Assignment NO -2

Date : / /20

Title - Data Types, Graphics & control structure in R

Problem statement :- To study & practice various commands using different data types, Graphics and control structures on R tool and study and practice of various control structures.

Pre - Lab :- A basic understanding of any of the programming language will help in executing simple R commands & control structures.

Theory :-

A. Data Types in R :-

In contrast to other programming language like C and java in R, the variables are not declared as some data type. The variable are assigned with R-objects & data type of R-object becomes the data type of variables. The frequently used ones are -

- vectors
- Lists
- Matrices
- Arrays
- Factors
- Data Frames

The simplest of these objects in the vector object and there are six data types of these atomic vectors, also termed as six classes of vectors.

Pooja

Data Types	Example
Logical	TRUE, FALSE
Numeric	12.3, 5, 999
Integer	2L, 34L, 0L
Complex	3+2i
character	'a', "good", "TRUE"
Raw	"Hello" is stored as 48 65 6C 6C 6f

1. Vectors :-

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

```
# create a vector
apple <- c('red', 'green', 'yellow')
print(apple)
# Get the class of the vector
print(class(apple))
```

2. List :

A list is an R-object which can contain many different types of elements inside it like vectors functions and even another list inside it.

```
# create a list
list1 <- list(c(2,5,3), 21.3, sin)
# print the list
print(list1)
```

Date : / / 20

3. Matrices -
A matrix is two dimensional rectangle dataset. It can created using a vector input to the matrix function.

```
# create Matrix
M = matrix (c ('a', 'b', 'c', 'b', 'a'), nrow = 2, ncol = 3,
                                        byrow = TRUE)

Print (M)
```

4. Arrays -
While matrices are confined to two dimensions, arrays can be of any number of dimensions. The array function takes a dim attribute which create the required number of dimension. In the below example we create an array with two element which are 3x3 matrix each.

```
#create an array
a ← array (c ('green', 'yellow'), dis = c (3,3,2))
print (a)
```

5. Factors :-
Factors are the r-object which are created using a vector. It stores the vector along with the distnict values of elements in vector as labels. The labels are always character irrespective of whether it is numeric or character or Boolean etc. in the put. vector. They are useful in statical modelling.

```
#Create vector.
apple-colors ← c ('green', 'green', 'yellow', 'red', 'green')
```

Pooja

Date : / / 20

```
#create a factor object
  factor_apple ← factor (apple_colors)
#print the factor
 print (factor_apple)
 print (nlevels (factor_apple))
```

6. Data Frames :-

Data frames are tabular data objects. Unlike a matrix in data frame each column can contain different modes of data. The first column can be numeric value while second column can character & third column can be logical. It is a list of vectors of equal length.

```
#create the data frame.
BMI ← data_frame(
gender= c("Male","Male","Female"),
height = c(152, 171.5, 165),
weight = c(81,93,78),
Age = c(42,38,26)
)
```

On vectors, lists and matrices airthemetic function can be performed using basic

7. strings :-

Any value written within a pair of single quote or double quotes in R is treated as a string Identranlly R stores every string within double quotes, even when you create them with single quote.

Pooja

Date : / / 20

Rules Applied in String Constructions :-

- The quotes at the beginning & end of a string should be both double quotes or both single quotes. They cannot be mixed.
- Double quotes can be inserted into a string starting and ending with single quotes.
- Single quote can be not be inserted into a string starting & ending with double quotes.
- Single quote can not be inserted into a string starting & ending with single quotes.

```
b <- "start and end with double quotes"
print(b)
```

String Manipulation functions used in R :-

Concatenation Strings - paste() function
Formating numbers & strings - format() function.
Counting number of characters in string - nchar()
changing the case - toupper() & tolower function
Extracting parts of a string - substring() function

B) Graphics

R programming language has numerous libraries to create charts & graphs.

1. Pie-chart :-

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 can represent in chart.

Pooja

Date : / /20

In R the pie-chart is created using the pie() function which takes positive numbers as a vector input.

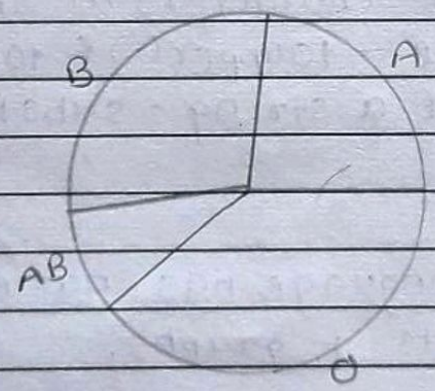
The basic syntax for creating pie chart using R-

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

Following is the description of parameter used-

- x is vector containing numeric values used in pie chart
- labels is used to give description to slices
- radius indicate the radius of circle of pie-chart (value between -1 and +1)
- main indicate the title of chart.
- col indicates the color palette
- clockwise is a logical value indicating if the slice are drawn clockwise or anti-clockwise.

A very simple pie-chart is created using just input vector and labels.



2. Bar charts:

A bar chart represent data in rectuangular bars with length of bar proportional to value of the variable. R uses the function barplot() to create bar charts to R can draw both vertical

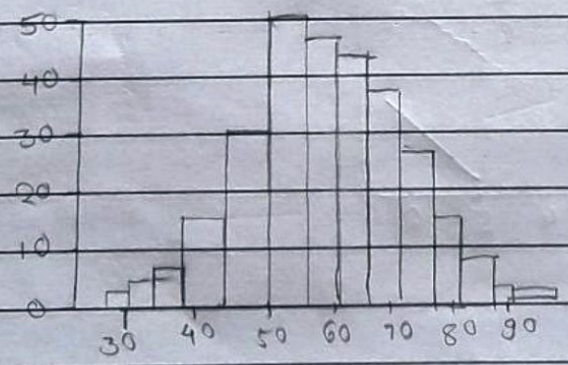
Pooja

Date : / / 20

is it groups the values into continuous range. Each bar in histogram represent the height of number of values present in that ranges.
R creates histogram using hist() function. This function takes a vector as an input and uses some more parameters to plot histogram. The basic syntax for creating a histogram is -

hist(v, main, xlab, xlim, ylim, breaks, col, border)

- v - is vector containing numeric value used
- main includes title of the chart.
- col is used to set color of bars
- boarder is used to set border color of each other.
- xlab is used to give description of x-axis.
- ylab is used to give desc
- xlim is used to specify the range values on x-axis
- ylim is used to specify the range values on y-axis.
- breaks is used to mention width of each other.



5. Line Graphs :-
A Line chart is graph that context a series of points by drawing line segments between them. These points are ordered in one of their

Pooja

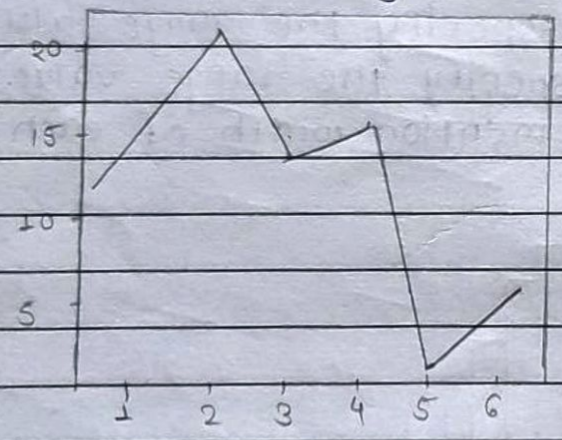
Date : / / 20

(usually the x-coordinate) value. Line carts are usually used in the identifying the trends in data The plot() function in R is used to create the line graph.

The basic syntax to create a line chart in R is plot(v, type, col, xlab, ylab)

Following is the description of parameters-

1. v is vector containing the numeric values.
2. type takes the value "p" to draw only the points "I" to draw only lines and "o" to draw both points and lines.
3. xlab is label for x-axis
4. ylab is label for y-axis.
5. main is the Title of the chart.
6. col is used to give colors to both point & lines.



6. Scatterplots:

Scatterplots show many points plotted in the Cartesian plane. Each point represents the values of two variables. One variable is chosen in horizontal axis. & in vertical axis..

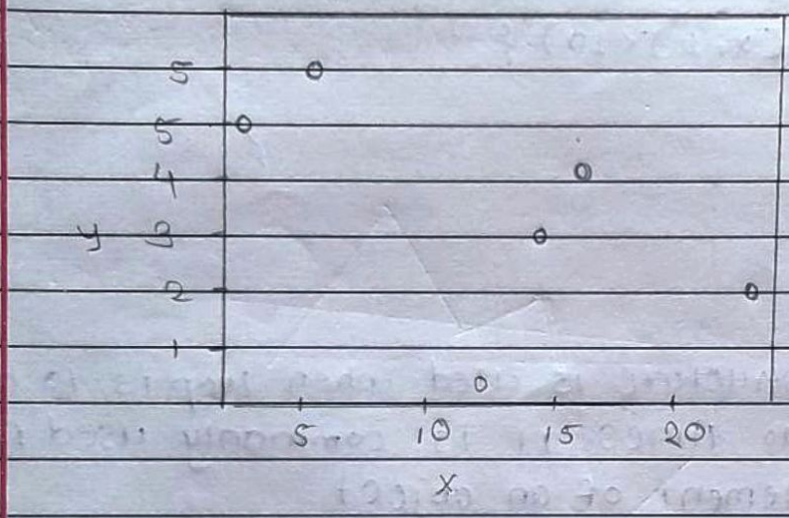
The basic syntax for creating scatterplot in R

Pooja

Date : / / 20

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

- x is data set whose values are horizontal coordinate
- y is data set whose values are vertical coordinate
- x lab is the label in horizontal axis.
- y lab is the label in vertical axis.
- x lim is limits of values of x used for plotting.
- y lim is limits of values of y used for plotting
- axes indicates whether both axes should be drawn



C. Control Structures in R :-

As the segments, a control structure controls flow of code written inside a function. A function is a set of multiple commands written to automate a repetive a coding task.

1. If else :

```
if (condition) {
    # do something
} else {
    # do something
}
```

Pooja

Date : / / 20

2. Vectorization with if else -

```
ifelse (x<-=10, "x less than 10", "x greater than 10")
```

3. Other valid ways of writing if else :-

```
if (sample (x,1) <10) {
        y<-5
} else {
        y<-0
}
y<- if (sample (x,1)<10) {
        5
} else {
    0
}
```

4. For :- This structure is used when loop is to be executed first No. times. It is commonly used for iterating over element of an object.

```
for (search condition>) {
  #do something
}
```

Examples -

```
for (i in 1:10) {
 print(i)
}
 x<- c("apples", "oranges", "bannas", "strawberries")
 for (i in x) {
   print(x[i])
}
```

Pooja

Date : / / 20

```
for (i in seq(x)){
   print(x[i])
}
for (i in 1:4) print(x[i])
```

5. while :

It begins by testing a condition, executes only if the condition is found to be true.

Below is the syntax :

```
i ← 1
while (i < 10) {
  print(i)
  i ← i + 1
}
```

6. Repeat :-

A Repeat loop is used to iterate over block of code multiple number of lines. There are no condition check in repeat loop to exit the loop.

We must ourselves put a condition explicitly inside the body of loop & use the break statement to exit the loop.

```
repeat {
  statement
  }
}
```

7. Next :- A next statement is useful when we want to skip the element iteration of a loop without terminating it.

Pooja

Date : / / 20

On encountering next, the R parser skips further evaluation. Below is the syntax

```
next
```

8. Break

A break statement is used inside a loop (repeat, for, while) to stop the iteration of loop the control outside of loop. In nested looping situation, where there is loop, this statement exist from innermost loop that evaluated.

Below is syntax.

```
break
```

Post-Lab :- Students will be able to execute various R commands & use control structure in R tool / R studio for application.

Conclusion - Thus exersied, Basic syntax, Data type, Variable, operators, vectors, Lists, Matrices, Data Frames, Factors, various types of graphs & control structures taking suitable examples.

Pooja

Assignment 3

Date : / / 20

Title - Data Preprocessing in R.

Problem statement :- To study & private various commands related to preprocessing of data.

Pre-lab :- A basic understanding of the preprocessing concepts of the data is required.

Theory :-

Data Preprocessing :-

Data preprocessing is a data mining technqiue that involves transforming raw data into an understable format. Real-World data in often incomplete, inconsistent, and/or lacking in certain behaviours or trends, & is likely to contain many errors. Data preprocessing is a proven method of resolving issues.

Data goes through a piece of steps during preprocessing.

- Data cleaning
- Data integration
- Data transformation
- Data reduction
- Data discretization.

- Data cleaning :- Data is cleansed through process such as missing values, smoothing the noisy data, or resolve the inconsistencies.

1. Fill in missing values (attribute or class value)
   - Ignore the tuple: usually done when class label

Pooja

Date : / / 20

missing.
- Use the attribute mean to fill in missing values.
- Use the attribute mean for all samples belong to the same class..
- Predict the missing values by using a learning algorithm : consider the attribute with the missings value as dependent variable & run a learning algorithm.

2. Identify outliers and smooth out noisy data:-
   - Binning
     - sort the attribute values & partition them into bins.
     - The smooth by bin means, bin median or bin boundaries.
   - clustering - group values in cluster & then detect and remove outliers
   - Regression - smooth by fitting the data into regression functions.
3. Correct inconsitent data - use domain knowldge or expert decision.

Data Integration: Data with different representation are put together & conflits within the data are resolved. R objects that reside in other R objects can require a lot of typing to access For example, to refer to a variable x in dataframe df one could type df$x. This is no problem when dataframe& variable names are short.
The attach() function in R can be used to make objects within dataframes accessible in R.

Pooja

Date : / / 20

The search() function can be used to list attached object & packages.
The detach() function is used to make objects within dataframe.
Users are cautioned that if there is already a variable called x in the local workspace, issuing attach(ds), may not mean that same x reference ds$x.

Data transformation :- Data is normalized, aggregated & generliazed.
If data to different sale are being employed by the user, it is recomnded to perform a normalizat? to make the data structure comparable. This is performed by the Normalization function.
The basic syntax is :-
Normalization (Data method = c("quantile", "Fisher-Yates", "standardize", "Range", "Q", "q", "F", "f", "S", "s", "R", "r"))
where
Data - A data matrix. It is assumed the rows are corresponding with the objects.
Method - A method of normalization. Should be one of the "Quantile", "Fisher-Yates" standardize", "Range"

The method "Quantile" refers to the Quantile-Normalization widely used in osmic data. The "Fisher-Yates" normallization has a similar approach as the Quantile-Normalization but does not rely on data, just on the No. of rows in matrix.

Pooja

Date : / / 20

"Range" computes the maximum & minimum value of the matrix & determines the range. Every value is then reduced by minimum & divided by range of data matrix. The latter normalizat° will result in values between 0 & 1.

Aggregation.
It is relatively easy to collaspe data in R using one or more by variables and defined function.
The format is

aggregate (x, by, FUN)

where x is the data object to be collasped, by is a list of variable that will crossed to form the new observation, FUN is scalar fun°n used to calculate summary statistics that will make up new observation

Data Discretization - Involves the reduction of a number of values of continuous attribute by dividing range of attribute intervals.
This function implements several basic unsupervised method to convert continuous variable into a categorial variables suitable for association rule mining.
syntax is -
discretize (x, method = "interval", categorize = 3, labels=NU
ordered=FALSE, onlycuts = FALSE, ...)
where
x is numeric vector (continuous variable)

Pooja

Date : / / 20

method - discretization method, Available are: "interval" (equal interval width), "frequency", "cluster" (k means austering) & "fixed" (categories specifies interval boundries)

Categories : No. of categories or a vector with boundries (all values outside the boundries)

labels - character vector, names for categories.

ordered : logical; return only factor with ordered levels?

onlycuts : logical; return only computed interval boundries?

Experimental setup :-

Commands used in R to import data from .csv file -

1. Data From *.csv (copy-and-paste)

select the table from excelfile. Copy, go to the R Console and type :

```
mydata <- read.table("clipboard", head = TRUE,
                     seq = "\t")

summary(mydata)
edit(mydata)
```

2) Data from *.csv (interactively)

```
mydata <- read.csv(file.choose(), header = TRUE)
```

3) Data from *.csv

```
mydata <- read.csv("C:\\mydata\\mydataFiles.csv",
                   header = TRUE)
```

Pooja

Date : / / 20

R commands for Missing values:

1. is.na(df) # checks the entire data set for NAs and return
2. rowSums (is.na(mydata)) # Number of missing rows
3. colsums (is.na(mydata)) # NO. of missing per column / variable
4. rowMeans (is.na(mydata) * length(mydata)
   #NO. of missing per row
   #length = num of variable / ele
   # convert to missing data
5. mydata[mydata$age == "f", "age"] <- NA
6. mydata[mydata$age = 999, "age"] <- NA
   #The function Complete.cases() return a logical vector indicating which cases are Complete.
7. mydata [! Complete.case(mydata),]
8. # The function, na.omit () return the object with listwise deletion.
9. # Creating new dataset without missing data.
10. mydata1 <- na.omit(mydata)

R Commands to remove data conflit:

attach (ds)
detach (ds)
search ()

R Commands for normalization:

Normalization (data, method = c ("quantile", "Fisher-Yates", "standardize", "Range", "q", "q", "F", "f", "S", "s", "R", "r"))

Pooja

Date : / / 20

R Command for discretization :

discretize (x, method = "interval", categorize = 3,
labels = NULL, ordered = FALSE, onlycuts = FALSE..)

Post lab - students will be able to preprocess the data using R could further used for decision making & analysis purpose.

Conclusion - thus exiercised various commands related to preprocessing in the data in R.

Pooja