|  |  |  |  |
| --- | --- | --- | --- |
| Date: | **27-07-2020** | Name: | **Varun G Shetty** |
| Course: | **Coursera** | USN: | **4AL17EC093** |
| Topic: | Basic Statestics | Semester & Section: | **6th & ‘B’** |
| GitHub Repository: | **Varunshetty4** |  |  |

**Report-**

R Language:

Basics: In its most basic form R can be used as a simple calculator. Consider the following arithmetic operators:

* Addition: +
* Subtraction: -
* Multiplication: \*
* Division: /
* Exponentiation: ^
* Modulo: %%

A basic concept in R programming is the **variable**. It allows you to store a value or an object in R. You can then later use this variable's name to easily access the value or the object that is stored within this variable. You use <- to assign a variable:

my\_variable <- 4

Data types: Some of R's most basic types to get started are:

* Decimals values like 4.5 are called **numerics**.
* Natural numbers like 4 are called **integers**. Integers are also numerics.
* Boolean values (TRUE or FALSE) are called **logical**.
* Text (or string) values are called **characters**.

You can avoid such embarrassing situations by checking the data type of a variable beforehand. You can do this as follows:

class(some\_variable\_name)

you can inspect it by typing [**ls()**](http://www.rdocumentation.org/packages/base/functions/ls) in the console

It is possible to transform your data from one type to the other. Next to the [**class()**](http://www.rdocumentation.org/packages/base/functions/class) function, you can use the as.\*() functions to enforce data to change types. For example,

var <- "3"

var\_num <- as.numeric(var)

as.integer("4.5")

as.numeric("three")

Vectors: In R, you create a vector with the combine function c(). You place the vector elements separated by a comma between the brackets. For example:

numeric\_vector <- c(1, 2, 3)

character\_vector <- c("a", "b", "c")

boolean\_vector <- c(TRUE, FALSE)

Indexing entails the use of square brackets [] to select elements from a vector. For instance, numeric\_vector[1] will select the first element of the vector numeric\_vector. numeric\_vector[c(1,3)] will select the first and the third element of the vector numeric\_vector.

Selection by Comparision: Sometimes you want to select elements from a vector in a more advanced fashion. This is where the use of logical operators may come in handy. The (logical) comparison operators known to R are: - < for less than - > for greater than - <= for less than or equal to - >= for greater than or equal to - == for equal to each other - != not equal to each other.

The nice thing about R is that you can use these comparison operators on vectors. For example, the statement c(4,5,6) > 5 returns: FALSE FALSE TRUE. In other words, you test for every element of the vector if the condition stated by the comparison operator is TRUE or FALSE.

Behind the scenes, R does an element-wise comparison of each element in the vector c(4,5,6) with the element 5. However, 5 is not a vector of length three. To solve this, R automatically replicates the value 5 to generate a vector of three elements, c(5, 5, 5) and then carries out the element-wise comparison.

Matrices: You can construct a matrix in R with the matrix() function. Consider the following example: matrix(1:9, byrow = TRUE, nrow = 3, ncol = 3)

In the matrix() function:

* The first argument is the collection of elements that R will arrange into the rows and columns of the matrix. Here, we use 1:9 which constructs the vector c(1, 2, 3, 4, 5, 6, 7, 8, 9).
* The argument byrow indicates that the matrix is filled by the rows. This means that the matrix is filled from left to right and when the first row is completed, the filling continues on the second row. If we want the matrix to be filled by the columns, we just place byrow = FALSE.
* The third argument nrow indicates that the matrix should have three rows.
* The fourth argument ncol indicates the number of columns that the matrix should have

The term factor refers to a statistical data type used to store categorical variables. The difference between a categorical variable and a continuous variable is that a categorical variable can belong to a limited number of categories. A continuous variable, on the other hand, can correspond to an infinite number of values.

It is important that R knows whether it is dealing with a continuous or a categorical variable, as the statistical models you will develop in the future treat both types differently.A good example of a categorical variable is the variable student\_status. An individual can either be "student" or "not student". This means that "student" and "not student" are two values of the categorical variable student\_status and every observation can be assigned one of these values. We can do this using the factor function.