Homework Week 1

Steven Simonsen

2024-01-18

Chapter 1 HW: Consider Figure 1.1 (page 2) with respect to some particular scientific research study, such as a research project that the reader might be undertaking. What is involved in the data analysis of the study at all its stages, from processing raw data to eventual communication of results? How much time is likely to be needed at each stage? What specifically needs to be done at each stage? Will R be sufficient for all the stages of analysis of the study? What other tools are required?

Considering Figure 1.1 with respect to a particular scientific research study, the gathering of raw data is the first stage of the study. In terms of involvement, the gathering of raw data can vary. For example, if the dataset is already readily available from an established source (ex: World Health Organization regarding life expectancy), then gathering the data from the source is straightforward. However, if the person conducting the research study must gather their own data, then parameters for the study must be established and executed. The amount of time spent collecting data outside of an existing data set can vary widely, depending on the type of study performed and the number of data points collected. By contrast, pulling raw data from an already existing raw data source is much faster. At this stage, the data may or may not be formatted in R and may also be structured or unstructured. The data may be present in the form of text, JSON, or SQL, just to name a few. The upcoming lifecycle stages will utilize the use of R much more heavily to produce a final result.

The next stage in the research study is data wrangling, and this stage can be very time consuming and laborious. The amount of time needed at this stage will vary depending on the complexity of the data and the amount of data manipulation needed to progress to tidy data. For this stage of the research study, R is not only sufficient, but preferred given the various packages R offers such as readr, dplyr, and tidyr (Andrews, 2021). At this stage, the use of numerous operations, commands, variables, data frames, and vectors may be needed to achieve the desired tidy data result. Ideally, tidy data is achieved in the form of a structured, tabular data set with rows and columns. Additional tools may include the need for R packages, existing functions, and custom functions created withing the author's code.

After tidy data has been achieved through data wrangling, the next step in the process is data visualization and exploration. As illustrated in Figure 1.1, this is an iterative process. Data visualization is defined and improved upon as further exploration of the data is conducted. The time it takes to explore and visualize the data depends upon the complexity of the tidy data. To elaborate, a study containing minimal data points, or a relatively small sample size will generally be less complex. However, a more simplistic data set may lead to false conclusions. As an example, a coin flip conducted ten times may indicate that a "heads" result occurs about 70% of the time. However, as additional coin flips are conducted, the law of large numbers is recognized, and "heads" will result much closer to 50% of the time. The exploration of the data at this stage will involve many of the same tools as the data wrangling stage. Additional tools may include the use of reading the data through the read_csv R command and viewing the data with functions such as summary, str, head, and tail. The visualization process in R can include various packages such as ggplot2 to display the data in the form of boxplots, scatterplots, and more. R should be solely sufficient in accomplishing this stage of the study.

After data visualization has been completed, data models are created to represent the data. Then, fitted models are created through inferences made within the data. As with data visualization and exploration, this process is iterative. The determination of the best fitted model to use for the data is conducted from inferences within the visualization created. Next, evaluations are made to determine if changes are required to the overall model, thus depicting a more accurate fitted model. The time spent within this stage of the study depends on the code used to fit the data to the model. As noted above, repetition is required as part of this process but can also result in overfitting the data to the model. Conversely, under-fitting is also a concern. Therefore, the model must accurately convey a well-defined relationship between the variables defined and the target results, resulting in a flexible probabilistic model (H2O.ai., n.d.). Again, R is a great choice for modelling the data due to the robust packages R has to offer.

Finally, communication is the final step in the research study and is vitally important. To demonstrate sound findings from previous steps in the research study, sound communication of the analysis must be conducted. A great way to communicate results is with R Markdown. R Markdown is available in RStudio and intertwines R code, plain text describing the analysis, and produces reports containing the visualizations from the study. The basic tools required for R Markdown include the use of knitr to translate the R to the desired output format (ex: PDF, HTML, etc.). The time spent within the communication phase depends on the complexity and volume of the findings. For example, if numerous dynamic documents such as plots, tables, and results are used in conjunction with narrative text, it could take significant time to create the communication document. Although knitr is used within the context of the final communication file(s), R is the underlying tool containing all analysis and code used to communicate the final results.

Sources

Andrews, M. (2021). Chapter 1: Data Analysis and Data Science. In Doing data science in R: An introduction for social scientists. essay, SAGE Publications Ltd.

What is model fitting and why is it important?. What is model fitting and why is it important? | H2O.ai. (n.d.). https://h2o.ai/wiki/model-fitting/

Chapter 2 HW: Install R and RStudio and then go through the steps in the guide, explicitly typing in all the code, using data or examples of their own choice. Copying and pasting any code is not recommended. Type all of your code in an RMarkdown and submit a knitted PDF. We will go over this in class.

```r
#Calculator commands
2+2 #addition
```

```
## [1] 4
```

```r
3-5 #subtraction
```

```
## [1] -2
```

```r
3*2 #multiplication
```

```
## [1] 6
```

```r
4/3 #division
```

```
## [1] 1.333333
```

```r
(2+2) ^ (3/3.5)
```

```
## [1] 3.281341
```

```r
#Equality/inequality operations
12==(6/2) #test for equality
```

```
## [1] FALSE
```

```r
(3*4) != (18-7) #test for inequality
```

```
## [1] TRUE
```

```r
3 < 10 #less than
```

```
## [1] TRUE
```

```r
(2*5) <= 10 #less than or equal
```

```
## [1] TRUE
```

```r
#Logical values and logical operations
TRUE & FALSE #logical and
```

```
## [1] FALSE
```

```r
TRUE | FALSE #logical or
```

```
## [1] TRUE
```

```r
!TRUE #logical not
```

```
## [1] FALSE
```

```r
(TRUE | !TRUE) & !FALSE
```

```
## [1] TRUE
```

```r
#Variables and assignment
(12/3.5)^2 + (1/2.5)^3 + (1+2+3)^0.33
```

```
## [1] 13.6254
```

```r
x <- (12/3.5)^2 + (1/2.5)^3 + (1+2+3)^0.33
x
```

```
## [1] 13.6254
```

```r
x^2
```

```
## [1] 185.6516
```

```r
x * 3.6
```

```
## [1] 49.05145
```

```r
#Vectors
primes <- c(2, 3, 5, 7, 11, 13)
primes + 1
```

```
## [1]  3  4  6  8 12 14
```

```r
primes / 2
```

```
## [1] 1.0 1.5 2.5 3.5 5.5 6.5
```

```r
primes == 3
```

```
## [1] FALSE  TRUE FALSE FALSE FALSE FALSE
```

```r
primes == 7
```

```
## [1] FALSE FALSE FALSE  TRUE FALSE FALSE
```

```r
#Indexing Vectors
primes[1]
```

```
## [1] 2
```

```r
primes[5]
```

```
## [1] 11
```

```r
primes[c(3, 5, 2)]
```

```
## [1]  5 11  3
```

```r
primes[-1]
```

```
## [1]  3  5  7 11 13
```

```r
primes[-2]
```

```
## [1]  2  5  7 11 13
```

```r
#Vector types
nation <- c('ireland', 'england', 'scotland', 'wales')
nation[1]
```

```
## [1] "ireland"
```

```r
nation[2:3]
```

```
## [1] "england"  "scotland"
```

```r
nation == 'ireland'
```

```
## [1]  TRUE FALSE FALSE FALSE
```

```r
class(primes)
```

```
## [1] "numeric"
```

```r
class(nation)
```

```
## [1] "character"
```

```r
class(nation == 'ireland')
```

```
## [1] "logical"
```

```r
#Data Frames
Df <- data.frame(name = c('billy', 'joe', 'bob'),
                 age = c(21, 29, 23))
Df
```

```
##    name age
## 1 billy  21
## 2   joe  29
## 3   bob  23
```

```r
#Indexing data frames
Df[3,2] #row 3, col 2
```

```
## [1] 23
```

```r
Df[c(1, 3), 2] #rows 1 and 3, col 2
```

```
## [1] 21 23
```

```r
Df[1,] #row 1, all cols
```

```
##    name age
## 1 billy  21
```

```r
Df[, 2] #all rows, col 2
```

```
## [1] 21 29 23
```

```r
Df$age
```

```
## [1] 21 29 23
```

```r
Df[['age']]
```

```
## [1] 21 29 23
```

```r
Df['age']
```

```
##   age
## 1  21
## 2  29
## 3  23
```

```r
#Functions
length(primes)
```

```
## [1] 6
```

```r
sum(primes)
```

```
## [1] 41
```

```r
mean(primes)
```

```
## [1] 6.833333
```

```r
median(primes)
```

```
## [1] 6
```

```r
sd(primes)
```

```
## [1] 4.400758
```

```r
var(primes)
```

```
## [1] 19.36667
```

```r
#Custom functions
my_mean <- function(x){sum(x)/length(x)}
my_mean(primes)
```

```
## [1] 6.833333
```

```r
#Writing R scripts and code comments
#Here is a data frame with four variables
#The variables are name, age, sex, and occupation
composites <- c(4, 6, 8, 9, 10, 12)
composites_plus_one <- composites + 1
composites_minus_one <- composites - 1
Df2 <- data.frame(name = c('jane', 'joe', 'billy'),
                  age = c(23, 27, 24),
                  sex = c('female', 'male', 'male'),
                  occupation = c('tinker', 'tailor', 'spy')
                  )

#Packages
install.packages("dplyr", repos = "http://cran.us.r-project.org")
```

```
## Installing package into 'C:/Users/steve/AppData/Local/R/win-library/4.3'
## (as 'lib' is unspecified)

## package 'dplyr' successfully unpacked and MD5 sums checked
##
## The downloaded binary packages are in
##  C:\Users\steve\AppData\Local\Temp\RtmpoJiZuF\downloaded_packages
```

```
install.packages(c("dplyr", "tidyr", "ggplot2"), repos = "http://cran.us.r-project.org")
```

```
## Installing packages into 'C:/Users/steve/AppData/Local/R/win-library/4.3'
## (as 'lib' is unspecified)

## package 'dplyr' successfully unpacked and MD5 sums checked
## package 'tidyr' successfully unpacked and MD5 sums checked
## package 'ggplot2' successfully unpacked and MD5 sums checked
##
## The downloaded binary packages are in
##  C:\Users\steve\AppData\Local\Temp\RtmpoJiZuF\downloaded_packages
```

```
install.packages("tidyverse", repos = "http://cran.us.r-project.org")
```

```
## Installing package into 'C:/Users/steve/AppData/Local/R/win-library/4.3'
## (as 'lib' is unspecified)

## package 'tidyverse' successfully unpacked and MD5 sums checked
##
## The downloaded binary packages are in
##  C:\Users\steve\AppData\Local\Temp\RtmpoJiZuF\downloaded_packages
```

```
library("tidyverse")
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.4     v readr     2.1.5
## v forcats   1.0.0     v stringr   1.5.1
## v ggplot2   3.4.4     v tibble    3.2.1
## v lubridate 1.9.3     v tidyr     1.3.0
## v purrr     1.0.2

## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

```
#Reading in data
library(readr)
getwd()
```

```
## [1] "C:/Users/steve/OneDrive/Documents/School/DSE5001 Intro to Data Science and Stats/Week 1"
```

```
test_data <- read_csv("weight.csv")
```

```
## Rows: 6068 Columns: 8
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (1): gender
## dbl (7): subjectid, height, height_selfreport, weight, weight_selfreport, ag...
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
test_data
```

```
## # A tibble: 6,068 x 8
##    subjectid gender height height_selfreport weight weight_selfreport   age
##        <dbl> <chr>   <dbl>             <dbl>  <dbl>             <dbl> <dbl>
##  1     10027 Male     178.              180.   81.5              81.7    41
##  2     10032 Male     170.              173.   72.6              72.6    35
##  3     10033 Male     174.              173.   92.9              93.0    42
##  4     10092 Male     166.              168.   79.4              79.4    31
##  5     10093 Male     191.              196.   94.6              96.6    21
##  6     10115 Male     172               175.   80.2              79.4    39
##  7     10117 Male     181               183.  116.              113.     32
##  8     10237 Male     185               188.   95.4              95.7    23
##  9     10242 Male     178.              178.   99.5              99.8    36
## 10     10244 Male     181.              183.   70.2              72.6    23
## # i 6,058 more rows
## # i 1 more variable: race <dbl>
```

```
glimpse(test_data)
```

```
## Rows: 6,068
## Columns: 8
## $ subjectid         <dbl> 10027, 10032, 10033, 10092, 10093, 10115, 10117, 102~
## $ gender            <chr> "Male", "Male", "Male", "Male", "Male", "Male", "Mal~
## $ height            <dbl> 177.6, 170.2, 173.5, 165.5, 191.4, 172.0, 181.0, 185~
## $ height_selfreport <dbl> 180.34, 172.72, 172.72, 167.64, 195.58, 175.26, 182.~
## $ weight            <dbl> 81.5, 72.6, 92.9, 79.4, 94.6, 80.2, 116.2, 95.4, 99.~
## $ weight_selfreport <dbl> 81.66969, 72.59528, 93.01270, 79.40109, 96.64247, 79~
## $ age               <dbl> 41, 35, 42, 31, 21, 39, 32, 23, 36, 23, 32, 28, 36, ~
## $ race              <dbl> 1, 1, 2, 1, 2, 1, 2, 1, 1, 1, 1, 1, 2, 1, 1, 2, 1, 1~
```