Cairo University
Faculty of Engineering
Computer Engineering Department

CMPN451
Spring 2022

Data Mining, Big Data and Analytics.

Lab 1 – RStudio and Introduction to R

Objectives:

By the end of this lab, the student should be able to:

- Install “RStudio” and be familiar with it.
- Use interactive learning tools such as Swirl.
- Know the basics of R language.

Introduction to R:

- R is a programming language and software environment for:
  - Statistical analysis.
  - Graphics representation and reporting.

- This programming language was named **R**, based on the first letter of first name of the two **R** authors (Robert Gentleman and Ross Ihaka), and partly a play on the name of the Bell Labs Language **S**, another popular statistical software.

- As illustrated in Table 1, the most popular languages for data science are **R** and **Python** (according to the famous portal KDnuggets). **R** is better for visualization and reporting while **Python** is more suited for building products.

- In this course, we are going to use **R** as a statistics/programming language for analytics and data mining. Also, we are going to use **RStudio** as an IDE for **R**.

Table 1: Most popular languages for data science in 2020

What programming/statistics languages you used for an analytics / data mining / data science work in 2020?	
Language used	🟥 % voters in 2020 (719 total) 🟦 % voters in 2019 (713 total) 🟪 % voters in 2018 (579 total)
R (352 voters in 2020)	🟥 49.0% 🟦 60.9% 🟪 52.5%
SAS (262)	🟥 36.4% 🟦 20.8% 🟪 19.7%
Python (252)	🟥 35.0% 🟦 38.8% 🟪 36.1%

SQL (220)	🟥 30.6% 🟦 36.6% 🟪 32.1%
Java (89)	🟥 12.4% 🟦 16.5% 🟪 21.2%
Unix shell/awk/sed (63)	🟥 8.8% 🟦 11.1% 🟪 14.7%
Pig Latin/ Hive/ other Hadoop-based languages (61)	🟥 8.5% 🟦 8.0% 🟪 6.7%
SPSS (58)	🟥 8.1% not asked not asked
MATLAB (45)	🟥 6.3% 🟦 12.5% 🟪 13.1%

Part 1: Install R and RStudio for Windows:

- **To install R:**
  1. Open an internet browser and go to www.r-project.org.
  2. Click the "**download R**" link in the middle of the page under "**Getting Started**"
  3. Select a CRAN location (a mirror site) and click the corresponding link.
  4. Click on the "**Download R for Windows/ (Mac) OS X**" link at the top of the page.
  5. Click on the "**install R for the first time**" link at the top of the page.
  6. Click "**Download R for Windows**" and save the executable file somewhere on your computer. Run the **.exe** file and follow the installation instructions.
  7. Now that **R** is installed, you need to download and install **RStudio**.

- **To install RStudio:**
  1. Go to www.rstudio.com and click on the "**Download RStudio**" button.
  2. Click on "**Download RStudio Desktop**."
  3. Click on the version recommended for your system, or the latest Windows version, and save the executable file. Run the **.exe** file and follow the installation instructions.

Part 2: Use interactive tools for learning.

- There are tons of resources to help you learn the different aspects of R starting from blogs, videos, tutorials and interactive tools.

- One of the easiest is “swirl” interactive tool through RStudio, which can assist you through your first steps in R.
- From RStudio Console , type command

  ```
  install.packages("swirl")
  ```

  then after downloading package and installation you can access it offline by typing these two commands:

  ```
  library("swirl")
  swirl()
  ```

- Your task is to learn as much as you can about R through your preferred way and get familiar with RStudio till next tutorial.

Part 3: Introduction to R

1. **Packages**:

   Packages are the fuel that drive the growth and popularity of **R**. **R** packages are bundles of *code*, *data*, *documentation*, and *tests* that are easy to share with others. R packages usually have no dependencies.

   You need to install them first through the command

   ```
   install.packages(“package_name”)
   ```

   then you need to include them in your code through

   ```
   library(“package_name”)
   ```

   You can search for the package name and get a view of its documentation in the Help Tab.

2. **Five Things to Remember About R:**
   1. (Almost) everything is an ***object.***
   2. (Almost) everything is a ***vector.*** For example:

      `a <- 3` is a 1x1 vector.

      `v <- c(1,2,3,4,5)` is a 5x1 vector.
   3. All commands are ***functions*.** For example:

      ```
      quit() or q() not q
      ```
   4. Same commands produce different output depending on imported package.
   5. Know your default arguments!

3. **Data Types in R:**

   Primitive (or atomic) data types in **R** are:
   - numeric (integer, double, complex)
   - character
   - logical
   - function

   Out of these, *vectors*, *lists*, *matrices* and *data frames* can be built.

	Linear	Rectangular
All same type	Vector	Matrix
Mixed types	List	Data Frame

Data Types	

Numbers, Strings	n <- 3
	s <- “columbus, ohio”

Vectors	levels <- c(“Wow”, “Good”,“Bad”)
	ratings <- c(“Bad”, “Bad”, “Wow”)

Factors and Lists	f <- factor(ratings, levels)
	l <- list(ratings=ratings,
	critics=c(“Siskel”,“Ebert”))

Functions	stdev <- function(x) sd(x)

4. R structured Types:

Data Types	R Code
Matrix - (n*m numeric data frame)	m <- matrix(c(1:3, 11:13), nrow = 2, ncol = 3, byrow = TRUE)
Table – contingency table	t <- table(dfm$factor_variable)
data frames – data sets	dfm <- read.csv(“CrimeRatesByStates2005.csv”)

Extracting data	ndfm <- dfm[1:3,]
	ndfm <- dfm[, 3:5]
	v <- dfm$salary

5. Basic R operations on vectors:

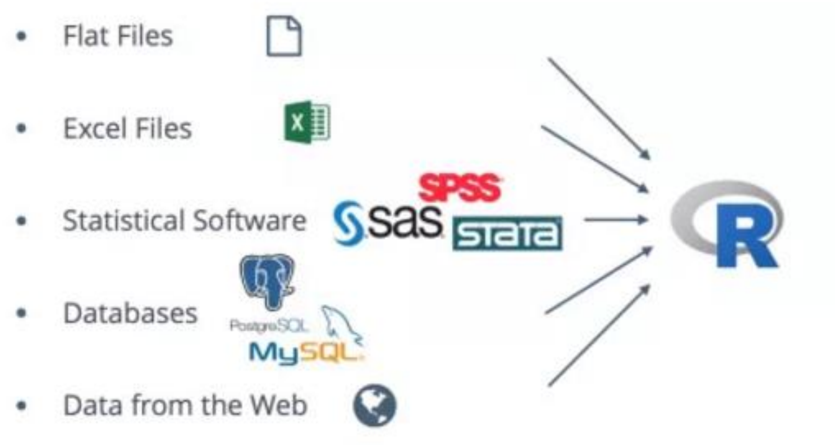
Function	R Code
Operations on Vectors	v <- c(1:10); w <- c(15:24) ; nv <- v * pi ; nw <- w * v

Vector transformations	radius <- sqrt(d$population)/ pi)
	t <- as.table(dfm$factor_variable)
	pct <- t/sum(t)* 100

Logical Vectors	v[v < 1000]
	ndf <- subset(dfm, d$population < 10000)
	nv <- v[c(1,2,3,5,8,13)]

Examining data structures	dim(dfm); attributes(dfm) ; class(dfm); typeof(dfm)

6. Import files :



- Flat Files
- Excel Files
- Statistical Software
- Databases
- Data from the Web

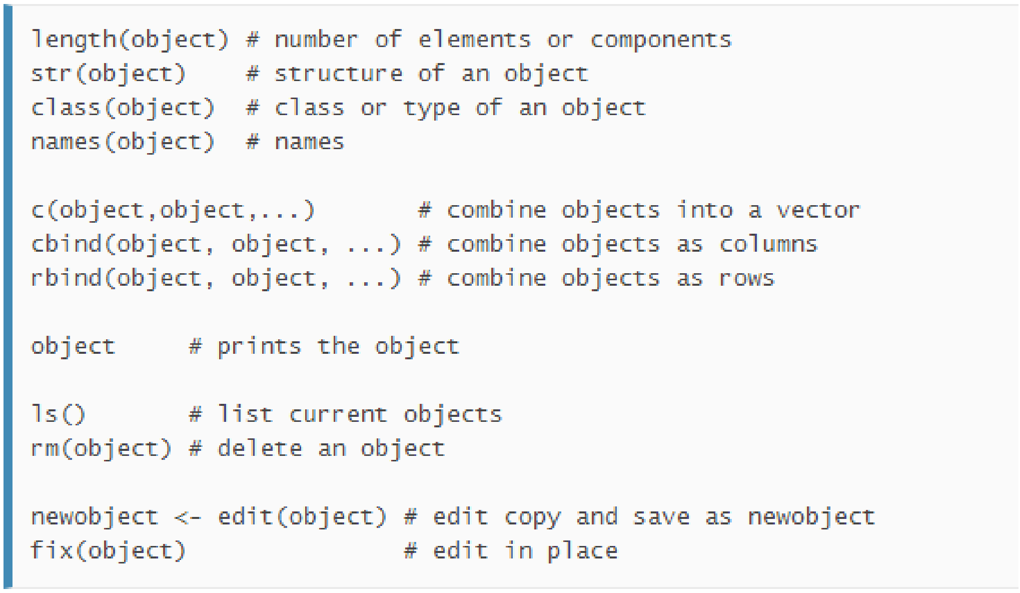
7. **Descriptive Statistics**:

Function	R Code
View the data	head(x); tail(x)
View a summary of the data	summary(x)
Compute basic statistics	sd(x); var(x); range(x); IQR(x)
Correlation	cor(x); cor(d$var1, d$var2)

8. Generic Functions:

Code	Function
Plot the variable x	plot (x)
Histogram of x	hist (x)
Internal structure of x	str (x)

9. **Useful functions:**



```
length(object) # number of elements or components
str(object)    # structure of an object
class(object)  # class or type of an object
names(object)  # names

c(object,object,...)       # combine objects into a vector
cbind(object, object, ...) # combine objects as columns
rbind(object, object, ...) # combine objects as rows

object     # prints the object

ls()       # list current objects
rm(object) # delete an object

newobject <- edit(object) # edit copy and save as newobject
fix(object)               # edit in place
```

Useful links and resources:

- https://www.r-bloggers.com/how-to-learn-r-2/
- http://swirlstats.com/
- https://www.datacamp.com/community/tutorials/r-data-import-tutorial?tap_a=5644-dce66f&tap_s=10907-287229#gs.9Zw03Cw
- http://www.r-tutor.com/r-introduction/basic-data-types
- http://www.statmethods.net/input/datatypes.html