Lab Class - 2 (Estimation and Testing)

ECO 204 (Spring 2025)
Instructor: Shaikh Tanvir Hossain

§

Here is the agenda for today's lab class,

- using getwd() and setwd() functions to get and set the working directory
- Loading a data set in R and picking a column from a data set
- Writing `if-else` statement, own `functions` & `AND`, `OR`, `NOT` operators in R
- Solving some applied problems from Chapter 8 and Chapter 9 of Anderson et al. (2020)
- Properly generate a RMarkdown file.
- Using `for-loop`, and see an applications of `For-loop` using confidence intervals.

Directory Functions and Loading a Data Set

1. It's always a good practice that before doing any session in R, you fix your working directory, so that you can easily access the files you need and also save the file in a specified place. This is for better file management.

   In R to set the working directory we can use the function `setwd()`, the input is the path of the directory that you prefer. If you are using windows, make sure your path has / (this is called slash) instead of \ (backslash). (**Important:** Do not use `setwd()` functions in a RMarkdwon file, run it before knitting the file).

   Loading a dataset is easy if you use RStudio, you can directly use the import option to import a data set. In this case RStudio automatically loads the package `readxl`, and then run the function `read_excel()` for you. (**Important:** Even though RStudio automatically run some codes, but it's a good practice to keep these codes into your RScript, so that you can reproduce the same result in future)

   When you have a data set in R, picking a column is easy, you can use the dollar sign ($) to pick a column. For example, if you have a data set called `Houston`, and you want to pick the column age, you can use `Houston$age`.

Writing if else statement, own functions, and AND, OR, NOT operators

`if-else` Statement

You should use If else statement whenever there are some conditions to check, the syntax for the `if-else` statement is

```
if (condition) {
  # code to be executed if condition is true
} else {
  # code to be executed if condition is false
}
```

For example,

```
#> Example of if-else statement
x <- 10
if (x > 5) {
  print("x is greater than 5")
} else {
  print("x is less than or equal to 5")
}
```

Writing Own Functions using the function `function`

When there are multiple copying and pasting, the elegant and neat way to solve the problem is to write a function. The syntax for writing a function is

```r
function_name <- function(arg1, arg2, ...) {
  # code to be executed
}
```

For example,

```r
#> Example of writing a function
add_func <- function(x, y) {
  return(x + y)
}

add_func(10, 20)
```

You can also test the default value of the argument, for example,

```r
#> Example of writing a function with default value
add_func <- function(x, y = 10) {
  return(x + y)
}

add_func(10)
```

AND, OR, NOT operators

The logical operators `AND`, `OR`, `NOT` in R are represented by `&`, `|`, and `!` respectively. For example,

```r
#> Example of AND, OR, NOT operators

myvec <- c(10, 20, 30)

#> AND operator
10 %in% myvec & 20 %in% myvec

#> OR operator
10 %in% myvec | 40 %in% myvec

#> NOT operator
!10 %in% myvec
```

In this problem `%in%` means the element is in the vector or not.

Some Problems From Chapter 8 and Chapter 9

References:

Anderson, D. R., Sweeney, D. J., Williams, T. A., Camm, J. D., Cochran, J. J., Fry, M. J. and Ohlmann, J. W. (2020), *Statistics for Business & Economics*, 14th edn, Cengage, Boston, MA.