Problem Set 1

Author Names

2023-09-14

Problem Set 1: Replication and Organization

See the problem set instructions in the README.md.

Note: In all of these r codeblocks, I set eval=FALSE. You will need to undo that.

I also ask you to save all figures to the output folder in addition to having them appear on this .Rmd file. I am encouraging you to **practice** using folders and filepaths.

1. Open the .gitignore and add *.csv*, .dta, *.Rdata, and any other data file extensions you have in mind.

This will prevent you from pushing any large data files to GitHub. GitHub cannot receive data files that are larger than 100MB. You can use Git Large File Storage to submit larger files, but even that has limits.

2. Create folders for data, code, documentation, output, and any necessary subfolders within this repository.

Reorganize the data files and code within these folders.

Lost points for this. I mean not sure how much we need to explain, we were confused how it worked once we did that and so we gave up on that.

3. Skim the paper and also reference the non-technical summary and a New York Times report. Briefly explain how the data were generated.

4. Correct the download_data.R to download the .csv to your preferred data folder.

Done

Note: If you correctly added *.csv to the .gitignore, you will not be able to push this csv to GitHub. It will not even appear as a file that you can stage to commit. This is good! You don't want to push large data files to GitHub.

```r
school_data <- read.csv(paste0(data,'CollegeAdmissions.csv'))
```

5. First, look at the data. Any quirks about it?

```r
school_data
```

6. Look at the variables `rel_apply` and `rel_attend`. What do these mean?

ANSWER HERE

7. Check the documentation for an explanation of what each row is – use that information to calculate the number of rows that there should be in the data. Does this match the number of rows in the dataset?

Answer how you calculated the number of rows and whether it matches the number of rows in the dataset.

```
some_function_to_count_rows(school_data)
```

8. What about the variable `rel_attend_cond_app`. What does this mean? Can you verify that it is calculated correctly?

Rel_att_cond_app is the ratio of test-score-reweighted absolute and relative attendance rate to test-score-reweighted relative application rate. We can verify that it is calculated correctly by manually solving for the ratio, taking one value from rel_attend and dividing it by rel_apply and comparing it with the listed value for rel_attend_cond_app.

```
# Choose an observation
chosen_index <- 2

# Check if the ratio is equal to rel_attend_cond_app for the chosen observation
verify <- data$rel_attend[chosen_index] / data$rel_apply[chosen_index] == data$rel_attend_cond_app[chos

# Print the result
cat(ifelse(verify, "Yes", "No"))
```

install.packages(reticulate) ## 9. The codebook mentions that most of these data are test-score-reweighted. What does that mean?

To create a more accurate representation of the relationship between test-scores, parental income, and attendance rates, the test scores were reweighted to account for their distribution among students who attend "Ivy-Plus" (Ivy League, Stanford, Duke, MIT) colleges. They first construct a score specific series, representing the weights of students with different parental incomes at a specific school. Then, a weighted average of attendance rates is calculated within each income group. The weights are determined by the distribution of test scores among students at each school. The measures for each college are combined into an overall average, using enrollment weights. Finally, the overall average is divided by the overall mean of the series. This creates a more accurate representation of how parental income influences attendance rates at these colleges, considering the distribution of students and accounting for variations. (Chetty et al, pg 16)

10. Replicate Figure 2b from the paper. I recommend using the package `ggplot2`.

You'll need to check the Appendix to get the exact group of schools. Interpret the graph. Save it to your output folder.

```
## Sourcing first
source('/Users/samuelcrawford/Desktop/Big Data/big-data-PS1MaggieSam2nd/housekeeping.r')

# Load necessary libraries
library(ggplot2)
library(readr)  # for read_csv
library(dplyr)  # for data manipulation

# Reading the data
file_path <- "../data/CollegeAdmissions.csv"  # Adjust this if your file path is different
data <- read_csv(file_path)
```

```
## Rows: 1946 Columns: 81
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr  (6): name, par_income_lab, public, tier, tier_name, test_band_tier
## dbl (75): super_opeid, par_income_bin, attend, stderr_attend, attend_level, ...
##
## 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.
```

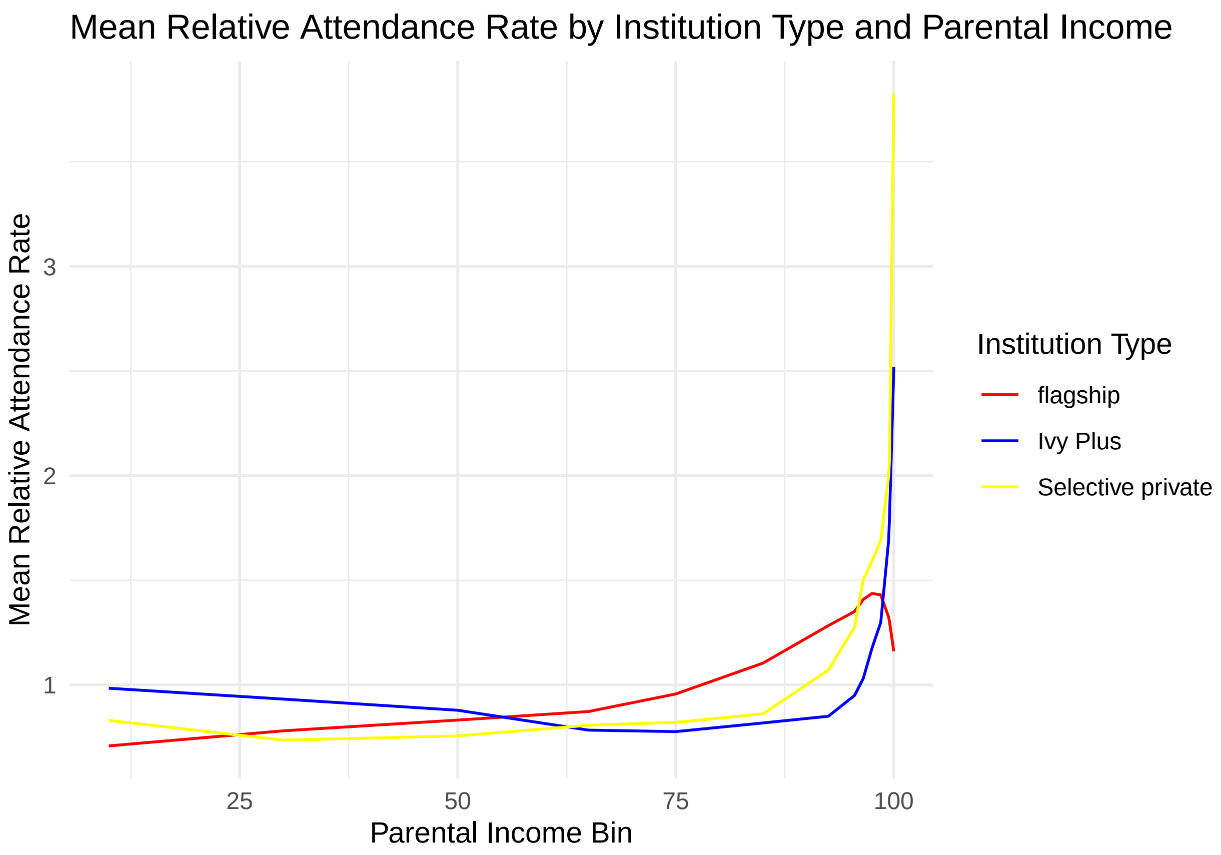
```
# Prepare the data for "flagship"
flagship_data <- data %>%
  filter(flagship == 1) %>%
  group_by(par_income_bin) %>%
  summarise(mean_rel_attend = mean(rel_attend, na.rm = TRUE))

# Prepare the data for "Ivy Plus" and "Other Selective Private" under "tier"
tier_data <- data %>%
  filter(tier %in% c("Ivy Plus", "Selective private")) %>%
  group_by(tier, par_income_bin) %>%
  summarise(mean_rel_attend = mean(rel_attend, na.rm = TRUE))
```

```
## `summarise()` has grouped output by 'tier'. You can override using the
## `.groups` argument.
```

```
# Combine the data
combined_data <- bind_rows(
  flagship_data %>% mutate(tier = "flagship"),
  tier_data
)

# Plotting
ggplot(combined_data, aes(x = par_income_bin, y = mean_rel_attend, color = tier)) +
  geom_line() +
  scale_color_manual(values = c("flagship" = "red", "Ivy Plus" = "blue", "Selective private" = "yellow")
  labs(title = "Mean Relative Attendance Rate by Institution Type and Parental Income",
       x = "Parental Income Bin",
       y = "Mean Relative Attendance Rate",
       color = "Institution Type") +
  theme_minimal()
```



```r
# Saving the plot to the output folder
output_path <- "output/mean_relative_attendance_rate_plot.png"  # Adjust this if your output path is di
ggsave(output_path, width = 10, height = 8)
```

11. Replicate Figures 3a and 3b from the paper with 95% CI bars. Save them to your output folder. Interpret these graphs.

```r
ggplot(data=school_data)
```

11. Replicate Figures 4a and 4b from the paper with 95% CI bars. Save them to your output folder. Interpret these graphs.

```r
ggplot(data=school_data)
```

12a. Plot the relative application, attendance, and matriculation rates for the schools in the NESCAC. Save it to your output folder. Interpret this graph.

Ok, firstly I'm really struggling with that Matriculation rate and how that is different from attendance rate. I've included something that might be what you are looking for, but it can't be used in relation to 12b.

```r
# Sourcing first

source('/Users/samuelcrawford/Desktop/Big Data/big-data-PS1MaggieSam2nd/housekeeping.r')
library(ggplot2)
library(readr)  # for read_csv
library(dplyr)  # for data manipulation
```

```r
# Reading the data
file_path <- "../data/CollegeAdmissions.csv"  # Adjust this if your file path is different
data <- read_csv(file_path)
```

```
## Rows: 1946 Columns: 81
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr  (6): name, par_income_lab, public, tier, tier_name, test_band_tier
## dbl (75): super_opeid, par_income_bin, attend, stderr_attend, attend_level, ...
##
## 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.
```

```r
# Define the list of colleges
selected_colleges <- c("Amherst College", "Bates College", "Bowdoin College",
                       "Colby College", "Connecticut College", "Hamilton College",
                       "Middlebury College", "Trinity College", "Tufts University",
                       "Wesleyan College", "Williams College")

# Filter the data for the selected colleges
filtered_data <- data %>%
  filter(name %in% selected_colleges)

# Create a scatter plot with lines for rel_apply
plot_apply <- ggplot(filtered_data, aes(x = par_income_bin, y = rel_apply, color = name, group = name))
  geom_point() +
  geom_line() +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1)) +
  ggtitle("rel_apply by par_income_bin")

# Create a scatter plot with lines for rel_attend
plot_attend <- ggplot(filtered_data, aes(x = par_income_bin, y = rel_attend, color = name, group = name
  geom_point() +
  geom_line() +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1)) +
  ggtitle("rel_attend by par_income_bin")
```

12b. Add 95% confidence intervals to the plot you just made.

```r
# Sourcing first
source('/Users/samuelcrawford/Desktop/Big Data/big-data-PS1MaggieSam2nd/housekeeping.r')

# Load necessary libraries
library(ggplot2)
library(readr)  # for read_csv
library(dplyr)  # for data manipulation

# Reading the data
file_path <- "../data/CollegeAdmissions.csv"  # Adjust this if your file path is different
data <- read_csv(file_path)
```

```
## Rows: 1946 Columns: 81
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr  (6): name, par_income_lab, public, tier, tier_name, test_band_tier
## dbl (75): super_opeid, par_income_bin, attend, stderr_attend, attend_level, ...
##
## 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.
```

```
# Define the list of colleges
selected_colleges <- c("Amherst College", "Bates College", "Bowdoin College",
                       "Colby College", "Connecticut College", "Hamilton College",
                       "Middlebury College", "Trinity College", "Tufts University",
                       "Wesleyan College", "Williams College")

# Filter the data for the selected colleges
filtered_data <- data %>%
  filter(name %in% selected_colleges)

# Calculate the confidence intervals for rel_attend and rel_apply
filtered_data <- filtered_data %>%
  mutate(
    lower_ci_attend = rel_attend - (1.96 * stderr_rel_attend),
    upper_ci_attend = rel_attend + (1.96 * stderr_rel_attend),
    lower_ci_apply = rel_apply - (1.96 * stderr_rel_apply),
    upper_ci_apply = rel_apply + (1.96 * stderr_rel_apply)
  )

# Create a scatter plot with error bars for rel_attend
plot_attend_ci <- ggplot(filtered_data, aes(x = par_income_bin, y = rel_attend, color = name, group = na
  geom_point() +
  geom_line() +
  geom_errorbar(aes(ymin = lower_ci_attend, ymax = upper_ci_attend), width = 0.6) +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1)) +
  ggtitle("rel_attend by par_income_bin with 95% CI")

# Create a scatter plot with error bars for rel_apply
plot_apply_ci <- ggplot(filtered_data, aes(x = par_income_bin, y = rel_apply, color = name, group = name
  geom_point() +
  geom_line() +
  geom_errorbar(aes(ymin = lower_ci_apply, ymax = upper_ci_apply), width = 0.6) +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1)) +
  ggtitle("rel_apply by par_income_bin with 95% CI")
```

13. Come up with your own cool visualization of the data. Save it to your output folder. Interpret this graph.

```
# Sourcing first
source('/Users/samuelcrawford/Desktop/Big Data/big-data-PS1MaggieSam2nd/housekeeping.r')

# Load necessary libraries
library(ggplot2)
```

```r
library(readr)  # for read_csv
library(dplyr)  # for data manipulation

# Reading the data
file_path <- "../data/CollegeAdmissions.csv"  # Adjust this if your file path is different
data <- read_csv(file_path)

# Define the list of colleges
selected_colleges <- c("Amherst College", "Bates College", "Bowdoin College",
                       "Colby College", "Connecticut College", "Hamilton College",
                       "Middlebury College", "Trinity College", "Tufts University",
                       "Wesleyan College", "Williams College")

# Filter the data for the selected colleges
filtered_data <- data %>%
  filter(name %in% selected_colleges)

# Create a scatter plot with lines for rel_att_cond_app
plot_rel_att_cond_app <- ggplot(filtered_data, aes(x = par_income_bin, y = rel_att_cond_app, color = nar
  geom_point() +
  geom_line() +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1)) +
  ggtitle("rel_att_cond_app by par_income_bin")
```

This is super interesting because the jump as wealth goes up is indicative of early decision I think. Since ED binds you to a contract, but you don't know financial assitance it means it makes up a larger proportion of high earners which is demonstrated by this chart.

14. Based on what you have seen so far, do you think that the admissions process at selective US colleges is meritocratic? What about at Bates and other NESCAC schools? Why or why not? What else would you want to know?

I think the chart that was done for Question 10 was fascinating. It shows to me that IVY PLUS are able to be more fair due to the fact that they have a more selective pool of applicants and their significant endowments. On the other hand, Selective Private clearly is more skewed towards the wealthy. Even if admissions doesn't account for income, there is stil bias and it is interesting to see that on a larger scale. On the NESCAC scale it was interesting to see the same thing take place where the "poorer" schools had higher relative attendence rates.

Hints:

Below I provide a list of handy functions/tips for R.

These `tidyverse` functions:

- `filter()` – subset the data
- `mutate()` – create new variables
- `group_by()` – group the data
- `summarise()` – calculate summary statistics
- `%>%` – pipe operator to chain together functions applied to the same data frame

And this is some useful base R:

- `c(1,2,3)` – create a vector
- `seq(1,10,by=2)` – create a sequence
- `rep(1,10)` – repeat a value
- `%in%` – check if a value is in a vector
- `signif` – round to a certain number of significant figures
- `stopifnot` – stop the code if a condition is not met. If a condition must be true, you want to stop the code before advancing.

And some useful ggplot2:

- `geom_point()` – scatterplot
- `geom_line()` – line plot
- `geom_errorbar()` – error bars
- `aes()` – the aesthetic function, which can be used to set the x and y axes, as well as the color, shape, etc. of the points
- You can save any ggplot object as a variable and then add layers to it. For example, you can save a scatterplot as `plot` and then add a line to it with `plot + geom_line()`.
- `ggsave()` – save a plot to a file.
  - You can specify the plot object and the filename.

```
plot<-ggplot(data=school_data, aes(x=rel_apply, y=rel_attend)) + geom_point()
ggsave(filename=paste0(output,'figure.pdf'),plot=plot)
```

  - If you don't specify a plot object, it will save the last plot you made:

```
ggplot(data=school_data, aes(x=rel_apply, y=rel_attend)) + geom_point()
ggplot(data=school_data, aes(x,y)) + geom_point()
#This saves the second ggplot
ggsave(filename=paste0(output,'figure.pdf'))
```

Basic statistics tip:

Use the rough formula that the upper and lower bounds of the confidence interval are:

$$\hat{\beta} \pm 1.96 \times SE(\hat{\beta})$$

where $\hat{\beta}$ is the estimated statistic and $SE(\hat{\beta})$ is the standard error of the statistic.