# STAT 413/613 HW 1

Yunting Chiu

2020-12-10

## Instructions

Admin elements:
1. Upload a photo (headshot) of yourself into your canvas profile
2. Review the Syllabus and the academic integrity code.
3. Fill in your information in the Student Info spreadsheet under the Canvas Collaboration site.

Analysis Elements: Rename the starter file under the analysis directory as `hw_01_yourname.Rmd` and use it for your solutions.
1. Modify the "author" field in the YAML header.
2. Stage and Commit R Markdown and HTML files (no PDF files).
3. **Push both .Rmd and HTML files to GitHub**.
- Make sure you have knitted to HTML prior to staging, committing, and pushing your final submission.
4. **Commit each time you answer a part of question, e.g. 1.1**
5. **Push to GitHub after each major question**, e.g., College Scorecard and World Bank Data
- **Committing and Pushing are graded elements for this homework.**
6. When complete, submit a response in Canvas

- Only include necessary code to answer the questions.
- Most of the functions you use should be from the tidyverse. Too much base R will result in point deductions.
- Use Pull requests and or email to ask me any questions. If you email, please ensure your most recent code is pushed to GitHub.

## Learning Outcomes:

- Operate with Git and GitHub.
- Apply concepts and methods from STAT 412/612.

## Canvas Picture, Syllabus, and Student Info

Review the Syllabus on Canvas and answer the following questions:

I, *enter your name* have:

1. Added a photo of myself (headshot) to my Canvas profile
2. Reviewed the syllabus and the associated policies on the following date:
3. Reviewed the American University policies on academic integrity, and understand how they apply to this course and agree to comply with them for this course
4. Filled in my information in the Student Info spreadsheet on Canvas collaborations

## College Scorecard

The data folder contains "college_score_200601.csv", a subset of the data in the College Scorecard database as of June 1, 2020. These data contain information on colleges in the United States. The variables include:

- `UNITID` and `OPEID`: Identifiers for the colleges.
- `INSTNM`: Institution name
- `ADM_RATE`: The Admission Rate.
- `SAT_AVE`: Average SAT equivalent score of students admitted.
- `UGDS`: Enrollment of undergraduate certificate/degree-seeking students
- `COSTT4_A`: Average cost of attendance (academic year institutions)
- `AVGFACSAL`: Average faculty salary
- `GRAD_DEBT_MDN`: The median debt for students who have completed
- `AGE_ENTRY`: Average age of entry
- `ICLEVEL`: Level of institution (1 = 4-year, 2 = 2-year, 3 = less than 2-year).
- `MN_EARN_WNE_P10`: Mean earnings of students working and not enrolled 10 years after entry.
- `MD_EARN_WNE_P10`: Median earnings of students working and not enrolled 10 years after entry.
- `FEMALE`: Share of female students
- `PCT_WHITE`: Percent of the population from students' zip codes that is White, via Census data

0. Libraries

```
library(tidyverse)
```

```
## -- Attaching packages --------------- tidyverse 1.3.0 --
```

```
## v ggplot2 3.3.2     v purrr   0.3.4
## v tibble  3.0.3     v dplyr   1.0.2
## v tidyr   1.1.2     v stringr 1.4.0
## v readr   1.3.1     v forcats 0.5.0
```

```
## -- Conflicts ------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(ggthemes)
library(readr)
library(dplyr)
```

1. Use a relative path and a readr function to load the data from data/college_score_200601.csv into a tibble.

```
collegeData <- read_csv(file = "../data/college_score_200601.csv")
```

```
## Parsed with column specification:
## cols(
##   UNITID = col_double(),
##   OPEID = col_double(),
##   MN_EARN_WNE_P10 = col_character(),
##   MD_EARN_WNE_P10 = col_character(),
##   INSTNM = col_character(),
##   STABBR = col_character(),
##   SAT_AVG = col_character(),
##   ADM_RATE = col_character(),
##   UGDS = col_character(),
##   COSTT4_A = col_character(),
##   AVGFACSAL = col_character(),
##   GRAD_DEBT_MDN = col_character(),
```

```
##   AGE_ENTRY = col_character(),
##   FEMALE = col_character(),
##   PCT_WHITE = col_character(),
##   ICLEVEL = col_double()
## )
```

```
head(collegeData)
```

```
## # A tibble: 6 x 16
##   UNITID  OPEID MN_EARN_WNE_P10 MD_EARN_WNE_P10 INSTNM STABBR SAT_AVG ADM_RATE
##    <dbl>  <dbl> <chr>           <chr>           <chr>  <chr>  <chr>   <chr>
## 1 100654 1.00e5 35500           31000           Alaba~ AL     957     0.8986
## 2 100663 1.05e5 48400           41200           Unive~ AL     1220    0.9211
## 3 100690 2.50e6 47600           39600           Amrid~ AL     NULL    NULL
## 4 100706 1.06e5 52000           46700           Unive~ AL     1314    0.8087
## 5 100724 1.00e5 30600           27700           Alaba~ AL     972     0.9774
## 6 100751 1.05e5 51600           44500           The U~ AL     1252    0.5906
## # ... with 8 more variables: UGDS <chr>, COSTT4_A <chr>, AVGFACSAL <chr>,
## #   GRAD_DEBT_MDN <chr>, AGE_ENTRY <chr>, FEMALE <chr>, PCT_WHITE <chr>,
## #   ICLEVEL <dbl>
```

2. If you used the default settings for reading in the data, 11 variables are probably type character when they should be numeric.

- Which ones?
- Ans: "MN_EARN_WNE_P10", "MD_EARN_WNE_P10", "SAT_AVG", "ADM_RATE", "UGDS", "COSTT4_A", "AVGFACSAL" "GRAD_DEBT_MDN", "AGE_ENTRY", "FEMALE", "PCT_WHITE". (total 11 variables)

```
map_chr(collegeData, class)
```

```
##          UNITID           OPEID MN_EARN_WNE_P10 MD_EARN_WNE_P10          INSTNM
##       "numeric"       "numeric"     "character"     "character"     "character"
##          STABBR         SAT_AVG        ADM_RATE            UGDS        COSTT4_A
##     "character"     "character"     "character"     "character"     "character"
##       AVGFACSAL   GRAD_DEBT_MDN       AGE_ENTRY          FEMALE       PCT_WHITE
##     "character"     "character"     "character"     "character"     "character"
##         ICLEVEL
##       "numeric"
```

- Why were they read in as type character?
- Ans: due to NA and PrivacySuppressed Text.

3. Fix these variables to be numeric in the tibble.

```
collegeData <- read_csv(file = "../data/college_score_200601.csv",
                        col_types = cols("MN_EARN_WNE_P10" = col_number(),
                            "MD_EARN_WNE_P10" = col_number(),
                            "SAT_AVG" = col_number(),
                            "ADM_RATE" = col_number(),
                            "UGDS" = col_number(),
                            "COSTT4_A" = col_number(),
                            "AVGFACSAL" = col_number(),
                            "GRAD_DEBT_MDN" = col_number(),
                            "AGE_ENTRY" = col_number(),
                            "FEMALE" = col_number(),
                            "PCT_WHITE" = col_number()), na = ".")
```

```
## Warning: 27004 parsing failures.
## row      col expected actual                                 file
##   3 SAT_AVG  a number   NULL '../data/college_score_200601.csv'
##   3 ADM_RATE a number   NULL '../data/college_score_200601.csv'
##   7 SAT_AVG  a number   NULL '../data/college_score_200601.csv'
##   7 ADM_RATE a number   NULL '../data/college_score_200601.csv'
##   8 SAT_AVG  a number   NULL '../data/college_score_200601.csv'
## ... ........ ........ ...... ..................................
## See problems(...) for more details.
```

```
map_chr(collegeData, class)
```

```
##          UNITID           OPEID MN_EARN_WNE_P10 MD_EARN_WNE_P10          INSTNM
##       "numeric"       "numeric"       "numeric"       "numeric"     "character"
##          STABBR         SAT_AVG        ADM_RATE            UGDS        COSTT4_A
##     "character"       "numeric"       "numeric"       "numeric"       "numeric"
##       AVGFACSAL   GRAD_DEBT_MDN       AGE_ENTRY          FEMALE       PCT_WHITE
##       "numeric"       "numeric"       "numeric"       "numeric"       "numeric"
##         ICLEVEL
##       "numeric"
```

```
#for test
# problems(collegeData) %>%
  # count(actual)
```

4. How is average faculty salary associated the median earnings of students ten years after initial enrollment? Create an appropriate plot and interpret the plot to justify your answer.

Explanation: The variables of "AVGFACSAL" and "MD_EARN_WNE_P10" have a positive relation, they will be both increased and decreased based on the same direction. Also, the blue line of this graphic has shown this tendency.

```
ggplot(data = collegeData, mapping = aes(x = AVGFACSAL, y = MD_EARN_WNE_P10))+
  geom_point(alpha = 0.2)+
  theme_bw()+
  geom_smooth(method = lm, se = FALSE)+
  labs(x = "Average Faculty Salary", y = " Median Earnings")+
  scale_y_log10()
```

```
## `geom_smooth()` using formula 'y ~ x'
```

```
## Warning: Removed 3440 rows containing non-finite values (stat_smooth).
```

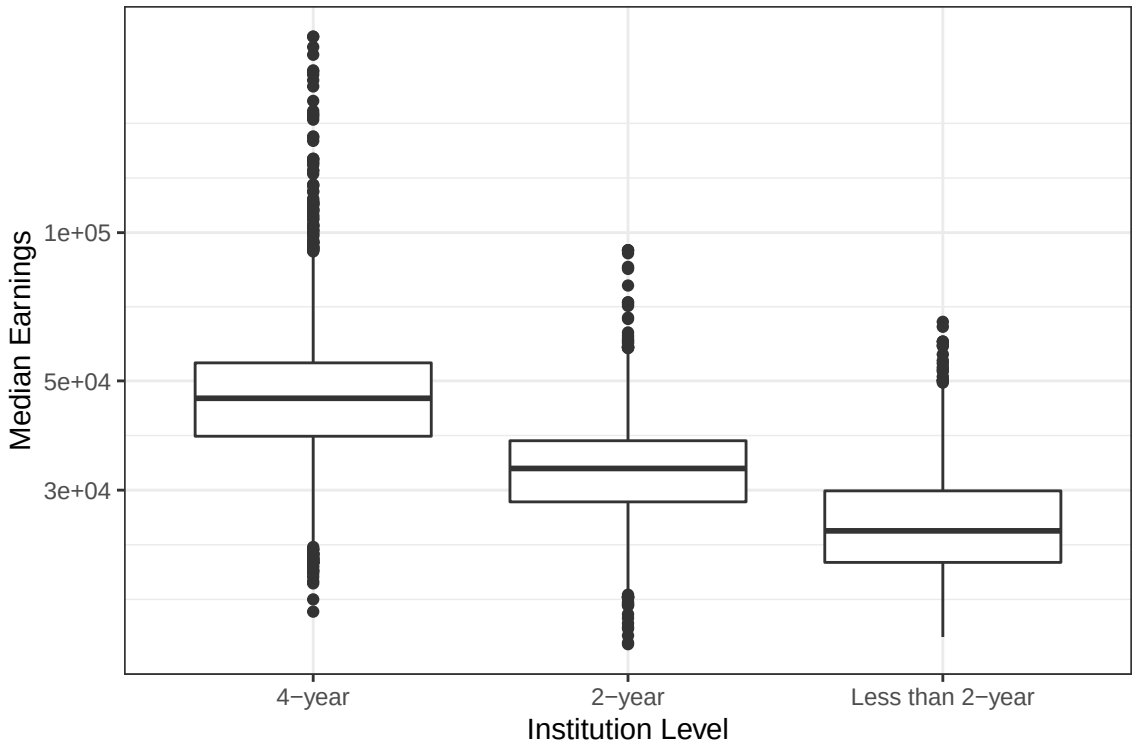
```
## Warning: Removed 3440 rows containing missing values (geom_point).
```


5. Does the level of the institution seem to be associated with the median earnings of students ten years after enrollment? Reproduce this plot in R to explore this relationship and interpret the plot:

Explanation: Yes, if the institution is 4-year, they got more earning. By contrast, If the institution is less than 2- year, they got fewer earning.

```
collegeData %>%
  mutate(ICLEVEL = as.factor(ICLEVEL),
         ICLEVEL = fct_recode(ICLEVEL,
                              "4-year" = "1",
                              "2-year" = "2",
                              "Less than 2-year" = "3"),
         ICELEVEL = fct_rev(ICLEVEL)) %>%
  ggplot(aes(x = ICLEVEL, y = MN_EARN_WNE_P10)) +
  scale_y_log10()+
  geom_boxplot()+
  theme_bw()+
  labs(x = "Institution Level", y = "Median Earnings")
```

```
## Warning: Removed 1989 rows containing non-finite values (stat_boxplot).
```




6. Plot the log of median earnings 10 years after enrollment for level 1 institutions as the Y axis against PCT_WHITE and, in a second plot, against FEMALE.

- Describe the relationship if any in each of the plots.

First plot: Even the line shows a **positive** correlation Second plot: Even the line shows a **negative** correlation and a curve would show more

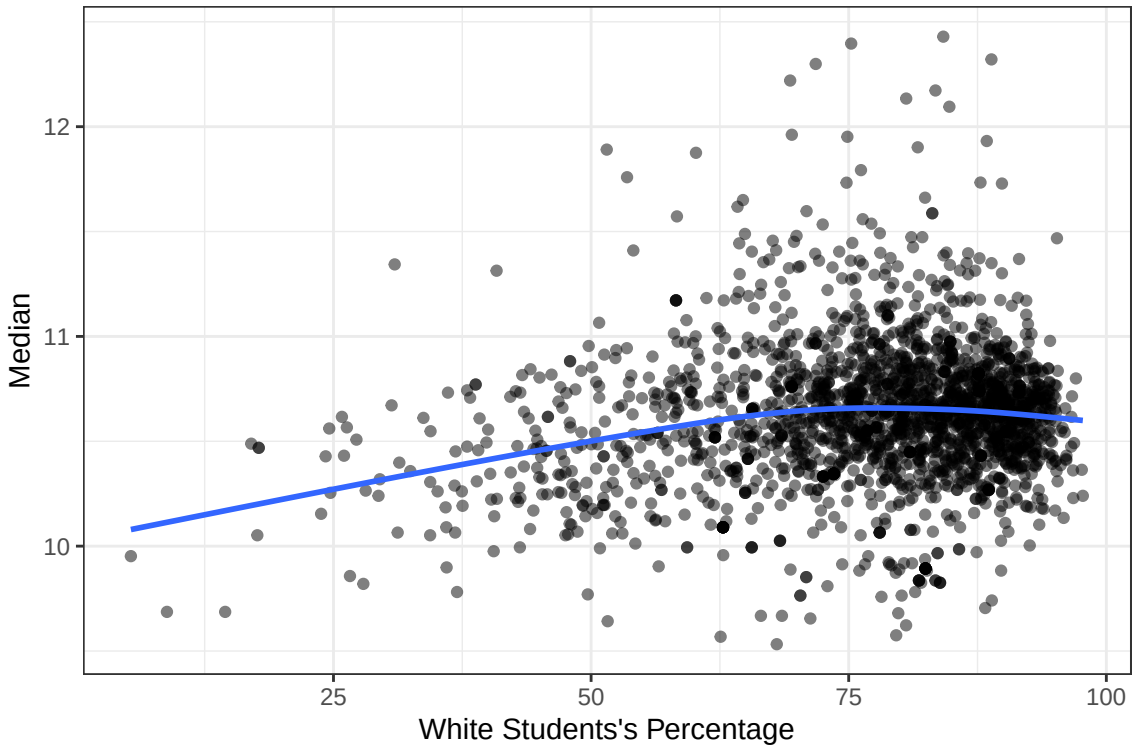
```
collegeData %>%
  filter(ICLEVEL == 1) -> levelone

#first plot
levelone %>%
  ggplot(mapping = aes(x = PCT_WHITE, y = log(MD_EARN_WNE_P10))) +
  geom_point(alpha = 0.5)+
  theme_bw()+
  geom_smooth(se = FALSE)+ # use curves to show local variation
  labs(x = "White Students's Percentage", y = "Median")
```

```
## `geom_smooth()` using method = 'gam' and formula 'y ~ s(x, bs = "cs")'
```

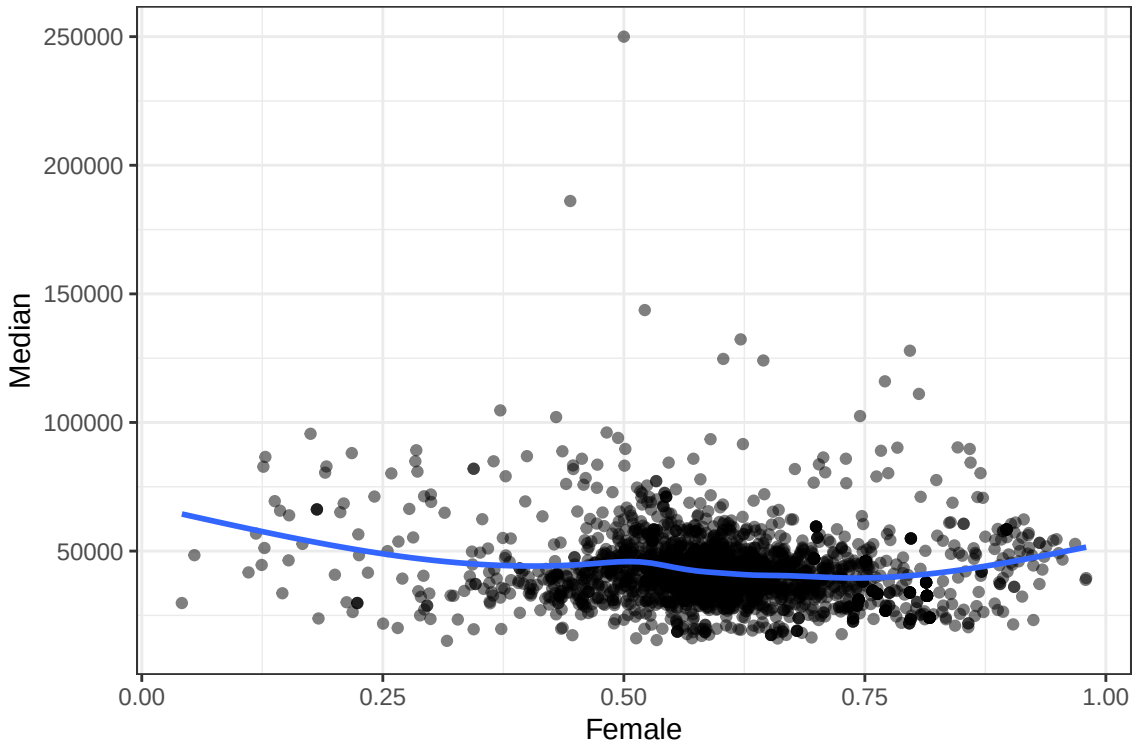
```
## Warning: Removed 973 rows containing non-finite values (stat_smooth).
```

```
## Warning: Removed 973 rows containing missing values (geom_point).
```




```
#second plot
levelone %>%
  ggplot(mapping = aes(x = FEMALE, y = MD_EARN_WNE_P10)) +
  geom_point(alpha = 0.5)+
  theme_bw()+
  geom_smooth(se = FALSE)+ # use curves to show local variation
  labs(x = "Female", y = "Median")
```

```
## `geom_smooth()` using method = 'gam' and formula 'y ~ s(x, bs = "cs")'
## Warning: Removed 798 rows containing non-finite values (stat_smooth).
## Warning: Removed 798 rows containing missing values (geom_point).
```




7. Create a scatter plot of the log of mean earnings 10 years after enrollment (Y) axis) compared to the log of median earnings 10 years after enrollment (X axis).

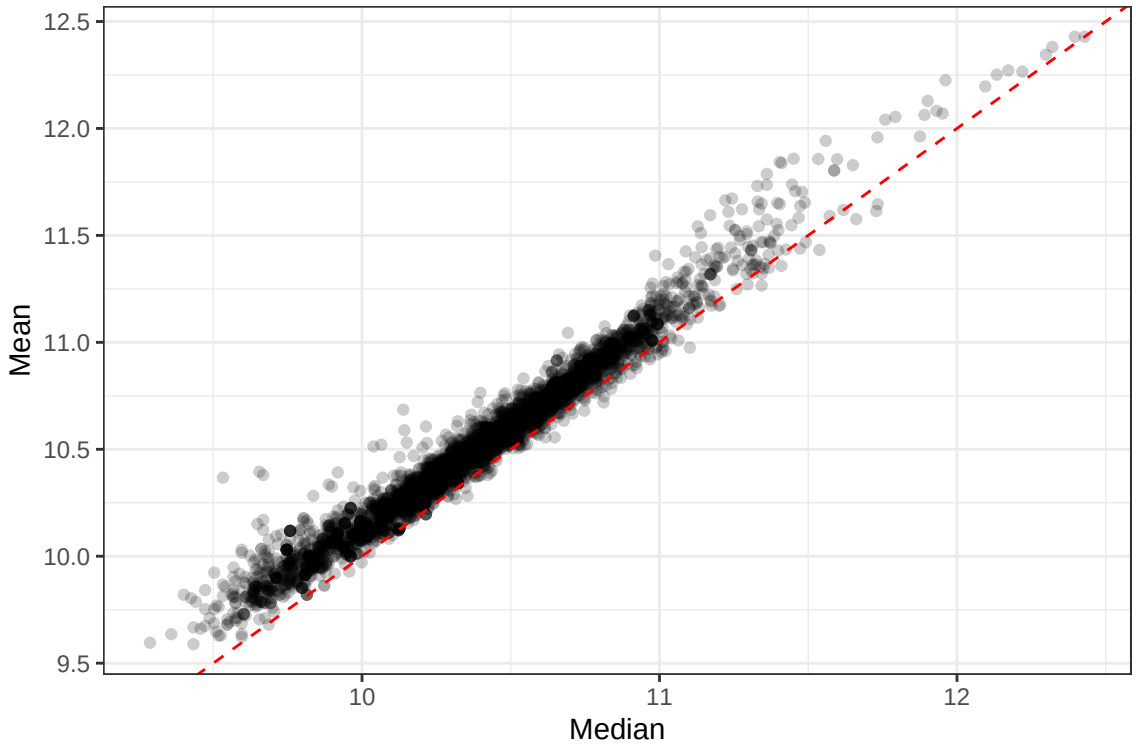
- Include an abline.
- Interpret the plot.

Ablines are usually used when you have a hypothesis X and Y have a specific ratio, here that is 1 (same rows), whereas smoother lines are more exploratory, so we use abline for this question.

Interpretation: Note that almost all of the means are above the abline and some are way above which you can interpret as the means are skewed to the right (greater) than the medians. This is not unexpected in many distributions associated with attributes like sales figures, salaries, prices etc..

```
collegeData %>%
  ggplot(mapping = aes(x = log(MD_EARN_WNE_P10), y = log(MN_EARN_WNE_P10))) +
  geom_point(alpha = 0.2)+
  geom_abline(color = "red", linetype = "dashed")+
  labs(x = "Median", y = "Mean")+
  theme_bw()
```

```
## Warning: Removed 1989 rows containing missing values (geom_point).
```




8. Compute a ranking of level 1 universities based on the ratio of median earnings 10 years after enrollment compared to median graduation debt.

We can use rank, not dense rank. That will ensure that if there are ties, the next rank will be the rank of the universities that are tied Plust the number of tied universities so in my example the one university would have a rank of 2001 not w2 ( So if 2000 universities are tied in ROI and one university is below it is rank 2, not 2001)

```
levelone %>%
  select(INSTNM, GRAD_DEBT_MDN, MD_EARN_WNE_P10) %>%
  mutate(ROI = MD_EARN_WNE_P10/GRAD_DEBT_MDN) %>% #Debt Ratio = debts / Assets
  arrange(desc(ROI)) -> DebtRatio

# remove NA in dataframe
MD_NewRanking <- DebtRatio[complete.cases(DebtRatio), ] #[row, column]
MD_NewRanking %>%
  mutate(U_rankings = (rank(-ROI))) -> MD_NewRanking  # adding ranking
MD_NewRanking %>%
  arrange(ROI) -> MD_NewRanking
head(MD_NewRanking)
```

```
## # A tibble: 6 x 5
##   INSTNM                          GRAD_DEBT_MDN MD_EARN_WNE_P10   ROI U_rankings
##   <chr>                                   <dbl>           <dbl> <dbl>      <dbl>
## 1 Martin University                       46769           24700 0.528      2293
## 2 Messenger College                       36884           19600 0.531      2292
## 3 Benedict College                        40000           25400 0.635      2291
## 4 Southwest University of Visual~         46212           30200 0.654      2290.
## 5 Southwest University of Visual~         46212           30200 0.654      2290.
## 6 Livingstone College                     35000           23400 0.669      2288
```

- Identify the top 5 best and the bottom 5 worst?

```
tail(MD_NewRanking, 5) # top 5 Universities
```

```
## # A tibble: 5 x 5
##   INSTNM                          GRAD_DEBT_MDN MD_EARN_WNE_P10   ROI U_rankings
##   <chr>                                   <dbl>           <dbl> <dbl>      <dbl>
## 1 Massachusetts Institute of Tec~         12500          104700  8.38          5
## 2 San Diego Mesa College                   4500           37800  8.4           4
## 3 Saint Augustine College                  2735           26300  9.62          3
## 4 California Institute of Techno~          8700           85900  9.87          2
## 5 SUNY Downstate Health Sciences~         12500          127900 10.2           1
```

```
head(MD_NewRanking, 5) # bottom 5 Universities
```

```
## # A tibble: 5 x 5
##   INSTNM                          GRAD_DEBT_MDN MD_EARN_WNE_P10   ROI U_rankings
##   <chr>                                   <dbl>           <dbl> <dbl>      <dbl>
## 1 Martin University                       46769           24700 0.528      2293
## 2 Messenger College                       36884           19600 0.531      2292
## 3 Benedict College                        40000           25400 0.635      2291
## 4 Southwest University of Visual~         46212           30200 0.654      2290.
## 5 Southwest University of Visual~         46212           30200 0.654      2290.
```

- What is American University's rank?

```
MD_NewRanking %>%
  filter(str_detect(INSTNM, "^American University$"))
```

```
## # A tibble: 1 x 5
##   INSTNM              GRAD_DEBT_MDN MD_EARN_WNE_P10   ROI U_rankings
##   <chr>                       <dbl>           <dbl> <dbl>      <dbl>
## 1 American University         23288           61000  2.62        402
```

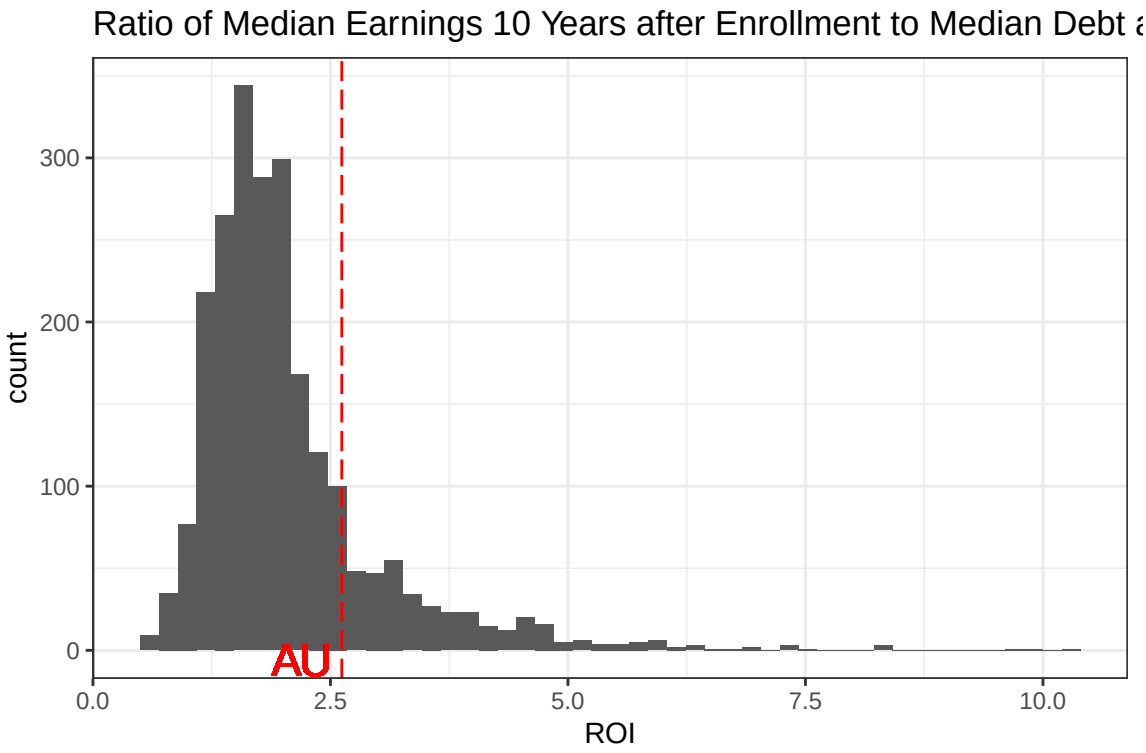
```
# AU's Rank is 402
```

- Extra Credit:
  - Reproduce the following plot so the AU line adjusts as the data adjusts:

```
MD_NewRanking %>%
  ggplot(aes(x = ROI))+
  geom_histogram(bins = 50)+
  theme_bw()+
  labs(title = "Ratio of Median Earnings 10 Years after Enrollment to Median Debt at Graduation (data f
  geom_vline(aes(xintercept = MD_NewRanking$ROI[MD_NewRanking$INSTNM == "American University"]),
             colour = "red", linetype = 5)+
  geom_text(x = 2.2, y = -6, label = "AU",
            size = 6, colour = "red")
```

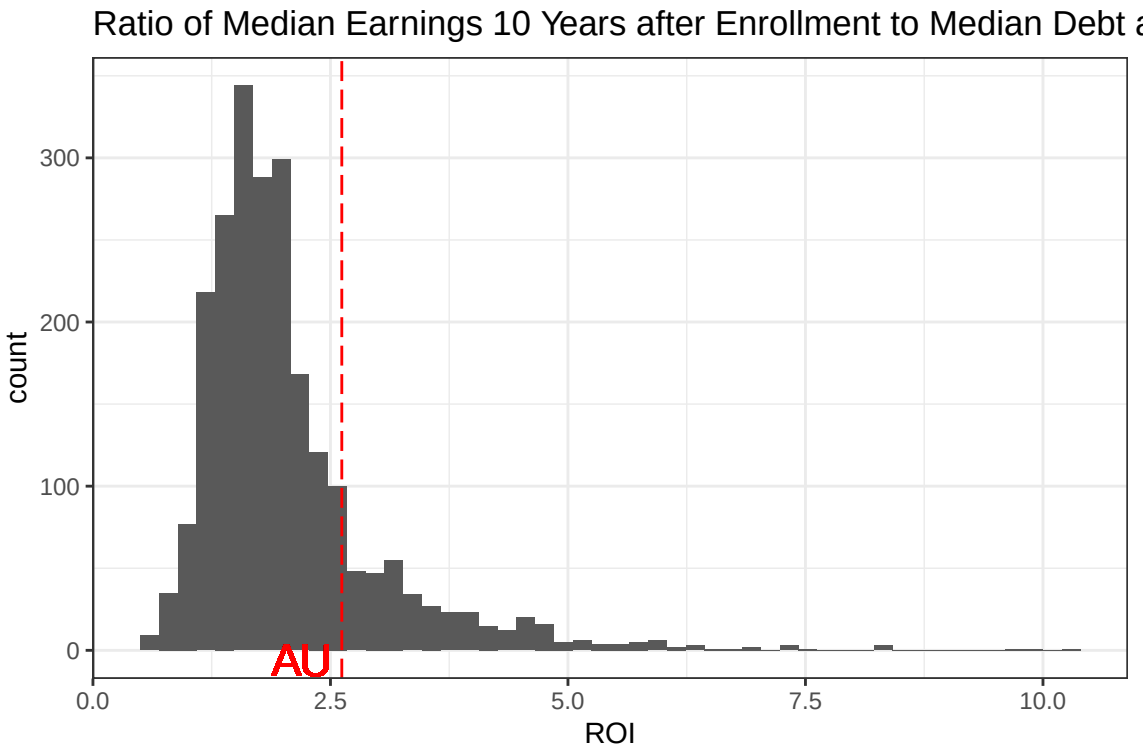
```
## Warning: Use of `MD_NewRanking$ROI` is discouraged. Use `ROI` instead.
```

```
## Warning: Use of `MD_NewRanking$INSTNM` is discouraged. Use `INSTNM` instead.
```




Another method with geom_vline

```
MD_NewRanking %>%
  ggplot(aes(x = ROI))+
  geom_histogram(bins = 50)+
  theme_bw()+
  labs(title = "Ratio of Median Earnings 10 Years after Enrollment to Median Debt at Graduation (data f
  geom_vline(aes(xintercept = ROI[INSTNM == "American University"]),
             colour = "red", linetype = 5)+
  geom_text(x = 2.2, y = -6, label = "AU",
            size = 6, colour = "red")
```




- What is AU's new ranking if the mean earnings are used?

```
# Restart using collegeData to practice tidy data
collegeData %>%
  mutate(ROI = MN_EARN_WNE_P10/GRAD_DEBT_MDN) %>%
  filter(ICLEVEL == 1, !is.na(MN_EARN_WNE_P10)) %>%
  arrange(ROI) %>%
  select(INSTNM, ROI) %>%
  filter(!is.na(ROI)) -> MN_NewRanking

MN_NewRanking %>%
  mutate(MN_NewRankings = rank(-ROI)) -> MN_NewRanking01
MN_NewRanking01 %>%
  filter(str_detect(INSTNM, "^American University$")) # Rank 408
```

```
## # A tibble: 1 x 3
##   INSTNM                ROI MN_NewRankings
##   <chr>               <dbl>          <dbl>
## 1 American University  2.91            408
```

## World Bank Data

The World Bank provides loans to countries with the goal of reducing poverty. The dataframes in the data folder were taken from the public data repositories of the World Bank.

- country.csv: Contains information on the countries in the data set.
  - The variables are:
    - `Country_Code`: A three-letter code for the country. Note not all rows are countries; some are regions.
    - `Region`: The region of the country.

    - `IncomeGroup`: Either `"High income"`, `"Upper middle income"`, `"Lower middle income"`, or `"Low income"`.
    - `TableName`: The full name of the country.
- fertility.csv: Contains the fertility rate information for each country for each year.
  - For the variables `1960` to `2017`, the values in the cells represent the fertility rate in total births per woman for that year.
  - Total fertility rate represents the number of children that would be born to a woman if she were to live to the end of her childbearing years and bear children in accordance with age-specific fertility rates of the specified year.
- life_exp.csv: Contains the life expectancy information for each country for each year.
  - For the variables `1960` to `2017`, the values in the cells represent life expectancy at birth in years for the given year.
  - Life expectancy at birth indicates the number of years a newborn infant would live if prevailing patterns of mortality at the time of its birth were to stay the same throughout its life.
- population.csv: Contains the population information for each country.
  - For the variables `1960` to `2017`, the values in the cells represent the total population in number of people for the given year.
  - Total population is based on the de facto definition of population, which counts all residents regardless of legal status or citizenship. The values shown are midyear estimates.

1. Use relative paths and a readr function to load these files into four tibbles.

```
country <- read_csv(file = "../data/country.csv")
```

```
## Parsed with column specification:
## cols(
##   `Country Code` = col_character(),
##   Region = col_character(),
##   IncomeGroup = col_character(),
##   TableName = col_character()
## )
```

```
head(country)
```

```
## # A tibble: 6 x 4
##   `Country Code` Region                    IncomeGroup         TableName
##   <chr>          <chr>                     <chr>               <chr>
## 1 ABW            Latin America & Caribbean High income         Aruba
## 2 AFG            South Asia                Low income          Afghanistan
## 3 AGO            Sub-Saharan Africa        Lower middle income Angola
## 4 ALB            Europe & Central Asia     Upper middle income Albania
## 5 AND            Europe & Central Asia     High income         Andorra
## 6 ARB            <NA>                      <NA>                Arab World
```

```
fertility <- read_csv(file = "../data/fertility.csv")
```

```
## Parsed with column specification:
## cols(
##   .default = col_double(),
##   `Country Name` = col_character(),
##   `Country Code` = col_character(),
##   `2018` = col_logical()
## )
```

```
## See spec(...) for full column specifications.
```

```r
head(fertility)
```

```
## # A tibble: 6 x 61
##   `Country Name` `Country Code` `1960` `1961` `1962` `1963` `1964` `1965` `1966`
##   <chr>          <chr>           <dbl>  <dbl>  <dbl>  <dbl>  <dbl>  <dbl>  <dbl>
## 1 Aruba          ABW              4.82   4.66   4.47   4.27   4.06   3.84   3.62
## 2 Afghanistan    AFG              7.45   7.45   7.45   7.45   7.45   7.45   7.45
## 3 Angola         AGO              7.48   7.52   7.56   7.59   7.61   7.62   7.62
## 4 Albania        ALB              6.49   6.40   6.28   6.13   5.96   5.77   5.58
## 5 Andorra        AND             NA     NA     NA     NA     NA     NA     NA
## 6 Arab World     ARB              6.95   6.97   6.99   7.01   7.02   7.02   7.02
## # ... with 52 more variables: `1967` <dbl>, `1968` <dbl>, `1969` <dbl>,
## #   `1970` <dbl>, `1971` <dbl>, `1972` <dbl>, `1973` <dbl>, `1974` <dbl>,
## #   `1975` <dbl>, `1976` <dbl>, `1977` <dbl>, `1978` <dbl>, `1979` <dbl>,
## #   `1980` <dbl>, `1981` <dbl>, `1982` <dbl>, `1983` <dbl>, `1984` <dbl>,
## #   `1985` <dbl>, `1986` <dbl>, `1987` <dbl>, `1988` <dbl>, `1989` <dbl>,
## #   `1990` <dbl>, `1991` <dbl>, `1992` <dbl>, `1993` <dbl>, `1994` <dbl>,
## #   `1995` <dbl>, `1996` <dbl>, `1997` <dbl>, `1998` <dbl>, `1999` <dbl>,
## #   `2000` <dbl>, `2001` <dbl>, `2002` <dbl>, `2003` <dbl>, `2004` <dbl>,
## #   `2005` <dbl>, `2006` <dbl>, `2007` <dbl>, `2008` <dbl>, `2009` <dbl>,
## #   `2010` <dbl>, `2011` <dbl>, `2012` <dbl>, `2013` <dbl>, `2014` <dbl>,
## #   `2015` <dbl>, `2016` <dbl>, `2017` <dbl>, `2018` <lgl>
```

```r
life_exp <- read_csv(file = "../data/life_exp.csv")
```

```
## Parsed with column specification:
## cols(
##   .default = col_double(),
##   `Country Name` = col_character(),
##   `Country Code` = col_character(),
##   `2018` = col_logical()
## )
## See spec(...) for full column specifications.
```

```r
head(life_exp)
```

```
## # A tibble: 6 x 61
##   `Country Name` `Country Code` `1960` `1961` `1962` `1963` `1964` `1965` `1966`
##   <chr>          <chr>           <dbl>  <dbl>  <dbl>  <dbl>  <dbl>  <dbl>  <dbl>
## 1 Aruba          ABW              65.7   66.1   66.4   66.8   67.1   67.4   67.8
## 2 Afghanistan    AFG              32.3   32.7   33.2   33.6   34.1   34.5   34.9
## 3 Angola         AGO              33.3   33.6   33.9   34.3   34.6   35.0   35.4
## 4 Albania        ALB              62.3   63.3   64.2   64.9   65.5   65.8   66.1
## 5 Andorra        AND             NA     NA     NA     NA     NA     NA     NA
## 6 Arab World     ARB              46.8   47.4   48.0   48.6   49.2   49.7   50.3
## # ... with 52 more variables: `1967` <dbl>, `1968` <dbl>, `1969` <dbl>,
## #   `1970` <dbl>, `1971` <dbl>, `1972` <dbl>, `1973` <dbl>, `1974` <dbl>,
## #   `1975` <dbl>, `1976` <dbl>, `1977` <dbl>, `1978` <dbl>, `1979` <dbl>,
## #   `1980` <dbl>, `1981` <dbl>, `1982` <dbl>, `1983` <dbl>, `1984` <dbl>,
## #   `1985` <dbl>, `1986` <dbl>, `1987` <dbl>, `1988` <dbl>, `1989` <dbl>,
## #   `1990` <dbl>, `1991` <dbl>, `1992` <dbl>, `1993` <dbl>, `1994` <dbl>,
## #   `1995` <dbl>, `1996` <dbl>, `1997` <dbl>, `1998` <dbl>, `1999` <dbl>,
## #   `2000` <dbl>, `2001` <dbl>, `2002` <dbl>, `2003` <dbl>, `2004` <dbl>,
## #   `2005` <dbl>, `2006` <dbl>, `2007` <dbl>, `2008` <dbl>, `2009` <dbl>,
## #   `2010` <dbl>, `2011` <dbl>, `2012` <dbl>, `2013` <dbl>, `2014` <dbl>,
```

```
## #   `2015` <dbl>, `2016` <dbl>, `2017` <dbl>, `2018` <lgl>
```

```
population <- read_csv(file = "../data/population.csv")
```

```
## Parsed with column specification:
## cols(
##   .default = col_double(),
##   `Country Name` = col_character(),
##   `Country Code` = col_character(),
##   `2018` = col_logical()
## )
## See spec(...) for full column specifications.
```

```
head(population)
```

```
## # A tibble: 6 x 61
##   `Country Name` `Country Code` `1960` `1961` `1962` `1963` `1964` `1965` `1966`
##   <chr>          <chr>           <dbl>  <dbl>  <dbl>  <dbl>  <dbl>  <dbl>  <dbl>
## 1 Aruba          ABW            5.42e4 5.54e4 5.62e4 5.67e4 5.70e4 5.74e4 5.77e4
## 2 Afghanistan    AFG            9.00e6 9.17e6 9.35e6 9.53e6 9.73e6 9.94e6 1.02e7
## 3 Angola         AGO            5.64e6 5.75e6 5.87e6 5.98e6 6.09e6 6.20e6 6.31e6
## 4 Albania        ALB            1.61e6 1.66e6 1.71e6 1.76e6 1.81e6 1.86e6 1.91e6
## 5 Andorra        AND            1.34e4 1.44e4 1.54e4 1.64e4 1.75e4 1.85e4 1.96e4
## 6 Arab World     ARB            9.25e7 9.50e7 9.77e7 1.00e8 1.03e8 1.06e8 1.09e8
## # ... with 52 more variables: `1967` <dbl>, `1968` <dbl>, `1969` <dbl>,
## #   `1970` <dbl>, `1971` <dbl>, `1972` <dbl>, `1973` <dbl>, `1974` <dbl>,
## #   `1975` <dbl>, `1976` <dbl>, `1977` <dbl>, `1978` <dbl>, `1979` <dbl>,
## #   `1980` <dbl>, `1981` <dbl>, `1982` <dbl>, `1983` <dbl>, `1984` <dbl>,
## #   `1985` <dbl>, `1986` <dbl>, `1987` <dbl>, `1988` <dbl>, `1989` <dbl>,
## #   `1990` <dbl>, `1991` <dbl>, `1992` <dbl>, `1993` <dbl>, `1994` <dbl>,
## #   `1995` <dbl>, `1996` <dbl>, `1997` <dbl>, `1998` <dbl>, `1999` <dbl>,
## #   `2000` <dbl>, `2001` <dbl>, `2002` <dbl>, `2003` <dbl>, `2004` <dbl>,
## #   `2005` <dbl>, `2006` <dbl>, `2007` <dbl>, `2008` <dbl>, `2009` <dbl>,
## #   `2010` <dbl>, `2011` <dbl>, `2012` <dbl>, `2013` <dbl>, `2014` <dbl>,
## #   `2015` <dbl>, `2016` <dbl>, `2017` <dbl>, `2018` <lgl>
```

2. These data are messy. The observational units in `fert`, `life`, and `pop` are locations in space-time (e.g. Aruba in 2017). Recall tidy data should have one observational unit per row.

- Tidy these three tibbles.
- Make sure the variable for `year` is a numeric.

```
fertility %>%
  pivot_longer(cols = "1960":"2018",names_to = "Year",
               values_to = "fertility_rate", values_ptypes = list(factor())) %>%
  mutate(Year = parse_number(Year)) -> fertilityTidy # make sure "Year is numeric
head(fertilityTidy)
```

```
## # A tibble: 6 x 4
##   `Country Name` `Country Code`  Year fertility_rate
##   <chr>          <chr>          <dbl>          <dbl>
## 1 Aruba          ABW             1960           4.82
## 2 Aruba          ABW             1961           4.66
## 3 Aruba          ABW             1962           4.47
## 4 Aruba          ABW             1963           4.27
## 5 Aruba          ABW             1964           4.06
```

```
## 6 Aruba          ABW             1965           3.84
```

```
life_exp %>%
  pivot_longer(cols = "1960":"2018",names_to = "Year",
                values_to = "life_expectancy", values_ptypes = list(factor())) %>%
  mutate(Year = parse_number(Year)) -> life_expTidy # make sure "Year is numeric
head(life_expTidy)
```

```
## # A tibble: 6 x 4
##   `Country Name` `Country Code`  Year life_expectancy
##   <chr>          <chr>          <dbl>           <dbl>
## 1 Aruba          ABW             1960            65.7
## 2 Aruba          ABW             1961            66.1
## 3 Aruba          ABW             1962            66.4
## 4 Aruba          ABW             1963            66.8
## 5 Aruba          ABW             1964            67.1
## 6 Aruba          ABW             1965            67.4
```

```
population %>%
    pivot_longer(cols = "1960":"2018",names_to = "Year",
                values_to = "population", values_ptypes = list(factor())) %>%
  mutate(Year = parse_number(Year)) -> populationTidy # make sure "Year is numeric
head(populationTidy)
```

```
## # A tibble: 6 x 4
##   `Country Name` `Country Code`  Year population
##   <chr>          <chr>          <dbl>      <dbl>
## 1 Aruba          ABW             1960      54211
## 2 Aruba          ABW             1961      55438
## 3 Aruba          ABW             1962      56225
## 4 Aruba          ABW             1963      56695
## 5 Aruba          ABW             1964      57032
## 6 Aruba          ABW             1965      57360
```

3. Combine the tibbles to create a new tibble which includes the fertility rate, population, and life expectancy in each year as well as the region for each country.

(Not a good idea to get rid of all rows that have an NA in one of the columns as you may not need that column in the analysis., It really shortens your data set unnecessarily)

```
country %>%
left_join(fertilityTidy, by = "Country Code") %>%
left_join(life_expTidy, by = c("Country Code", "Year", "Country Name")) %>%
left_join(populationTidy, by = c("Country Code", "Year", "Country Name")) %>%
  rename(Country = "Country Name") -> WBdata

# remove NA in dataframe
WBdata_noNA <- WBdata[complete.cases(WBdata), ] #[row, column]
head(WBdata_noNA)
```

```
## # A tibble: 6 x 9
##   `Country Code` Region IncomeGroup TableName Country  Year fertility_rate
##   <chr>          <chr>  <chr>       <chr>     <chr>   <dbl>          <dbl>
## 1 ABW            Latin~ High income Aruba     Aruba    1960           4.82
## 2 ABW            Latin~ High income Aruba     Aruba    1961           4.66
## 3 ABW            Latin~ High income Aruba     Aruba    1962           4.47
## 4 ABW            Latin~ High income Aruba     Aruba    1963           4.27
```

```
## 5 ABW            Latin~ High income Aruba     Aruba    1964           4.06
## 6 ABW            Latin~ High income Aruba     Aruba    1965           3.84
## # ... with 2 more variables: life_expectancy <dbl>, population <dbl>
```

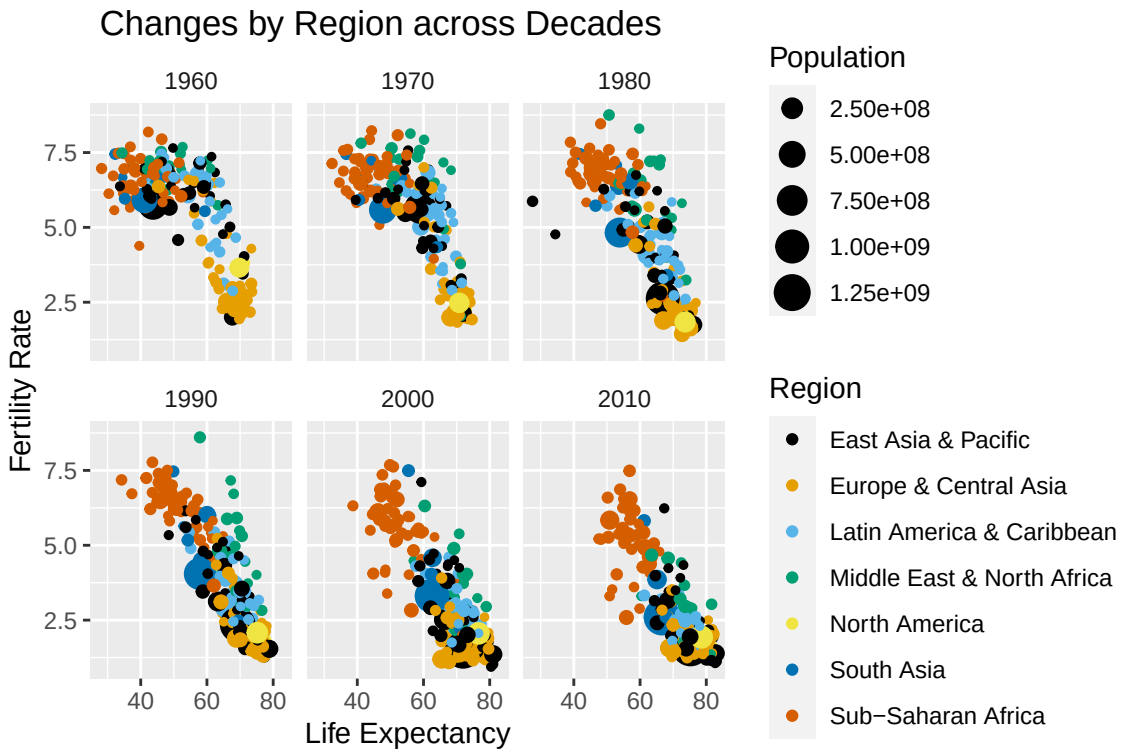
```
nrow(WBdata_noNA) # check rows of data frame
```

```
## [1] 11291
```

4. Make a scatterplot of fertility rate vs life expectancy, color-coding by region and annotating size by the population.

- Include only the years 1960, 1970, 1980, 1990, 2000, and 2010.
- Facet by these years.
- Your final plot should look like this (Each element of the formatting is graded):
- Hint: use `ggthemes`
- **Interpret the plot in one sentence**.
- As time goes by, the life expectancy of people has been increased in the world. (should also discuss fertility rate)

```
WBdata_noNA %>%
  filter(Year == 1960 | Year == 1970 | Year == 1980 | Year == 1990 | Year == 2000 | Year == 2010) %>%
  rename("Population" = population) %>%
 ggplot(mapping = aes(x = life_expectancy, y = fertility_rate ,
                      color = Region, size = Population), na.rm = TRUE)+
 geom_point()+
  ggtitle(" Changes by Region across Decades")+
xlab("Life Expectancy")+
  ylab("Fertility Rate")+
  scale_color_colorblind() +
facet_wrap(~Year)+
theme(strip.background = element_blank(), strip.placement = "outside")
```



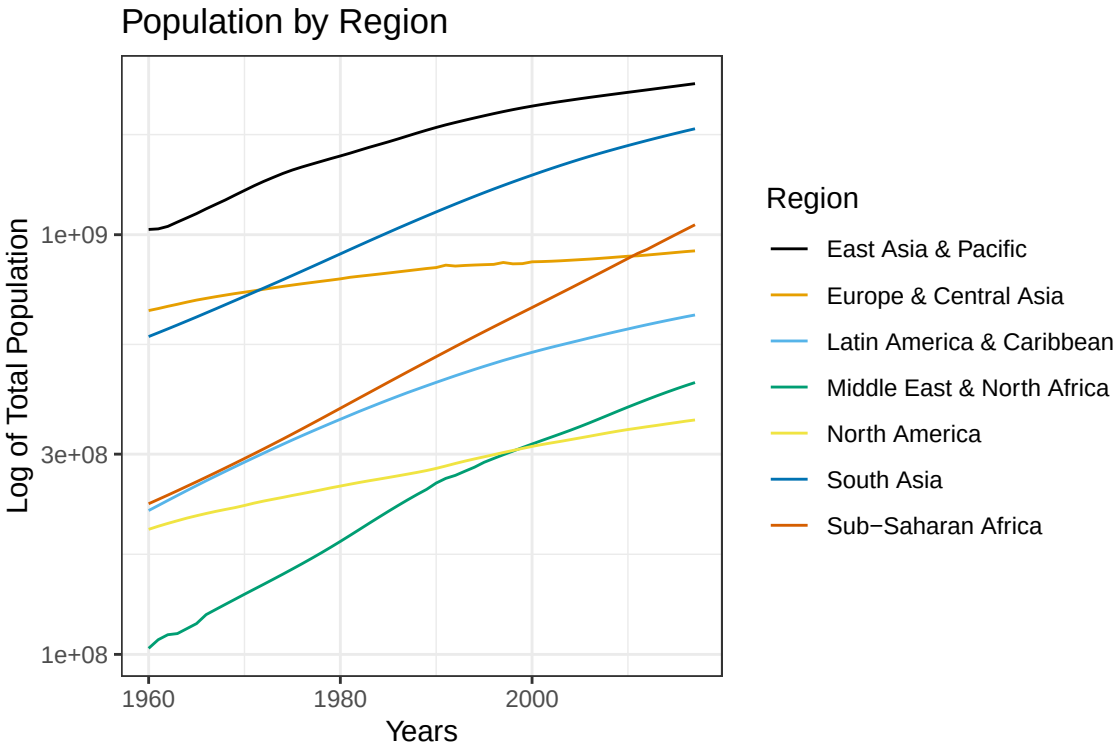

5. Calculate the total population for each region for each year.

- Exclude 2018.

- Make a line plot of year versus log of total population, color-coding by region.
- Your final plot should look like this:
- **Interpret the plot in one sentence**.
- The population of the world is rapidly increasing

```
WBdata_noNA %>%
  select(Region, Year, population) %>%
  filter(Year != 2018) %>%
  group_by(Year, Region) %>%
  mutate(ttl_population = sum(population), na.rm = TRUE) -> WBdata_noNA_no2018

  ggplot(data = WBdata_noNA_no2018, mapping = aes(x = Year, y = ttl_population, color = Region))+
    geom_line()+
    labs(x = "Years", y = "Log of Total Population")+
    ggtitle("Population by Region")+
    theme_bw()+
    scale_y_log10()+
    scale_color_colorblind()
```

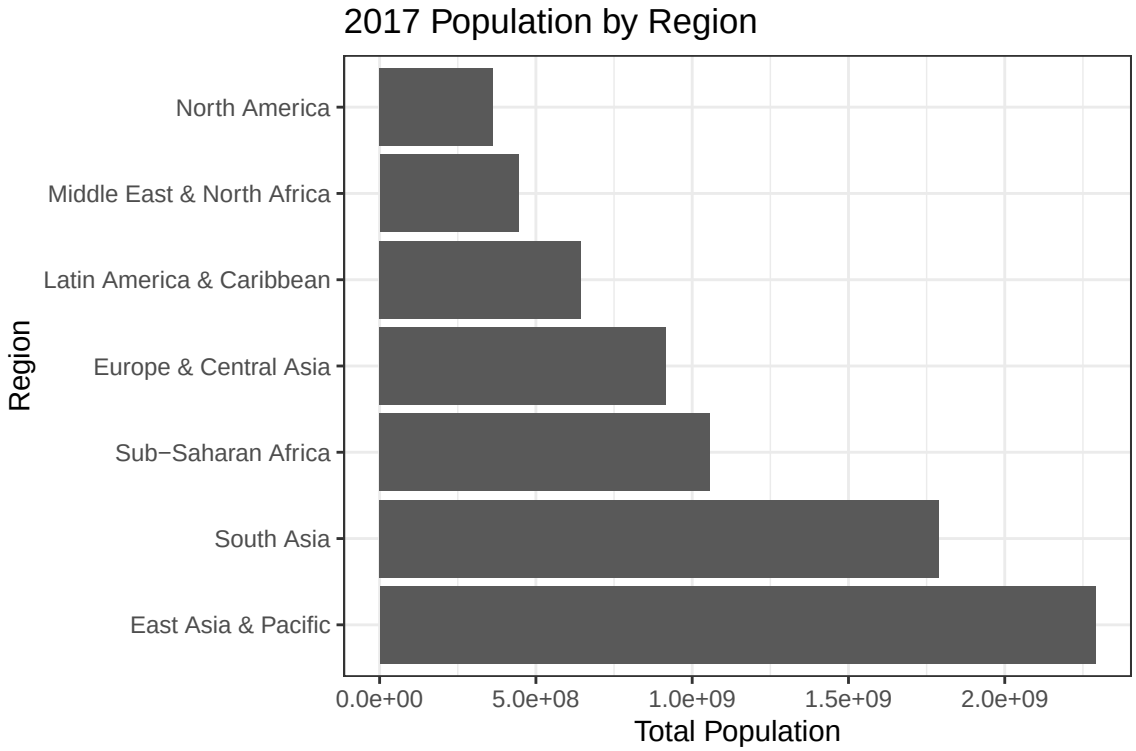



6. Make a bar plot of population vs region for the year 2017.

- Order the bars on the $y$-axis in **decreasing** order of population.
- Your final plot should look like this:

```
WBdata_noNA %>%
  select(Region, population, Year) %>%
  filter(Year == 2017) %>%
  group_by(Region) %>%
  summarise(ttl_population = sum(population)) %>%
  ggplot(mapping = aes(x = reorder(Region, -ttl_population), y = ttl_population))+
  geom_bar(stat = "identity")+
  coord_flip()+
    ggtitle("2017 Population by Region")+
    xlab("Region")+
    ylab("Total Population")+
  theme_bw()
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```


2017 Population by Region
North America
Middle East & North Africa
Latin America & Caribbean
Europe & Central Asia
Sub−Saharan Africa
South Asia
East Asia & Pacific
Region
0.0e+00
5.0e+08
1.0e+09
1.5e+09
2.0e+09
Total Population