Viewer
Zoom Export Publish
setosa versicolor virginica
Petal.Width
Petal.Length
Sepal.Width
Sepal.Length
Scatter Plot Matrix
14:55

RStudio

File Edit Code View Plots Session Build Debug Profile Tools Help

```
243 image(diagrt.1,xlab= Bill Diagonal ,ylab= Right Height )
244 contour(diagrt.1,xlab="Bill Diagonal",ylab="Right Height",add
245 points(Swiss$diagon,Swiss$right,pch=as.character(Swiss$genu),
246
247 Genuine <- Swiss$genu
248 Genuine[Genuine==0] <- "Forged"
249 Genuine[Genuine==1] <- "Real"
250 xyplot(Swiss$diagon~Swiss$bottom|Genuine,groups=Genuine)
251
252 splom(~Swiss[,1:6],groups=Genuine,auto.key=list(columns=2))
253
254 pairs.image <- function(x) {
255   pairs(x,panel=function(x,y) {
256     foo <- bin2(cbind(x,y),nbin=c(75,75))
257
```

```
> coplot(upoz~safb|inbh*v500,number=c(4,4),overlap=.25,panel=functi
on(x,y,...) panel.smooth(x,y,span=.6,...),data=Ozdata)
>
>
> library(ash)
> Swiss = read.csv("http://course1.winona.edu/bdeppa/DSCI%20415/Dat
a/Swiss.csv")
> names(Swiss)
[1] "leng"   "left"   "right"  "bottom" "top"    "diagon"
[7] "genu"
>
> d1 = ash1(bin1(Swiss$diagon,nbin=50),3)
> hist(Swiss$diagon,nclass=20,prob=T,col="pink",main="Histogram of
 Bill Diagonal (mm)")
> lines(d1)
>
> 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="pink")
>
```

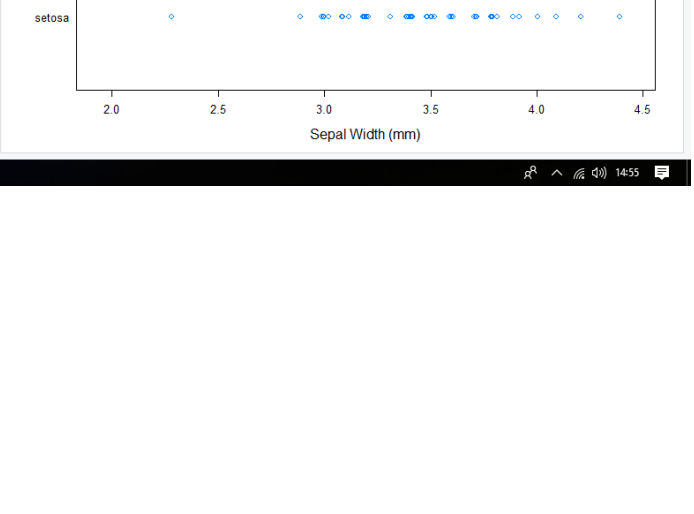


RStudio

File Edit Code View Plots Session Build Debug Profile Tools Help

```
243 image(diagrt.1,xlab= Bill Diagonal ,ylab= Right Height )
244 contour(diagrt.1,xlab="Bill Diagonal",ylab="Right Height",add
245 points(Swiss$diagon,Swiss$right,pch=as.character(Swiss$genu),
246
247 Genuine <- Swiss$genu
248 Genuine[Genuine==0] <- "Forged"
249 Genuine[Genuine==1] <- "Real"
250 xyplot(Swiss$diagon~Swiss$bottom|Genuine,groups=Genuine)
251
252 splom(~Swiss[,1:6],groups=Genuine,auto.key=list(columns=2))
253
254 pairs.image <- function(x) {
255   pairs(x,panel=function(x,y) {
256     foo <- bin2(cbind(x,y),nbin=c(75,75))
257
```

```
> coplot(upoz~safb|inbh*v500,number=c(4,4),overlap=.25,panel=functi
on(x,y,...) panel.smooth(x,y,span=.6,...),data=Ozdata)
>
>
> library(ash)
> Swiss = read.csv("http://course1.winona.edu/bdeppa/DSCI%20415/Dat
a/Swiss.csv")
> names(Swiss)
[1] "leng"   "left"   "right"  "bottom" "top"    "diagon"
[7] "genu"
>
> d1 = ash1(bin1(Swiss$diagon,nbin=50),3)
> hist(Swiss$diagon,nclass=20,prob=T,col="pink",main="Histogram of
 Bill Diagonal (mm)")
> lines(d1)
>
> 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="pink")
>
```



RStudio

File Edit Code View Plots Session Build Debug Profile Tools Help

graphicsinr.R

```
239 diagrt.1 <- ash2(diagrt.bin,m=c(5,5))
240 persp(diagrt.1,xlab="Diagonal Length",ylab="Right Height",zla
241
242
243 image(diagrt.1,xlab="Bill Diagonal",ylab="Right Height")
244 contour(diagrt.1,xlab="Bill Diagonal",ylab="Right Height",add
245 points(Swiss$diagon,Swiss$right,pch=as.character(Swiss$genu),
246
247 Genuine <- Swiss$genu
248 Genuine[Genuine==0] <- "Forged"
249 Genuine[Genuine==1] <- "Real"
250 xyplot(Swiss$diagon~Swiss$bottom|Genuine,groups=Genuine)
251
252
```

252:60 (Top Level) R Script

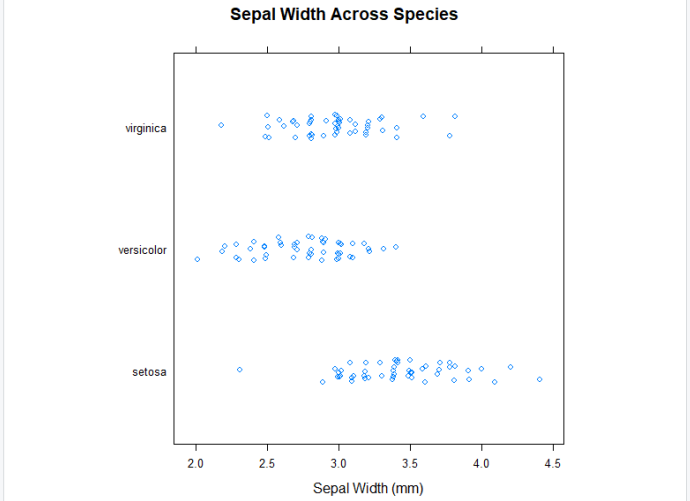
Console Terminal Jobs

```
> coplot(upoz~safb|inbh*v500,number=c(4,4),overlap=.25,panel=functi
on(x,y,...) panel.smooth(x,y,span=.6,...),data=Ozdata)
>
>
> library(ash)
> Swiss = read.csv("http://course1.winona.edu/bdeppa/DSCI%20415/Dat
a/Swiss.csv")
> names(Swiss)
[1] "leng"   "left"   "right"  "bottom" "top"    "diagon"
[7] "genu"
>
> d1 = ash1(bin1(Swiss$diagon,nbin=50),3)
> hist(Swiss$diagon,nclass=20,prob=T,col="pink",main="Histogram of
 Bill Diagonal (mm)")
> lines(d1)
>
> 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="pink")
>
```

Environment History Connections

Files Plots Packages Help Viewer

Zoom Export Publish



14:56

RStudio

File Edit Code View Plots Session Build Debug Profile Tools Help

graphicsinr.R

```
239 diagrt.1 <- ash2(diagrt.bin,m=c(5,5))
240 persp(diagrt.1,xlab="Diagonal Length",ylab="Right Height",zla
241
242
243 image(diagrt.1,xlab="Bill Diagonal",ylab="Right Height")
244 contour(diagrt.1,xlab="Bill Diagonal",ylab="Right Height",add
245 points(Swiss$diagon,Swiss$right,pch=as.character(Swiss$genu),
246
247 Genuine <- Swiss$genu
248 Genuine[Genuine==0] <- "Forged"
249 Genuine[Genuine==1] <- "Real"
250 xyplot(Swiss$diagon~Swiss$bottom|Genuine,groups=Genuine)
251
252
```

252:60 (Top Level) R Script

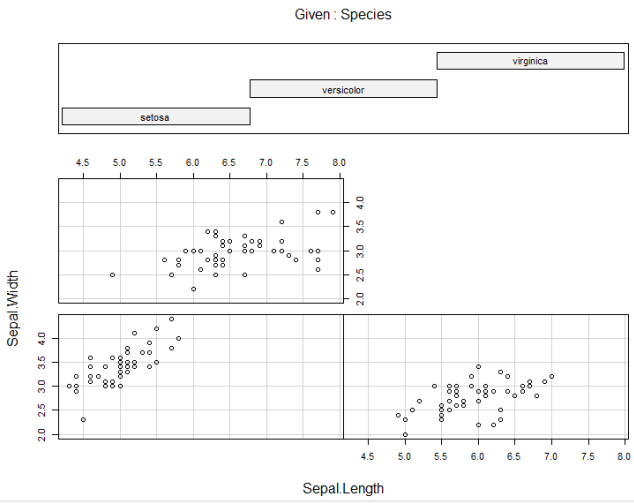
Console Terminal Jobs

```
> coplot(upoz~safb|inbh*v500,number=c(4,4),overlap=.25,panel=functi
on(x,y,...) panel.smooth(x,y,span=.6,...),data=Ozdata)
>
>
> library(ash)
> Swiss = read.csv("http://course1.winona.edu/bdeppa/DSCI%20415/Dat
a/Swiss.csv")
> names(Swiss)
[1] "leng"   "left"   "right"  "bottom" "top"    "diagon"
[7] "genu"
>
> d1 = ash1(bin1(Swiss$diagon,nbin=50),3)
> hist(Swiss$diagon,nclass=20,prob=T,col="pink",main="Histogram of
 Bill Diagonal (mm)")
> lines(d1)
>
> 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="pink")
>
```

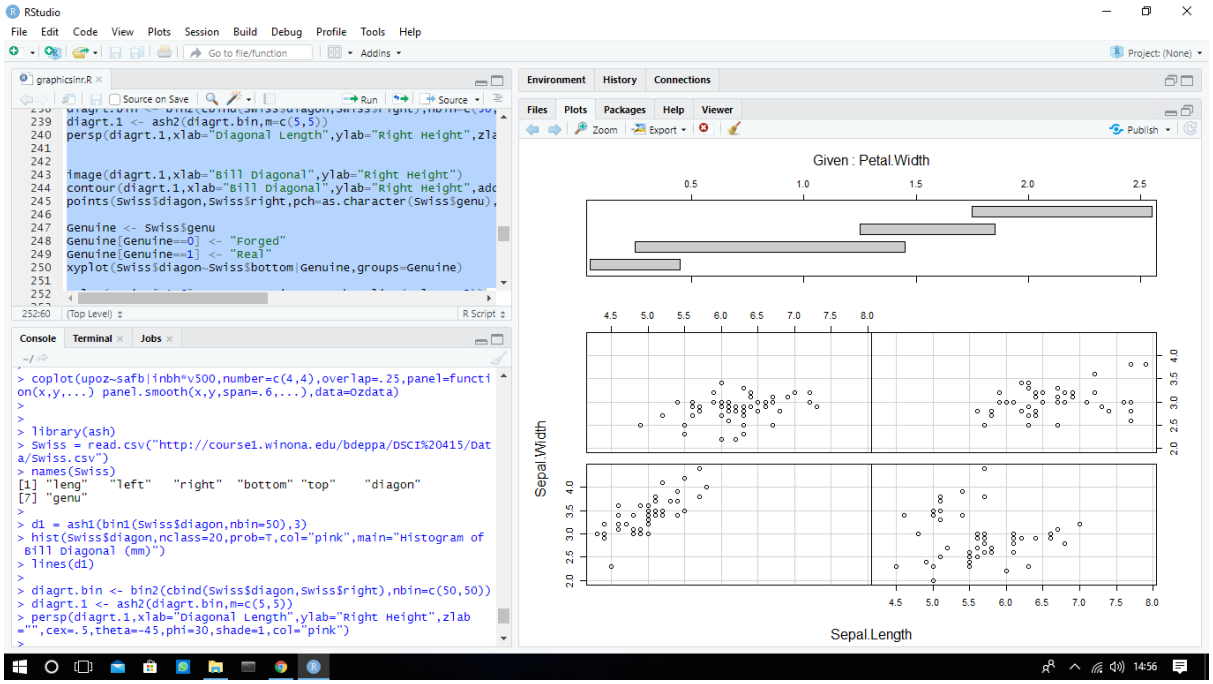
Environment History Connections

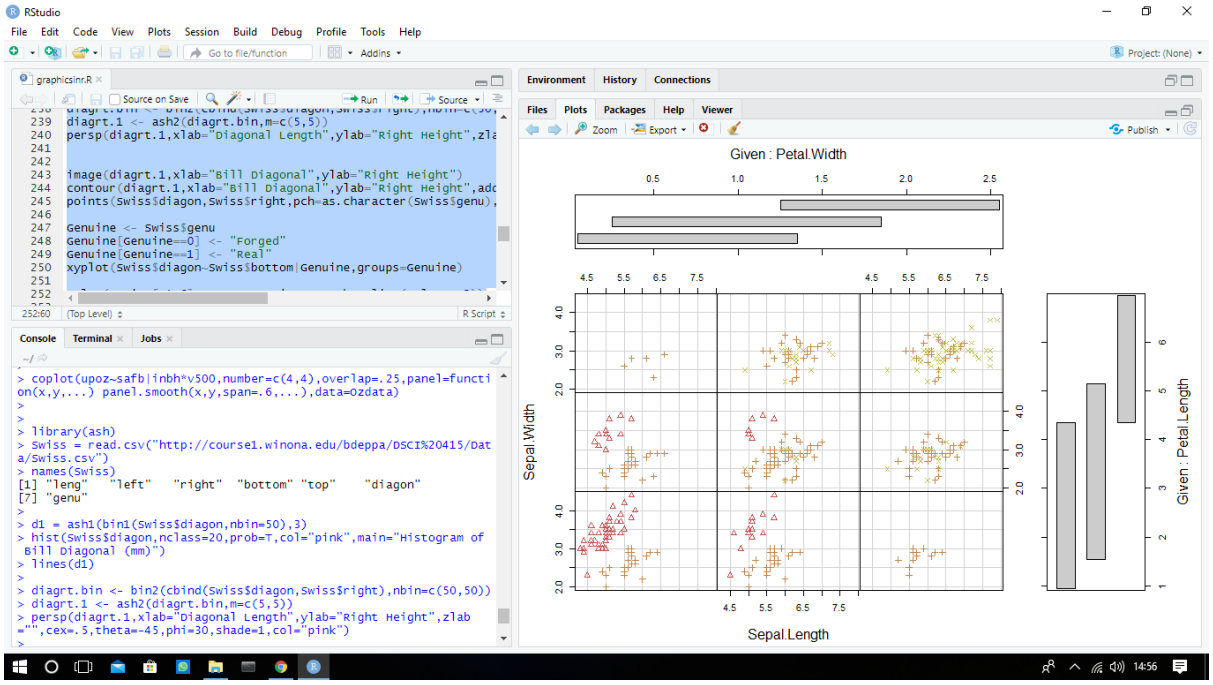
Files Plots Packages Help Viewer

Zoom Export Publish



14:56


RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
graphicsinr.R
diagrt.1 <- ash2(diagrt.bin,m=c(5,5))
persp(diagrt.1,xlab="Diagonal Length",ylab="Right Height",zla
image(diagrt.1,xlab="Bill Diagonal",ylab="Right Height")
contour(diagrt.1,xlab="Bill Diagonal",ylab="Right Height",add
points(Swiss$diagon,Swiss$right,pch=as.character(Swiss$genu),
Genuine <- Swiss$genu
Genuine[Genuine==0] <- "Forged"
Genuine[Genuine==1] <- "Real"
xyplot(Swiss$diagon~Swiss$bottom|Genuine,groups=Genuine)
Console Terminal Jobs
> coplot(upoz~safb|inbh*v500,number=c(4,4),overlap=.25,panel=functi
on(x,y,...) panel.smooth(x,y,span=.6,...),data=Ozdata)
> library(ash)
> Swiss = read.csv("http://course1.winona.edu/bdeppa/DSCI%20415/Dat
a/Swiss.csv")
> names(Swiss)
[1] "leng" "left" "right" "bottom" "top" "diagon"
[7] "genu"
> d1 = ash1(bin1(Swiss$diagon,nbin=50),3)
> hist(Swiss$diagon,nclass=20,prob=T,col="pink",main="Histogram of
Bill Diagonal (mm)")
> lines(d1)
> 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="pink")
Environment History Connections
Files Plots Packages Help Viewer
Given : Petal.Width
Sepal.Width
Sepal.Length

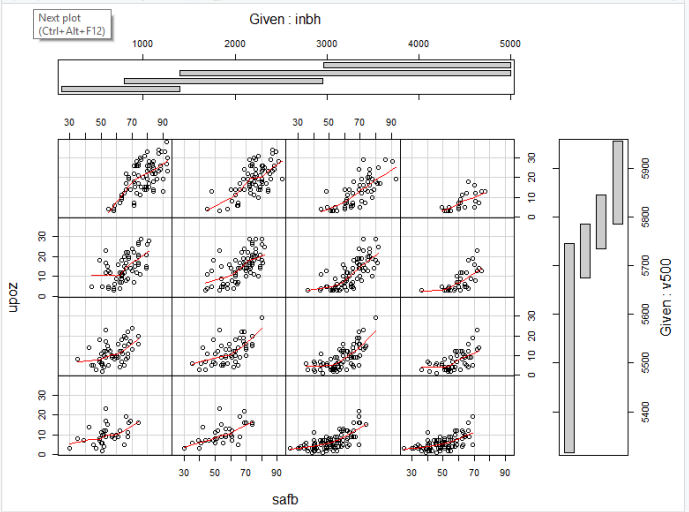

RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
graphicsinr.R
diagrt.1 <- ash2(diagrt.bin,m=c(5,5))
persp(diagrt.1,xlab="Diagonal Length",ylab="Right Height",zla
image(diagrt.1,xlab="Bill Diagonal",ylab="Right Height")
contour(diagrt.1,xlab="Bill Diagonal",ylab="Right Height",add
points(Swiss$diagon,Swiss$right,pch=as.character(Swiss$genu),
Genuine <- Swiss$genu
Genuine[Genuine==0] <- "Forged"
Genuine[Genuine==1] <- "Real"
xyplot(Swiss$diagon~Swiss$bottom|Genuine,groups=Genuine)
Console Terminal Jobs
> coplot(upoz~safb|inbh*v500,number=c(4,4),overlap=.25,panel=functi
on(x,y,...) panel.smooth(x,y,span=.6,...),data=Ozdata)
> library(ash)
> Swiss = read.csv("http://course1.winona.edu/bdeppa/DSCI%20415/Dat
a/Swiss.csv")
> names(Swiss)
[1] "leng" "left" "right" "bottom" "top" "diagon"
[7] "genu"
> d1 = ash1(bin1(Swiss$diagon,nbin=50),3)
> hist(Swiss$diagon,nclass=20,prob=T,col="pink",main="Histogram of
Bill Diagonal (mm)")
> lines(d1)
> 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="pink")
Environment History Connections
Files Plots Packages Help Viewer
Given : Petal.Width
Sepal.Width
Given : Petal.Length
Sepal.Length

RStudio

File Edit Code View Plots Session Build Debug Profile Tools Help

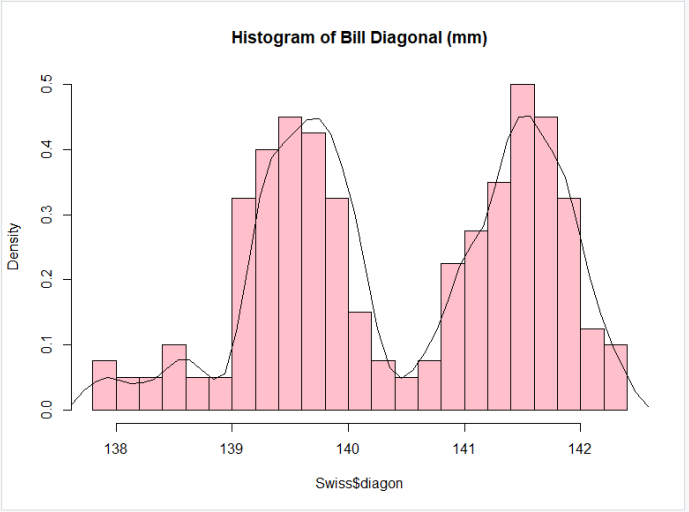
```
239 diagrt.1 <- ash2(diagrt.bin,m=c(5,5))
240 persp(diagrt.1,xlab="Diagonal Length",ylab="Right Height",zla
241
242
243 image(diagrt.1,xlab="Bill Diagonal",ylab="Right Height")
244 contour(diagrt.1,xlab="Bill Diagonal",ylab="Right Height",add
245 points(Swiss$diagon,Swiss$right,pch=as.character(Swiss$genu),
246
247 Genuine <- Swiss$genu
248 Genuine[Genuine==0] <- "Forged"
249 Genuine[Genuine==1] <- "Real"
250 xyplot(Swiss$diagon~Swiss$bottom|Genuine,groups=Genuine)
251
252
```

```
> coplot(upoz~safb|inbh*v500,number=c(4,4),overlap=.25,panel=functi
on(x,y,...) panel.smooth(x,y,span=.6,...),data=Ozdata)
>
>
> library(ash)
> Swiss = read.csv("http://course1.winona.edu/bdeppa/DSCI%20415/Dat
a/Swiss.csv")
> names(Swiss)
[1] "leng"   "left"   "right"  "bottom" "top"    "diagon"
[7] "genu"
>
> d1 = ash1(bin1(Swiss$diagon,nbin=50),3)
> hist(Swiss$diagon,nclass=20,prob=T,col="pink",main="Histogram of
 Bill Diagonal (mm)")
> lines(d1)
>
> 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="pink")
>
```



RStudio

File Edit Code View Plots Session Build Debug Profile Tools Help

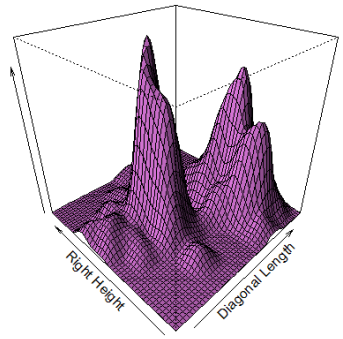


RStudio

File Edit Code View Plots Session Build Debug Profile Tools Help

```
234  d1 = ash1(bin1(Swiss$diagon,nbin=50),3)
235  hist(Swiss$diagon,nclass=20,prob=T,col="pink",main="Histogram
236  lines(d1)
237
238  diagrt.bin <- bin2(cbind(Swiss$diagon,Swiss$right),nbin=c(50,
239  diagrt.1 <- ash2(diagrt.bin,m=c(5,5))
240  persp(diagrt.1,xlab="Diagonal Length",ylab="Right Height",zla
241
242
243  image(diagrt.1,xlab="Bill Diagonal",ylab="Right Height")
244
```

```
> 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))
> persp(diagrt.1,xlab="Diagonal Length",ylab="Right Height",zlab
="",cex=.5,theta=-45,phi=30,shade=1,col="purple")
> persp(diagrt.1,xlab="Diagonal Length",ylab="Right Height",zlab
="",cex=.5,theta=-45,phi=30,shade=1,col="violet")
> persp(diagrt.1,xlab="Diagonal Length",ylab="Right Height",zlab
="",cex=.5,theta=-45,phi=30,shade=2,col="violet")
> persp(diagrt.1,xlab="Diagonal Length",ylab="Right Height",zlab
="",cex=.5,theta=-45,phi=30,shade=0.5,col="violet")
>
```

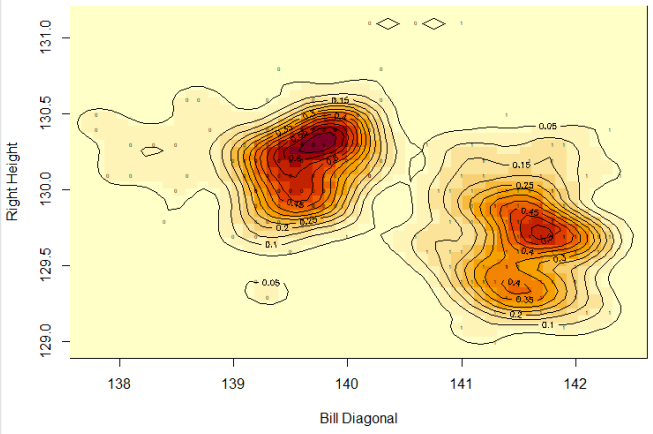


RStudio

File Edit Code View Plots Session Build Debug Profile Tools Help

```
298    points(kodiak.crab$longit,kodiak.crab$latit,pch=".")
299  }
300  nr.pp = nrec/npots
301  crab.plot(lat,long,y=nr.pp,year=83,yname="Number of Recruits
302
303  detach(survey.crab)
304  detach(kodiak.crab)  # if you attach, be sure to detach when
305
306  attach(City)         # This command readies the data frame cit
307
308
```

```
>
> 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) {
```

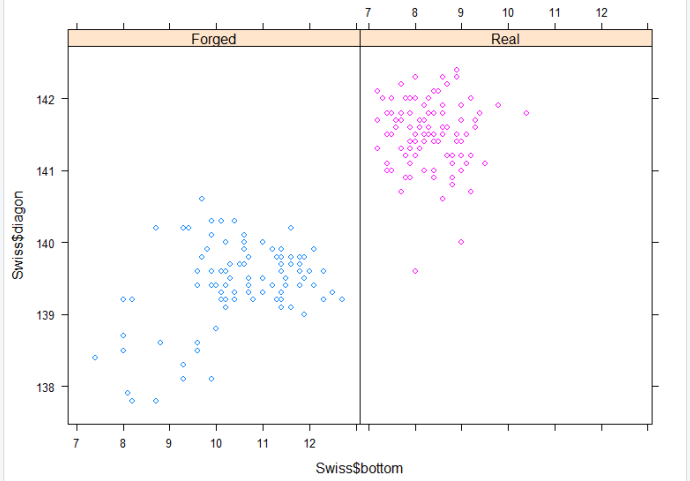


RStudio

File Edit Code View Plots Session Build Debug Profile Tools Help

```
  points(kodiak.crab$longit,kodiak.crab$latit,pch=".")
}
nr.pp = nrec/npots
crab.plot(lat,long,y=nr.pp,year=83,yname="Number of Recruits

detach(survey.crab)
detach(kodiak.crab)  # if you attach, be sure to detach when

attach(City)         # This command readies the data frame cit
```

```
> 
> 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) {
```

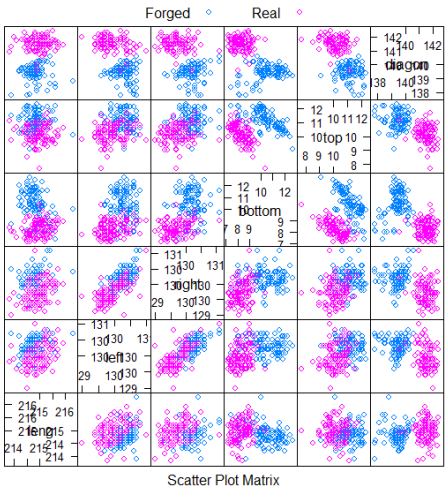


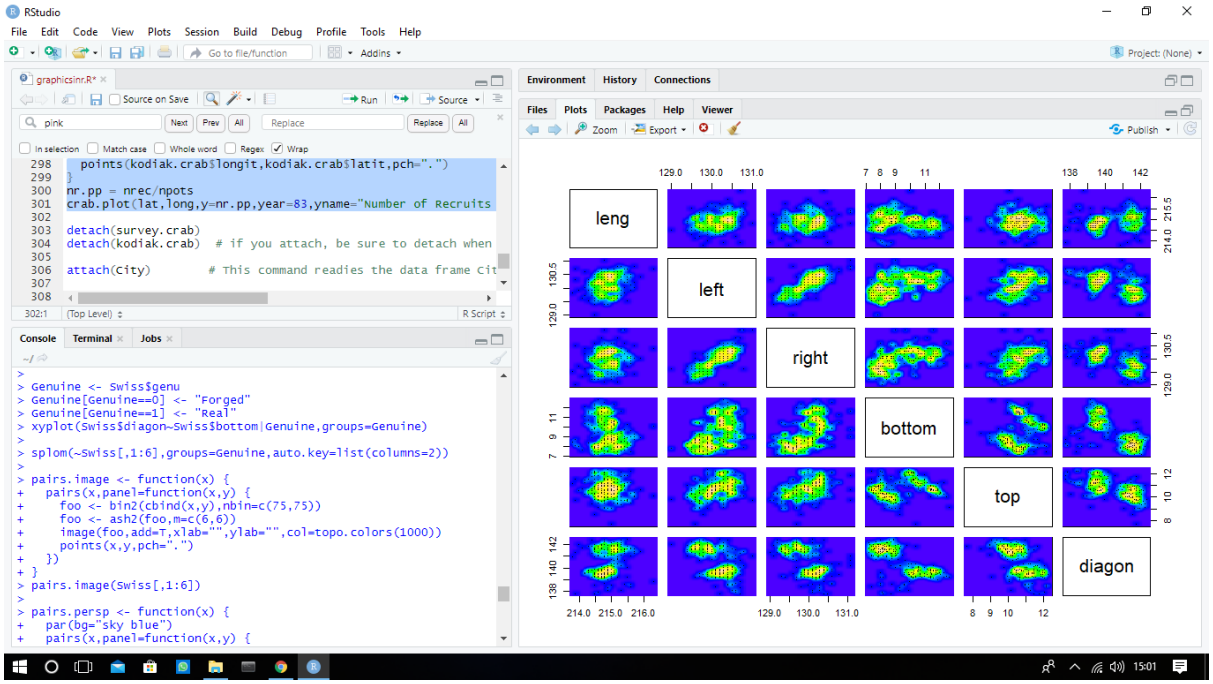
RStudio

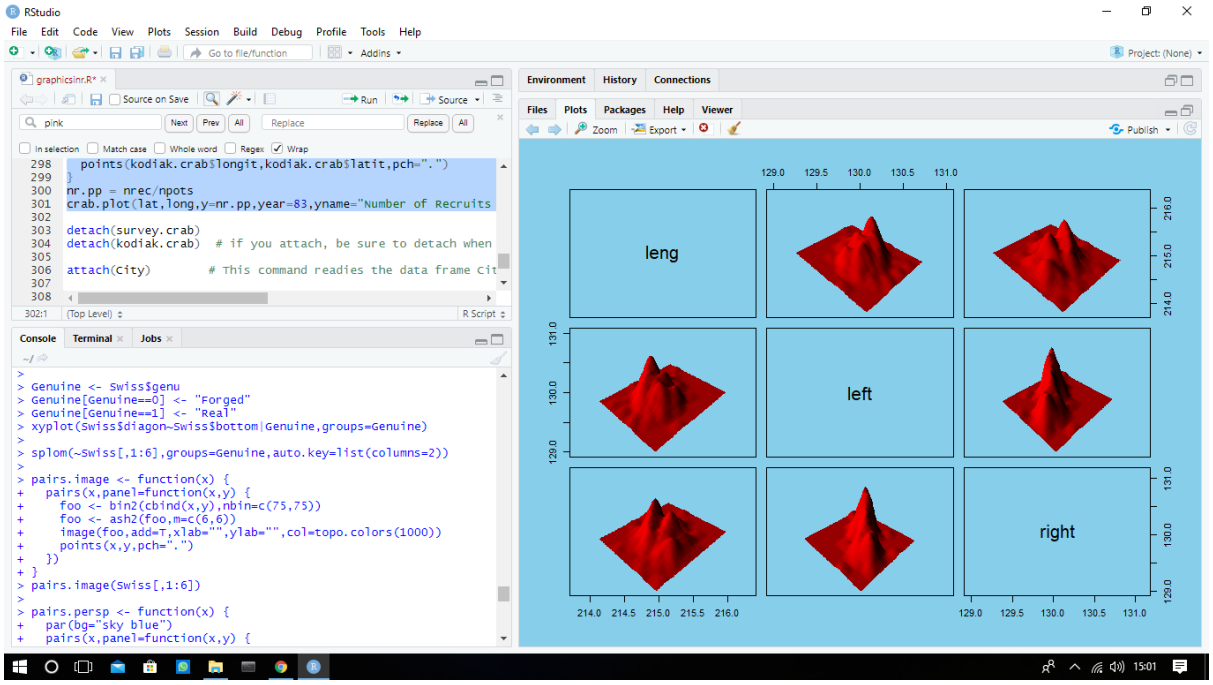
File Edit Code View Plots Session Build Debug Profile Tools Help

```
  points(kodiak.crab$longit,kodiak.crab$latit,pch=".")
}
nr.pp = nrec/npots
crab.plot(lat,long,y=nr.pp,year=83,yname="Number of Recruits

detach(survey.crab)
detach(kodiak.crab)  # if you attach, be sure to detach when

attach(City)         # This command readies the data frame cit
```

```
> 
> 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) {
```




RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
graphicsinr.R*
Source on Save
Run
Source
pink
Next Prev All
Replace
Replace All
In selection Match case Whole word Regex Wrap
298 points(kodiak.crab$longit,kodiak.crab$latit,pch=".")
299 }
300 nr.pp = nrec/npots
301 crab.plot(lat,long,y=nr.pp,year=83,yname="Number of Recruits
302
303 detach(survey.crab)
304 detach(kodiak.crab) # if you attach, be sure to detach when
305
306 attach(City) # This command readies the data frame cit
307
308
302:1 (Top Level) R Script
Console Terminal Jobs
> 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) {
Environment History Connections
Files Plots Packages Help Viewer
Zoom Export Publish
leng
left
right
bottom
top
diagon
15:01

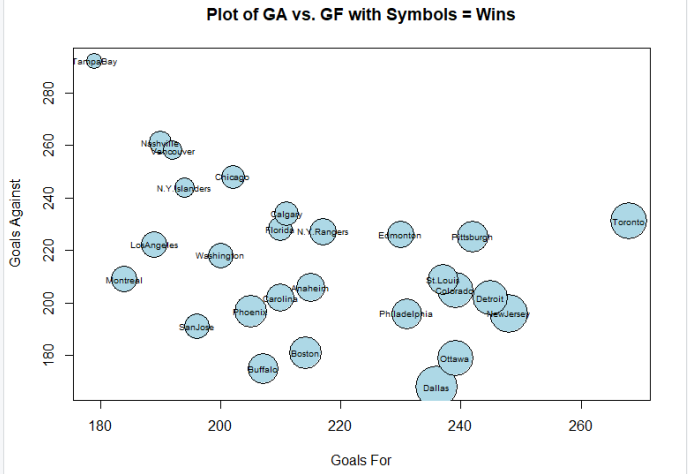

RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
graphicsinr.R*
Source on Save
Run
Source
pink
Next Prev All
Replace
Replace All
In selection Match case Whole word Regex Wrap
298 points(kodiak.crab$longit,kodiak.crab$latit,pch=".")
299 }
300 nr.pp = nrec/npots
301 crab.plot(lat,long,y=nr.pp,year=83,yname="Number of Recruits
302
303 detach(survey.crab)
304 detach(kodiak.crab) # if you attach, be sure to detach when
305
306 attach(City) # This command readies the data frame cit
307
308
302:1 (Top Level) R Script
Console Terminal Jobs
> 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) {
Environment History Connections
Files Plots Packages Help Viewer
Zoom Export Publish
leng
left
right
15:01

RStudio

File Edit Code View Plots Session Build Debug Profile Tools Help

```
298   points(kodiak.crab$longit,kodiak.crab$latit,pch=".")
299   }
300   nr.pp = nrec/npots
301   crab.plot(lat,long,y=nr.pp,year=83,yname="Number of Recruits
302
303   detach(survey.crab)
304   detach(kodiak.crab)  # if you attach, be sure to detach when
305
306   attach(City)         # This command readies the data frame cit
307
308
```

```
> 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) {
```

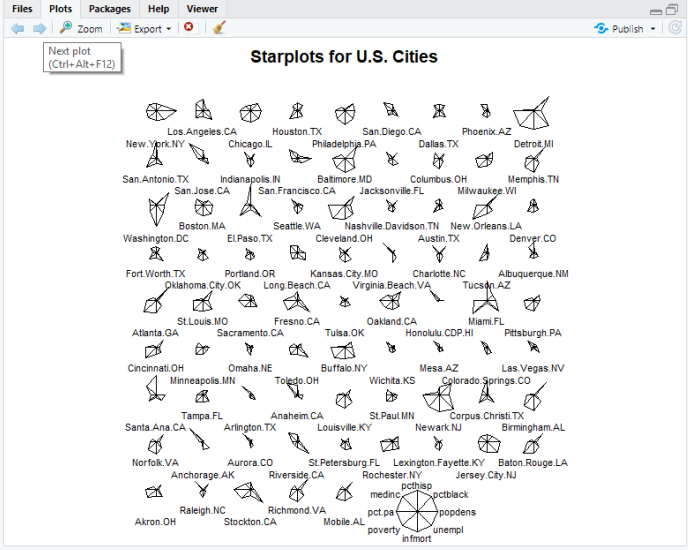


RStudio

File Edit Code View Plots Session Build Debug Profile Tools Help

	pop90	pop80	growth	area	popdens	blackpop	hispc
New.York.NY	7311966	7071639	3.4	309	23671	2102512	1
Los.Angeles.CA	3489779	2968528	17.6	469	7436	487674	1
Chicago.IL	2768483	3005072	-7.9	227	12185	1087711	5
Houston.TX	1690180	1595138	6.0	540	3131	457990	4
Philadelphia.PA	1552572	1688210	-8.0	135	11492	631936	8
San.Diego.CA	1148851	875538	31.2	324	3546	104261	2
Dallas.TX	1022497	904599	13.0	342	2986	296994	2
Phoenix.AZ	1012230	789704	28.2	420	2411	51053	1
Detroit.MI	1012110	1203368	-15.9	139	7297	777916	2
San.Antonio.TX	966437	785940	23.0	333	2902	65884	5
San.Jose.CA	801331	629400	27.3	171	4678	36790	2
Indianapolis.IN	746538	700807	6.5	362	2064	165570	7
San.Francisco.CA	728921	678974	7.4	47	15609	79039	1
Baltimore.MD	726096	786741	-7.7	81	8986	435768	7
Jacksonville.FL	661177	540920	22.2	759	871	160283	1
Columbus.OH	642987	565021	13.8	191	3368	142748	6
Milwaukee.WI	617043	636297	-3.0	96	6421	191255	3
Memphis.TN	610275	646174	-5.6	256	2384	334737	4
Washington.DC	585221	638432	-8.3	61	9531	399604	3

Showing 1 to 23 of 77 entries, 72 total columns

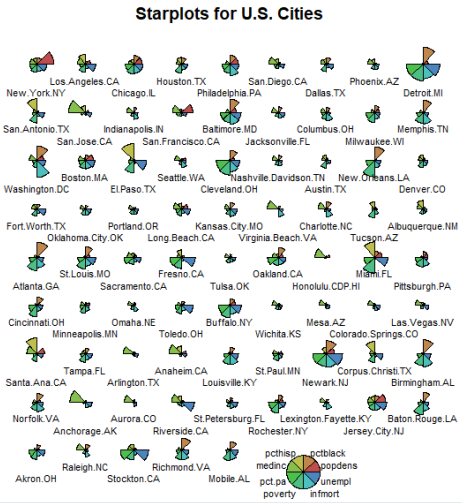


RStudio

File Edit Code View Plots Session Build Debug Profile Tools Help

```
318 palette("default")
319
320 library(MASS)
321
322 parcoord(city.mat)
323 names(olives)
324 olive.mat = olives[,3:9]
325 palette(rainbow(12,s=0.6,v=0.75))
326 parcoord(olive.mat,col=as.numeric(olives$Area.Name))
327
328
329
330
```

```
[66] "ctrend"    "zeroday"   "freeze"    "hot"       "ddays"
[71] "high.jul"  "low.jan"
>
> view(City)
>
> city.mat <-cbind(popdens,pctblack,pcthisp,medinc,pct.pa,poverty,i
nfmort,unempl)
> stars(city.mat, key.loc = c(15,1.25),main = "Starplots for U.S. C
ities",label=row.names(City),cex=.7)
>
> palette(rainbow(12, s = 0.6, v = 0.75))
> stars(city.mat,len=.80,key.loc = c(15,1.25),main = "Starplots for
 U.S. Cities",draw.segments = TRUE,label=row.names(City),cex=.7)
>
> palette("default")
>
> library(MASS)
>
> parcoord(city.mat)
> names(olives)
 [1] "Region.Name" "Area.Name"   "Palmitic"    "Palmitoleic"
```

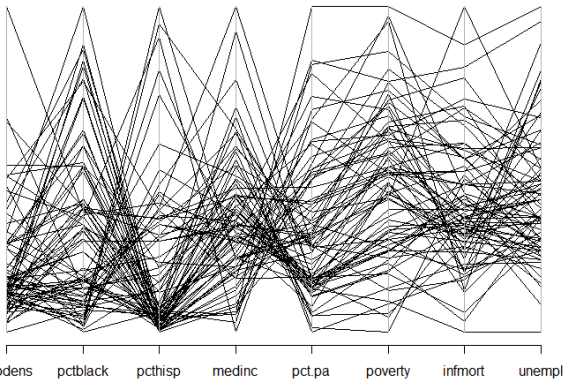


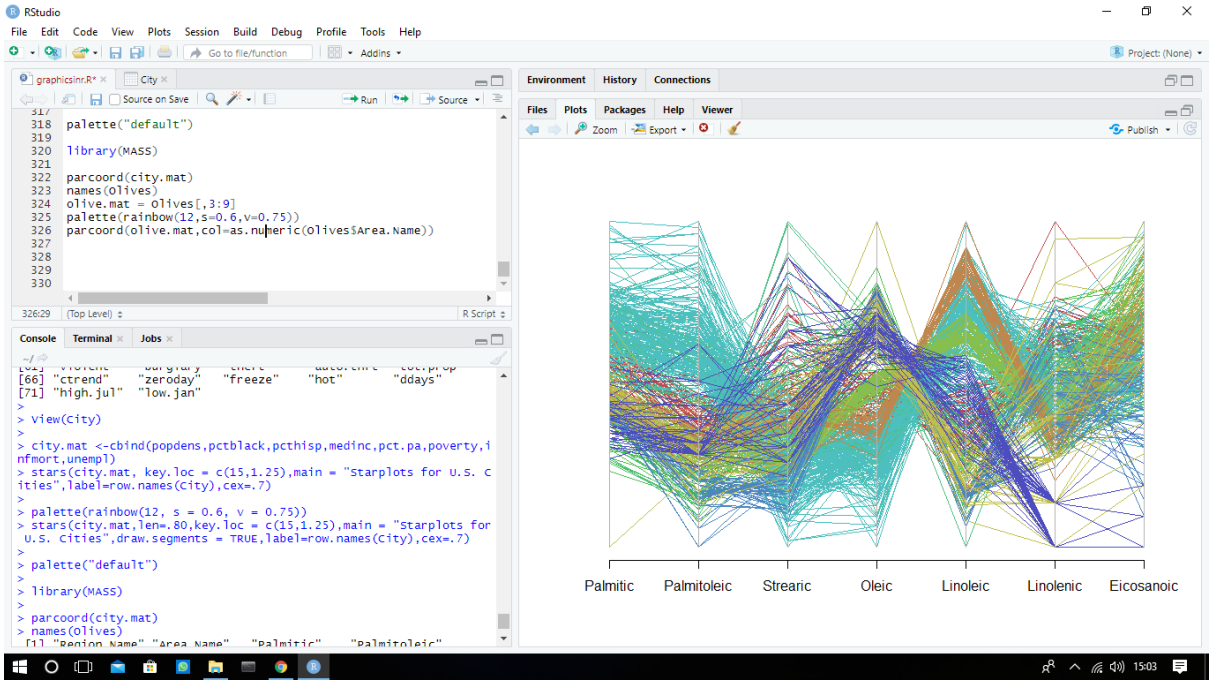
RStudio

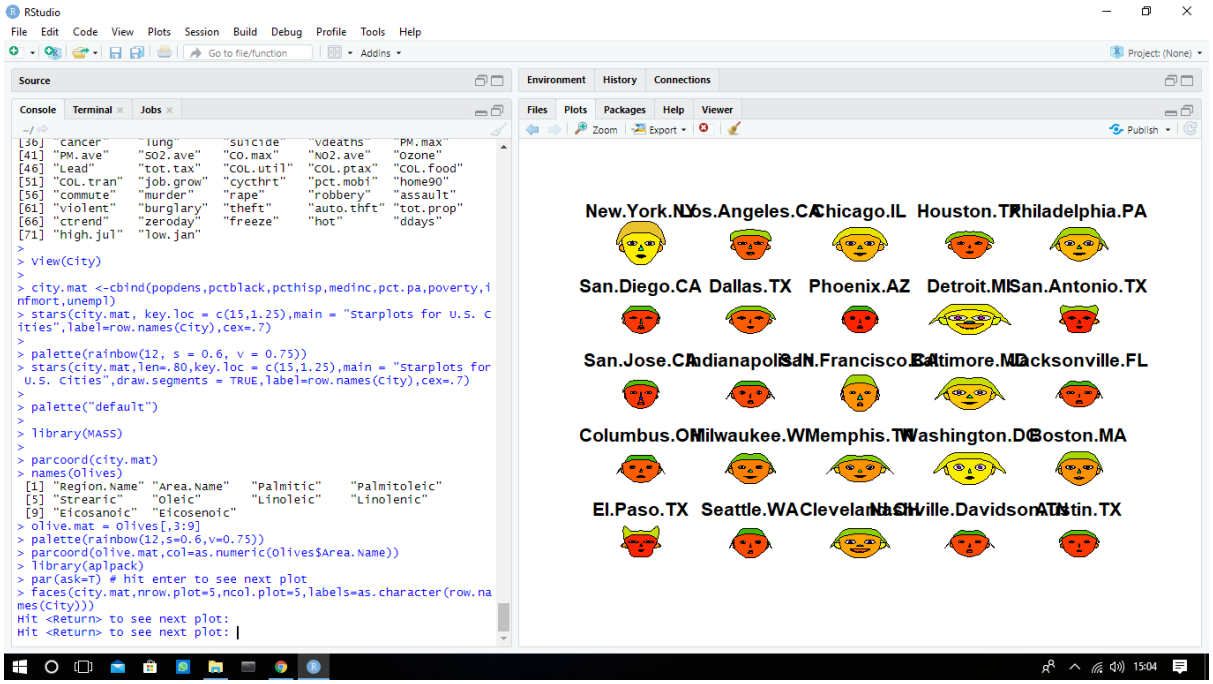
File Edit Code View Plots Session Build Debug Profile Tools Help

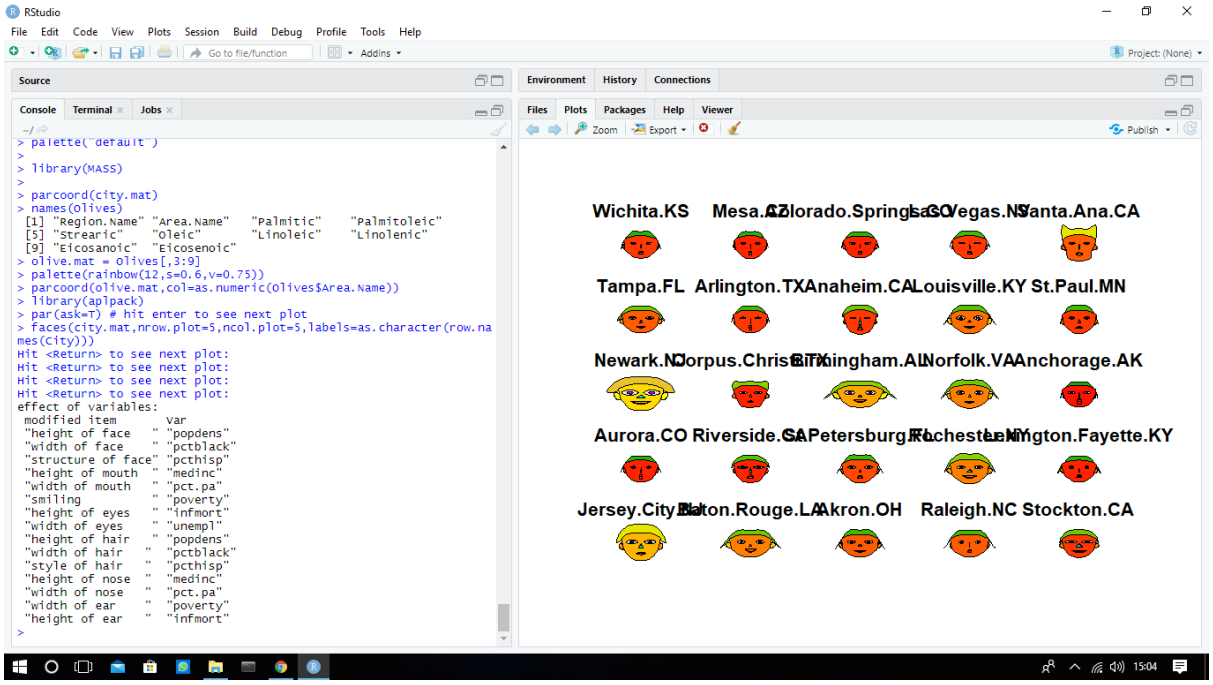
```
318 palette("default")
319
320 library(MASS)
321
322 parcoord(city.mat)
323 names(olives)
324 olive.mat = olives[,3:9]
325 palette(rainbow(12,s=0.6,v=0.75))
326 parcoord(olive.mat,col=as.numeric(olives$Area.Name))
327
328
329
330
```

```
[66] "ctrend"    "zeroday"   "freeze"    "hot"       "ddays"
[71] "high.jul"  "low.jan"
>
> view(City)
>
> city.mat <-cbind(popdens,pctblack,pcthisp,medinc,pct.pa,poverty,i
nfmort,unempl)
> stars(city.mat, key.loc = c(15,1.25),main = "Starplots for U.S. C
ities",label=row.names(City),cex=.7)
>
> palette(rainbow(12, s = 0.6, v = 0.75))
> stars(city.mat,len=.80,key.loc = c(15,1.25),main = "Starplots for
 U.S. Cities",draw.segments = TRUE,label=row.names(City),cex=.7)
>
> palette("default")
>
> library(MASS)
>
> parcoord(city.mat)
> names(olives)
 [1] "Region.Name" "Area.Name"   "Palmitic"    "Palmitoleic"
```




RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
graphicsInr.R*
City
Source on Save
Run
Source
palette("default")
library(MASS)
parcoord(city.mat)
names(olives)
olive.mat = olives[,3:9]
palette(rainbow(12,s=0.6,v=0.75))
parcoord(olive.mat,col=as.numeric(olives$Area.Name))
Console
Terminal
Jobs
[66] "ctrend" "zeroday" "freeze" "hot" "ddays"
[71] "high.jul" "low.jan"
> view(City)
> city.mat <-cbind(popdens,pctblack,pcthisp,medinc,pct.pa,poverty,infmort,unempl)
> stars(city.mat, key.loc = c(15,1.25),main = "Starplots for U.S. Cities",label=row.names(City),cex=.7)
> palette(rainbow(12, s = 0.6, v = 0.75))
> stars(city.mat,len=.80,key.loc = c(15,1.25),main = "Starplots for U.S. Cities",draw.segments = TRUE,label=row.names(City),cex=.7)
> palette("default")
> library(MASS)
> parcoord(city.mat)
> names(olives)
Environment
History
Connections
Files
Plots
Packages
Help
Viewer
Zoom
Export
Publish
Palmitic
Palmitoleic
Strearic
Oleic
Linoleic
Linolenic
Eicosanoic





Conclusion : Thus we have studied basic and advanced graphics in R.

Experiment No : 9

Aim: Basic of correlation, simple and multiple regression in R

Theory:

Correlation and linear regression each explore the relationship between two quantitative variables. Both are very common analyses.

Correlation determines if one variable varies systematically as another variable changes. It does not specify that one variable is the dependent variable and the other is the independent variable. Often, it is useful to look at which variables are correlated to others in a data set, and it is especially useful to see which variables correlate to a particular variable of interest.

The three forms of correlation presented here are Pearson, Kendall, and Spearman. The test determining the *p*-value for Pearson correlation is a parametric test that assumes that data are bivariate normal. Kendall and Spearman correlation use nonparametric tests.

In contrast, linear regression specifies one variable as the independent variable and another as the dependent variable. The resultant model relates the variables with a linear relationship.

The tests associated with linear regression are parametric and assume normality, homoscedasticity, and independence of residuals, as well as a linear relationship between the two variables.

Appropriate data

- For Pearson correlation, two interval/ratio variables. Together the data in the variables are bivariate normal. The relationship between the two variables is linear. Outliers can detrimentally affect results.
- For Kendall correlation, two variables of interval/ratio or ordinal type.
- For Spearman correlation, two variables of interval/ratio or ordinal type.
- For linear regression, two interval/ratio variables. The relationship between the two variables is linear. Residuals are normal, independent, and homoscedastic. Outliers can affect the results unless robust methods are used.

Hypotheses

- For correlation, null hypothesis: The correlation coefficient (*r*, *tau*, or *rho*) for the sampled population is zero. Or, there is no correlation between the two variables.
- For linear regression, null hypothesis: The slope of the fit line for the sampled population is zero. Or, there is no linear relationship between the two variables.

Interpretation

- For correlation, reporting significant results as “Variable A was significantly correlated to Variable B” is acceptable. Alternatively, “A significant correlation

between Variable A and Variable B was found." Or, "Variables A, B, and C were significantly correlated."

- For linear regression, reporting significant results as "Variable A was significantly linearly related to Variable B" is acceptable. Alternatively, "A significant linear regression between Variable A and Variable B was found."

Output:

Code:

```
install.packages("ggpubr")
library("ggpubr")

my_data <- mtcars
head(my_data, 6)

ggscatter(my_data, x = "mpg", y = "wt",
        add = "reg.line", conf.int = TRUE,
        cor.coef = TRUE, cor.method = "pearson",
        xlab = "Miles/(US) gallon", ylab = "Weight (1000 lbs)")

x <- c(151, 174, 138, 186, 128, 136, 179, 163, 152, 131)
y <- c(63, 81, 56, 91, 47, 57, 76, 72, 62, 48)

# Apply the lm() function.
relation <- lm(y~x)

print(relation)

x <- c(151, 174, 138, 186, 128, 136, 179, 163, 152, 131)
y <- c(63, 81, 56, 91, 47, 57, 76, 72, 62, 48)

# Apply the lm() function.
relation <- lm(y~x)

print(summary(relation))


# The predictor vector.
x <- c(151, 174, 138, 186, 128, 136, 179, 163, 152, 131)

# The resposne vector.
y <- c(63, 81, 56, 91, 47, 57, 76, 72, 62, 48)

# Apply the lm() function.
relation <- lm(y~x)

# Find the weight of a person with height 170.
```

```
a <- data.frame(x = 170)
result <-  predict(relation,a)
print(result)

# Create the predictor and response variable.
x <- c(151, 174, 138, 186, 128, 136, 179, 163, 152, 131)
y <- c(63, 81, 56, 91, 47, 57, 76, 72, 62, 48)
relation <- lm(y~x)

# Give the chart file a name.
png(file = "linearregression.png")

# Plot the chart.
plot(y,x,col = "blue",main = "Height & Weight Regression",
    abline(lm(x~y)),cex = 1.3,pch = 16,xlab = "Weight in Kg",ylab = "Height in cm")

# Save the file.
dev.off()

input <- mtcars[,c("mpg","disp","hp","wt")]
print(head(input))
input <- mtcars[,c("mpg","disp","hp","wt")]

# Create the relationship model.
model <- lm(mpg~disp+hp+wt, data = input)

# Show the model.
print(model)

# Get the Intercept and coefficients as vector elements.
cat("# # # # The Coefficient Values # # # ","\n")

a <- coef(model)[1]
print(a)

Xdisp <- coef(model)[2]
Xhp <- coef(model)[3]
Xwt <- coef(model)[4]

print(Xdisp)
print(Xhp)
print(Xwt)
```

Screenshot:

RStudio

File Edit Code View Plots Session Build Debug Profile Tools Help

Go to file/function | Addins | Project: (None)

Console Terminal Jobs

```
>
> cor.test(mtcars$qsec, mtcars$cyl, method = "spearman")

        Spearman's rank correlation rho

data:  mtcars$qsec and mtcars$cyl
S = 8578.7, p-value = 0.0006196
alternative hypothesis: true rho is not equal to 0
sample estimates:
       rho
-0.5723509

Warning message:
In cor.test.default(mtcars$qsec, mtcars$cyl, method = "spearman") :
  Cannot compute exact p-value with ties
>
> 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'
>
```

Environment History Connections

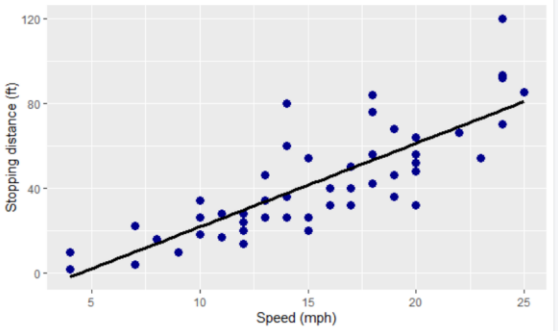
Import Dataset | List | Global Environment

Data

banner	List of 7
dat	10 obs. of 3 variables
h	626 obs. of 70 variables
loafercreek	Formal class SoilProfileCollection
m	int [1:2, 1:5] 1 2 3 4 5 6 7 8 9 10

Files Plots Packages Help Viewer

Zoom | Export | Publish



RStudio

File Edit Code View Plots Session Build Debug Profile Tools Help

```
* indicates p < .05. ** indicates p < .01.


> 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")

<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>-0.4
34</td><td>-0.708</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)


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.

>
```

Environment History Connections

Import Dataset | List | Global Environment

Data

banner	List of 7
dat	10 obs. of 3 variables
h	626 obs. of 70 variables
loafercreek	Formal class SoilProfileCollection
m	int [1:2, 1:5] 1 2 3 4 5 6 7 8 9 10
mtcars	32 obs. of 11 variables
product_test	150 obs. of 18 variables
to_correlate	32 obs. of 4 variables
w	150 obs. of 36 variables
x	List of 5
xlist	List of 3

Values

clay	num [1:453] 16.7 17 19.6 23.6 21.5 ...
codeframe_likes	Named num [1:10] 1 2 3 4 5 6 7 8 98 99
i	7
n	chr [1:6] "A" "Bt1" "Bt2" "Bt3" "Cr" "R"
overall_liking_scale	Named num [1:7] 1 2 3 4 5 6 7
p	chr [1:6] "^A|BA$" "Bt1|Bw" "Bt$|Bt2" "Bt3|CBt$|BCt" "Cr"...
tab_cpct_sig	function (value)
tab_means_sig	function (value)
vars	chr [1:4] "hzdepm" "clay" "phfield" "total_frags_pct"
y	int [1:3] 10 11 12
z	chr [1:3] "Sarah" "Tracy" "Jon"

Files Plots Packages Help Viewer

RStudio

File Edit Code View Plots Session Build Debug Profile Tools Help

Go to file/function | Addins | Project: (None)

Console Terminal Jobs

```
> ggplot(to_correlate, aes(x=cyl, y=qsec)) + geom_jitter
(width=0.1) + stat_smooth(method="lm", se=FALSE)
Error in ggplot(to_correlate, aes(x = cyl, y = qsec)) :
  could not find function "ggplot"
> library(ggplot2)

Attaching package: 'ggplot2'

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

    vars

> library("ggplot2")
>  install.packages("ggplot")
WARNING: Rtools is required to build R packages but is no
t currently installed. Please download and install the ap
propriate version of Rtools before proceeding:

https://cran.rstudio.com/bin/windows/Rtools/
Installing package into 'C:/Users/varun/Documents/R/win-l
ibrary/3.6'
(as 'lib' is unspecified)
Warning in install.packages :
  package 'ggplot' is not available (for R version 3.6.3)
>
> 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'
>
```

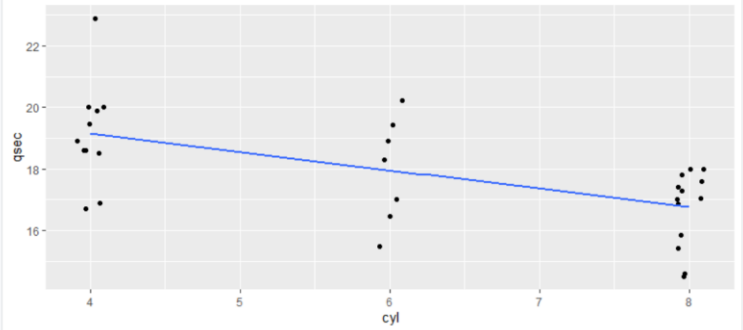
Environment History Connections

Import Dataset | List | Global Environment

Data	
banner	List of 7
dat	10 obs. of 3 variables
h	626 obs. of 70 variables
loafercreek	Formal class SoilProfileCollection
m	int [1:2, 1:5] 1 2 3 4 5 6 7 8 9 10

Files Plots Packages Help Viewer

Zoom | Export | Publish



RStudio

File Edit Code View Plots Session Build Debug Profile Tools Help

Go to file/function | Addins | Project: (None)

Console Terminal Jobs

```
package 'tidyselect' successfully unpacked and MD5 sums checked
package 'BH' successfully unpacked and MD5 sums checked
package 'plogr' successfully unpacked and MD5 sums checked
package 'dplyr' successfully unpacked and MD5 sums checked

The downloaded binary packages are in
        C:\Users\varun\AppData\Local\Temp\RtmpUDOB8m\downloaded_packages
>
> library(magrittr)
>
> library(dplyr)

Attaching package: 'dplyr'

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
>
```

Environment History Connections

Import Dataset | List | Global Environment

Data	
banner	List of 7
dat	10 obs. of 3 variables
h	626 obs. of 70 variables
loafercreek	Formal class SoilProfileCollection
m	int [1:2, 1:5] 1 2 3 4 5 6 7 8 9 10
mtcars	32 obs. of 11 variables
product_test	150 obs. of 18 variables
to_correlate	32 obs. of 4 variables
w	150 obs. of 36 variables
x	List of 5
xlist	List of 3
Values	
clay	num [1:453] 16.7 17 19.6 23.6 21.5 ...
codeframe_likes	Named num [1:10] 1 2 3 4 5 6 7 8 98 99
i	7
n	chr [1:6] "A" "Bt1" "Bt2" "Bt3" "Cr" "R"
overall_liking…	Named num [1:7] 1 2 3 4 5 6 7
p	chr [1:6] "^A|BA$" "Bt1\|Bw" "Bt$|Bt2" "Bt…
tab_cpct_sig	function (value)
tab_means_sig	function (value)
vars	chr [1:4] "hzdepm" "clay" "phfield" "tota…
y	int [1:3] 10 11 12
z	chr [1:3] "Sarah" "Tracy" "Jon"

Files Plots Packages Help Viewer

Conclusion: Thus, we have studied the basics of correlation, simple and multiple regression in R.

Experiment No: 10

Aim: Clustering in R - k-means

Theory:

K Means Clustering is an unsupervised learning algorithm that tries to cluster data based on their similarity. Unsupervised learning means that there is no outcome to be predicted, and the algorithm just tries to find patterns in the data. In k means clustering, we have to specify the number of clusters we want the data to be grouped into. The algorithm randomly assigns each observation to a cluster and finds the centroid of each cluster. Then, the algorithm iterates through two steps:

- Reassign data points to the cluster whose centroid is closest.
- Calculate the new centroid of each cluster.

These two steps are repeated until the within-cluster variation cannot be reduced any further. The within-cluster variation is calculated as the sum of the euclidean distance between the data points and their respective cluster centroids.

Clustering Types:

Clustering can be broadly divided into two subgroups:

- Hard clustering: in hard clustering, each data object or point either belongs to a cluster completely or not. For example in the Uber dataset, each location belongs to either one borough or the other.
- Soft clustering: in soft clustering, a data point can belong to more than one cluster with some probability or likelihood value. For example, you could identify some locations as the border points belonging to two or more boroughs.

The clustering algorithm that we are going to use is the K-means algorithm, which we can find in the package stats. The K-means algorithm accepts two parameters as input:

- The data;
- A **K** value, which is the number of groups that we want to create.

Conceptually, the K-means behaves as follows:

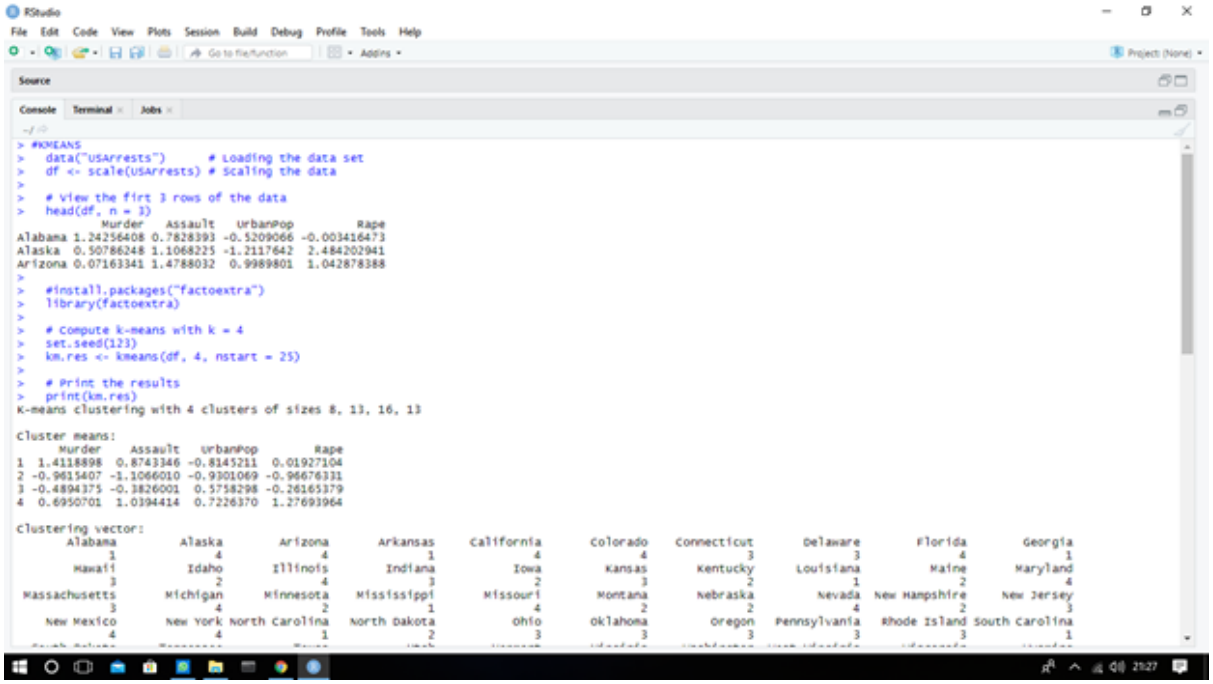
1. It chooses K centroids randomly;
2. Matches each point in the data (in our case, each mammal) with the closest centroid in an n-dimensional space where n is the number of features used in the clustering (in our example, 5 features — water, protein, fat, lactose, ash). After this step, each point belongs to a group.
3. Now, it recalculates the centroids as being the mean point (vector) of all other points in the group.
4. It keeps repeating the steps 2 and 3 until either when the groups are stabilized, that is, when no points are reallocated to another centroid or when it reaches the maximum number of iterations (the stats library uses 10 as default).

Output :

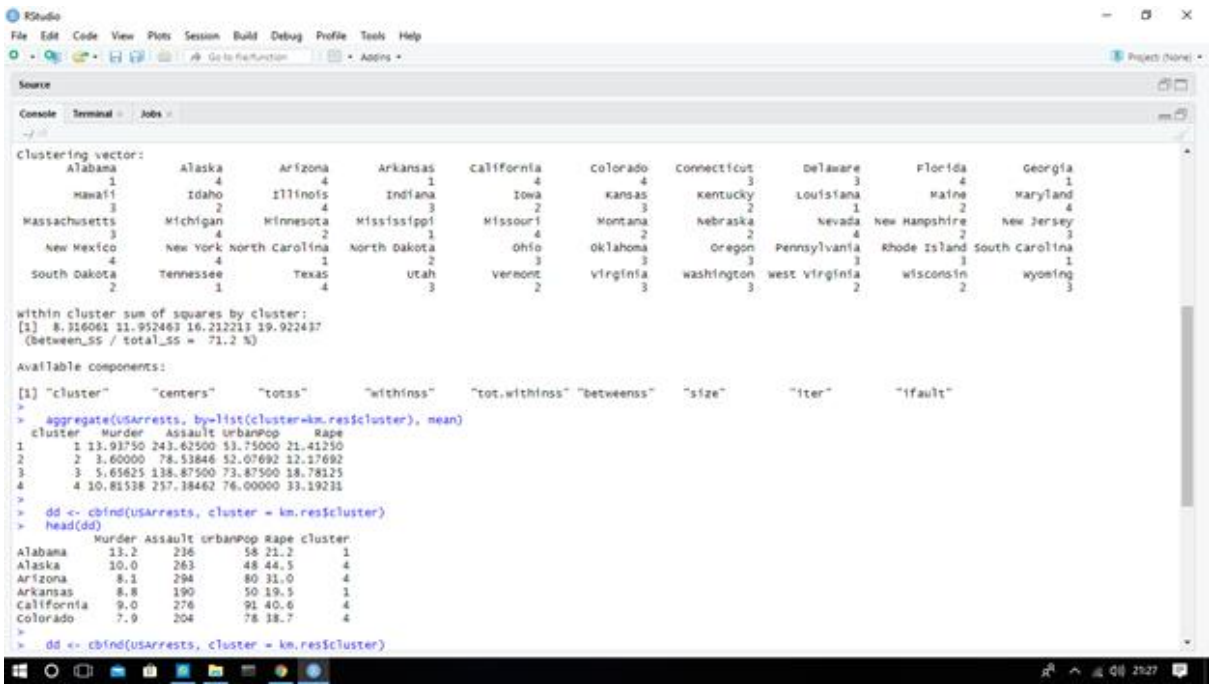
Code :

```
> #Author DataFlair
> library(tidyverse)
> library(cluster)
> library(factoextra)
> library(gridExtra)
> data('USArrests')
> d_frame <- USArrests
> d_frame <- na.omit(d_frame) #Removing the missing values
> d_frame <- scale(d_frame)
> head(d_frame)
```

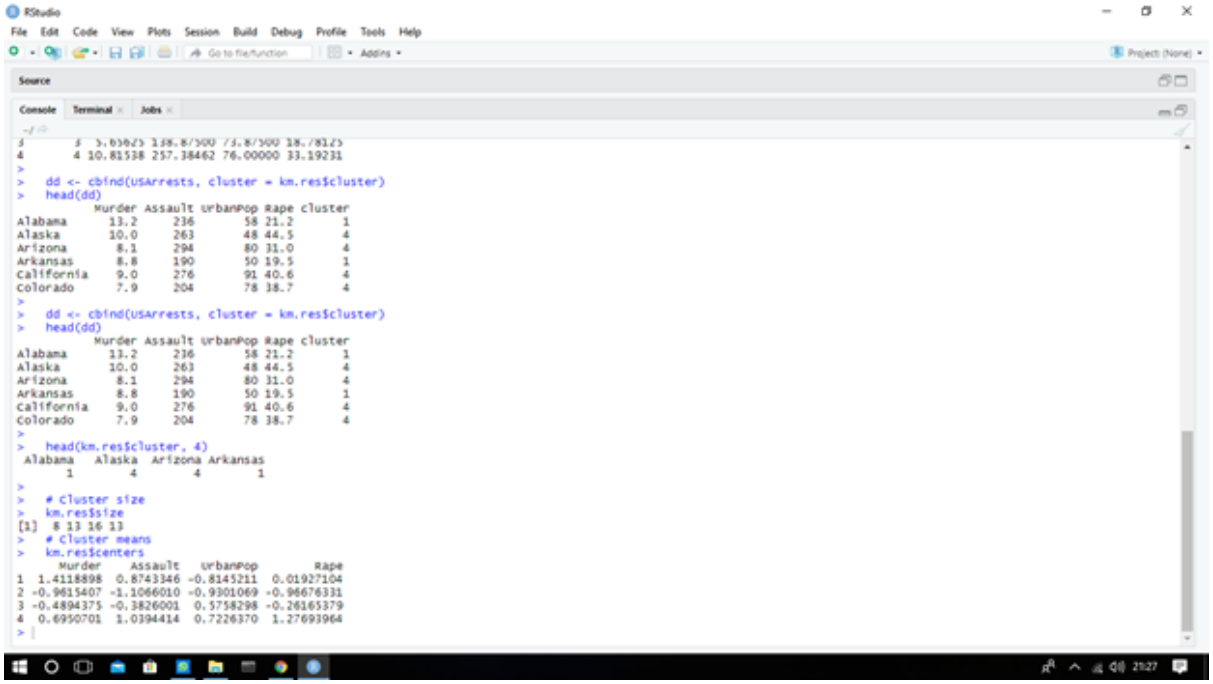
Screenshots :

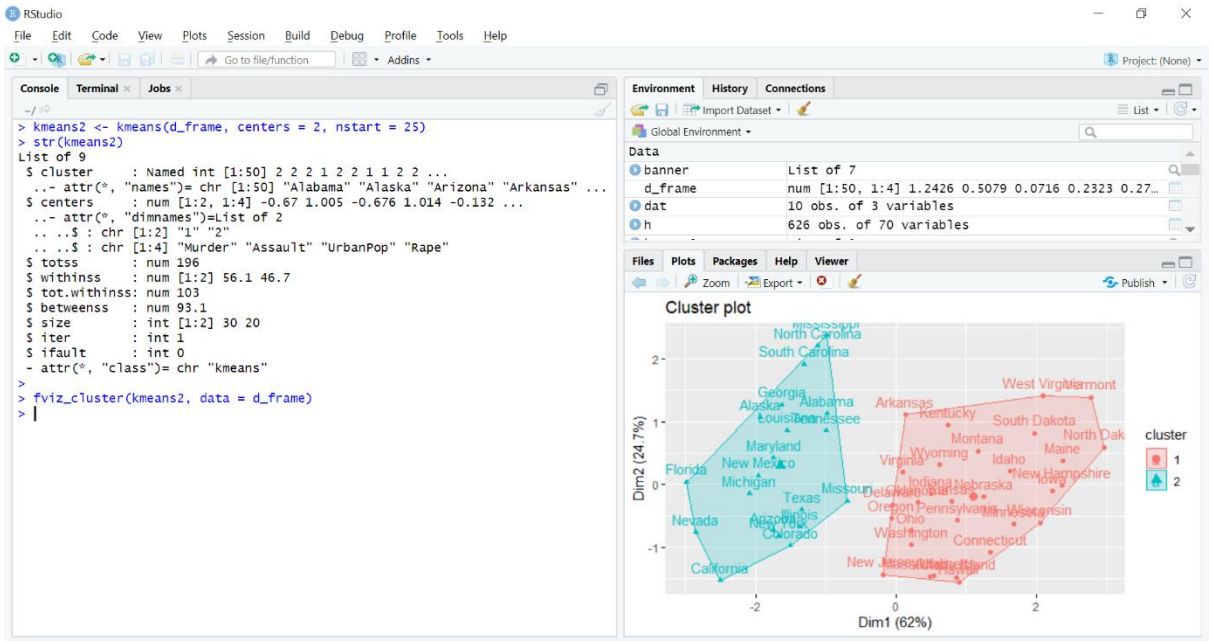


```
> #KMEANS
>   data("USArrests")      # Loading the data set
>   df <- scale(USArrests) # Scaling the data
>
>   # View the firt 3 rows of the data
>   head(df, n = 3)
            Murder   Assault   UrbanPop         Rape
Alabama 1.24256408 0.7828393 -0.5209066 -0.003416473
Alaska  0.50786248 1.1068225 -1.2117642  2.484202941
Arizona 0.07163341 1.4788032  0.9989801  1.042878388
>
>   #install.packages("factoextra")
>   library(factoextra)
>
>   # Compute k-means with k = 4
>   set.seed(123)
>   km.res <- kmeans(df, 4, nstart = 25)
>
>   # Print the results
>   print(km.res)
K-means clustering with 4 clusters of sizes 8, 13, 16, 13

Cluster means:
      Murder    Assault   UrbanPop        Rape
1  1.4118898  0.8743346 -0.8145211  0.01927104
2 -0.9615407 -1.1066010 -0.9301069 -0.96676331
3 -0.4894375 -0.3826001  0.5758298 -0.26165379
4  0.6950701  1.0394414  0.7226370  1.27693964

Clustering vector:
       Alabama         Alaska        Arizona       Arkansas     California       Colorado    Connecticut       Delaware        Florida        Georgia
             1              4              4              1              4              4              3              3              4              1
        Hawaii          Idaho       Illinois        Indiana           Iowa         Kansas       Kentucky      Louisiana          Maine       Maryland
             3              2              4              3              2              3              2              1              2              4
 Massachusetts       Michigan      Minnesota    Mississippi       Missouri        Montana       Nebraska         Nevada  New Hampshire     New Jersey
             3              4              2              1              4              2              2              4              2              3
    New Mexico       New York North Carolina   North Dakota           Ohio       Oklahoma         Oregon   Pennsylvania   Rhode Island South Carolina
             4              4              1              2              3              3              3              3              3              1
```



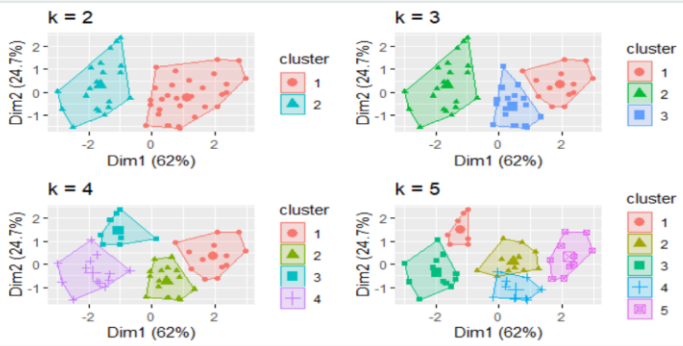
```
Clustering vector:
       Alabama         Alaska        Arizona       Arkansas     California       Colorado    Connecticut       Delaware        Florida        Georgia
             1              4              4              1              4              4              3              3              4              1
        Hawaii          Idaho       Illinois        Indiana           Iowa         Kansas       Kentucky      Louisiana          Maine       Maryland
             3              2              4              3              2              3              2              1              2              4
 Massachusetts       Michigan      Minnesota    Mississippi       Missouri        Montana       Nebraska         Nevada  New Hampshire     New Jersey
             3              4              2              1              4              2              2              4              2              3
    New Mexico       New York North Carolina   North Dakota           Ohio       Oklahoma         Oregon   Pennsylvania   Rhode Island South Carolina
             4              4              1              2              3              3              3              3              3              1
  South Dakota      Tennessee          Texas           Utah        Vermont       Virginia     Washington  West Virginia      Wisconsin        Wyoming
             2              1              4              3              2              3              3              2              2              3

Within cluster sum of squares by cluster:
[1]  8.316061 11.952463 16.212213 19.922437
 (between_SS / total_SS =  71.2 %)

Available components:

[1] "cluster"      "centers"      "totss"        "withinss"     "tot.withinss" "betweenss"    "size"         "iter"         "ifault"
>
>   aggregate(USArrests, by=list(cluster=km.res$cluster), mean)
  cluster   Murder   Assault UrbanPop     Rape
1       1 13.93750 243.62500 53.75000 21.41250
2       2  3.60000  78.53846 52.07692 12.17692
3       3  5.65625 138.87500 73.87500 18.78125
4       4 10.81538 257.38462 76.00000 33.19231
>
>   dd <- cbind(USArrests, cluster = km.res$cluster)
>   head(dd)
           Murder Assault UrbanPop Rape cluster
Alabama      13.2     236       58 21.2       1
Alaska       10.0     263       48 44.5       4
Arizona       8.1     294       80 31.0       4
Arkansas      8.8     190       50 19.5       1
California    9.0     276       91 40.6       4
Colorado      7.9     204       78 38.7       4
>
>   dd <- cbind(USArrests, cluster = km.res$cluster)
```

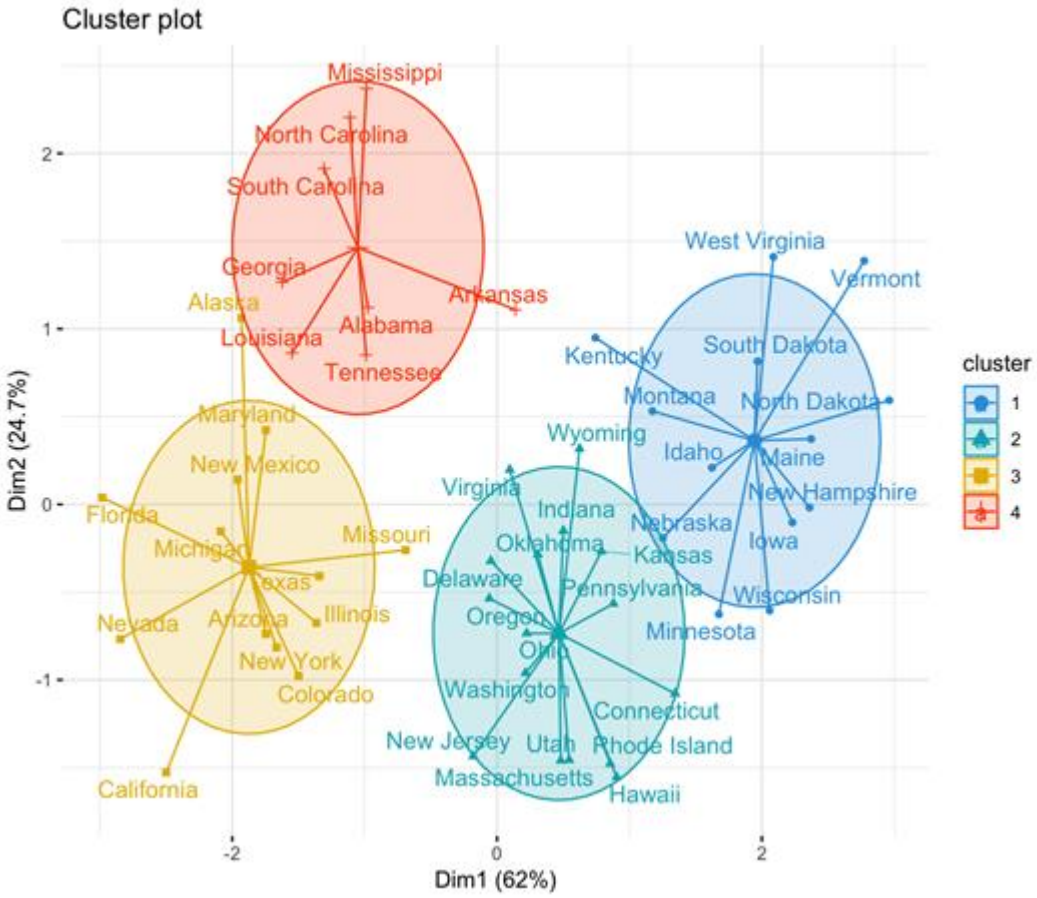

RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
Source
Console Terminal Jobs
4 4 10.81538 257.38462 76.00000 33.19231
> dd <- cbind(USArrests, cluster = km.res$cluster)
> head(dd)
Murder Assault UrbanPop Rape cluster
Alabama 13.2 236 58 21.2 1
Alaska 10.0 263 48 44.5 4
Arizona 8.1 294 80 31.0 4
Arkansas 8.8 190 50 19.5 1
California 9.0 276 91 40.6 4
Colorado 7.9 204 78 38.7 4
> dd <- cbind(USArrests, cluster = km.res$cluster)
> head(dd)
Murder Assault UrbanPop Rape cluster
Alabama 13.2 236 58 21.2 1
Alaska 10.0 263 48 44.5 4
Arizona 8.1 294 80 31.0 4
Arkansas 8.8 190 50 19.5 1
California 9.0 276 91 40.6 4
Colorado 7.9 204 78 38.7 4
> head(km.res$cluster, 4)
Alabama Alaska Arizona Arkansas
1 4 4 1
> # Cluster size
> km.res$size
[1] 8 13 16 13
> # Cluster means
> km.res$centers
Murder Assault UrbanPop Rape
1 1.4118898 0.8743346 -0.8145211 0.01927104
2 -0.9615407 -1.1066010 -0.9301069 -0.96676331
3 -0.4894375 -0.3826001 0.5758298 -0.26165379
4 0.6950701 1.0394414 0.7226370 1.27693964


RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
Console Terminal Jobs
> 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)
Environment History Connections
Import Dataset
Global Environment
Data
banner List of 7
d_frame num [1:50, 1:4] 1.2426 0.5079 0.0716 0.2323 0.27...
dat 10 obs. of 3 variables
h 626 obs. of 70 variables
Files Plots Packages Help Viewer
Zoom Export Publish
Cluster plot
North Carolina
South Carolina
Georgia
Alaska
Alabama
Louisiana
Tennessee
Arkansas
Kentucky
West Virginia
Vermont
South Dakota
North Dakota
Montana
Maine
Wyoming
Idaho
New Hampshire
Virginia
Indiana
Nebraska
Iowa
Maryland
New Mexico
Florida
Michigan
Missouri
Texas
Nevada
Illinois
Arizona
Colorado
Oregon
Pennsylvania
Ohio
Washington
Connecticut
California
Hawaii
cluster
1
2
Dim2 (24.7%)
Dim1 (62%)
2
1
0
-1
-2
0
2

```
> 
> library(tidyverse)
-- Attaching packages --------------------------------------- tidyverse 1.3.0 --
v ggplot2 3.3.0     v purrr   0.3.3
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()
x dplyr::slice()  masks aqp::slice()
> 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)  #Removing the missing values
> 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
>
```

```
> kmeans3 <- kmeans(d_frame, centers = 3, nstart = 25)
> kmeans4 <- kmeans(d_frame, centers = 4, nstart = 25)
> kmeans5 <- kmeans(d_frame, centers = 5, nstart = 25)
>
> #Comparing the Plots
> plot1 <- fviz_cluster(kmeans2, geom = "point", data = d_fram
e) + ggtitle("k = 2")
> plot2 <- fviz_cluster(kmeans3, geom = "point", data = d_fram
e) + ggtitle("k = 3")
> plot3 <- fviz_cluster(kmeans4, geom = "point", data = d_fram
e) + ggtitle("k = 4")
> plot4 <- fviz_cluster(kmeans5, geom = "point", data = d_fram
e) + ggtitle("k = 5")
> grid.arrange(plot1, plot2, plot3, plot4, nrow = 2)
>
```





Conclusion : This, we have studied clustering in R - k-means

Experiment No. 11

Aim : Classification on large and noisy dataset with R - logistic regression and naive bayes.

Theory :

In statistics, the logistic model (or logit model) is used to model the probability of a certain class or event existing such as pass/fail, win/lose, alive/dead or healthy/sick. This can be extended to model several classes of events such as determining whether an image contains a cat, dog, lion, etc. Each object being detected in the image would be assigned a probability between 0 and 1 and the sum adding to one.

Logistic regression is a statistical model that in its basic form uses a logistic function to model a binary dependent variable, although many more complex extensions exist. In regression analysis, logistic regression (or logit regression) is estimating the parameters of a logistic model (a form of binary regression).

R makes it very easy to fit a logistic regression model. The function to be called is glm() and the fitting process is not so different from the one used in linear regression. In this post I am going to fit a binary logistic regression model and explain each step.

Naïve Bayes classification is a kind of simple probabilistic classification method based on Bayes' theorem with the assumption of independence between features. The model is trained on a training dataset to make predictions by predict() function. This article introduces two functions naiveBayes() and train() for the performance of Naïve Bayes classification.

Output:

Code:

```
training_data <- read.csv("aps_failure_training_set.csv", skip = 20,na.strings = "na")
test_data <- read.csv("aps_failure_test_set.csv",skip = 20, na.strings = "na")

dim(training_data)
dim(test_data)

library(dplyr)
library(caret)
library(caretEnsemble)
library(mice)
library(doParallel)
library(car)

install.packages("car")

glimpse(training_data)
```

```
glimpse(test_data)

summary(training_data$class)
summary(test_data$class)

options(digits = 2)
prop.table(table(training_data$class))
prop.table(table(test_data$class))

options(scipen = 999)
summary_df <- do.call(cbind, lapply(training_data[,
                                    2:ncol(training_data)], summary))
summary_df_t <- as.data.frame(round(t(summary_df),0))
names(summary_df_t)[7] <- paste("Missing_values")
summary_df_t_2 <- summary_df_t %>%
  mutate(obs = nrow(training_data),
         Missing_prop = Missing_values / obs)
print(summary_df_t_2)

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)

#replicate our sets
training_data_bind <- training_data
test_data_bind <- test_data
#create a new column "set" to label the observations
training_data_bind$set <- "TRAIN"
test_data_bind$set <- "TEST"
#merge them into 1 single set
full_dataset <- rbind(training_data_bind, test_data_bind)
dim(full_dataset)

set.seed(123)
imputed_full <- mice(full_dataset,
                     m=1,
                     maxit = 5,
                     method = "pmm",
                     seed = 500)

full_imputed <- complete(imputed_full, 1)
dim(full_imputed)

#subset the full_imputed_filtered dataset
```

```
training_data_imp <- subset(training_data)
test_data_imp <- subset(test_data)
#drop the "set" column, we don't need it anymore
training_data_imp$set <- NULL
test_data_imp$set <- NULL
#check dimensions
dim(training_data_imp)
dim(test_data_imp)

install.packages("DMwR")
library(DMwR)
knnImputation(training_data, k = 10, scale = T, meth = "weighAvg",distData = NULL)

cooksd <- cooks.distance(glm(class ~ .,
                        family = "binomial",
                        data = training_data_imp))

plot(cooksd,
    pch="*",
    cex=2,
    main="Influential Obs by Cooks distance")
abline(h = 4*mean(cooksd, na.rm=T), col="red")

outliers <- rownames(training_data_imp[cooksd > 4*mean(cooksd, na.rm=T), ])
print(outliers)

sum((correlation > 0.5 | correlation < -0.5) & correlation < 1) / (162*162)
sum((correlation > 0.7 | correlation < -0.7) & correlation < 1) / (162*162)
sum((correlation > 0.9 | correlation < -0.9) & correlation < 1) / (162*162)

registerDoParallel(3)
getDoParWorkers()
set.seed(123)
my_ctrl <- trainControl(method = "cv",
                    number = 5,
                    classProbs = TRUE,
                    savePredictions = "final",
                    na.action = na.pass,
                    index = createResample(training_data_imp$class, 3),
                    sampling = "up",
                    allowParallel = TRUE)

model_list <- caretList(class ~ .,
                    data = training_data_imp,
                    methodList = c("glm", "nb"),
                    metric = "Kappa",
                    tuneList = NULL,
                    na.action = na.pass,
                    continue_on_fail = FALSE,
                    preProcess = c("center", "scale"),
```

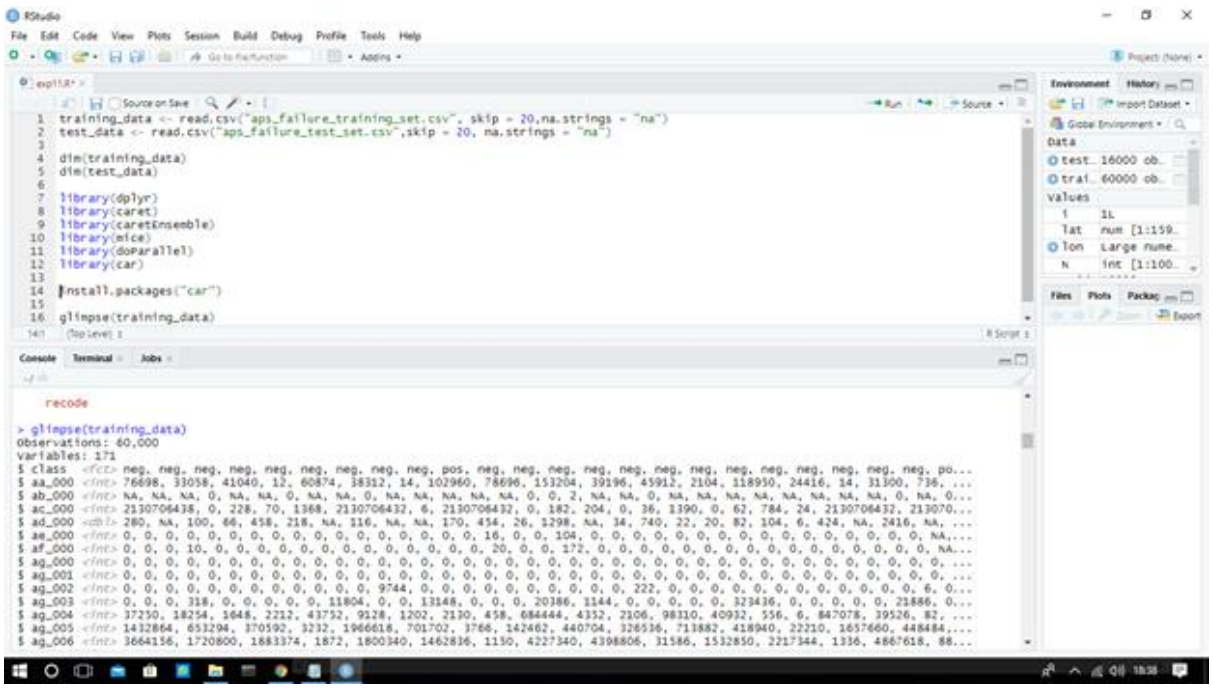
trControl = my_ctrl)

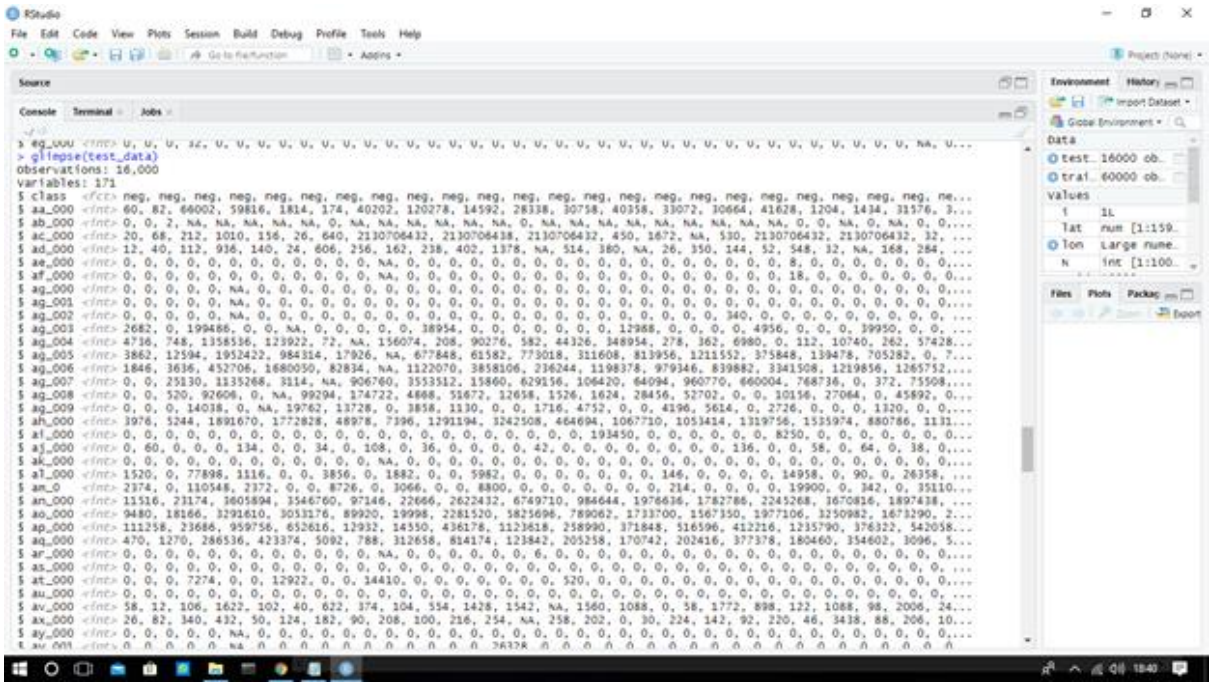
model_list$glm
model_list$nb

Screenshots:

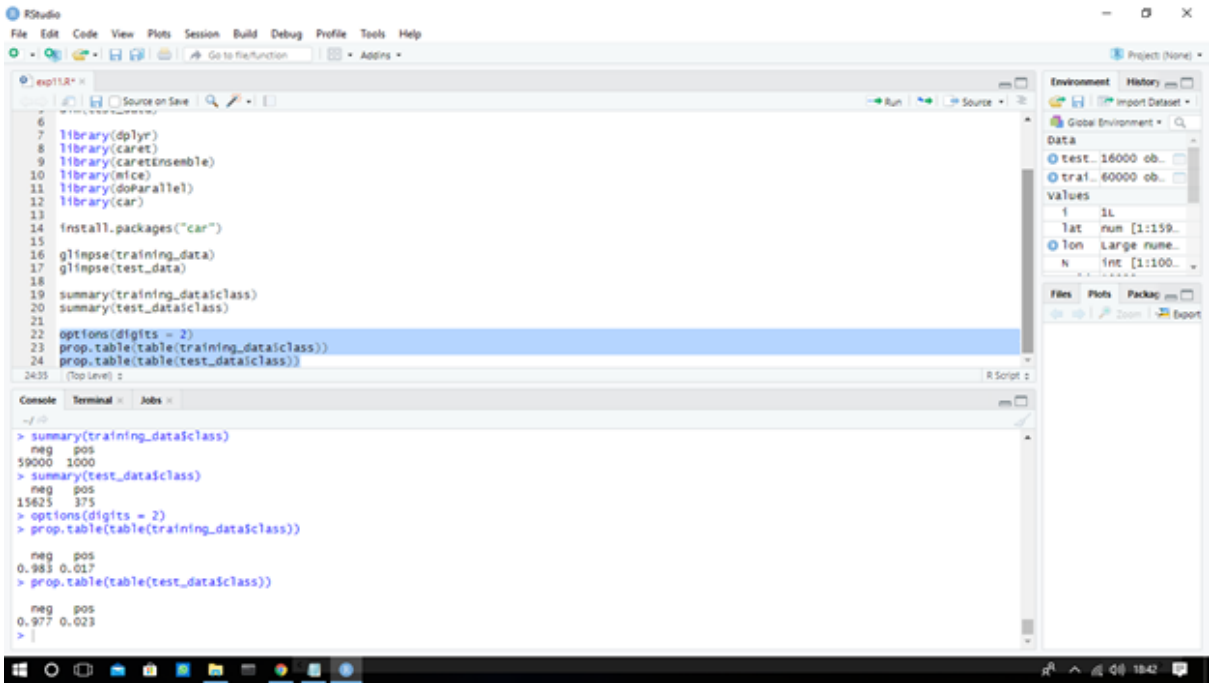
STEPS:

1. Download the dataset from https://archive.ics.uci.edu/ml/machine-learning-databases/00421/
2. Load the data set and View Data (str or dplyr's glimpse)

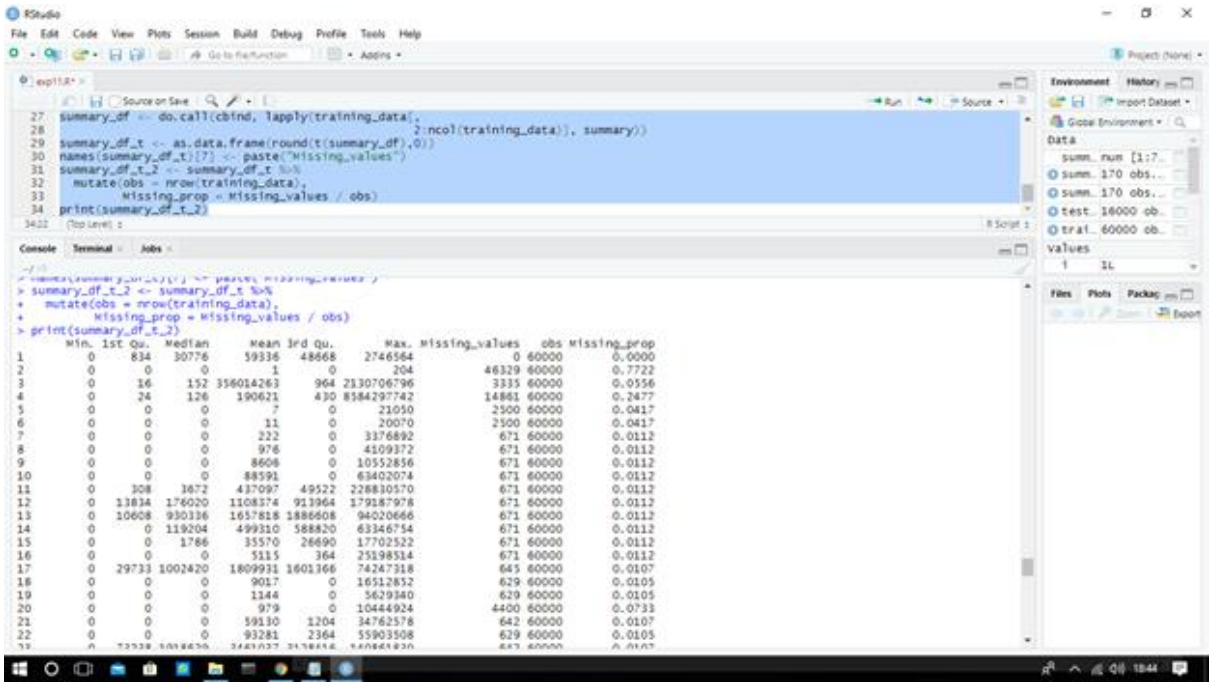


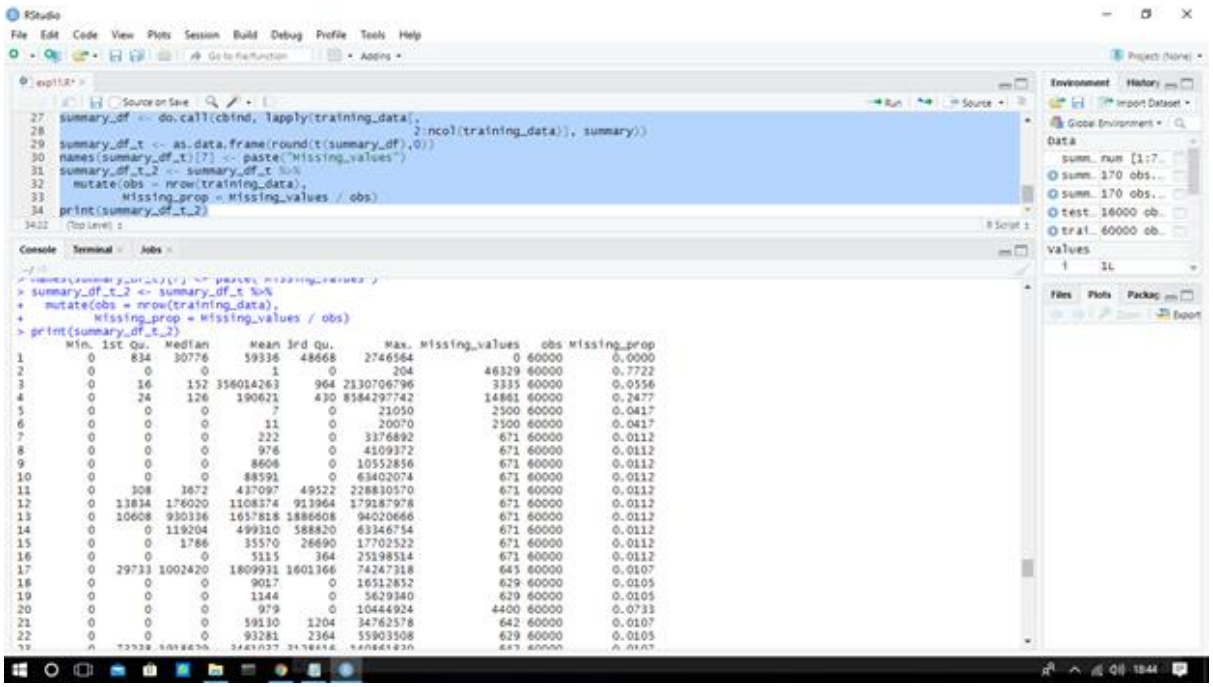


3. Check How many neg and pos do we have in each set and check proportions

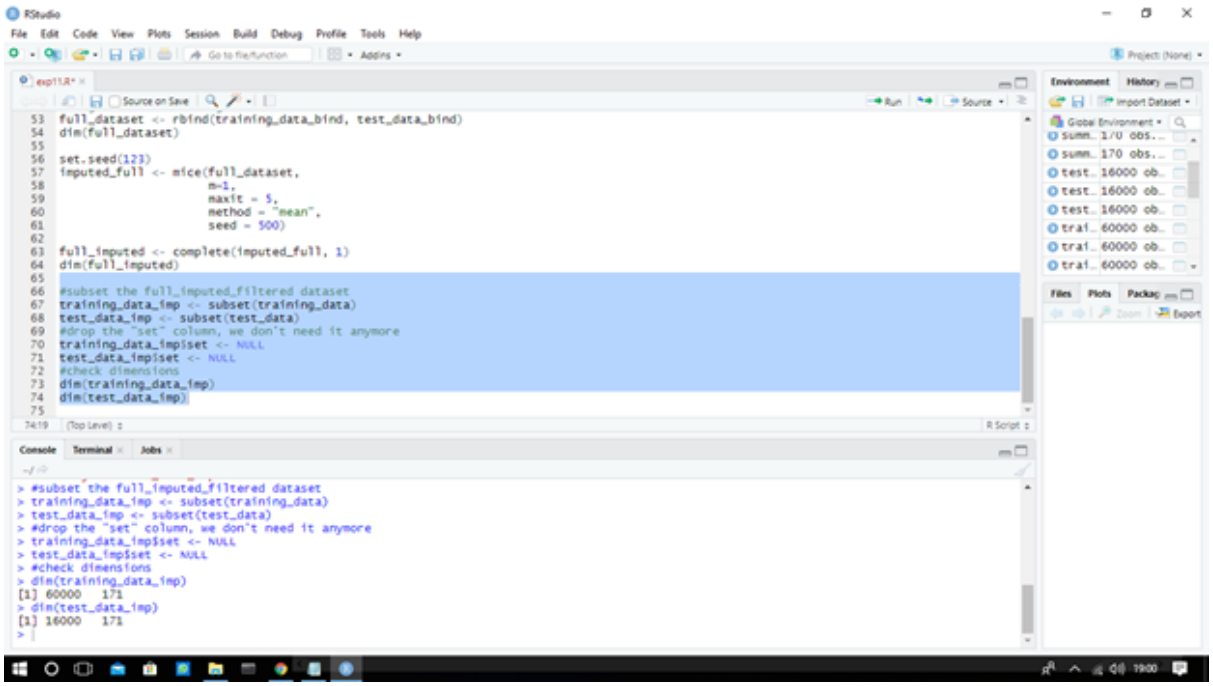


4.Rest of the features. Summary statistics. Understand your data

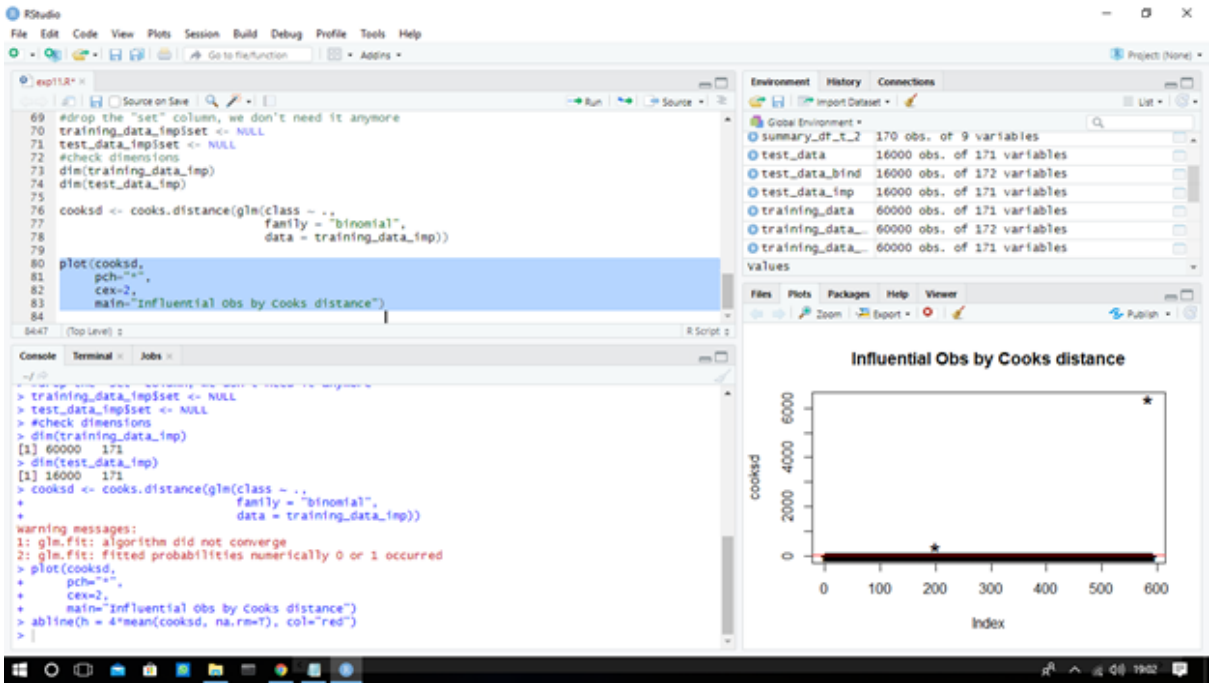


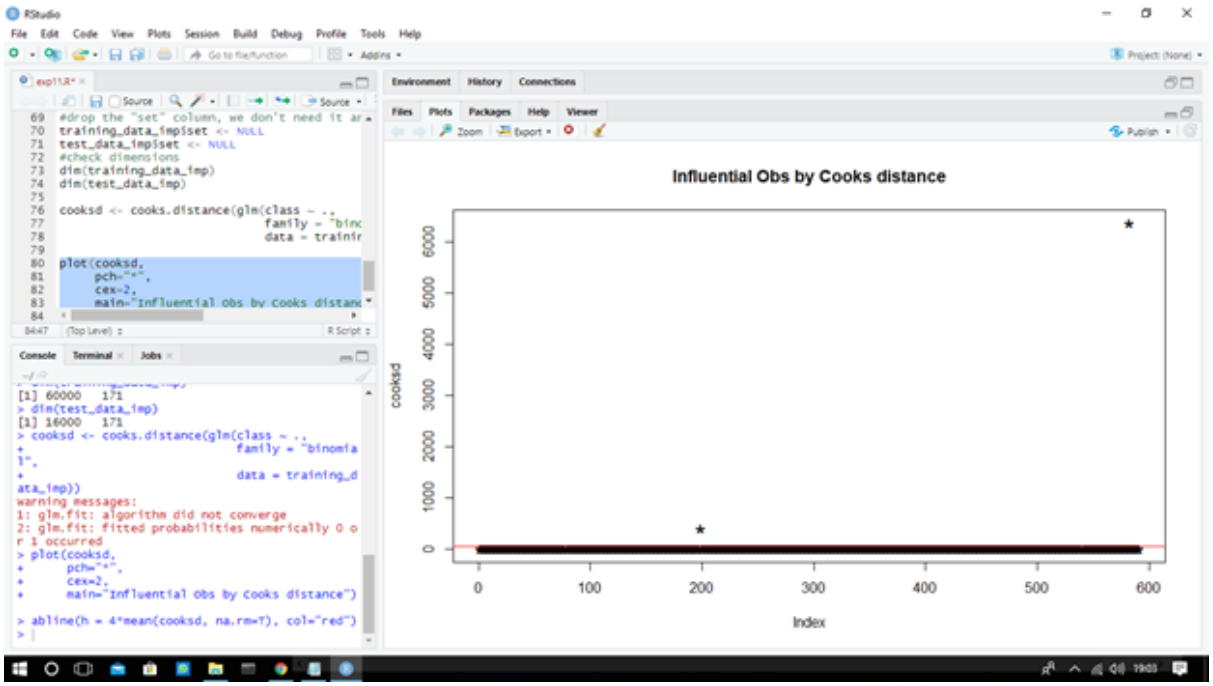


5. Our features present more than 8% missing values on average. Because it's a lot of information we don't want to lose, our approach is going to be to impute them. Impute the full dataset using mean imputation

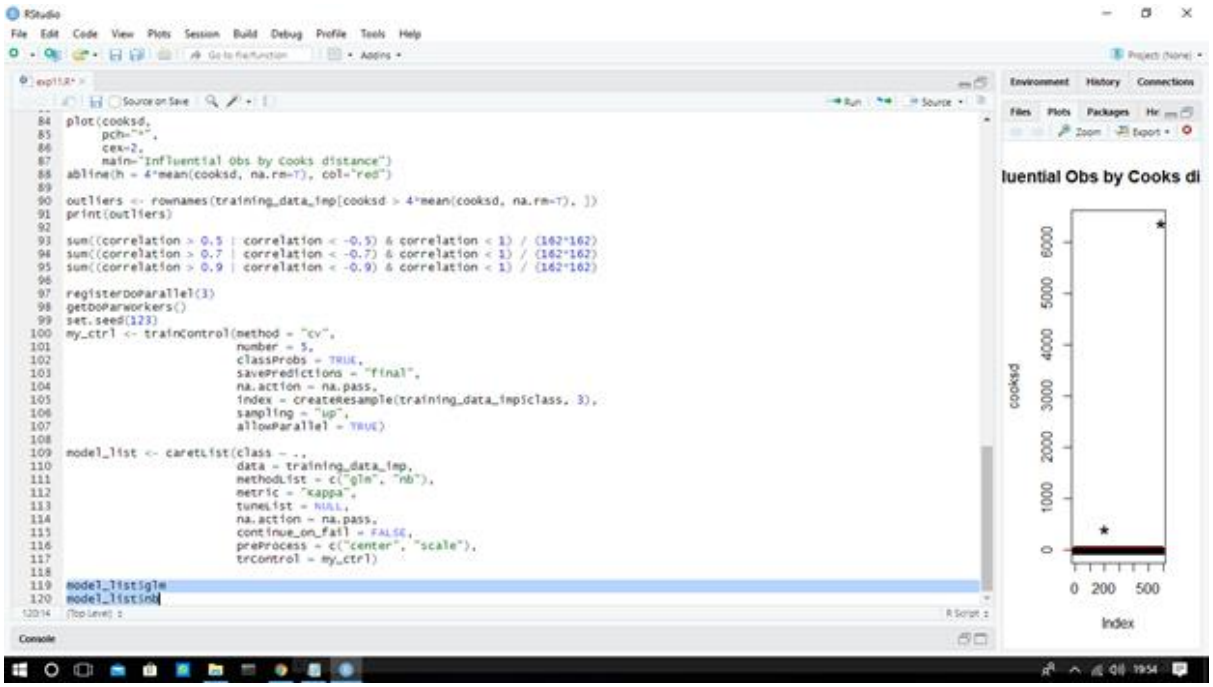


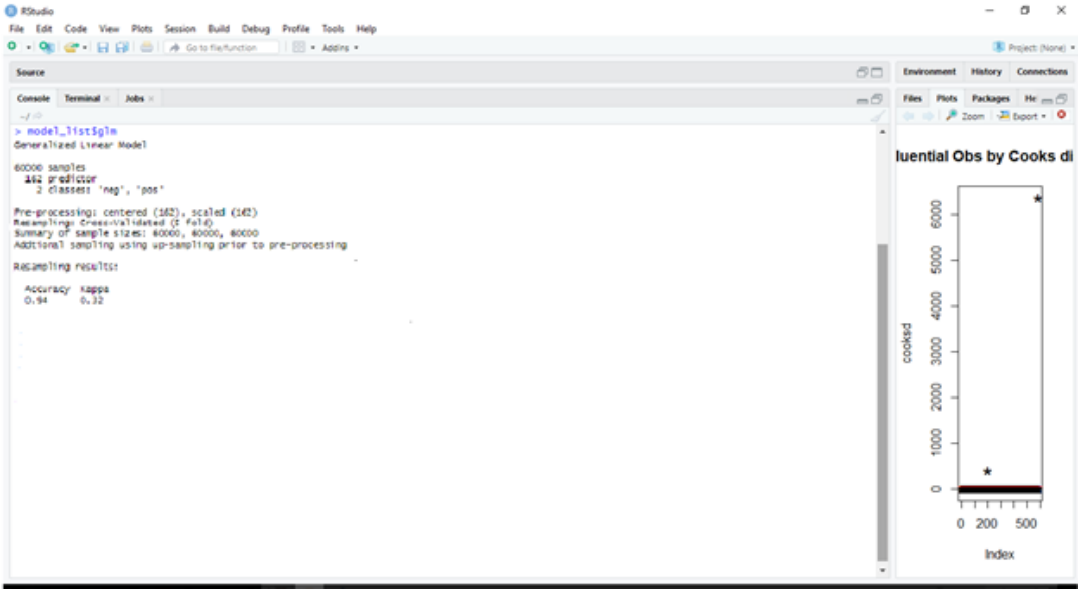
6. Check for outliers and other inconsistent data points. Box-plots. Cooks' distance. DBSCAN

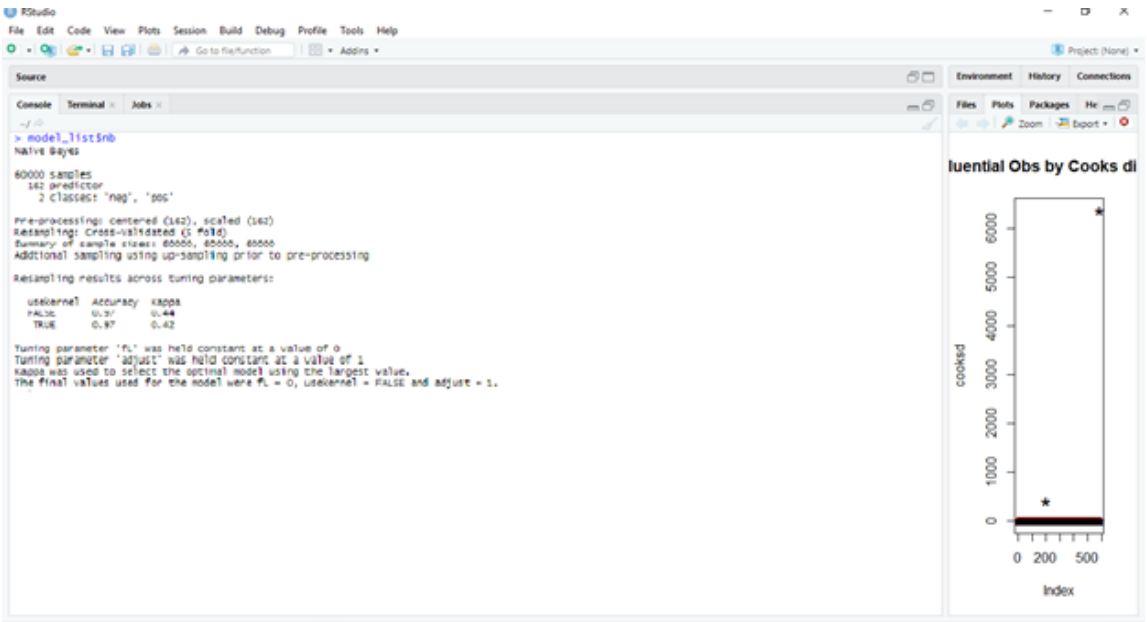




7. We'll use caretEnsemble's caretList()to train both at the same time and with the same resampling .







Conclusion: Thus, we have studied classification on large and noisy dataset with R - logistic regression and naive bayes.