Data Analytics using R

Dr Akhter Raza

Intro of S—Programming

S was developed by John Chambers and others at the old Bell Telephone Laboratories, originally part of AT&T Corp. S was initiated in 1976 as an internal statistical analysis environment

In 1988 the system was rewritten in C and began to resemble that we have today. The book Statistical Models in S by Chambers and Hastie documents the statistical analysis functionality. Version 4 of the S language was released in 1998 and is the version we use today.

Intro of R— Programming

The R came a bit after S. One key limitation of the S was its availability in commercial package, S-PLUS. In 1991, R was created by Ross Ihaka and Robert Gentleman in the Department of Statistics at the University of Auckland. In 1993 the first announcement of R was made to the public. Ross's and Robert's experience developing R is documented in a 1996 paper in the Journal of Computational and Graphical Statistics

Design of R

R system is available from R Archive Network, also known as CRAN. CRAN also hosts many add-on packages that can be used to extend the functionality of R.

The R system is divided into 2 conceptual parts:

1. “base” R system that you download from CRAN
2. Everything else.

Base R

The “base” R system contains the base package which is required to run R and contains the most fundamental functions.

The other packages contained in the “base” system include ***utils, stats, datasets, graphics, grDevices, grid, methods, tools, parallel, compiler, splines, tcltk, stats4.***

Other R packages

There are also “Recommended” packages: ***boot, class, cluster, codetools, foreign, KernSmooth, lattice, mgcv, nlme, rpart, survival, MASS, spatial, nnet, Matrix.***

There are over 4000 packages on CRAN that have been developed by users and programmers around the world.

R functionalities

- R is case sensitive
- R is open source
- Available for both windows and MAC
- Packages can be downloaded from CRAN

  https://cran.r-project.org/
- R manuals can be downloaded from cran
- R can be installed on USB

R Basics

```
> data(cars)

> str(cars)

> head(cars)

> plot(cars)
```

Histogram

```
> x11()   # new graphical window

> hist(cars$speed)
```

Installing packages

> install.packages(“name of package”)

Or use menu item to install the package both options are correct and same

> tools > install packages

Loading package into R session

require(name of package)

or

library(name of package)

No quotes is needed here in package name as we have in intall.package

Require and library commands are identical

Saving and Closing R

- Saving workspace

  All variables and their data is to be saved in a file

  ```
  q()   #quit from R session
  ```

- Save history   all typed command in R console are saved in a file

R data structures

Vectors and factors

creating vectors using combine function

```
gpa <- c(3.45, 2.79, 3.22, 2.98)
```

checking structure of gpa vector

```
> str(gpa)
```

accessing individual elements (like 3rd element)

```
> gpa[3]
```

Handling data

Accessing range of elements (1st to 3rd)

```
> gpa[1:3]
```

Creating character vectors using combine

```
> fst_name <- c(“ajmal”,”zafar”,”Saem”,”Jea”)
```

Checking structure of gpa vector

```
> str(gpa)
```

Handling data

Accessing range of elements (1st to 3rd)

```
> gpa[1:3]
```

Creating character vectors using combine

```
> fst_name <- c(“ajmal”,”zafar”,”Saem”,”Jea”)
```

Checking structure of fst_name vector

```
> str(fst_name)
```

Handling data

Creating Boolean vectors using combine

```
> pass <- c(true, false,  true, true)
```

Checking structure of pass vector

```
> str(pass)
```

Creating factor vector

Creating gender

```
gender <- c(“M”, “M” , “M”  , “F” )
```

Checking structure of pass vector

```
str(gender)
```

Gender is a character to convert it into factor

```
gender <- factor(gender)
```

it overwrite the gender into factor

Creating factor vector

Defining levels into a factor

```
gender <- factor(gender, levels=c(“M”,”F” ))
```

Checking structure of gendere vector

```
str(gender)
```

Creating a dataframe

To check how many vectors are defined in memory using list objects

```
ls()
```

Creating a data frame

```
df1 <- data.frame(fst_name, gender, gpa, pass)
```

```
str(df1)
```

Creating a dataframe

To access individual elements of a data frame

```
df1[row, col]
```

Command will show 2nd row 4th col

```
df1[2,4]
```

Following command will show entire 2nd row

```
df1[2,]
```

Data frame

Data frame change character var into factor. As you can check the fst_name variable which was actually into character data frame automatically convert it into factor. So we have to change it into character explicitly

```
df1 <- data.frame(fst_name, gender, gpa, pass,
         stringsAsFactors = FALSE)

str(df1)
```

Exploring Data into R

Exploring Data V9

copy your csv file into current folder

load the csv file using read.csv command

```
df3 <- read.csv("usedcars.csv")
str(df3)
```

shows structure of usedcars.csv data

```
head(df3)
```

shows first 6 rows of usedcars.csv data

```
tail(df3)
```

show last 6 rows of usedcars.csv data

Exploring Data into R

# again read this file with one option

```
df4 <- read.csv("usedcars.csv", stringsAsFactors = FALSE)
```

# compare structure of df3 and df4

```
str(df4)
```

we can see that now model, color and transmission variables are not factors now they are characters

Creating Subsets of Data

```
yellow_cars <- subset(df4, color %in% "Yellow")
```

Creates subset of yellow cars

```
high_mileage <- subset(df4, mileage > 100000)
```

Creates subset of cars whose mileage> 100000

Subset of Manual cars

```
auto_cars <- subset(df4, transmission %in% "AUTO")
```

```
man_cars <- subset(df4, transmission %in% "MANUAL")
```

```
head(df4$color)
```

Data frames elements

```
df4$color[3]
```

Shows first 3 values of color variable

```
df4$color[3:5]
```

Color of cars from 3rd to 5th row

```
table(df4$color)
```

Frequency table of color variable

Data frames elements

```
df4[ , ]
```

shows all rows and all columns it is similar as df4

```
df4[1, ]
```

shows first rows and all colums it is similar as df4

```
df4[c(1,3,5) , 3 ]
```

shows only price of first, third and fifth car

Data frames elements

```
df4[c(1,3,5) , "price" ]
```

3rd column name is price we can use column name instead of its serial number

```
df4[c(1,3,5) , c(3,5) ]
```

shows price and color columns of first, third and fifth car

Data frames elements

```
df4[c(1,3,5) , c("price", "color" )]
```

shows price and color columns of first, third and fifth car

```
df4[c(1,3,5) , -c(3,5) ]
```

shows except 4rd and 5th columns of first, third and fifth car

Frequency table of transmission

```
table(df4$transmission)
```

how many cars having automatic and manual transmission

```
table(df4$transmission)/length(df4$transmission)
```

Shows probabilities/proportion of each type of car

Pie and bar charts

```
pie(table(df4$transmission))
```

making pie chart of transmission variable

```
barplot(table(df4$transmission))
```

Create bar chart of transmission variable

Univariate Descriptive measures

```
mean(df4$mileage)
median(df4$mileage)
var(df4$mileage)
sd(df4$mileage)
range(df4$mileage)
hist(df4$mileage)
boxplot(df4$mileage)
boxplot(df4$mileage,horizontal = TRUE)
```

Univariate Descriptive measures

```
plot(mileage ~ transmission, data = df4)
```

Create box 2 box plots of mileage of auto and manual cars seperately

```
tapply(df4$mileage, df4$transmission, mean)
tapply(df4$mileage, df4$transmission, sd)
```

Find mean and standard dev of mileage variable seperately for manual and automatic cars

Visualization of data

```
hist(df4$price)
```

Histogram of price variable

```
boxplot(df4$price)
```

Boxplot of price variable

```
boxplot(df4$price,horizontal = TRUE)
```

Boxplots drawn horizontalay

```
plot(df4$mileage, df4$price)
```

Scatterplot between mileage and price variables

Visualization of data

```
plot(price ~ mileage, data=df4,
        pch=as.integer(transmission))
```

pch parameter is the plotting character on the graph using triangles and circles for plotting depending on transmission

Visualization of data

Scatterplot of price and mileage variable seperated for transmission variable legend and color parameters are used

```
plot(price ~ mileage, data=df4,
        pch=as.integer(transmission),col=as.integer(transmission)+2)

legend("topright",legend = c("AUTO","MANUAL"),pch=c(1,2),col=c(3,4))
```

Multivariate summaries

# V-13

plotting scatter matrix

```
pairs(df4)
```

Returns error can anyone explain why?

Df4 contains string variables that must be eliminated to draw matrix scatter

```
pairs(df4[, -c(2,5,6)])
cor(df4[, -c(2,5,6)])
```

Multivariate summaries

```
aggregate(price ~ transmission, data=df4,
    FUN=mean)
```

Returns mean price of cars seperately by transmission

```
tapply(df4$price, df4$transmission,
    FUN=mean)
```

the same thing can be done by using tapply function

Multivariate summaries

```
aggregate(price ~ transmission + color,
          data=df4, FUN=mean)
```

aggregate can be used transmission and color of car basis

```
table(df4$transmission, df4$color)
```

creating contingency table using transmission and color. Crosstabulation b/w two categorical variables

Multivariate summaries

```
table(df4$transmission,
          df4$color)/length(df4$transmission)
```

Returns proportions

```
round(table(df4$transmission,
          df4$color)/length(df4$transmission),1)
```

Returns probabilities/proportions rounded to one decimal place

Multivariate summaries

```
plot(price ~ color, data= df4)
```

Draw box plot of prices using different colors

This command will not work b/c color variable is not factor now we convert color variable into factor

```
df4$color <- factor(df4$color)
plot(price ~ color, data= df4)
```

Questions?