Predictive Analytics Hands-on coding, 2018

Section 1

Project Data Sets: Iris flower, Breast cancer, Credit risk

Before even starting, install R software and R studio and a folder to save your R script files on that platform

Let's start

Part one

- Assigning values to variables
- Data types, vectors, arrays, and matrices
- There are two broad sets of data types: atomic vectors and compound vectors
- There are basically five data types in R programming under the atomic vector category
- numeric or numbers, characters or strings, factors, logical, and complex
- And there are four compound data types: data frame, lists, array, and matrix
- The primary data object in R is a vector; even when we assign a single-digit number to any alphabet, it is a single element vector in R
- All data objects contain a mode and a length
- The mode determines the kind of data stored in the object
- and the length determines the number of elements contained in that object
- The **c()** function in R implies concatenating of various elements in a vector

```
# integer
a <- 233


# double
f <- 2.99


# string
b <- 'hello world, data analytics'


# boolean
c <- TRUE
```

Predictive Analytics Hands-on coding, 2018

Part one task

1. Create a vector **x1 with values (2.5,1.4,6.3,4.6,9.0)** in it and find the class, mode, length of **x1** vector.
2. Find class, mode, length of
   - **x2**<-c(TRUE,FALSE,TRUE,FALSE,FALSE)
   - **x3**<-c("DataMining","Statistics","Analytics","Projects","MachineLearning")

Part two

The **factor** is another form of data where various categories listed in the vector are known as levels, in the preceding example, x3 is a factor vector.

The as.**factor()** command is used to convert a character vector into a factor data type. After applying that, it indicates there are five levels such as Analytics, DataMining, MachineLearning, Projects, and Statistics.

Part two task

Convert **x3 vector** into a factor data type and find its class, mode and **print it**

Part three

Dataframes are another popular form of the data type in the R programming language that includes all different data types. A dataframe is a list that contains multiple vectors of the same length and different types of data.

If you **simply import a dataset from a spreadsheet**, the data type by default becomes dataframe. Later on, the data type for individual variables can be changed.

So, dataframe can be defined as a matrix that contains columns of different data types. In the preceding script, the dataframe x contains three different data types: numeric, logical, and character. Most real-world datasets contain different data types.

example, in a retail store, information about customers is stored in a database. This includes customer ID, purchase date, amount purchased, whether part of any loyalty program or not, and so on.

Part three task

Put x1,x2,x3 into one dataframe **x**. Find the class of x.

Note: One important point about vectors: all elements of a vector should be of the same type. If not, R will forcibly convert that by coercion. For example, in a numeric vector, if one element contains a character value, the vector type will change from numeric to character.

```
X1 <- c(x1,"cat")
class(x1)
```

Tip: R is case sensitive, so "cat" is different from "CAT". Hence, please be careful while assigning object names to the vectors.

Part four

A list is an ordered collection of objects that can contain arbitrary objects. Elements of a list can be accessed using the double square bracket. Those collections of objects are not necessarily of the same type.

Part four task

Create a list of following details

custid=112233

custname="John R"

mobile="989-101-1011"

email=JohnR@gmail.com

Print the list. Access its elements via indexes from one square bracket

In the preceding example, the customer ID and mobile number are of numeric data type; however, the customer name and e-mail ID are of character data type. There are basically four elements in the preceding list. To extract elements from a list, we use double square brackets, and if we need to extract only a sublist from the list, we can use a single square bracket.

Predictive Analytics Hands-on coding, 2018

Part five

Lists can be combined using the **cbind()** function, that is, the column bind function

Part five task

Create two lists , mylist1 and mylist 2 with same variables in it. Combine these two list into a new list mylist

Part six

There are different file formats; among them, CSV or text format is the best for the R programming platform. However, we can import from other file formats.

Part six task

Read any csv file using **read.csv ()** function and find the names of columns using **names()** function

Before using the above function set your working directory using **setwd()**

Tip: If you are using the read.csv command, there is no need to write the header True and separator as comma, but if you are using the read.table command, it is mandatory to use

Part 7

There are various types of data such as numeric, factor, character, logical, and so on.

Changing one data type to another if the formatting is not done properly is not difficult at all using R. Before changing the variable type, it is essential to look at the data type it currently is

Part 7 task

Find the data types of x1, x2, x3 vectors using is.numeric(), is.character(), is.vector(), is.matrix(), is.data.frame(). Then change data types of some of the vectors using as.numeric(), as.vector(), as.data.frame(), as.matrix() functions

Predictive Analytics Hands-on coding, 2018

Part 8

Sorting and merging are two important concepts in data management. The object can be a single vector or it can be a data frame or matrix. To sort a vector in R, the **sort()** command is used. A decreasing order option can be used to change the order to ascending or descending.

The order command is used to sort the data, where ascending or descending order can be set for multiple variables. Descending order can be executed by putting a negative sign in front of a variable name.

Part 8 task

Sort **petal length** variable in the **iris dataset** once in ascending order then in descending order

5 Things To Remember

Assignment: R uses the arrow operator (**<-**) for assignment, not a single equals (**=**)

Case Sensitive: The R language is case sensitive, meaning that C() and c() are two different function calls

Help: You can get help on any operator or function using the **help()** function or the question mark operator**?**

How To Quit: You can exit the R interactive environment by calling the question function **q()**