Exercise 3

Contents

```
knitr::opts_chunk$set(include = TRUE)
knitr::opts_chunk$set(comment = NA)
library(rvest)
library(httr)
library(magrittr)
library(tidyverse)
library(rio)
library(glue)
```

For this exercise, you will not be working with a single dataset, but will instead practice new skills using both your NLSY data from last seminar, as well as online data concerning movies.

1. Revisit your NLSY97 dataset from last week

```
nlsy97 <-import("nlsy97.rds")
```

   a. Create an indicator for sex using a vectorized conditional statement.

```
nlsy97 <- nlsy97 %>% mutate(female = ifelse(sex == 2, 1, 0))
```

   b. Recode the schooltype variable into text values, corresponding to:

      - "Public" if the value is 1
      - "Private, religious" if the value is 2
      - "Private, non-religious" if the value is 3
      - "Other" if the value is 4.

```
nlsy97 <- nlsy97 %>% mutate(schooltype = case_when(
  schooltype == 1 ~ "Public",
  schooltype == 2 ~ "Private, religious",
  schooltype == 3 ~ "Private, non-religious",
  schooltype == 4 ~ "Other"
  ))
```

2. Load the IMDB Top 250 Movies

   a. Scrape the data from the "Top 250 Movies as rated by IMDb users" from https://www.imdb.com/chart/top

```
# IMDB Top 250 Movies
top250_basic <- read_html("https://www.imdb.com/chart/top/") %>%
   html_table() %>% as.data.frame()
```

   b. Notice that IMDB scrapes the data in Swedish by fault. To get the data in English, use **html_session()** in place of **read_html()**, adding the option:

```
add_headers("Accept-Language"="en-US, en;q=0.5")
```

   - You may need to load the **httr** package to use **add_headers()**.

```
top250_eng.pre <- html_session("https://www.imdb.com/chart/top/",
                               add_headers("Accept-Language"="en-US, en;q=0.5")) %>%
                    html_table %>% as.data.frame()
```

c. Keep only the columns "Rank. . . Title" and "IMDb.Rating", suitably renaming them.

```
top250_eng <- top250_eng.pre %>% select("Rank...Title", "IMDb.Rating") %>%
  rename(Title = Rank...Title,
         Rating = IMDb.Rating) # Keep only Title and Year Columns
```

d. Create a ranking variable by extracting the values that appear before the dot in the title column.

```
top250_eng$Ranking <- top250_eng$Title %>% str_extract("[0-9]+(?=(.\n))")
```

e. Create a year variable, by extracting the numbers inside a parenthesis from the title column.

```
top250_eng$Year <- str_extract(top250_eng$Title, "(?<=[:punct:])[:digit:]+")
```

f. Redefine the title variable by extracting the string information that appear after the dot in the title column.

```
top250_eng$Title <- top250_eng$Title %>% str_extract("(?<=(.\n)).+")
```

g. Trim the white space on both sides of the title.

```
top250_eng$Title %<>% str_trim(side = "both")
```

h. View data frame

```
head(top250_eng)
```

```
# A tibble: 6 x 4
  Title                    Rating Ranking Year
  <chr>                     <dbl> <chr>   <chr>
1 The Shawshank Redemption    9.2 1       1994
2 The Godfather               9.2 2       1972
3 The Godfather: Part II      9   3       1974
4 The Dark Knight             9   4       2008
5 12 Angry Men                8.9 5       1957
6 Schindler's List            8.9 6       1993
```

3. Get the box office statistics for the top 500 all-time US box office earners

    a. Using a loop, create a list of the top 500 box office hits taking advantage of the fact that each 100 movies is listed on the following pages:

    - https://www.boxofficemojo.com/alltime/domestic.htm?page=1
    - https://www.boxofficemojo.com/alltime/domestic.htm?page=2
    - https://www.boxofficemojo.com/alltime/domestic.htm?page=3
    - https://www.boxofficemojo.com/alltime/domestic.htm?page=4
    - https://www.boxofficemojo.com/alltime/domestic.htm?page=5
    - You may need to experiment with one of the pages first to ensure that you get the right dataframe from each iteration of the loop.

```
domesticgross <- list()
for(i in 1:5){
  domesticgross[[i]] <-read_html(
```

```r
    glue("https://www.boxofficemojo.com/alltime/domestic.htm?page={i}")) %>%
    html_nodes(xpath = "//table") %>% html_table(fill=TRUE)  %>%
    extract2(6)
}
```

b. Form a single dataframe out of all the observations from the list you created.

```r
topearners <- domesticgross %>% bind_rows()
```

c. Ensure that the column names are correctly treated as column names and not as observations.

```r
colnames(topearners) <-topearners[1,]
topearners %<>%  filter(Rank !="Rank")
```

d. Rename the Title and Lifetime earnings appropriately and keep only the title, studio, and gross earnings variables.

```r
topearners %<>% rename("Title" = "Title(click to view)",
                       "Gross"="Lifetime Gross") %>%
  select(Title,Studio, Gross)
```

e. View the box office earnings dataset

```r
head(topearners)
```

```
# A tibble: 6 x 3
  Title                        Studio Gross
  <chr>                        <chr>  <chr>
1 Star Wars: The Force Awakens BV     $936,662,225
2 Avatar                       Fox    $760,507,625
3 Black Panther                BV     $700,059,566
4 Avengers: Infinity War       BV     $678,815,482
5 Titanic                      Par.   $659,363,944
6 Jurassic World               Uni.   $652,270,625
```

4. Create a dataset with both IMDB performance and earnings

   a. Join the box office earnings and IMDB top 250 datasets, keeping all variables and only the observations that found in both datasets.

```
expensive_movies <-inner_join(top250_eng, topearners, by="Title")
```

   b. Remove the dollar sign and commas from the gross earnings variable.

```
expensive_movies$Gross %<>% str_replace_all("[$,]+","")
```

   c. Vectorize the as.numeric() function to convert the ranking, gross earnings, and year variables into numeric.

```
expensive_movies[,c("Ranking","Gross","Year")] %<>% map(as.numeric)
```

   d. Create a new variable equal to the log of gross earnings

```
expensive_movies %<>%  mutate(logearnings = log(Gross))
```

   f. Write your own OLS function (producing coefficients) and use it to run a regression of log earnings on rating and year (with a constant).

   - In matrix notation, the formula for $\hat{\beta}_{OLS}$ is:

$$\hat{\beta}_{OLS} = (X'X)^{-1}(X'y)$$

   - You will need some more matrix multiplication operators for this:
     - solve(A) yields the inverse of matrix A.
     - t(A) provides the transpose of matrix A.

```
myols <- function(depvar,indvars) {
  x <- indvars %>% as_tibble()
  x %<>% as_tibble() %>% mutate(Constant = 1) %>% as.matrix()
  y <- depvar %>% as.matrix()
  beta_myols <- t(solve(t(x) %*% x) %*% (t(x) %*% y))
  colnames(beta_myols) <-colnames(x)
  rownames(beta_myols) <- "Estimate"
  beta_myols <- t(beta_myols)
  return(beta_myols)
}

myols(expensive_movies$logearnings,expensive_movies[,c("Rating","Year")])
```

```
            Estimate
Rating   0.197996254
Year     0.006279776
Constant 5.035941990
```