Assignment 1

Due at 11:59pm on September 17.

Turn in this assignment as an HTML or PDF file to ELMS. Make sure to include the R Markdown or Quarto file that was used to generate it. You should include the questions in your solutions. You may use the qmd file of the assignment provided to insert your answers.

Git and GitHub

1) Provide the link to the GitHub repo that you used to practice git from Week 1. It should have:

- ☒ Your name on the README file.
- ☒ At least one commit with your name, with a description of what you did in that commit.
- Link to the repo.

Reading Data

Download both the Angell.dta (Stata data format) dataset and the Angell.txt dataset from this website: https://stats.idre.ucla.edu/stata/examples/ara/applied-regression-analysis-by-fox-data-files/

2) Read in the .dta version and store in an object called `angell_stata`.

```
angell_stata <- read_stata(
  "https://stats.oarc.ucla.edu/stat/stata/examples/ara/angell.dta")
glimpse(angell_stata)
```

```
Rows: 43
Columns: 5
$ city   <chr> "Rochester", "Syracuse", "Worcester", "Erie", "Milwaukee", "Bri~
$ morint <dbl> 19.0, 17.0, 16.4, 16.2, 15.8, 15.3, 15.2, 14.3, 14.2, 14.1, 14.~
$ ethhet <dbl> 20.6, 15.6, 22.1, 14.0, 17.4, 27.9, 22.3, 23.7, 10.6, 12.7, 39.~
$ geomob <dbl> 15.0, 20.2, 13.6, 14.8, 17.6, 17.5, 14.7, 23.8, 19.4, 31.9, 18.~
$ region <chr> "E", "E", "E", "E", "MW", "E", "E", "MW", "E", "MW", "MW", "W",~
```

3) Read in the .txt version and store it in an object called `angell_txt`.

```
angell_txt <- read_table2(
  "https://stats.idre.ucla.edu/wp-content/uploads/2016/02/angell.txt",
  col_names = FALSE, # there are not column names in the first row
  col_types = list("c", "n", "n", "n", "c")) # we set the class types for each variable
```

4) What are the differences between `angell_stata` and `angell_txt`? Are there differences in the classes of the individual columns?

- .dta files are proprietary to Stata, a statistical software package, while .txt files are general open source files that can be opened through any text reader. DTA files, on the other hand need a special package that can open this file, and these cannot be open through text file editors unless there is a plug in such as in visual studio code.
- Stata data files preserve the classes for each variable, additionally they also include label(s) for each value (i.e., "What is your Gender":Gender 1=male, 2=female), while text files do not and require additional code when reading in this type of file.
- Another issue with text files is that the practitioner needs to know if the first row contains variable names or not, otherwise it must be identified in the argument as well as names must be provided at some point.

5) Make any updates necessary so that `angell_txt` is the same as `angell_stata`.

- When possible, we would like to minimize the amount of code and processing time while reading in data. Therefore, I changed the classes for each variable during the read in data phase. Next I use the same column names from the stata file to name the unnamed columns in the txt file.

```
angell_txt<- angell_txt |>
  set_names(colnames(angell_stata))

glimpse(angell_txt)
```

```
Rows: 43
Columns: 5
$ city   <chr> "Rochester", "Syracuse", "Worcester", "Erie", "Milwaukee", "Bri~
$ morint <dbl> 19.0, 17.0, 16.4, 16.2, 15.8, 15.3, 15.2, 14.3, 14.2, 14.1, 14.~
$ ethhet <dbl> 20.6, 15.6, 22.1, 14.0, 17.4, 27.9, 22.3, 23.7, 10.6, 12.7, 39.~
$ geomob <dbl> 15.0, 20.2, 13.6, 14.8, 17.6, 17.5, 14.7, 23.8, 19.4, 31.9, 18.~
$ region <chr> "E", "E", "E", "E", "MW", "E", "E", "MW", "E", "MW", "MW", "W",~
```

6) Describe the Ethnic Heterogeneity variable. Use descriptive statistics such as mean, median, standard deviation, etc. How does it differ by region?

- Ethnic heterogeneity is the percent of nonwhite and foreign-born white residents in a city.
- Here we see that the average ethnic heterogeneity is 31.4 percent (median = 23.7) with a standard deviation of 20.4 percent, and widely ranges between 10.6 and 84.5 percent.

```
angell_stata |>
  summarise(across(ethhet, list(min, mean, sd, median, max))) |> # summary statistics
  mutate(across(where(is.numeric), round, digits=1)) |> # round 1 decimal point
  rename(min_ethhet=1, mean_ethhet=2, sd_ethhet=3, median_ethhet=4, max_ethhet=5) # rename ou
```

```
# A tibble: 1 x 5
  min_ethhet mean_ethhet sd_ethhet median_ethhet max_ethhet
       <dbl>       <dbl>     <dbl>         <dbl>      <dbl>
1       10.6        31.4      20.4          23.7       84.5
```

- When we look at ethnic heterogeneity across regions, we find that the South tends to be more ethnically heterogeneous at 20.7 percent than other regions, yet the standard deviation of 52.5 percent suggest that there are cities within this region that are vastly different on this variable.
- Non Southern regions tend to be more similar on ethnic heterogeneity with some variation, While the West appears to be skewed based on the Mean (12.3) being distinct from the Median (4.2).

```
angell_stata |>
  group_by(region) |>
  reframe(across(ethhet, list(min, mean, sd, median, max))) |>
  mutate(across(where(is.numeric), round, digits=1)) |>
  rename(min_ethhet=1, mean_ethhet=2, sd_ethhet=3, median_ethhet=4, max_ethhet=5) |>
  arrange(desc(mean_ethhet))
```

```
# A tibble: 4 x 6
  min_ethhet mean_ethhet sd_ethhet median_ethhet max_ethhet ethhet_5
  <chr>            <dbl>     <dbl>         <dbl>      <dbl>    <dbl>
1 S                 20.7      52.5          21.4       53.8     84.5
2 W                 12.3      16.5           4.2       16.1     23.9
3 MW                10.7      21.7           9.1       19.2     39.7
4 E                 10.6      23.5          10.8       22.1     45.8
```

Describing Data

R comes also with many built-in datasets. The "MASS" package, for example, comes with the "Boston" dataset.

7) Install the "MASS" package, load the package. Then, load the Boston dataset.

```
require(MASS)
data(Boston)
glimpse(Boston)
```

```
Rows: 506
Columns: 14
$ crim    <dbl> 0.00632, 0.02731, 0.02729, 0.03237, 0.06905, 0.02985, 0.08829,~
$ zn      <dbl> 18.0, 0.0, 0.0, 0.0, 0.0, 0.0, 12.5, 12.5, 12.5, 12.5, 12.5, 1~
$ indus   <dbl> 2.31, 7.07, 7.07, 2.18, 2.18, 2.18, 7.87, 7.87, 7.87, 7.87, 7.~
$ chas    <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,~
$ nox     <dbl> 0.538, 0.469, 0.469, 0.458, 0.458, 0.458, 0.524, 0.524, 0.524,~
$ rm      <dbl> 6.575, 6.421, 7.185, 6.998, 7.147, 6.430, 6.012, 6.172, 5.631,~
$ age     <dbl> 65.2, 78.9, 61.1, 45.8, 54.2, 58.7, 66.6, 96.1, 100.0, 85.9, 9~
$ dis     <dbl> 4.0900, 4.9671, 4.9671, 6.0622, 6.0622, 6.0622, 5.5605, 5.9505~
$ rad     <int> 1, 2, 2, 3, 3, 3, 5, 5, 5, 5, 5, 5, 5, 4, 4, 4, 4, 4, 4, 4, 4,~
$ tax     <dbl> 296, 242, 242, 222, 222, 222, 311, 311, 311, 311, 311, 311, 31~
$ ptratio <dbl> 15.3, 17.8, 17.8, 18.7, 18.7, 18.7, 15.2, 15.2, 15.2, 15.2, 15~
$ black   <dbl> 396.90, 396.90, 392.83, 394.63, 396.90, 394.12, 395.60, 396.90~
$ lstat   <dbl> 4.98, 9.14, 4.03, 2.94, 5.33, 5.21, 12.43, 19.15, 29.93, 17.10~
$ medv    <dbl> 24.0, 21.6, 34.7, 33.4, 36.2, 28.7, 22.9, 27.1, 16.5, 18.9, 15~
```

8) What is the type of the Boston object?

- The type of the Boston object is a list

```
typeof(Boston)
```

```
[1] "list"
```

9) What is the class of the Boston object?

- The class of the Boston object is a data frame

```
class(Boston)
```

```
[1] "data.frame"
```

10) How many of the suburbs in the Boston data set bound the Charles river?

- First we need to call the codebook to find the variables of interest using the help function:
- We find that *chas* is a dummy variable with 1 = bounds the Charles river, 0 = does not.
- We assume that rows corresponds to suburbs.

```
?Boston
```

- We can use the dplyr::count() function to get a count of how many suburbs bound the Charles river, and we find that 35 of the 506 bound the river.

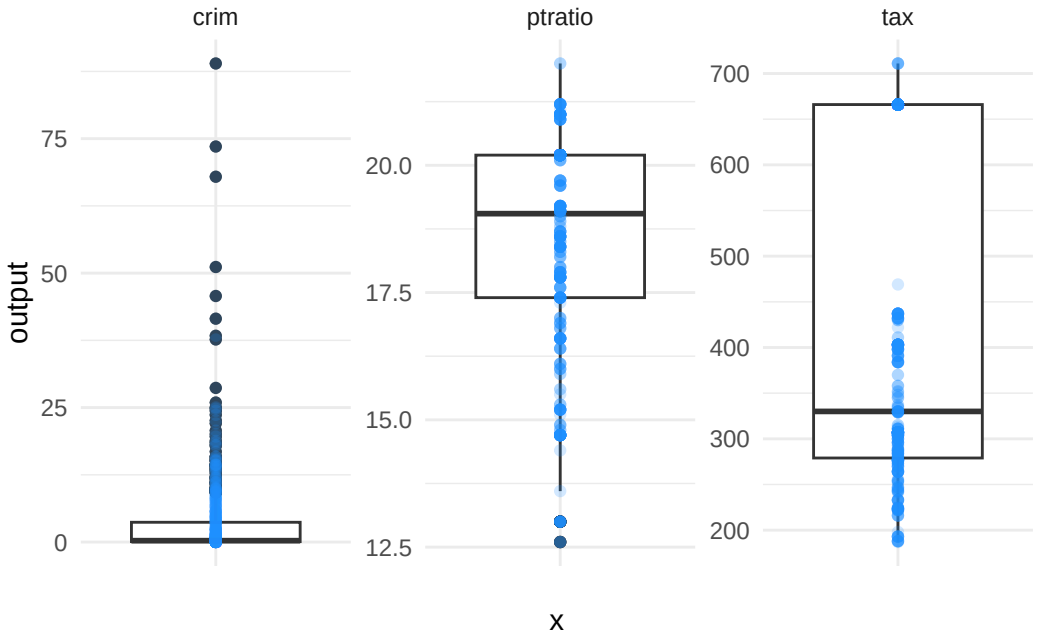
```
Boston |> count(chas)
```

```
  chas   n
1    0 471
2    1  35
```

11) Do any of the suburbs of Boston appear to have particularly high crime rates? Tax rates? Pupil-teacher ratios? Comment on the range of each variable.

- Using boxplots we can see that several suburbs have an above average crime rate per capita with a range between 0 to 89 reported crimes per, most likely, 1000. There are at least two suburbs that have high full value property tax rate per $10,000 with a range of 187 to 711. Finally, there is a lot of variation in the pupil-teacher ratio raging from 12.6 to 22.0 students per teacher, with at least 3 suburbs that have below average ratios.

```
# plot summary of variables
Boston |>
  dplyr::select(crim,  ptratio, tax) |>
  pivot_longer(everything(), names_to = "indicator", values_to="output") |>
  ggplot(aes(x="", y=output)) +
  geom_boxplot() +
  geom_point(col="dodgerblue", alpha=.2) +
  facet_wrap(~indicator, scales = "free_y") +
  theme_minimal()
```




```
# calculate range for each variable
Boston |>
  dplyr::select(crim,  ptratio, tax) |>
  summarise(across(where(is.double), list(range)))
```

```
    crim_1 ptratio_1 tax_1
1  0.00632      12.6   187
2 88.97620      22.0   711
```

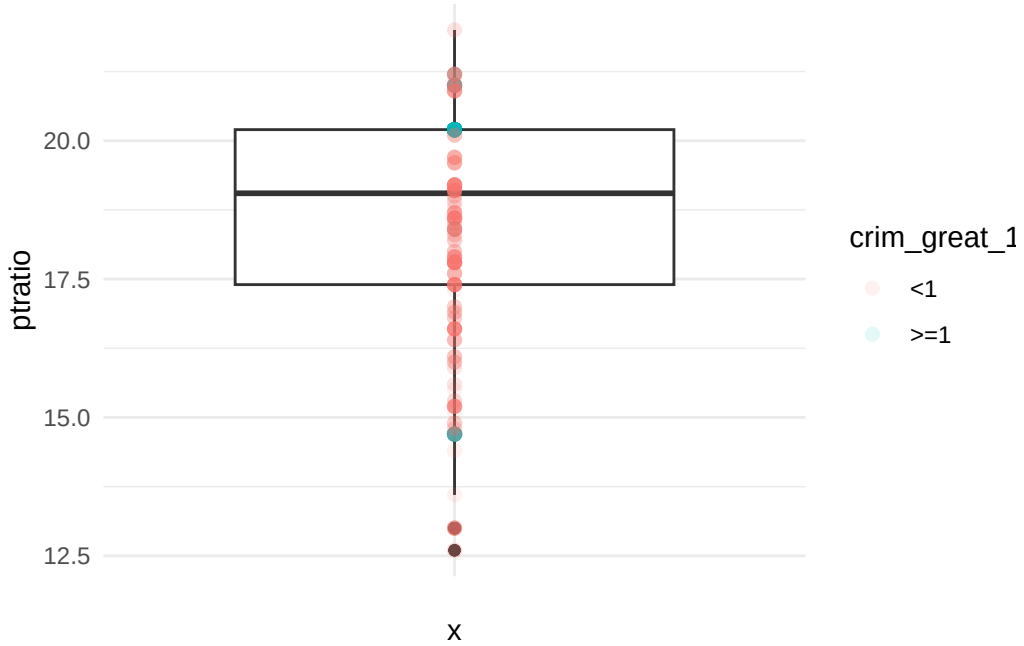
12) Describe the distribution of pupil-teacher ratio among the towns in this data set that have a per capita crime rate larger than 1. How does it differ from towns that have a per capita crime rate smaller than 1?

- The pupil-teacher ratio between towns with >= 1 crime rates and those with < 1 does not differ much with respect to the means of 14.7, and 12.6 (for towns < 1). We find that for towns with >= 1 crime rate there are only 5 distinct pupil-teacher ratio values, suggesting that we may need to consider other ways to compare these two groups.

```
Boston |>
  mutate(crim_great_1 = ifelse(crim >= 1.00, ">=1", "<1")) |>
  group_by(crim_great_1) |>
  reframe(across(ptratio, list(min, mean, sd, median, max))) |> # summary statistics of the c
  rename(min_ptratio=2, mean_ptratio=3, sd_ptratio=4, median_ptratio=5, max_ptratio=6)
```

```
# A tibble: 2 x 6
  crim_great_1 min_ptratio mean_ptratio sd_ptratio median_ptratio max_ptratio
  <chr>              <dbl>        <dbl>      <dbl>          <dbl>       <dbl>
1 <1                  12.6         18.0       2.06           18.3        22
2 >=1                 14.7         19.3       2.12           20.2        21.2
```

```
Boston |>
  mutate(crim_great_1 = ifelse(crim >= 1.00, ">=1", "<1")) |>
  ggplot(aes(x="", y=ptratio)) +
  geom_boxplot() +
  geom_point(aes(col=crim_great_1), alpha=.1, size=2) +
  theme_minimal()
```



Writing Functions

13) Write a function that calculates 95% confidence intervals for a point estimate. The function should be called `my_CI`. When called with `my_CI(2, 0.2)`, the function should print out "The 95% CI upper bound of point estimate 2 with standard error 0.2 is 2.392. The lower bound is 1.608."

```
# function for 95% and print out.
my_CI <- function(point_est, stand_error){

  upper = point_est + (1.96*stand_error)
  lower = point_est - (1.96*stand_error)

  message(str_c(
    str_c(
    "The 95% CI upper bound of point estimate ", point_est,
    " with standard error ", stand_error, " is ",  upper, "."),
    str_c(
    " The lower bound is ", lower, "."), sep="\n"
  ))
}

my_CI(2, 0.2)
```

```
The 95% CI upper bound of point estimate 2 with standard error 0.2 is 2.392.
 The lower bound is 1.608.
```

*Note: The function should take a point estimate and its standard error as arguments. You may use the formula for 95% CI: point estimate +/- 1.96*standard error.*

Hint: Pasting text in R can be done with: `paste()` *and* `paste0()`

14) Create a new function called `my_CI2` that does that same thing as the `my_CI` function but outputs a vector of length 2 with the lower and upper bound of the confidence interval instead of printing out the text. Use this to find the 95% confidence interval for a point estimate of 0 and standard error 0.4.

```
# function for 95% and print out.
my_CI2 <- function(point_est, stand_error){

  CI_range = c(point_est - (1.96*stand_error),
         point_est + (1.96*stand_error))
```

```
  return(CI_range)
}

my_CI2(0, 0.4)
```

```
[1] -0.784  0.784
```

15) Update the `my_CI2` function to take any confidence level instead of only 95%. Call the new function `my_CI3`. You should add an argument to your function for confidence level.

```
my_CI3 <- function(point_est, stand_error, CI){

  conf_level = qnorm(CI)

  CI_range = c(point_est - (conf_level*stand_error),
         point_est + (conf_level*stand_error))

  return(CI_range)
}

my_CI3(0, 0.4, .975) # note for CI .95, use formula 1-(1- .95)/2, for .99 use 1-(1- .99)/2
```

```
[1] -0.7839856  0.7839856
```

Hint: Use the `qnorm` *function to find the appropriate z-value. For example, for a 95% confidence interval, using* `qnorm(0.975)` *gives approximately 1.96.*

16) Without hardcoding any numbers in the code, find a 99% confidence interval for Ethnic Heterogeneity in the Angell dataset. Find the standard error by dividing the standard deviation by the square root of the sample size.

```
eth_hetero <- angell_stata |>
  summarise(mean_ethhet = mean(ethhet), se_ethhet = sd(ethhet)/sqrt(n()) )

  my_CI3(eth_hetero[[1]], eth_hetero[[2]], .995)
```

```
[1] 23.35425 39.38993
```

17) Write a function that you can `apply` to the Angell dataset to get 95% confidence intervals. The function should take one argument: a vector. Use if-else statements to output NA and avoid error messages if the column in the data frame is not numeric or logical.

```
CI_fun <- function(var_dat){


  if(is.numeric(var_dat) | is.logical(var_dat) | is.double(var_dat)){
    print(is.na(var_dat))

    mean_var = mean(var_dat, na.rm=TRUE)
    se_var = sd(var_dat, na.rm = TRUE) / sqrt(length(var_dat))
    lower = round(mean_var - 1.96 * se_var, 2)
    upper = round(mean_var + 1.96 * se_var, 2)

    print(str_c("95% CI [", lower, ", ", upper, "]"))

  } else {
    message("Skipping non numeric variable")
  }
}

lapply(angell_stata,  CI_fun)
```

```
Skipping non numeric variable

 [1] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
[13] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
[25] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
[37] FALSE FALSE FALSE FALSE FALSE FALSE FALSE
[1] "95% CI [10.13, 12.27]"
 [1] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
[13] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
[25] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
[37] FALSE FALSE FALSE FALSE FALSE FALSE FALSE
[1] "95% CI [25.27, 37.47]"
 [1] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
[13] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
[25] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
[37] FALSE FALSE FALSE FALSE FALSE FALSE FALSE
[1] "95% CI [24.67, 30.52]"
```

```
Skipping non numeric variable


$city
NULL

$morint
[1] "95% CI [10.13, 12.27]"

$ethhet
[1] "95% CI [25.27, 37.47]"

$geomob
[1] "95% CI [24.67, 30.52]"

$region
NULL
```