Introduction to R

Teaching Assistant: Parshan Pakiman Course: IDS 572

College of Business Administration ◇ University of Illinois at Chicago

If you find any issue in this document, please reach out to Parshan Pakiman at ppakim2@uic.edu.

1 Plan for the Second Week

1.1 Generating and Loading Data

In this course, our goal is to run analytics on different datasets to extract knowledge from data. Either we access a dataset, or we want to work with synthetic data. Let's begin with the latter case. We can use R to generate synthetic data. Section 3.4 of R for Beginners provides a comprehensive explanation of how to generate data in R. Below is a shortlist of basic R codes for data generation in R, which we learn in a lab session.

```
> x <- 1:30
> seq(length=9, from=1, to=5)
> c(1, 1.5, 2, 2.5, 3, 3.5, 4, 4.5, 5)
> rep(1, 30)
> gl(2, 6, label=c(“Male”, “Female”))
> data <- expand.grid(h=c(60,80), w=c(100, 300), sex=c(“Male”, “Female”))
> x = rnorm(10)
> matrix(data=5, nr=2, nc=2)
> x <- 1:4; n <- 10
> M <- c(10, 35); y <- 2:4
> data.frame(x, n)
> library(tidyverse);
> read_csv(“a,b,c \n 1,2,3 \n 4,5,6”)
> heights <- read_csv(“folder/name.csv”)    # loads a csv file with given name and path.
> parse_datetime(“2010-10-01T2010”)
> date <- parse_date(“2010-10-01”)
```

Run the above lines in your R environment, and answer the following questions:

- How to change the name of columns of variable `data`?
- What is the mean and standard deviation of vector `x`? What distribution random variable `x` follows?
- What is the type of the `date` variable?

Moreover, solve the following exercises:

- Generate $N = 100$ samples from a truncated normal distribution over the range $[-3, 10]$ that has the mean of 3 and the standard deviation of 1. Compute the empirical mean and standard deviation of this distribution using the samples, and report the error between actual mean and standard deviation and the empirical ones. How error changes as your increase N?

- From R for Data Science, solve Question 7 of Section 11.3.5 and Questions 1 and 5 of Section 11.2.2.

Next, read Sections 3.5.7 and 3.5.8 of R for Beginners.

1.2 For Loop and If-Else in R

Loops if-else statements are important parts of every programming language. Below are related R codes:

```
> x <- 5
> if(x > 0){
+ print(“Positive number”)
+ }
> x <- -5
> if(x > 0){
+ print(“Non-negative number”)
+ }
+ else {
+ print(“Negative number”)
+ }
> x <- -5
> if(x > 0) print(“Non-negative number”) else print(“Negative number”)    # one line of code
> x = c(1,2,3);
> for (i in 1:length(x)) if (x[i] == 1) y[i] <- 0 else y[i] <- 1    # one line of code
> y
```

For more examples, read Section 3.5.3 of R for Beginners as well as Sections 21.3.2 to 21.3.5 of R for Data Science. The `ifelse` statement in R is quite useful. Below is an example copied from R Documentation available at [LINK]

```
> x <- c(6:-4)
> sqrt(x)    # gives warning
> sqrt(ifelse(x >= 0, x, NA))    # no warning
> ifelse(x >= 0, sqrt(x), NA)    # this also gives the warning!
> ifelse(x %% 2 == 0,“even”, “odd”)
> library(tidyverse)
> diamonds    # we work diamonds dataset in the later sections
> high_price = ifelse(diamonds$price>=15000,1,0)
> hist(high_price)    # we will learn hist() function in detail in next sections
```

You can also see the video titled “ifelse” under the “ Lunchbreak Lessons” section of LinkedIn Learning video series [LINK].

1.3 Functions

In the upcoming assignments, you may need to call a function multiple times for different parameters to tune your model. Thus, learning how to write a function in R is a good investment! To start, let's define a function to normalize a dataset as follows:

```
> # Following code is obtained from https://www.statology.org/how-to-normalize-data-in-r
> min_max_norm <- function(x) {   # min-max normalization function
+ (x - min(x)) / (max(x) - min(x))
+ }
> data = iris[1:4]
> # Approach 1 to normalize data (direct call function ):
> min_max_norm(data[,1])
> min_max_norm(data[,2])
> min_max_norm(data[,3])
> min_max_norm(data[,4])
> # Approach 2 to normalize data (for lo0p):
> for (col in 1:ncol(data)) {
+ print(min_max_norm(data[,col]))
+ }
> # Approach 3 to normalize data (lapply):
> data_norm <- as.data.frame(lapply(data, min_max_norm))
> print(data_norm )
> # view first six rows of normalized iris dataset
> head(data_norm )
> head(data)
```

To complete our discussion on functions, read Section 6.3 of R for Beginners. Next, answer following questions:

- Write a function to standardize a numeric vector be deducting the mean of this vector from its elements and then divide its elements by its standard deviation. For example, for input vector `x<-c(2,4,6)`, your function should return the standardized vector `[-1 0 1]`.
- Generate a synthetic dataset and apply function `myfun` on page 67 of R for Beginners to your data.
- Consider an asset with a fixed price p. Let $D_1, D_2, \ldots, D_n$ be n samples from a truncated normal distribution on range $[1, 100]$. Define a function that computes expected revenue of selling this asset. Specifically, write a function that receives the price level and the list demand data, and it returns both expected and total revenues defined as

$$\text{Expected revenue=} \quad p \times \frac{1}{n} \times (\sum_{i=1}^{n} D_i)$$

$$\text{Total revenue=} \quad p \times (\sum_{i=1}^{n} D_i)$$

- Define a function called `Fibonacci` that receives a positive integer `N` and returns the N-th element of the Fibonacci sequence. This sequence is defined as follows”

$$F_n := \begin{cases} 0 & \text{if } n = 0 \\ 1 & \text{if } n = 1 \\ F_{n-1} + F_{n-2} & \text{if } n \geq 2 \end{cases}$$