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 Third Week

1.1 Objects

There are different objects in R such as vector, factor, matrix, etc. Read Section 3.5.1 of R for Beginners for how to define objects in R. Below is an R script which defines some R objects. Try to run this script in your machine.

```
> factor(1:3, labels=c("A", "B", "C"))    # factor.
> matrix(1:6, 2, 3, byrow=TRUE)    # matrix.
> x <- 1:4; n <- 10;
> data.frame(x, n)    # data frame.
> L1 <- list(x, x^2);    # list.
> x <- 3; y <- 2.5; z <- 1
> exp1 <- expression(x / (y + exp(z)))    # expression.
> exp1
> eval(exp1)   # evaluating an expression.
```

Next, we need to learn how to change type of an object to different type, referred to as type casting. For example, casting a string to a date. To better learn this concept, read Section 3.5.2 of R for Beginners and answer the following questions:

- Cast factors `factor(1:3)` and `factor(c("Male", "Female"))` to a numeric list.
- Convert matrix `matrix(1:6, 2, 3, byrow=TRUE)` to a data frame and convert it back to the same matrix.
- Convert list of strings `x <- c("1jan1960", "2jan1960", "31mar1960", "30jul1960")` to a list of dates given by `x_date <- "1960-01-01" "1960-01-02" "1960-03-31" "1960-07-30"`. What are the types of these objects (e.g., run codes `typeof(x)` and `typeof(x_date)`)? Are their types different?

1.2 Handling Date in R

In many datasets, we need to deal with dates and times. There are several packages to handle dates in R. We use lubridate package in this course to handle dates in datasets. To start, please visit website [LINK] and run R scripts in your R environment. I summarized some of the important lubridate codes below:

```
> library(lubridate)
> now()    # shows current time
> x <- as.Date("2012-03-1")    # type casting: string to date
> y <- as.Date("2012-02-28")    # type casting: string to date
> # Now that we have date object, we can do asthmatics with dates, e.g., deducting two dates
> x - y
> x <- c("2015-07-01", "2015-08-01", "2015-09-01")    # list of strings
> as.Date(x)    # type casting: string to date
> # Below are more asthmatics that one can do with date objects
> year(x)
> month(x, label = TRUE)
> month(x, label = TRUE, abbr = FALSE)
> x <- ymd(x)
> update(x, year = c(2013, 2014, 2015))
> x + years(1) - days(c(2, 9, 21))
> # One can also create a sequence of dates between two start and end dates
> seq(as.Date("2015/1/1"), as.Date("2021/12/30"), by = "quarter")
> seq(ymd("2021-01-01"), ymd("2021-01-14"), by = "2 days")
```

Below is another useful example from [LINK].

```
> x <- seq(as.Date("2000-02-29"), as.Date("2004-10-04"), by = "1 month")
> # has many "yyyy-mm-29", but a few "yyyy-03-01" in the non-leap years
> y <- ifelse(as.POSIXlt(x)$mday == 29, x, NA)
> head(y)    # not what you expected ... ==> need restore the class attribute:
> class(y) <- class(x)
> y
> # This is a case where it is better not to use ifelse(), but rather the more clear code
> y2 <- x
> y2[as.POSIXlt(x)$mday != 29] <- NA
> stopifnot(identical(y2, y)) # which gives the same as ifelse()+class() hack
```

To further learn the date manipulation in R via the lubridate package, take a look at the subsection titled "Working with dates and times in R" of Section 4 of the video series [LINK] on LinkedIn Learning.

1.3 Basic Plots in R

It is useful to start data mining projects by doing basic statistical analyses. Let's start by running following codes on your machine to load the `diamonds` dataset from `tidyverse` library.

```
> library(tidyverse)
> diamonds
> str(diamonds)
```

Having data loaded, we run the following codes to learn the properties of this dataset:

```
> summary(diamonds)
> boxplot(diamonds$carat)
> hist(x$price)
> plot(diamonds$cut)
> cor(diamonds$carat,diamonds$depth)
> cor(diamonds$carat,diamonds$price)
> x = diamonds[c('carat','depth','table','price','x','y','z')]
> cor(x)
> corrplot(cor(x),method="number")
```

Once you learned the above codes, it is time to learn `ggplot2` library. From R for Data Science e-book, work on the following:

1. Read Sections 7.5.1 and 7.3.1.
2. Solve Questions 2, 4, and 6 of Section 7.5.1.1. Also, solve Questions 1 and 3 in Section 7.3.4.
3. Read Sections 7.5.2 and 7.5.3.
4. Solve Questions 2 and 3 of section 7.5.3.1.
5. Read Sections 3.2.2, 3.2.3, and 3.3.
6. Solve all questions in Section 3.2.4, and Questions 1, 2, and 3 of Section 3.3.1.
7. Read Section 7.4 of R for Data Science.
8. Solve both questions in section 7.4.1.
9. Take a look at Section 3.6 of R for Data Science.

There are many online resources that you can use to learn `ggplot2`. I recommend learning this library as we go forward and as you need to visualize something to deliver an insight. The LinkedIn Learning video series [LINK] provides a comprehensive resource for learning `ggplot2`.

1.4 Data Transformation

Data transformation is needed in most data mining projects. We learn the basics of data transformation in R using `dplyr` package by mainly focusing on Section 5 of R for Data Science. To this end, we go over the following steps in order:

1. Read Sections 5.1.2 and 5.1.3 to load dataset `nycflights13`.
2. To learn function `filter()`, read Section 5.2 including Subsections 5.2.1, 5.2.2, and 5.2.3.
3. Solve Questions 1, 2, and 3 of Section 5.2.4.

4. To order a dataset by a column, we can use function `arrange()` of `dplyr`. Read Section 5.3 and solve all questions in Section 5.3.1.

5. Read Section 5.4 to learn how a subset of a large dataset can be extracted. Solve Exercises 5.4.1.

6. In machine learning, we often need to come up with new features that are not in the generic dataset. Hence, we need to learn how new columns can be generated and added to a given data. Read Section 5.5 to learn `mutate()` function to add new columns that are functions of existing column.

7. Read Subsections 5.6.1, 5.6.2, 5.6.3, and 5.6.5. Solve Question 2, 5, 7, and 8 of Section 5.7.1.

1.5 RMarkdown

RMarkdown allows you to create a PDF file that embeds R codes, explanations related to each code block, plots, and tables. RMarkdown is a useful tool to write a report which includes both your code and your text. In this course, you should turn in your assignments using RMarkdown. We do not cover this topic in class, but learning it probably does not take more than half an hour. Feel free to use any resources to understand how to create and to knit RMarkdown files. You can take a look at Section 2 of the LinkedIn Learning video series available at [LINK]. Also, you can take a look at the video titled "Use .Rmd for documentation" under the " Lunchbreak Lessons" subsection of the LinkedIn Learning video series [LINK].