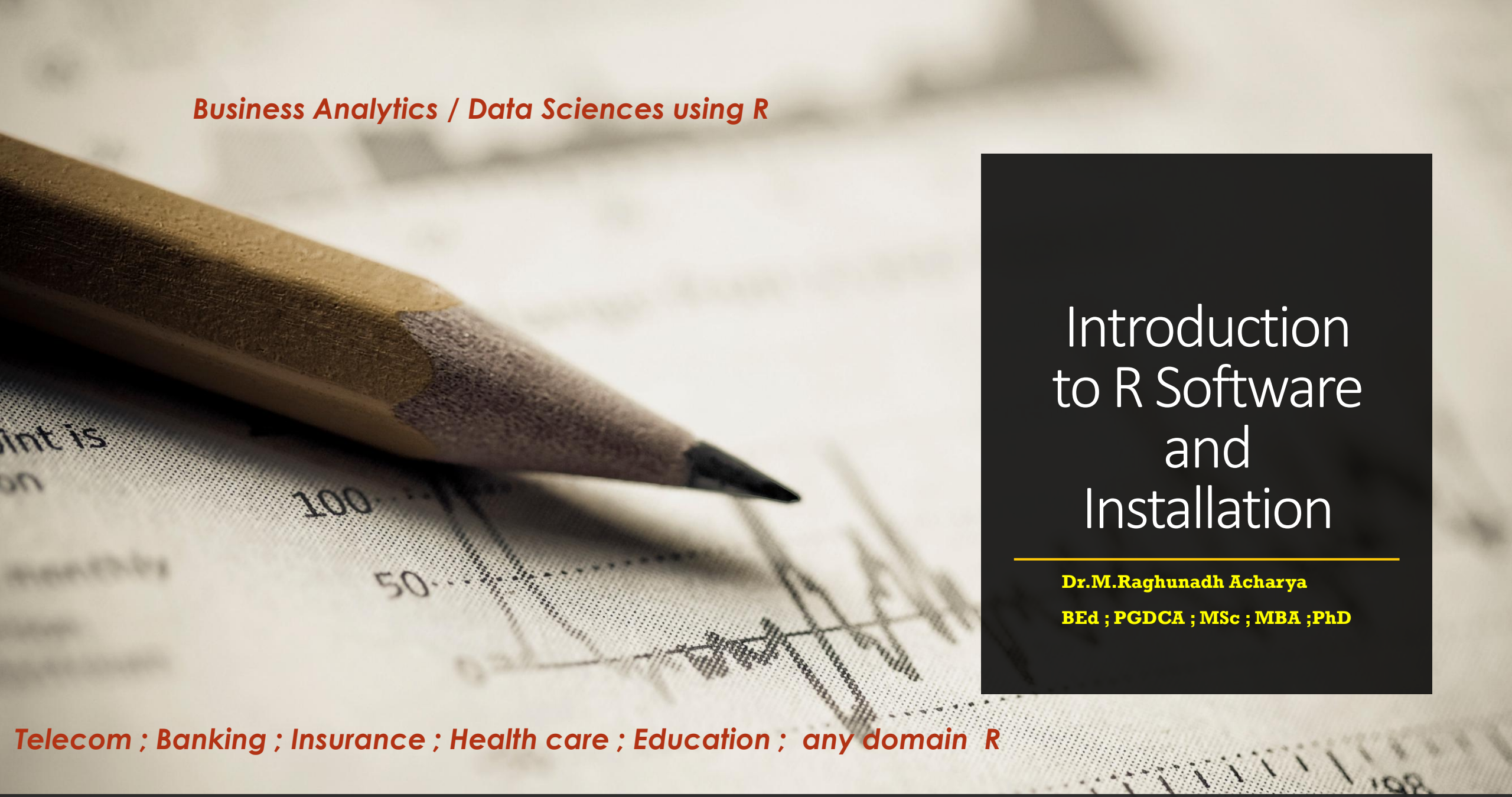
Business Analytics / Data Sciences using R



Introduction to R Software and Installation

Dr.M.Raghunadh Acharya

BEd ; PGDCA ; MSc ; MBA ;PhD

Telecom ; Banking ; Insurance ; Health care ; Education ; any domain R

Introduction to R Software and Installation

- R is a magic software very user friendly simple yet very powerful.

Dr. M.Raghunadh Acharya

- ***SPSS, SAS and many other paid softwares are like muggles. They are limited in their ability to change their environment. They have to rely on algorithms that have been developed for them. The way they approach a problem is constrained by how SAS/SPSS employed programmers thought to approach them. And they have to pay money to use these constraining algorithms.***

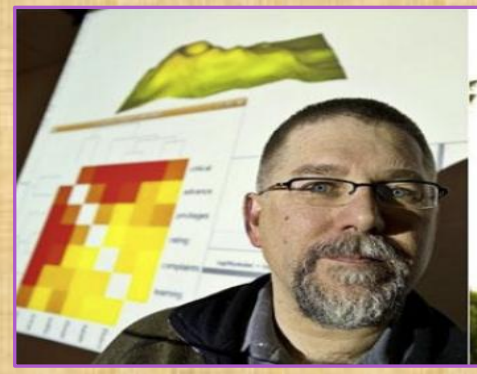
Matthew Keller

Introduction to R

- R is a multifaceted software and language consists of packages
- It has many built in functions through which one can execute the statistical functions
- One can develop an application using R software
- It is similar to JAVA and C++ and object oriented.
- R is FOSS (Free Open source) or GNU(General Public License) GPL.
- The initiators of R is a product of Bell labs it is S and then became R.

Introduction to R
History …

- R is extension of S language
- 1991: Created in New Zealand by Ross Ihaka and Robert Gentleman.
- 1993: First announcement of R.
- 1995: Martin Machler convinces Ross and Robert to use the GNU General Public License to make R free software.



Robert Gentleman

Ross Ihaka

Introduction to R
History …

- 1996: A public mailing list is created (R - help and R - devel)
- 1997: The R Core Group is formed (containing some people associated with S-PLUS). The core group controls the source code for R.
- 2000: R version 1.0.0 is released.
- 2020: R version R version 4.0.2 (2020 – 06 - 22) -- “Taking Off Again" is released
- Runs on almost any standard computing platform/OS (even on the Android)
- Frequent releases (bugfix releases); active development

Introduction to R
Advantages of R …

- R is free
- Open Source
- The array of packages
- Interface with other languages and scripting capabilities
- Quality plotting and graphics
- R role in academics
- Very active development
- Machine learning operations

Introduction to R
Advantages of R …

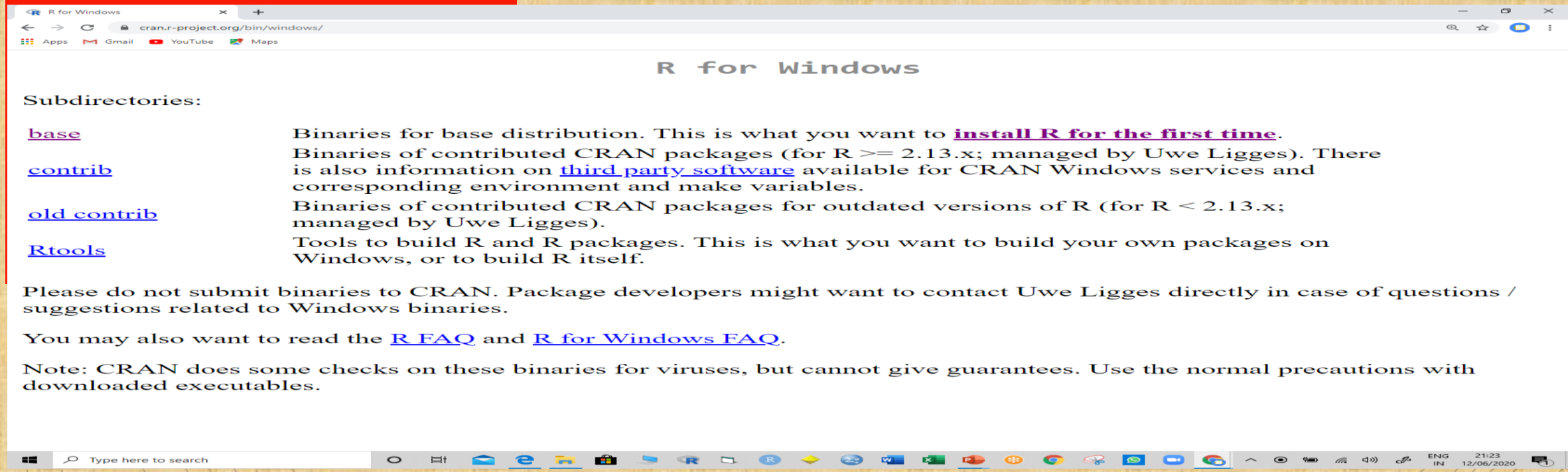
- Apart from AI , Graphs , advanced Graphs many other packages are available.
- There are about **16664** packages on CRAN that have been developed by users and programmers around the world
- People often make packages available on their personal websites; there is no absolute way to keep track of how many packages are available in this fashion

Installation of R and R Studio

Use the Link https://cran.r-project.org/bin/windows/

OR one can use The basic R system: R console (GUI) & packages

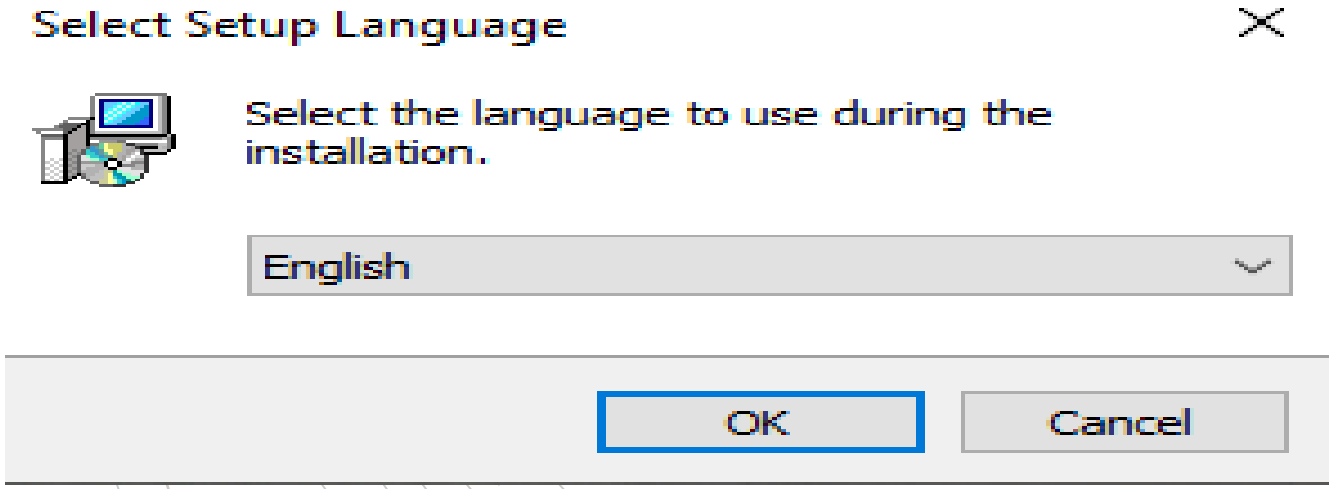
http://cran.us.r-project.org/

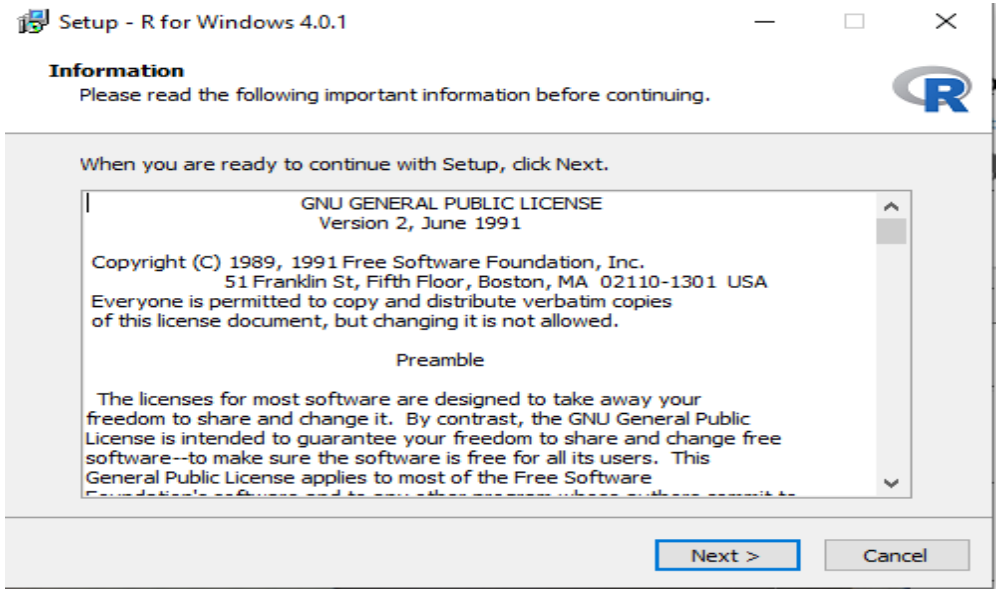


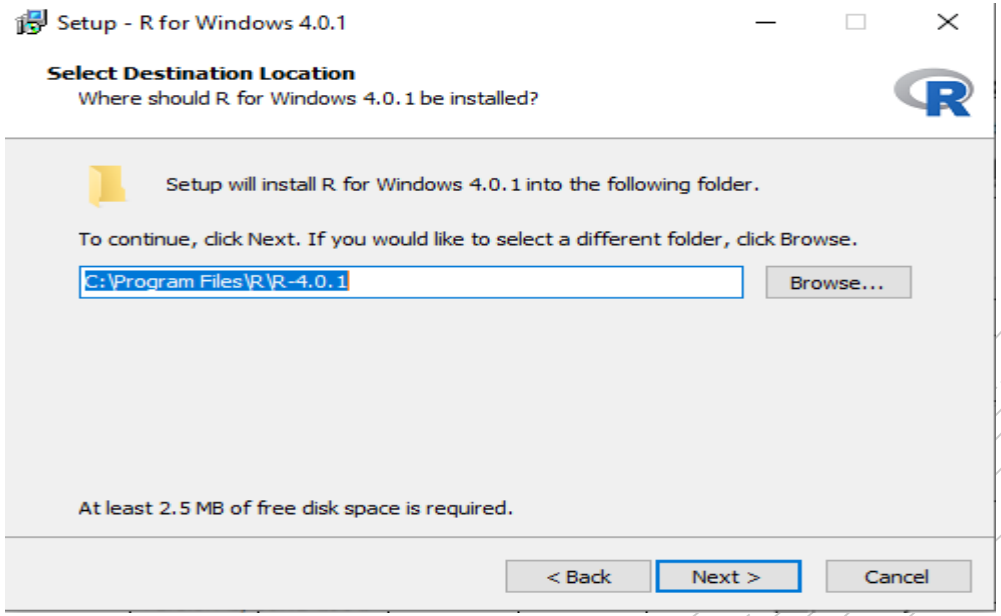
Then click on **install R for the first time** After Downloading the Software proceed as follows

Installation of R and R Studio

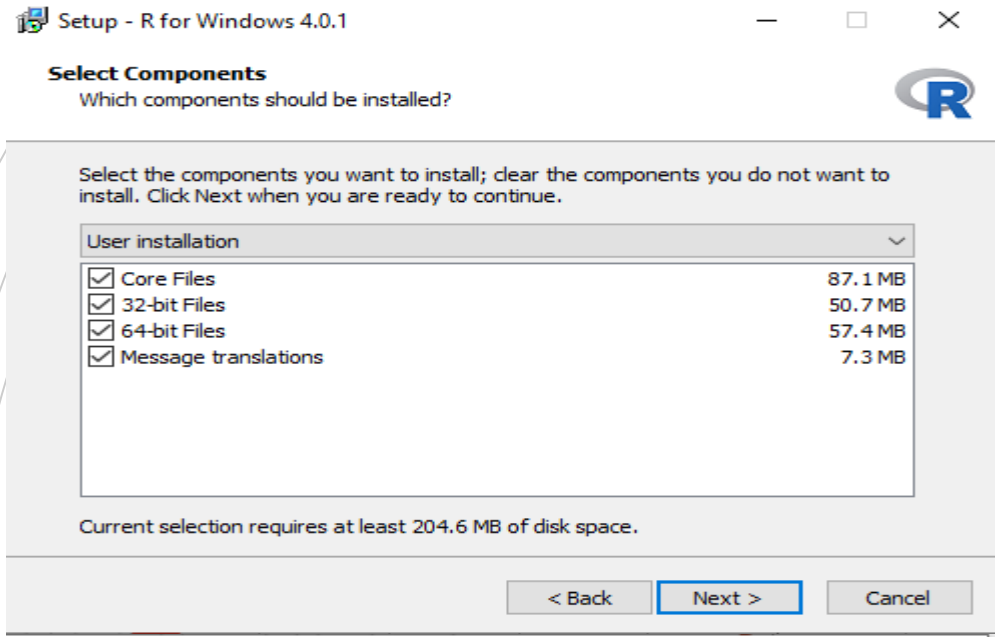
Choose Yes and Press Enter

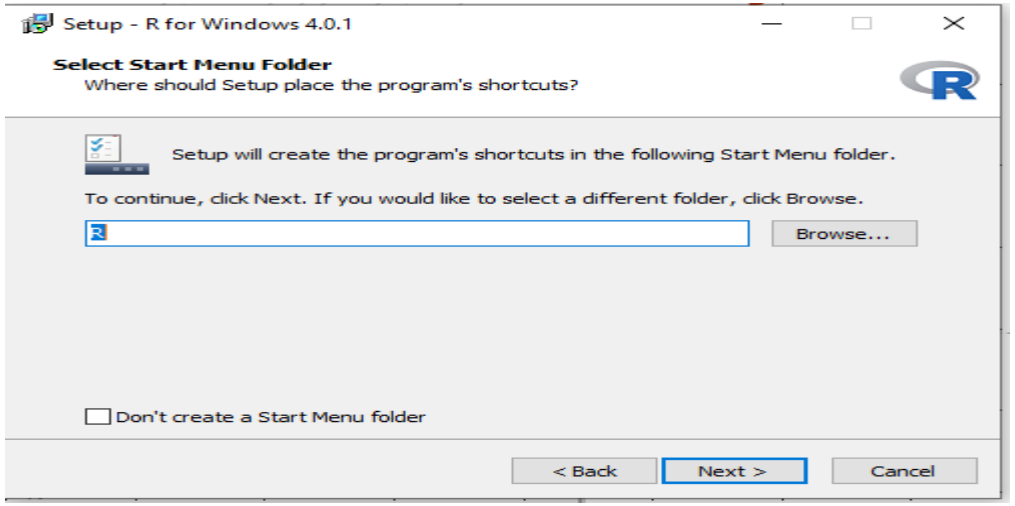


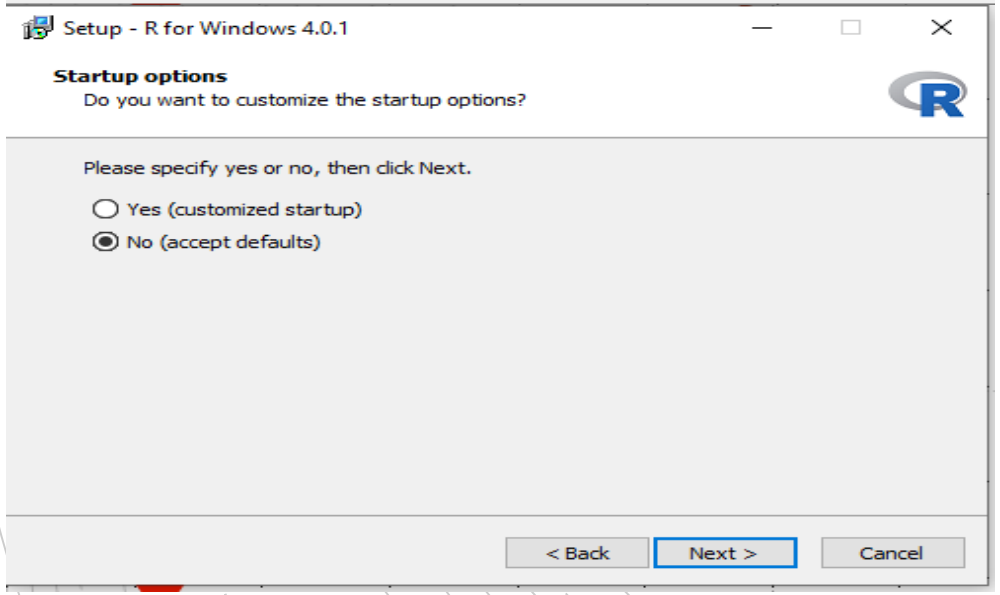


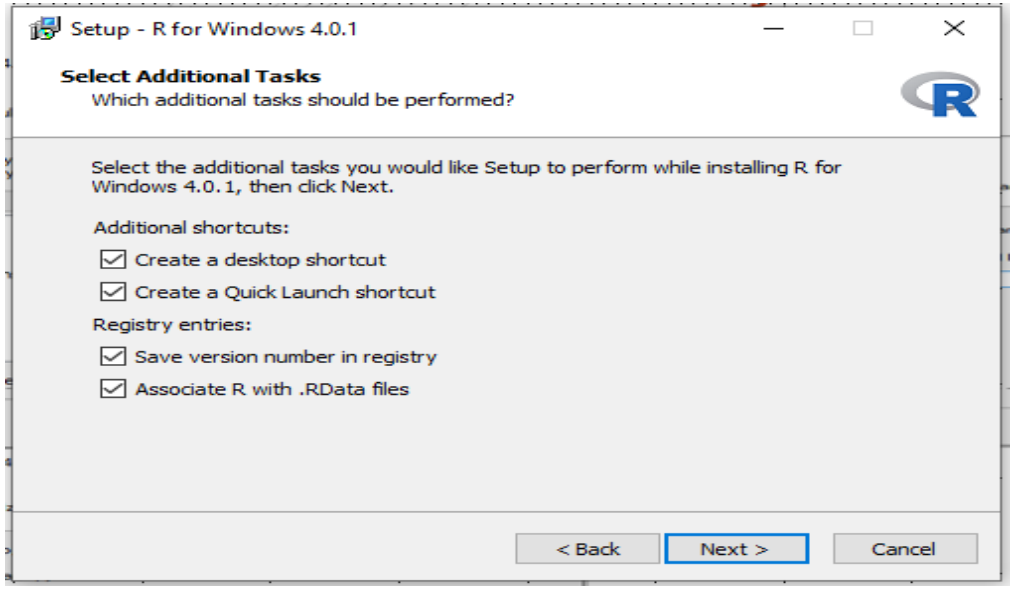


Installation of R and R Studio

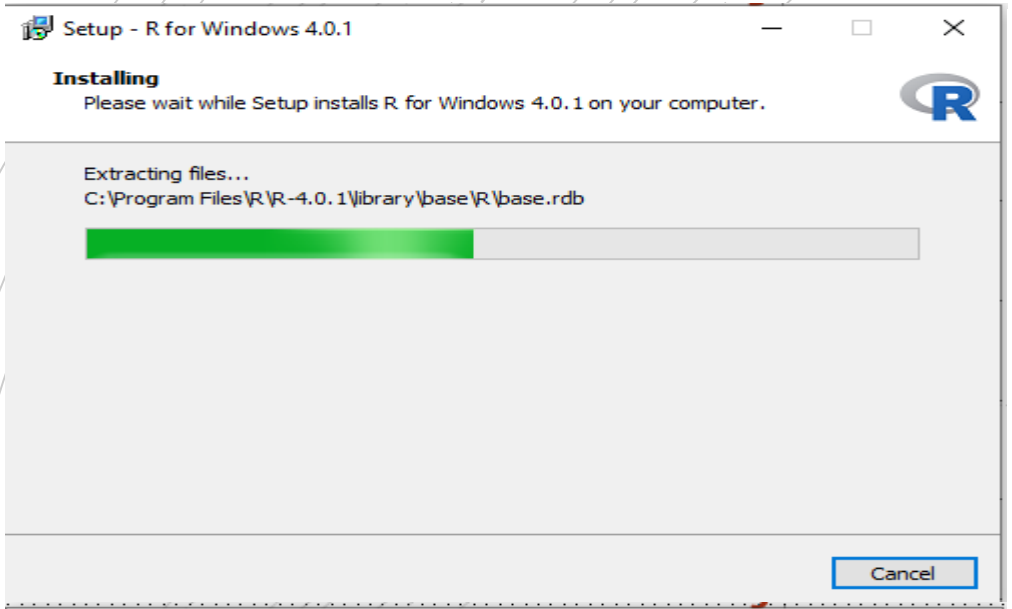








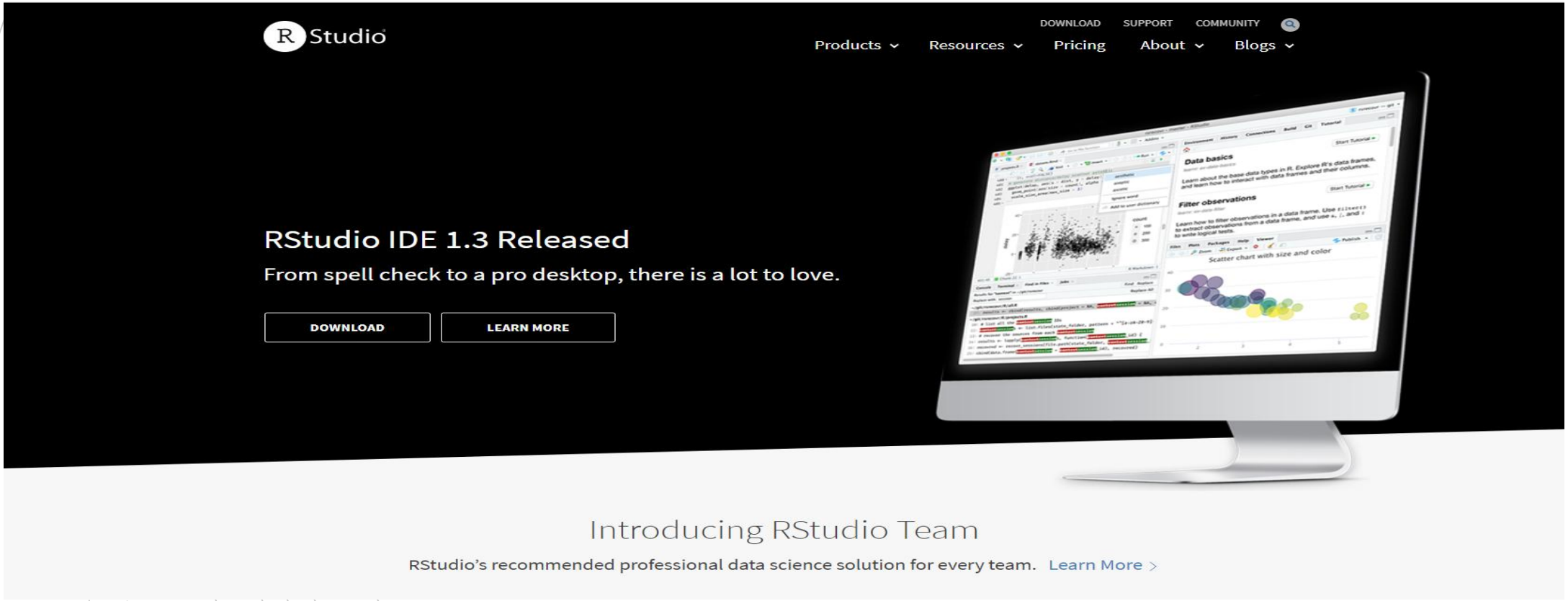
Installation of R and R Studio




```
R version 4.0.2 (2020-06-22) -- "Taking Off Again"
Copyright (C) 2020 The R Foundation for Statistical Computing
Platform: x86_64-w64-mingw32/x64 (64-bit)

R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

  Natural language support but running in an English locale

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

[Previously saved workspace restored]
```

Installation of R and R Studio

Use the Link https://www.rstudio.com/



Then click on **Download**

Follow next slide

Architecture of R Studio

- **R Studio** has four windows , they are used for copying the code reading the output , running the code etc., they are as follows,
- The four panes / windows of R Studio are
- 1. Source
- 2. Console
- 3. Environment / History
- 4. File / Plots / Packages / Help

Architecture of R Studio

- R Studio appears as follows ,
-

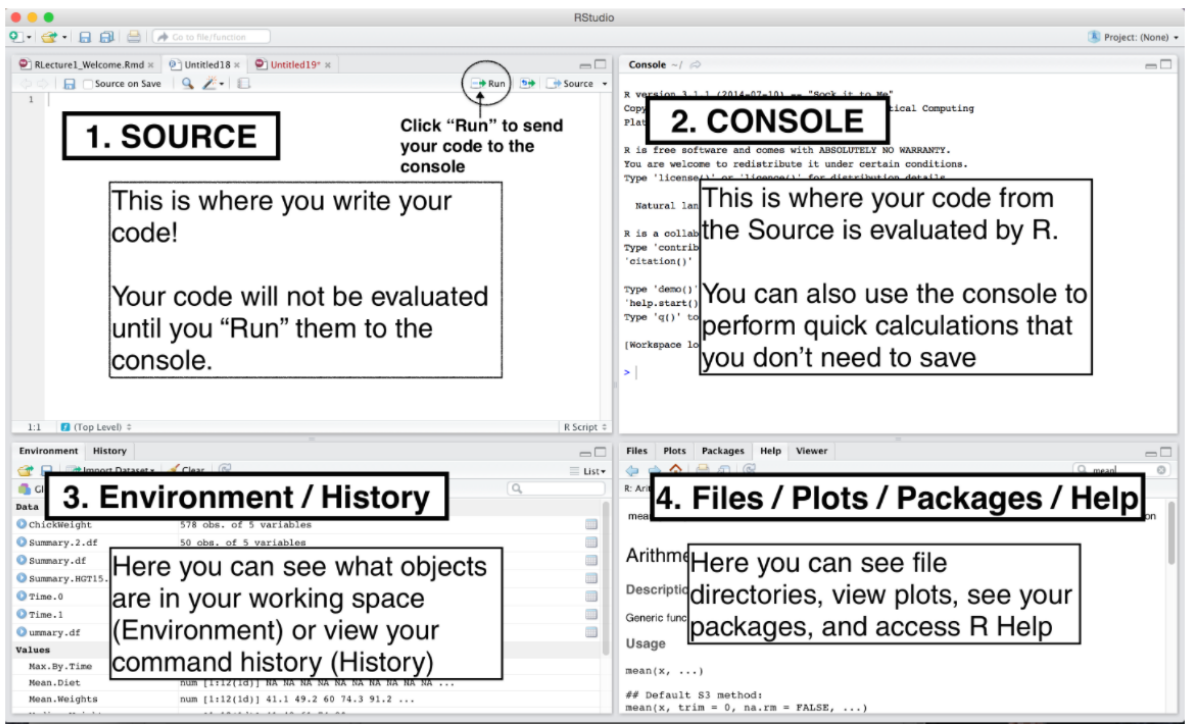
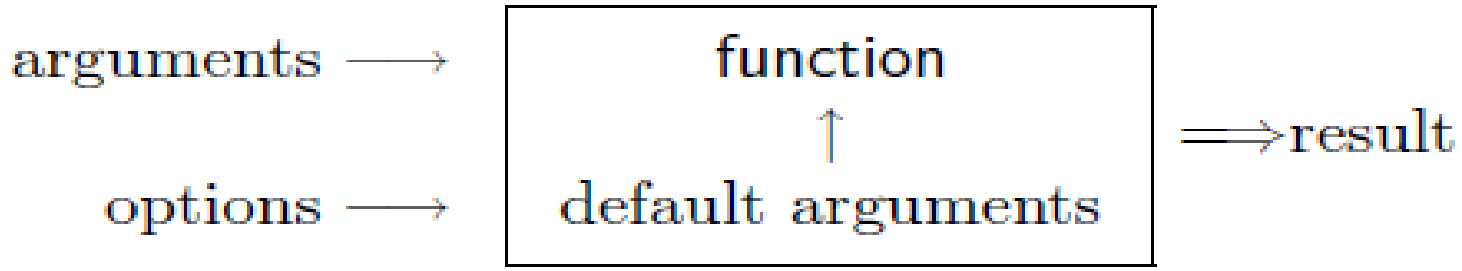


Figure 1.7: The four panes of RStudio.

Introduction to R

- Many people had contributed to development of R.
- One can have installation and other in CRAN (Comprehensive R Archive Network)
- An R software may be thought as follows



Introduction to R

- Another interesting feature is on line help the help function will give help on different functions.
- Syntax …. > ? Topic …. Will give the help from local software…
- > ?? Topic …. Will give the help from internet
- R provides scripting and interfacing facilities as well hence one can develop a customized software.
- R can be used as calculator.
- The R has command prompt >

```
[1]  8
> sqrt((19-1)^2)
[1]  18
> 2-3
[1]  -1
> 3*4-2
[1]  10
>
```

Basics of R

- Assignment operator
- R uses assignment operator <- for assigning a value for data object
- x <- 2 this indicates x is assigned with a value of 2.

  x takes the value 2
- 3 -> x this indicates x is assigned with a value of 3.

  value 3 is given to x.

  commands are separated by “;” or they can

  be in new line.

Basics of R

- Assignment operator
- > assign(“x”,3);x ;
- > assign(“Y”,”R”);R
- > class(x);class(Y)
- > z = scan() then press enter and enter the values and to terminate press double enter.

There are many ways to assign values to a variable , the above are few.

Systematic … R.. Operators

- > x<-10 ; y <- 21 ; x + y ; x - y ; x / y ; x * y ; y %% x ;
- [1] 31
- [1] -11
- [1] 0.4761905
- [1] 210
- [1] 1
- > 3.4 %/% 2
- [1] 1
- > 3.4 %% 2
- [1] 1.4

Basics of R

- For commenting # is used anything after # is ignored by the interpreter.

Logical Operators ;

- Comparison operators > greater than, < less than >= greater than or equals to <= less than or equals to == equals to != not equals to ! Logical not & logical and | logical or
- Data structures or Vectors
- Like any programming language R has vectors and data structures.
- A vector is a series of observations / elements of same type.

Systematic … R.. Operators

R has many operators to carry out different mathematical and logical operations.

Operators in R can mainly be classified into the following categories.

Type of operators in R

Arithmetic operators
Relational operators
Logical operators
Assignment operators

Systematic … R.. Operators

R Arithmetic Operators

These operators are used to carry out mathematical operations like addition and multiplication. Here is a list of arithmetic operators available in R.

Arithmetic Operators in R

Operator	Description
+	Addition
–	Subtraction
*	Multiplication
/	Division
^	Exponent
%%	Modulus (Remainder from division)
%/%	Integer Division

Systematic … R.. Operators

Relational Operators in R

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

Systematic … R.. Operators

- > x<-10 ; y <- 21
- > x>y
- [1] FALSE
- > x<y
- [1] TRUE
- > # equals to ==
- > x == y
- [1] FALSE
- > x!= y
- [1] TRUE

Systematic … R.. Operators

- Vector based Operators …
- The speciality of R programming is the operations on numbers can be applied to Vectors,
- Vector is array of numbers , Row Vector will have only one row , column Vector will have one column , a vector can have multiple rows and / or columns that will be called as Matrix.
- Let us observe some examples

Systematic … R.. Operators ; Operations on Vectors

- > x<-c(10,12,15,20) ; y <- c(2,6,8,9)
- > x + y ; x -y ; x /y ; x * y ; x + 100;
- [1] 12 18 23 29
- [1] 8 6 7 11
- [1] 5.000000 2.000000 1.875000 2.222222
- [1] 20 72 120 180
- [1] 110 112 115 120

Systematic … R.. Operators

- > x<-c(10,12,15,20) ; y <- c(2,6,8,9)
- > x > y ; x < y ; x != y ; x == y
- [1] TRUE TRUE TRUE TRUE
- [1] FALSE FALSE FALSE FALSE
- [1] TRUE TRUE TRUE TRUE
- [1] FALSE FALSE FALSE FALSE

Systematic … R.. Operators

R Logical Operators

Logical operators are used to carry out Boolean operations like `AND`, `OR` etc.

Logical Operators in R

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

Further …basics of R

Now hope you have become bit free with R ; let us proceed with some formal understanding of R

R is a language with multiple packages developed by many persons , to execute a code or syntax to achieve a specific task a package must be Installed and invoked ,

the process is as follows ,

Programmatic Driven/Syntax Driven

Syntax:

install.packages('name of the package')

Example:

install.packages('car')

Further …basics of R

After installation one need to invoke the package ,

To **activate** the package

Syntax:

library(name of the package installed)

Example:

library(car)

Note: If no errors/warning messages,

it does mean that installation is successful

Please try installing the package , **tidyverse**

Further …basics of R

Both the above can be achieved by the key word , **require**(tidyverse)

The others with respect to ***installation of Packages***,

To **see** the list of packages installed

Syntax:

installed.packages()

Example:

installed.packages()

To **update** the installed packages

Syntax:

update.packages()

Example:

update.packages('car')

Further …basics of R

To get the documentation on a function

Syntax:

help('Name of the function')

Example:

help('read.csv')

To search the help system

Syntax:

help.search('Function Name')

Example:

help.search('')

To see the list of functions with appropriate name

Syntax:

apropos('part of a function')

Example:

apropos('rea')

Further …basics of R

To get an example of a function

Syntax:

example('Function Name')

Example:

example(dim)

1)To set the working directory

Syntax:

setwd('path')

Example:

setwd('C:\\Users\\Desktop\\Sales Data.csv')

1)To set the working directory

Syntax:

setwd('path')

Example:

setwd('C:\\Users\\Desktop\\Sales Data.csv')

Further …basics of R

2)To know the existing working directory

Syntax:

getwd()

Example:

getwd() ; The working directory is just a file path on your computer that sets the default location of any files you read into R, or save out of R.

3)To choose a file using browse option

Syntax:

file.choose()

Example:

file.choose()

Further …basics of R

- Naming of the object / variable
- R is case sensitive hence one need to be careful in defining and
- execution of the commands.
- R is expression language with very simple syntax.
- A name or object can start with any alphabet or with special character _ or. then one can use any alpha numeric it depends on operating system and country of usage
- – Calculator; Expression Evaluator; vectors matrices working
- with vectors and matrices
- R as a calculator and R as tool to evaluate expression
- > 3 + 4 ; 10 – 2 ^ 10 – 210 + 3200; 50 ^ 10 + 200

Further …basics of R

- > m_name<-“Raghunadh”; s_name<-“Acharya”
- > paste(m_name, s_name)
- > x<-10
- > x ^ 7 + 10 – 2 * x
- *The output of the above expression is* ***9999990***

Data Types in R

1) Vector
2) List
3) Data Frame
4) Matrix
5) Array
6) Factor

Further …basics of R

1)Vector

- The simplest object in R is vector object.

This object covers 6 types of atomic vectors (six classes of vectors).

These 6 classes are logical, numeric, character, complex, integer & raw

2)List

- List is an object which contain many different types of elements like vectors, functions and even a list inside

3)Matrices

- A matrix is a 2-dimensional rectangular dataset.Created using a vector input to the matrix function

Further …basics of R

4)Arrays

- Array is n-dimensional rectangular dataset.

5)Data Frame

- An object with data arranged in rows and columns.

6)Factors

- An object which stores the categorical data.

This stores the nominal values as a vector of integers in the range of 1 to k.

Further …basics of R

- > # defining vectors and working with vectors
- > x is a vector taking different predefined values
- > x<-c(20,30,40,50,60,90) ; y<-c(21,43,54,65,29,92) ; assign("z",c(23,32,34,45,54))
- > x; y ; z
- Creating character vector v2<-c("r","m")
- > v2

Further …basics of R

- To Create the tables .. In R
- For example to create a table of the form ,

C1	C2
1	20
2	45
3	60

```
> data.frame(C1 = c(1,2,3), C2 = c(20,45,60))
  C1 C2
1  1 20
2  2 45
3  3 60
```

- Assigning the table to another variable

```
> tbl = data.frame(C1 = c(1,2,3), C2 = c(20,45,60))
> tbl
  C1 C2
1  1 20
2  2 45
3  3 60
```

Further …basics of R

- **To Create the tables .. In R**
- **For example to create a table of the form ,**

C1	C2
1	20
2	45
3	60

- **> tbl = data.frame(C1 = c(1,2,3),C2=c(30,45,60)**
- **> tbl**

```
  C1 C2
1  1 20
2  2 45
3  3 60
```

- **Naming the Rows and Columns**
- **rownames(tbl) = c("R1","R2","R3"); tbl**

```
   C1 C2
R1  1 20
R2  2 45
R3  3 60
```

Further …basics of R

- **Naming the column names**
- **colnames(tbl) = c(“T1”,”T2”)**

```
    T1 T2
R1   1 20
R2   2 45
R3   3 60
```

- **Modifying and Sub Setting of Vectors**
- One can modify the vectors just by giving values or values by conditions etc.,

```
> x<-c(12,13,20,23,45)
> x[3]<-34
> x
[1] 12 13 34 23 45
> x[x<15]<-50
> x
[1] 50 50 34 23 45
```

Further …basics of R

- *Creating Data in R*
- *1)Write the data into an object*
- *Syntax:*
- *object name<-c()*
- *Example:*
- *abc<-c(1,2,3,4,'B School','MBA')*
- *abc*

Further …basics of R

- 2) Creating a series of numbers
- Syntax:
- seq(start_num,end_num,by=number)
- Example:
- seq(1,100,0.8)
- 3) To repeat a number(s) by n times
- Syntax:
- rep(req_num,no.of times)
- Example:
- rep(12,10)
- rep('MBA',10)

Generate 50 Even numbers ; and 50 Odd Numbers

Further …basics of R

- 4)To generate random numbers in a specified range
- Syntax:
- sample(dataset, sample size)
- Example:
- sample(1:100,10)

```
> sample(1:100,10,replace = T)
 [1] 56  2 26 86 79 96 86  7 12 53
> x= c(2,5,6,8,9,12,15,20)
> sample(x,10,replace = F)
Error in sample.int(length(x), size, replace, prob) :
  cannot take a sample larger than the population when 'replace = FALSE'
> sample(x,10,replace = T)
 [1]  9  8  6  6  6 15 12 12 20  2
```

Further …basics of R

- 5)To create continuous random numbers
- Syntax:
- rnorm(number of values)
- Example:
- rnorm(10)
- 6) To select capital letters
- Syntax:
- LETTERS[start_num:end_num]
- Example:
- LETTERS[2:9]

Further …basics of R

- 7)To select small letters
- Syntax:
- letters[start_num:end_num]
- Example:
- letters[2:9]
- 8)To select the months
- Syntax:
- month.abb[start_num:end_num]
- Example:
- month.abb[5:12]

Further …basics of R

- ***Creating Matrices***
- **Matrices are numbers or symbols arranged in the form rows and columns**
- **Syntax**
- **To create a matrix**
- **Syntax:**
- **matrix(object name, nrow= ,ncol=, byrow=TRUE)**
- **Example:**
- **M = matrix(c('a','a','b','c','b','a'), nrow = 2, ncol = 3, byrow = TRUE)**

Create a matrix with the numbers

2	-3	10
6	8	11

Further …basics of R

```
> datamat = matrix(c(2,-3,10,6,8,11),nrow = 2 , ncol = 3 , byrow = T)
> datamat
     [,1] [,2] [,3]
[1,]    2   -3   10
[2,]    6    8   11
> colnames(datamat)=c("I","II","III")
> datamat
     I II III
[1,] 2 -3  10
[2,] 6  8  11
```

- To create an array
- Syntax:
- array(object name, dim=c())
- Example:
- array(c('green','yellow','red','blue','black'),dim = c(3,3,2))

Further …basics of R

- Data Conversion and Data Conversion Validation and Data Exploration
- Refer to Note Pad file R in Notepad by Dr.R
- Data Exploration , data exploration will take up the data in different ways ,
- Refer to Note Pad file R in Notepad by Dr.R
- to check the sample files in R
- data()

Further …basics of R … Importing and Exporting files

- Importing and Exporting the Files …. Flat Files , Excel Files , SPSS Files , SAS Files , etc.,
- Importing and exporting Flat files
- Generally for importing …. Key word is Read
- For exporting it is Write

Further …basics of R

Reading a note pad file

- file.choose( ) for choosing the location / path of the file,
- Read.scv **or** read.delim **or** read.table can be used for reading the note pad files

Reading an EXCEL file,

Further …basics of R

```
#1. BASICS
#=============================================================
 ============================
#1. Basic Arithmetic
2+3
5 %% 2
round(3.4, 0)
abs(-2.4)
```

Further …basics of R

```
# 2 Basic Character
> rn = "raghunadh"
> an <- "acharya"
> paste0(rn,an)
[1] "raghunadhacharya"
> paste(rn, an)
[1] "raghunadh acharya"
#=========================================================
#1 declaring vectors
x <- c(1,2,3)
y <- c(5,4,6)
x+y
2*x
```

Further …basics of R

 # 3 Sub setting vector

 # 1. User defined order

```
 y <- c(5,4,6)
#  y has three elements , first element is 5 ; second element is 4 and the third element is 6 ;
  Any vector value can be read by the vector as it is or one can call by place / position
  > y<-c(5,4,6)
  > y[2]
   [1] 4
  > y[c(2,1,3)]
     [1] 4 5 6
 # alternatively
 mashi <- c(2,1,3)
  y[mashi]
  [1] 4 5 6
```

Further …basics of R

```
# Selecting values of a vector based on a mathematical operator
> y
[1] 5 4 6
> y[y>4]
[1] 5 6
#  Display the values of a Vector by Order
> order(y, decreasing = T) # displays the positions
[1] 3 1 2
> y[order(y, decreasing = T)] # displays the values
[1] 6 5 4
```

Further …basics of R

```
> y[order(y)]  # default is ascending or increasing
> # sorting the values in a vector
> v<-c(12,9,25,16,3)
> v
[1] 12  9 25 16  3
> sort(v, decreasing = TRUE)
[1] 25 16 12  9  3
> sort(v)
[1]  3  9 12 16 25
```

Further …basics of R

Replacing the variables in a **Table**

> age= c(10,20,30); name=c('T','Q','W'); gen=c("M","F","M");

- **tabl = data.frame(name,age,gen)**
- **# to remove a variable**
- **> tabl**

name age gen

1 T 10 M

2 Q 20 F

3 W 30 M

Further …basics of R

```
> tabl[-2]
  name gen
1    T   M
2    Q   F
3    W   M
> tabl$age = age
> tabl
  name age gen
1    T  10   M
2    Q  20   F
3    W  30   M
```

Further …basics of R

```
# adding a row
 > tabl
  name age gen
1    T  10   M
2    Q  20   F
3    W  30   M
> # adding a row to an existing table
> v = c("E",30,"M")
> rbind(tabl,v)
  name age gen
1    T  10   M
2    Q  20   F
3    W  30   M
4    E  30   M
```

Further …basics of R

 # adding a column

many ways of adding a column , can be done using cbind or otherwise,

tabl

name age gen

1 T 10 M

2 Q 20 F

3 W 30 M

Any variable in a table or array can be recalled as,

$variable

Further …basics of R

Adding a new column or a new variable is

```
> tabl$presentage = age + 2
> tabl
  name age gen presentage
1    T  10   M         12
2    Q  20   F         22
3    W  30   M         32
```

The same can be achieved by cbind

Further …basics of R

```
cbind(tabl,age-10)

  name age gen presentage age - 10
1    T  10   M         12       0
2    Q  20   F         22      10
3    W  30   M         32      20
```

Matrices in … R…

Matrices are the objects which are elements are represented in a two-dimensional structure where mostly the numeric elements are present for doing various computation.

Syntax for Creating Matrix,

```
matrix(value, nrow, ncol, byrow, dimnames)
```

The parameters are as follows ,

1. **value** - specifies the vector input, which is each entry specifies the elements in a matrix.
2. **nrow** - specifies the size of the row in the matrix.

Matrices in … R…

3. ncol - specifies the size of the column in the matrix.

4. byrow - The boolean value consists of either TRUE or FALSE. If TRUE, the elements of the matrix are arranged in the row, whereas FALSE will arrange the element by column-wise.

5. dimnames - specifies the name given to each element of rows and columns of a matrix.

```
> matri<-matrix(c(-2,4,10,4,5,6,-2,5),nrow = 2,ncol = 4 )
> matri
     [,1] [,2] [,3] [,4]
[1,]   -2   10    5   -2
[2,]    4    4    6    5
```

Matrices in … R…

```
> matri<-matrix(c(-2,4,10,4,5,6,-2,5),nrow = 2,ncol = 4 ,byrow = TRUE)
> matri
     [,1] [,2] [,3] [,4]
[1,]   -2    4   10    4
[2,]    5    6   -2    5
> matri1<-matrix(c(-2,4,10,4,5,6,-2,5),nrow = 2,ncol = 4 ,byrow = FALSE)
> matri
     [,1] [,2] [,3] [,4]
[1,]   -2   10    5   -2
[2,]    4    4    6    5
```

Matrices in … R…

```
> # naming the Rows and Columns in a Matrix
> matri<-matrix(c(-2,4,10,4,5,6,-2,5),nrow = 2,ncol = 4 ,byrow =
  FALSE,dimnames=(list(c("R1","R2"),c("C1","C2","C3","C4"))))
> matri
   C1 C2 C3 C4
R1 -2 10  5 -2
R2  4  4  6  5
```

Matrices in … R…

Further on Matrices …

- **x = c(-1,-2,10,12,15,20,21,5,6,10,15,20)**
- **# creating a 2 by 6 and 6 by 2 matirces ; length(x);**
- **mat = matrix(x,nrow = 2, ncol = 6, byrow = T)**

```
> mat
     [,1] [,2] [,3] [,4] [,5] [,6]
[1,]   -1   -2   10   12   15   20
[2,]   21    5    6   10   15   20
mat2 = matrix(x,nrow = 2, ncol = 6, byrow = F)
> mat2
     [,1] [,2] [,3] [,4] [,5] [,6]
[1,]   -1   10   15   21    6   15
[2,]   -2   12   20    5   10   20
>
```

Generating a Sequence of Numbers… R

```
> 1:20
 [1]  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20
> pi
[1] 3.141593
> pi:10
[1] 3.141593 4.141593 5.141593 6.141593 7.141593 8.141593 9.141593
> 15:1
 [1] 15 14 13 12 11 10  9  8  7  6  5  4  3  2  1
```

Generating a Sequence of Numbers… R

```
> seq(1,20)
 [1]  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20
> seq(from = 1 , to = 20)
 [1]  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20
> seq(from = 1 , to = 20,by = 0.5)
 [1]  1.0  1.5  2.0  2.5  3.0  3.5  4.0  4.5  5.0  5.5  6.0  6.5  7.0  7.5  8.0  8.5  9.0  9.5
10.0 10.5 11.0 11.5 12.0 12.5 13.0 13.5 14.0 14.5 15.0 15.5 16.0 16.5 17.0 17.5 18.0
18.5 19.0 19.5 20.0
> rep(2,20)
 [1] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
```

Generating a Sequence of Numbers… R

```
> rep(c(2,3,4),10)
 [1] 2 3 4 2 3 4 2 3 4 2 3 4 2 3 4 2 3 4 2 3 4 2 3 4 2 3 4 2 3 4
> # generating even numbers ; syntax of sequence ; seq( from = a to = b , by = c)
> seq(from = 0, to = 100 , by = 2)
 [1]   0   2   4   6   8  10  12  14  16  18  20  22  24  26  28  30  32  34  36  38  40  42  44  46
48  50  52  54  56  58  60  62  64  66  68  70  72  74  76  78  80  82  84  86  88
[46]  90  92  94  96  98 100
> # generating odd numbers
> seq(1,100,2)
 [1]  1  3  5  7  9 11 13 15 17 19 21 23 25 27 29 31 33 35 37 39 41 43 45 47 49 51 53 55
57 59 61 63 65 67 69 71 73 75 77 79 81 83 85 87 89 91 93 95 97 99
```

Importing and Exporting the data in … R

Importing and Exporting a note pad file …

Usually for importing the commands in R are **read** ; and for Exporting the commands in R are **write**

Importing a note pad file or flat file , they are many ways in r for reding the contents or importing a notepad file in R

read.table, read.del, read.csv are the commonly used commands ,

Example : to read a note pad file

```
note <- read.table(“ file location or path ”)
```

Importing and Exporting the data in … R

Example : > flat<-**read.table**("C:\\Users\\drach\\OneDrive\\Desktop\\raghu.txt")

> **flat**

```
  V1  V2
1  Name age
2 Raghunadh  80
3  Ramu  35
4  Savitha  42
```

> flat <- **read.table**("C:\\Users\\drach\\OneDrive\\Desktop\\raghu.txt",header = TRUE)

> **flat**

```
  Name        age
1 Raghunadh  80
2 Ramu  35
3 Savitha  42
```

One can read using the key words read.csv ; read.delim

Importing and Exporting the data in … R

Writing in to a note pad format ;

write.table(mtcars,"C:\\Users\\drach\\OneDrive\\Desktop\\raghu1.txt")

Reading an **EXCEL file** ,

- Install
- `install.packages("readxl")`
- Load
- **`library`**`("readxl")`

Reading an **EXCEL file** ,

> exel <- **read_excel**("C:\\Users\\drach\\OneDrive\\Desktop\\Example.xlsx",sheet = "R")

Importing and Exporting the data in … R

```
> exel
# A tibble: 4 x 2
  Name       Age
  <chr>    <dbl>
1 Raghunadh    90
2 Vasundara    43
3 Ramesh       28
4 Keerthana    32
```

Importing and Exporting the data in … R

Writing in to an EXCEL format ;

For reading and writing there is another way of doing it by means of

Installing a package called ; **require('xlsx')** or install.packages('xlsx') and library('xlsx') **;**

Writing in to EXCEL file ,

- install.packages('xlsx')
- library(openxlsx)
- write.xlsx(mtcars,"C:\\Users\\drach\\OneDrive\\Desktop\\Exx.xlsx",colnames=T ,rownames=T)
- write.xlsx(mtcars,"C:\\Users\\drach\\OneDrive\\Desktop\\Exl.xlsx",colnames=T, ro wnames=T ,sheetName = "raghu")

Importing and Exporting the data in … R

Writing by some other format ;

with out java requirement ,

- install.packages("writexl") ; > library(writexl) ;

Writing the file to EXCEL format

- write_xlsx(iris,"C:\\Users\\drach\\OneDrive\\Desktop\\telco.xlsx")

Importing and Exporting the data in … R

The error can be resolved by installing java software

let us look in to importing and exporting IBM – SPSS files ,

```
> install.packages('tidyverse’)
> library(haven)
```

Importing a sample file of spss

```
> bankloan<-read_sav("C:\\Program Files (x86)\\IBM\\SPSS\\Statistics\\20\\Samples\\English\\bankloan.sav")
> bankloan
# A tibble: 850 x 12
    age  ed employ address income debtinc creddebt othdebt   default preddef1 preddef2 preddef3
  <dbl>      <dbl+lbl> <dbl>   <dbl> <dbl>   <dbl>   <dbl>   <dbl> <dbl+lbl>   <dbl>   <dbl>   <dbl>
1    41 3 [Some college]              17     12   176     9.3  11.4     5.01    1 [Yes]   0.808   0.789     0.213
2    27 1 [Did not complete high school]    10      6    31    17.3   1.36    4.00    0 [No]    0.198   0.128     0.437
3    40 1 [Did not complete high school]    15     14    55     5.5   0.856   2.17    0 [No]    0.0100  0.00299   0.141
```

Importing and Exporting the data in … R

```
# to see first ten records one can use
➢ head(bankloan,10)
# to see last ten records one can use
➢ tail(bank,10)
> # arranging the data in a proper table from
> bank<-data.frame(bankloan)
> head(bank,10)
   age ed employ address income debtinc  creddebt   othdebt default   preddef1   preddef2  preddef3
1   41  3     17      12    176     9.3 11.359392  5.008608       1 0.80839433 0.788640432 0.2130434
2   27  1     10       6     31    17.3  1.362202  4.000798       0 0.19829748 0.128445387 0.4369030
3   40  1     15      14     55     5.5  0.856075  2.168925       0 0.01003611 0.002986778 0.1410226
4   41  1     15      14    120     2.9  2.658720  0.821280       0 0.02213828 0.010273265 0.1044222
5   24  2      2       0     28    17.3  1.787436  3.056564       1 0.78158831 0.737884820 0.4369030
6   41  2      5       5     25    10.2  0.392700  2.157300       0 0.21670894 0.328194657 0.2335771
7   39  1     20       9     67    30.6  3.833874 16.668126       0 0.18596011 0.179255713 0.8170917
8   43  1     12      11     38     3.6  0.128592  1.239408       0 0.01470865 0.010572668 0.1133576
9   24  1      3       4     19    24.4  1.358348  3.277652       1 0.74804120 0.619443452 0.6639040
10  36  1      0      13     25    19.7  2.777700  2.147300       0 0.81505701 0.797227451 0.5155301
```

Importing and Exporting the data in … R

To write in to SPSS file / Exporting an SPSS file

- **write_sav**(mtcars,"C:\\Users\\drach\\OneDrive\\Desktop\\bank.sav")
- **Sample files in R** the sample files in R can be viewed by the command
- **> data( ) …**

```
Data sets in package 'datasets':

AirPassengers                           Monthly Airline Passenger Numbers 1949-1960
BJsales                                 Sales Data with Leading Indicator
BJsales.lead (BJsales)                  Sales Data with Leading Indicator
BOD                                     Biochemical Oxygen Demand
CO2                                     Carbon Dioxide Uptake in Grass Plants
ChickWeight                             Weight versus age of chicks on different diets
DNase                                   Elisa assay of DNase
EuStockMarkets                          Daily Closing Prices of Major European Stock Indices, 1991-1998
Formaldehyde                            Determination of Formaldehyde
HairEyeColor                            Hair and Eye Color of Statistics Students
Harman23.cor                            Harman Example 2.3
Harman74.cor                            Harman Example 7.4
Indometh                                Pharmacokinetics of Indomethacin
InsectSprays                            Effectiveness of Insect Sprays
JohnsonJohnson                          Quarterly Earnings per Johnson & Johnson Share
```

To check the specific file for example … to check … mtcars …

```
> mtcars
                   mpg cyl  disp  hp drat    wt  qsec vs am gear carb
Mazda RX4         21.0   6 160.0 110 3.90 2.620 16.46  0  1    4    4
Mazda RX4 Wag     21.0   6 160.0 110 3.90 2.875 17.02  0  1    4    4
Datsun 710        22.8   4 108.0  93 3.85 2.320 18.61  1  1    4    1
Hornet 4 Drive    21.4   6 258.0 110 3.08 3.215 19.44  1  0    3    1
Hornet Sportabout 18.7   8 360.0 175 3.15 3.440 17.02  0  0    3    2
Valiant           18.1   6 225.0 105 2.76 3.460 20.22  1  0    3    1
Duster 360        14.3   8 360.0 245 3.21 3.570 15.84  0  0    3    4
Merc 240D         24.4   4 146.7  62 3.69 3.190 20.00  1  0    4    2
Merc 230          22.8   4 140.8  95 3.92 3.150 22.90  1  0    4    2
Merc 280          19.2   6 167.6 123 3.92 3.440 18.30  1  0    4    4
```

Importing and Exporting the data in … R

Reading and Writing an SPSS file using another method

The following is another method for reading the data ….

```
> library(foreign)
> let<-read.spss("C:\\Program Files (x86)\\IBM\\SPSS\\Statistics\\20\\Samples\\English\\telco.sav")
re-encoding from UTF-8
> # in the above we gave the path manually , but path can be copied as follows
> file.choose()
[1] "C:\\Program Files (x86)\\IBM\\SPSS\\Statistics\\20\\Samples\\English\\telco.sav"
```

Viewing Data… in … R…

Some basic functions to check and view data

dim will give the number of records and variables , the number of rows and columns

```
> dim(mtcars)
[1] 32 11
```

names will give you the variable names

```
> names(mtcars)
 [1] "mpg"  "cyl"  "disp" "hp"   "drat" "wt"   "qsec" "vs"   "am"   "gear" "carb“
> ls(mtcars)
 [1] "am"   "carb" "cyl"  "disp" "drat" "gear" "hp"   "mpg"  "qsec" "vs"   "wt"
```

- **ls() … gives the variable / vector / file names in the R directory**

```
>  ls()
  [1] "a"           "accuracy_test"  "ad"          "add"         "adver"       "age_trans"
  [7] "alpha"       "an"             "ana"         "ancova"      "anova"       "ANOVA"
 [13] "anova_tway"  "aov"            "b"           "ban"         "bank"        "bankloan"
```

Viewing data… in … R…

```
> # the function attach will attach the file to the present r working directory
> mpg
Error: object 'mpg' not found
> attach(mtcars)
The following objects are masked _by_ .GlobalEnv:
cyl, disp, hp
> mpg
 [1] 21.0 21.0 22.8 21.4 18.7 18.1 14.3 24.4 22.8 19.2 17.8 16.4 17.3 15.2 10.4 10.4 14.7 32.4 30.4 33.9
      21.5
[22] 15.5 15.2 13.3 19.2 27.3 26.0 30.4 15.8 19.7 15.0 21.4
```

Viewing Data… in … R…

The function str will give a brief description of the file

```
> str(mtcars)
'data.frame':   32 obs. of  11 variables:
$ mpg : num  21 21 22.8 21.4 18.7 18.1 14.3 24.4 22.8 19.2 ...
$ cyl : num  6 6 4 6 8 6 8 4 4 6 ...
$ disp: num  160 160 108 258 360 ...
$ hp  : num  110 110 93 110 175 105 245 62 95 123 ...
$ drat: num  3.9 3.9 3.85 3.08 3.15 2.76 3.21 3.69 3.92 3.92 ...
$ wt  : num  2.62 2.88 2.32 3.21 3.44 ...
$ qsec: num  16.5 17 18.6 19.4 17 ...
$ vs  : num  0 0 1 1 0 1 0 1 1 1 ...
$ am  : num  1 1 1 0 0 0 0 0 0 0 ...
$ gear: num  4 4 4 3 3 3 3 4 4 4 ...
$ carb: num  4 4 1 1 2 1 4 2 2 4 ...
```

Viewing Data… in … R…

```
> # class function will five the type of varibale ; and the type of file in possible cases
> class(mtcars)
[1] "data.frame"
> class(mpg)
[1] "numeric"
> class(bank)
[1] "data.frame“
```

print … will display the file with all records.

head … will display the specified records …. tail … will display the specified records

Tables… in … R…

Tables in R :

Data Frame is way of expressing the data in the form of a table ,

for example : > # creating two vectors using the command ; combine or concatenate

```
> x<-c(3,4,5,6,8)
> y<-c("a","b","c","d","e")
> data.frame(x,y)
  x y
1 3 a
2 4 b
3 5 c
4 6 d
5 8 e
> table(x,y)
   y
x   a b c d e
  3 1 0 0 0 0
  4 0 1 0 0 0
  5 0 0 1 0 0
  6 0 0 0 1 0
```

Tables… in … R…

As the table is not giving meaningful information , table should be used for appropriate data,

Let us consider the data , bankloan ,

```
> names(bank)
```

```
[1] "age"     "ed"      "employ"   "address"  "income"   "debtinc"  "creddebt" "othdebt"
"default" [10] "preddef1" "preddef2" "preddef3“
```

```
> table(ed,default)
```

ed	default 0	1
1	293	79
2	139	59
3	57	30
4	24	14
5	4	1

This table is giving numbers with respect to the variables in row and in column

Tables… in … R…

```
> # let us observe another function margin.table , as the name suggets it will give the details in
    margin
> margin.table(tab)
[1] 700
> margin.table(tab,1)
ed
  1   2   3   4   5
372 198  87  38   5
> margin.table(tab,2)
default
  0   1
517 183
```

Tables… in … R…

- **# letus observe another form of tables prop.table**

```
> tab
   default
ed    0   1
   1 293  79
   2 139  59
   3  57  30
   4  24  14
   5   4   1
```

- **This is a cross tab between ed and default**
- **Let observe some examples of prop.table**

```
> # over all percentage
> prop.table(tab)
    default
ed             0           1
   1 0.418571429 0.112857143
   2 0.198571429 0.084285714
   3 0.081428571 0.042857143
   4 0.034285714 0.020000000
   5 0.005714286 0.001428571
```

Tables… in … R…

```
> # row wise percentages
> prop.table(tab,1)
   default
ed          0         1
  1 0.7876344 0.2123656
  2 0.7020202 0.2979798
  3 0.6551724 0.3448276
  4 0.6315789 0.3684211
  5 0.8000000 0.2000000
> # Colmn wise Percentage
> prop.table(tab,2)
   default
ed          0         1
  1 0.566731141 0.431693989
  2 0.268858801 0.322404372
  3 0.110251451 0.163934426
  4 0.046421663 0.076502732
  5 0.007736944 0.005464481
```

Selecting a Sample …in R

Selecting a sample from the given data set

```
> index <- sample(1:nrow(iris), 5)
> index
[1] 114  88  51  32  11
> iris[index,]
    Sepal.Length Sepal.Width Petal.Length Petal.Width    Species
114          5.7         2.5          5.0         2.0  virginica
88           6.3         2.3          4.4         1.3 versicolor
51           7.0         3.2          4.7         1.4 versicolor
32           5.4         3.4          1.5         0.4     setosa
11           5.4         3.7          1.5         0.2     setosa
```

Selecting a Sample …in R

```
> # the keyword set.seed will generate the same random sample
> sample(c(1:10))
 [1]  8  7  6  9  3  2  1  4 10  5
> set.seed(10)
> sample(c(1:10))
 [1]  9  7  8  6  3  2 10  5  4  1
> set.seed(10)
> sample(c(1:10))
 [1]  9  7  8  6  3  2 10  5  4  1
```

Selecting a Sample …in R

```
> mtcars[sample(nrow(mtcars),10),]
                mpg cyl  disp  hp drat    wt  qsec vs am gear carb
Ferrari Dino   19.7   6 145.0 175 3.62 2.770 15.50  0  1    5    6
Merc 450SLC    15.2   8 275.8 180 3.07 3.780 18.00  0  0    3    3
Camaro Z28     13.3   8 350.0 245 3.73 3.840 15.41  0  0    3    4
Mazda RX4 Wag  21.0   6 160.0 110 3.90 2.875 17.02  0  1    4    4
Hornet 4 Drive 21.4   6 258.0 110 3.08 3.215 19.44  1  0    3    1
Datsun 710     22.8   4 108.0  93 3.85 2.320 18.61  1  1    4    1
Maserati Bora  15.0   8 301.0 335 3.54 3.570 14.60  0  1    5    8
Mazda RX4      21.0   6 160.0 110 3.90 2.620 16.46  0  1    4    4
Porsche 914-2  26.0   4 120.3  91 4.43 2.140 16.70  0  1    5    2
Merc 280       19.2   6 167.6 123 3.92 3.440 18.30  1  0    4    4
```

Selecting a Sample …in R

Selecting two samples from the given data set

One the Control Group (70%) ; the other remaining ,Test Group(30 %)

```
> index<-sample(2,nrow(tel),replace = TRUE,prob = c(.70,.30))
> index
 [1] 1 2 1 2 2 2 1 1 1 1 2 1 1 1 1 2 2 2 2 2 2 1 1 2 1 2 1 2 2 1 1 1 1 2 1 1
[41] 1 1 1 1 1 1 2 1 1 2 1 1 1 1 2 2 2 1 1 1 1 1 1 1 1 1 2 2 1 1 2 2 2 1 1 2 2 1 1
```

this indicates 1st record in 1st sample , 2nd record in second sample etc.,

Selecting part of a file based on a condition

```
> mtcars[(mpg>30),]
                mpg cyl disp  hp drat    wt  qsec vs am gear carb
Fiat 128       32.4   4 78.7  66 4.08 2.200 19.47  1  1    4    1
Honda Civic    30.4   4 75.7  52 4.93 1.615 18.52  1  1    4    2
Toyota Corolla 33.9   4 71.1  65 4.22 1.835 19.90  1  1    4    1
Lotus Europa   30.4   4 95.1 113 3.77 1.513 16.90  1  1    5    2
```

Control Group and Test Group …in R

```
> # Selecting a Control Group , Control Group has 70% of the data
> Controlgroup<-sample(tel[index == 1,],)
> head(Controlgroup,10)
   ebill retire internet equipten longmon voice longten age ed equip region equipmon  loglong wireless confer
1      0      0        0     0.00    3.70     0   37.45  44  4     0      2     0.00 1.308333        0      0
3      0      0        0     0.00   18.15     0 1300.60  52  1     0      3     0.00 2.898671        0      1
7      1      0        1     0.00   10.90     0  504.50  22  2     0      3     0.00 2.388763        0      0
8      1      0        1  1820.90    6.05     1  239.55  35  2     1      2    50.10 1.800058        1      1
9      0      0        0     0.00    9.75     0  449.05  59  4     0      3     0.00 2.277267        0      1
10     0      0        0     0.00   24.15     0 1659.70  41  1     0      1     0.00 3.184284        0      0
11     1      0        1   110.10    4.85     0   17.25  33  4     1      2    26.15 1.578979        0      0
13     0      0        0     0.00    8.55     0  308.70  38  2     0      1     0.00 2.145931        0      0
14     1      0        1  2590.95   15.60     1  825.35  54  4     1      2    46.70 2.747271        1      1
15     0      0        0     0.00    4.40     0   36.80  46  1     0      2     0.00 1.481605        0      0
   tollmon  logtoll multline employ    lninc churn custcat marital callcard forward cardten callid address
1     0.00       NA        0      5 4.158883     1       1       1        1       1     110      0       9
3    18.00 2.890372        0     29 4.753590     0       3       1        1       0    2150      1      24
7     0.00       NA        1      4 2.944439     1       2       1        1       0     415      0       2
8    45.00 3.806662        1     10 4.330733     0       4       0        1       1     880      1       5
9    28.50 3.349904        1     31 5.111988     0       3       1        1       1     505      1       7
10    0.00       NA        1     22 4.276666     0       2       1        1       0    1155      0      21
11    0.00       NA        0      5 4.828314     1       1       0        0       0       0      0      10
```

Control Group and Test Group …in R

```
> Testgroup<-sample(tel[index == 2,],)
> head(Testgroup,10)
   wireten longmon wireless  logtoll  logwire cardten callwait address employ wiremon age tollfree marital
2   380.35    4.40        1 3.032546 3.575151     125        1       7      5    35.7  33        1       1
4     0.00    9.45        0       NA       NA       0        0      12      0     0.0  33        0       0
5     0.00    6.30        0       NA       NA       0        0       9      2     0.0  30        0       1
6     0.00   11.80        0 2.957511       NA     570        1      17     16     0.0  39        1       0
12    0.00    7.10        0 3.091042       NA     145        1      14     15     0.0  35        1       0
18    0.00    6.65        0 2.917771       NA       0        1       3      6     0.0  48        1       0
19    0.00    1.05        0       NA       NA       0        0       3      3     0.0  24        0       0
20  910.10    6.70        1       NA 3.645450     610        1       3      2    38.3  29        0       1
21   78.20    3.75        1       NA 2.928524       0        1       7      1    18.7  30        0       0
22    0.00   20.70        0       NA       NA    1505        0      17     24     0.0  52        0       1
   reside ebill ed forward equipmon longten region equip retire gender tollten equipten cardmon callcard churn
2       6     0  5       1      0.0   42.00      3     0      0      0  211.45      0.0   15.25        1     1
4       1     0  2       0      0.0  288.80      2     0      0      1    0.00      0.0    0.00        0     1
```

Control Group and Test Group …in R

```
> # checking for randomness
> dim(Controlgroup)
[1] 704  42
> dim(Testgroup)
[1] 296  42
> tabc<-table(Controlgroup$ed)
> prop.table(tabc)

         1          2          3          4          5
0.22301136 0.28125000 0.19176136 0.23295455 0.07102273
> tabt<-table(Testgroup$ed)
> prop.table(tabt)

         1          2          3          4          5
0.15878378 0.30067568 0.25000000 0.23648649 0.05405405
```

Decision Trees … in R

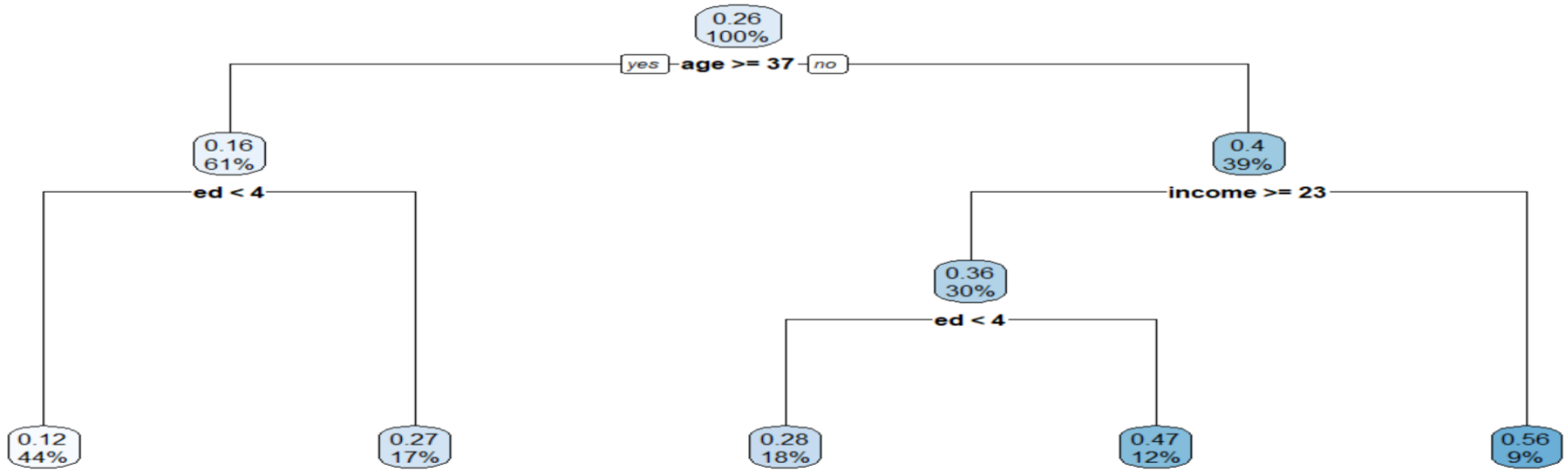
- Construction of decisiontree
- dtree = rpart(churn~age+ed+income,data = Controlgroup,method = "class")

```
> dtree = rpart(churn~age+ed+income,data = Controlgroup)
> sumamry(dtree)
Error in sumamry(dtree) : could not find function "sumamry"
> summary(dtree)
Call:
rpart(formula = churn ~ age + ed + income, data = Controlgroup)
  n= 704

          CP nsplit rel error    xerror       xstd
1 0.07052337      0 1.0000000 1.0026573 0.04200859
2 0.01557651      1 0.9294766 0.9340421 0.04156947
3 0.01459128      2 0.9139001 0.9727988 0.04334230
4 0.01263651      3 0.8993088 0.9759237 0.04436261
5 0.01000000      4 0.8866723 0.9739124 0.04483162
```

- this is not much clear let us observe it in the form of a Diagram

Graphical form of a Decision Tree …in R

- > install.packages('rpart.plot')
- > library(rpart.plot)
- > rpart.plot(dtree)



Interpretation of a Graph in Decision tree …in R

- Interpretation of the Graph
- There are .26 or 26% of the customers who are churning
- First rule is based on age , if age is >= 37 years ; the chance of churning is .16 or 16 % are churning ; and in the data there are 61 % of the people who are having age less greater than or equals to 37 years.

  If the age is less than 37 years ; the chance of churning is .4 or 40% of the people are churning ; and in the data 39 % of the customers are of age less than 37 years.

Interpretation of a Graph in Decision tree …in R

- Interpretation of the Graph
- Further the age is grater than or equals to 37 years and Level of education is less than 4 than the chance of churning is 12 % and there are 44 % of the people are present.
- If Level of education is more than or equals to 4 then the chance of churning is 27 % and there are 17 % of the people are present.
- On the similar lines the decision tree can be read.

Predicting … in R

- Predicting on whether the customer is churning or not making use of the rules of decision tree.
- We developed the decision tree rules based on the Control Group which has 70% of the records.
- Let us predict the churning behaviour on Test Group making use of the rules constructed.
- install.packages('caret')
- library('caret')
- # caret is Classification And REgression Training

Predicting … in R

- Let us predict the churning behaviour in the test group ;
- > pre<-**predict**(dtree,newdata = Testgroup,type = "class")
- Let us examine the out put

```
> pre
  1   3   7   8   9  10  11  13  14  15  16  17  25  26  28  30  33  34  35  36  37  39
  0   0   1   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
 40  41  42  43  44  45  46  48  49  51  52  53  54  58  59  60  61  62  63  64  65  66
  0   0   0   0   1   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
 67  70  71  75  76  79  80  81  82  83  84  86  87  90  91  92  93  94  95  96  98  99
  0   0   0   0   0   0   0   0   0   0   0   0   1   0   0   0   0   0   1   1   0   0
```

- Let us represent the predicted in terms of a table
- > table(pre)
- pre
- 0 1
- 271 25

Predicted values in table form …in R

- Let us check the percentages
- tabp<-table(pre)
- > prop.table(tabp)
- pre
- 0 1
- 0.91554054 0.08445946
- The percentage of churning in the control group

Predicted values in table form …in R

- > table(churn)
- churn
- 0 1
- 523 181
- > tabc<-table(churn)
- > prop.table(tabc)
- churn
- 0 1
- 0.7428977 0.2571023

Confusion Matrix …in R

- **Confusion Matrix** – the matrix which will give a comparison between actual and predicted values.
- > table(churn,pre)
- pre
- churn 0 1
- 0 191 12
- 1 80 13

	Yes	No	
Yes	191	12	203
No	80	13	93
	271	25	

	Yes	No	
Yes	0.940887	0.059113	1
No	0.860215	0.139785	1
	1.801102	0.198898	
	Correct clasification		0.540336
	Wrong Clasification		0.459664