R Programming for Data Science

Mrs.Rohini Sudhir Patil[1], Mrs. Kavita Ganesh Kurale[2]

[1]HOD, Computer Engineering Dept., Sant Gajanan Maharaj Rural Polytechnic, Mahagaon, Maharashtra, India
[2]Sr.Lecturer, Computer Engineering Dept., Sant Gajanan Maharaj Rural Polytechnic, Mahagaon, Maharashtra, India

---***---

Abstract – *Data science is a process of extracting knowledge from different structure and unstructured data using algorithms, mathematical methods. Data Science is concept related to big data, machine learning, artificial intelligence and data mining. Data Mining consists of data in statistics format which is extracted from useful information of data. Difference between Data mining and R programming is that, data mining is a concept related to the large data set while R is object oriented programming language to process such big amount of data sets. Everything stored in Object form. This object consists of information related to object and functions which are required to process that data. While considering the applications of R programming it includes finance, banking, healthcare, social media, manufacturing, E-commerce etc. Advantage of R is it is fee of cost and it provides huge number of packages and libraries that contain mathematical, statistical and other methods. R is very simple language which provided powerful tool for data mining and data processing. These tools are applicable in almost each and every filed of research.*

Key Words: **Data Science, Big Data, Data Mining, Artificial Intelligence**

1. INTRODUCTION

R is flexible programming language used for statistical operations. R is open source programming language and it is available freely for any kind of operating system. Unique and beneficial aspect of R is it is user-contributed Software. There are huge number of users who have contributed statistical methods required for statistical analysis. One of advantage of using R is statistical analyses is done by series of steps. Here intermediate results are being stored in 34-Osborne , where the objects are later “interrogated” for the information of interest .While other programs e.g. SAS and SPSS prints large amount of output on the monitor. As results stored in object form, these results can use later for another analysis. There are lots of applications available in real world using R. As IoT devices are increasing day by day they are creating data in terabytes and this terabyte data is used for making certain decisions. Here comes data science where Programming language line R can be used. Data Scientist can gather real time data, perform predictive and statistical analysis, create data in visual format and provide actionable results to stakeholders. Another use of R in predictive analytics and machine learning. It provides various package for common ML tasks like linear and non-linear regression, linear and non-linear classification, decision trees etc. Nowadays researcher implementing machine learning algorithms in fields like retail, marketing, health care, entertainment, finance etc.

1.1 ROLE OF R IN DATA SCIENCE

As R is open Source, it is freely available. R's interfaces allows to integrate with other systems and applications.

R is Programming language which provides objects, functions and operators which allows users to create visual based data, model data and explore it.

R is used in data analysis field where in data science it is used to handle, analyze and store data.

R is an environment used for statistical analysis of data. R provides various statistical and graphical capabilities.

The R Foundation provides notes that can be helpful for classification, statistical tests, clustering, and linear and nonlinear modeling.

1.2 R PACKAGES FOR DATA SCIENCE

1. DBI package is used for basic communication between R and database management systems.

2. Stringr are tools that work with character strings and regular expressions.

3. Dplyr offers functions for summarizing, connecting and rearranging datasets.

4. Lubridate facilitates working with dates and times across various periods.

5. Ggplot2 is used to produce visually appealing plots and graphics.

6. Roll package used for three-dimensional, interactive visualizations with R in which we can rotate and zoom in on specific parts of visualization.

7. RandomForest is used in a machine learning.

8. Caret is used for regression models and training classification.

9. Shiny is for data science that helps to create web applications.

10. Xtable this package provides code in HTML or latex form that can be paste in R for final document.

11. Ggmap is used to download map areas form Google Maps and insert them into ggplots.

12. Xts includes tools used for working with time series datasets.

13. XML assists in working with XML documents.

14. Httr assists in working with http connections.

15. Devtools helps you create your own R package.

2. R DATA TYPES

While writing any program we require to store information and for that purpose we need variables. Variables are nothing but reserved memory locations used to store values.

We required to store data in different format like character, integer, Floating point, Boolean etc. Depending upon type of variable memory is allocated. Like other programming languages like C, C++ and Java in R programming there is no requirement of declaring data type. Here variables are assigned with R Objects. There are various types of R-Objects are available. Some are as follows-

2.1 Vector-

When we want to create vector with more than one element, we should use c() function which means to combine the elements into a vector.

```
# Create a vector.
R>fruit <- c('apple','grapes',"banana")
R>print(fruit)
Output-[1] "apple" "grapes" "banana"

# Get the class of the vector.

R>print(class(fruit))

Output-[1] "character"
```

2.2 List-

List- List contains different types of data like vectors, characters, strings, numbers, logical values, another list, matrix, function etc. List is created using list() function.

```
# Create a list.
R>list1 <- list(c(10,4.5,'xyz'),100.50,sin)

# Print the list.
R>print(list1)

 Output-

 [[1]]

 [1] "10" "4.5" "xyz"

 [[2]]

 [1] 100.5

 [[3]]

 function (x) .Primitive("sin")
```

2.3 Matrix-

Matrix is a two dimensional data structure in R programming where it is created using matrix() function. Matrix is similar to vector but it contains the extra attribute i.e. dimension.

```
# Create a matrix.
R>matrixEX = matrix( c('a','b','c','1','2','3'), nrow = 2, ncol = 3, byrow = TRUE)
R>print(matrixEX)

 Output-

 [,1] [,2] [,3]

 [1,] "a" "b" "c"

 [2,] "1" "2" "3"
```

2.4 Arrays

In R matrices store data in two dimensions but arrays can be used to store any number of dimensions. 'dim' attribute of array function is used to create required number of dimension. .In the example below we create an array with three elements which are 3x3 matrices each.

```
# Create an array.
R>arrayEx <- array(c('mango','apple','grapes'),dim = c(3,3,2))
R>print(arrayEx)

 Output-

 , , 1

  [,1] [,2] [,3]

 [1,] "mango" "mango" "mango"

 [2,] "apple" "apple" "apple"

 [3,] "grapes" "grapes" "grapes"

 , , 2

  [,1] [,2] [,3]

 [1,] "mango" "mango" "mango"

 [2,] "apple" "apple" "apple"

 [3,] "grapes" "grapes" "grapes"
```

2.5 Factors

For creating factor type of data we require vector. It is used to store vector values as well as distinct values of elements in the vector as labels. The labels values are character form only. Factors are useful for statistical modelling
factor() function is used to create factor and nlevels() functions returns the count of levels.

```
# Create a vector.
R>week_days <-
c('sunday','tuesday','friday','tuesday','sunday','friday','sunday
')

# Create a factor object.
R>factor_week <- factor(week_days)

# Print the factor.
R>print(week_days)
R>print(nlevels(factor_week))
```

Output-

```
[1] "sunday" "tuesday" "friday" "tuesday" "sunday" "friday"
"sunday"

[1] 3
```

2.6 Data Frames

Data frame is used to store different modes of data means first column can contain data in numeric form while second can be logical and third can be character. Mainly data frames are data objects in tabular form.
data.frame () function is used to create Data Frames.

```
# Create the data frame.
R>dataFrameDemo <- data.frame(
 gender = c ("Male", "Male","Female"),
 height = c (154, 170, 163.2),
 weight = c (80,91,72),
 Age = c (41,36,25)
)
R>print(dataFrameDemo)
```

Output-

```
gender height weight Age

1 Male 154.0 80 41

2 Male 170.0 91 36

3 Female 163.2 72 25
```

3. CHARTS AND GRAPHS IN R

R Programming language provides huge libraries to create charts and graphs.

3.1 Pie-chart-

A pie-chart is used to represent values as slices of a circle with different colors. The slice is displayed by using labels.

Syntax for creating basic pie chart is-

pie(x, labels, radius, main, col, clockwise)

Where, x is a vector which contains values in numeric form.

-labels is used to provide description to the slices.

-radius indicates the radius of the pie chart. (value between −1 and +1).

-main is used for assigning the title of the chart.

-col is used for the color palette.

-clockwise is a logical value represents whether the slices are drawn clockwise or anti clockwise.

Ex.

```
# Create data for the graph.

R>x <- c (25, 60, 12, 48)

R>labels <- c ("Delhi","Kolkata","Pune","Mumbai")

R>piepercent<- round(100*x/sum(x), 1)

# Give the chart file a name.

r>png (file = "city_percentage.jpg")

# Plot the chart.

R>pie(x, labels = piepercent, main = "City pie chart",col =
rainbow(length(x)))

R>legend("topright", c("Delhi","Kolkata","Pune","Mumbai"),
cex = 1,

 fill = rainbow(length(x)))

# Save the file.

R>dev.off()
```

Output-

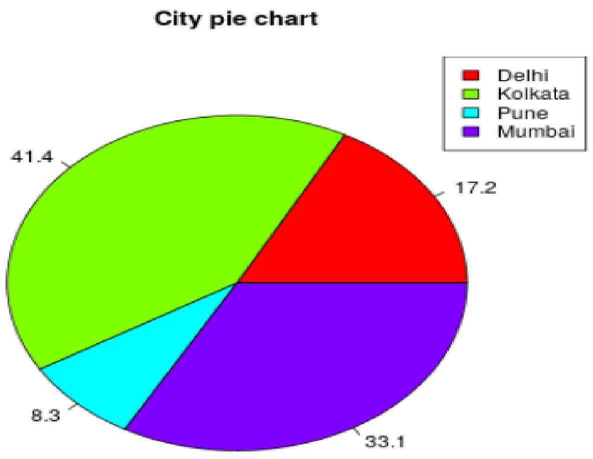

Fig -1: Pie chart example

3.2 Bar-chart-

A bar chart is used to represent data in rectangular bars. Length of bar is proportional to value of data or variable. barplot() function is used to create bar charts. There are two options available to draw charts i.e horizontal and vertical. There is facility to give different color for each bar.

Syntax for bar chart is-

barplot(H,xlab,ylab,main, names.arg,col)

Where, H is used to provide numeric values in bar chart.

xlab- is the label for x axis.

ylab- is the label for y axis.

main- is the heading of the bar chart.

names.arg- is a vector of martix of names appearing under each bar.

col- is used to assign colors to the bars.

Ex.

```
# Create the data for the chart

R>S <- c(100,45,50,30,25)

R>D <- c("June","July","August","September","October")

# Give the chart file a name

R>png(file = "barchart_revenue.png")

# Plot the bar chart

R>barplot(S,names.arg=D,xlab="Month",ylab="Revenue",col
="Yellow",

R>main="Revenue chart",border="red")

# Save the file

R>dev.off()
```

Output-

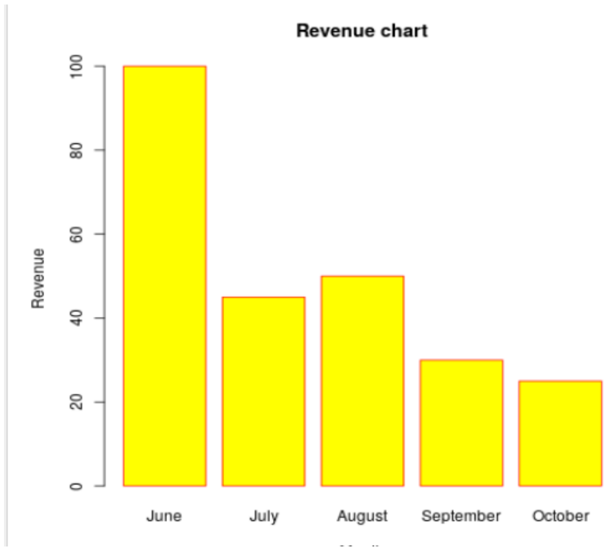

Fig -2: Bar chart example

3.3 Boxplot-

Data can be well distributed with the help of Boxplots. Boxplots divides data set into three parts named quartiles. This graph is used for representing minimum value, maximum value, median etc. from the data set. It is also used for comparison of data which is distributed throughout in data set.

Function boxplot() is used to create Boxplots.

Syntax for Boxplot is-

boxplot(x, data, notch, varwidth, names, main)

Where, x- is a vector or a formula to be calculated.

data - is the data frame.

notch - is a logical value. Always Set as TRUE to draw a notch.

varwidth- is a logical value. True value should be set to draw width of the box proportionate to the sample size.

names- are used to give group labels that will be printed under each boxplot.

main- is used to give a title to the graph.

Ex.

```
# Give the chart file a name.
```

R>png(file = "boxplot.png")

# Plot the chart.

R>boxplot(mpg ~ cyl, data = mtcars, xlab = "Number of Cylinders",

R> ylab = "Miles Per Gallon", main = "Mileage Data")

# Save the file.

R>dev.off() Output-

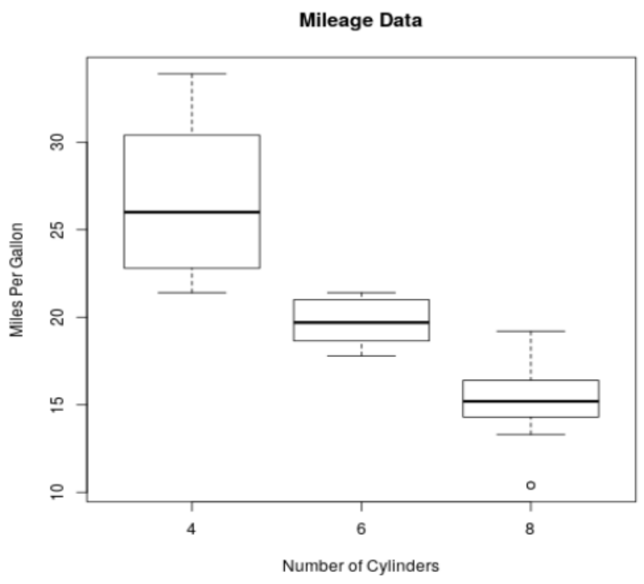

Fig -3: Boxplot example

3.4 Scatterplot-

Scatterplots are used to show multiple points plotted in the Cartesian plane. Two types of variables are used .One variable is selected as horizontal axis and one as vertical axis.

The function plot() is used to draw simple scatterplot.

Syntax for Scatterplot is-

plot(x, y, main, xlab, ylab, xlim, ylim, axes)

Where, x- is used to display values of data set in horizontal coordinates.

y- is used to display values of data set in vertical coordinates.

main- is used to give heading of the graph.

xlab -is used give label in the horizontal axis.

ylab - is used to give label in the vertical axis.

xlim - is the limits of the values of x axis used for plotting.

ylim - is the limits of the values of y axis used for plotting.

axes - indicates whether both axes should be drawn on the plot or not.

Ex. Here the data set "mtcars" is used which is available in the R environment to create a basic scatterplot. Consider the columns "wt" and "mpg" in mtcars.

R>sample <- mtcars[,c('wt','mpg')]

R>print(head(sample))

Above code will print-

wt mpg

Mazda RX4 2.620 21.0

Mazda RX4 Wag 2.875 21.0

Datsun 710 2.320 22.8

Hornet 4 Drive 3.215 21.4

Hornet Sportabout 3.440 18.7

Valiant 3.460 18.1

The following script will create a scatterplot graph for the relation between wt (weight) and mpg(miles per gallon).

# Get the input values.

R>sample <- mtcars [,c('wt','mpg')]

# Give the chart file a name.

R>png(file = "scatterplot.png")

# Plot the chart for cars with weight between 2.5 to 5 and mileage between 15 and 30.

R>plot(x = sample$wt,y = sample$mpg,

R> xlab = "Weight",

r> ylab = "Milage",

R> xlim = c(2.5,5),

R>ylim = c(15,30),

R> main = "Weight vs Milage"

)

# Save the file.

R> dev.off()

Output-

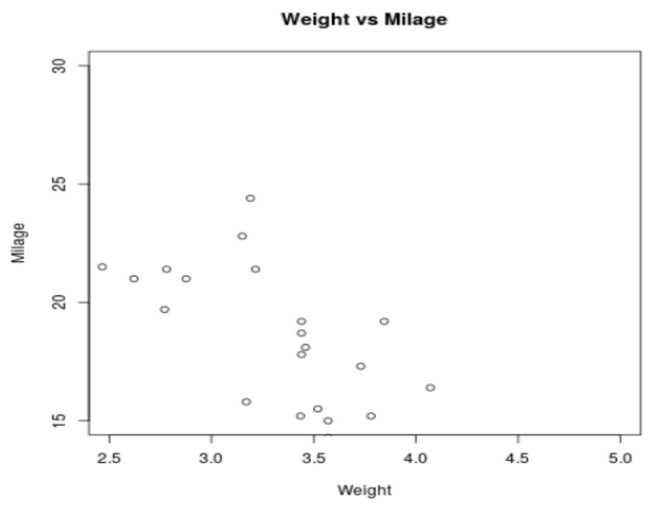

Fig -4: Scatterplot example

4. CONCLUSION

R Programming is rapidly growing in the field of data science. Data science requires statistical tools add methods that are used for data analysis and data interpretation. So R programming is rapidly growing. R Language is open source and easy to access freely. It includes machine learning. It provides huge library of packages. For these reasons today world is working with data science and R programming. It will benefit for developer and industry too.

REFERENCES

1. R Programming for data science by Roger D. Peng, 2015-07-20

2. R language in Data Mining Techniques and Statistics, January 2013,American Journal of Software Engineering and Applications 2(1):7

3. A Survey on Data Science Technologies & Big Data Analytics, T. Giri Babu Dr. G. Anjan Babu Research Scholar, Department of Computer Science Associate Professor, Tirupati, Andhra Pradesh, India, ISSN: 2277 128X