Problem set 8

Your name here

Due 11/29/2022 at 5pm

*Note: Start with the file **ps8_2022.Rmd** (available from the github repository at https://github.com/UChicago-pol-methods/IntroQSS-F22/tree/main/assignments). Modify that file to include your answers. Make sure you can "knit" the file (e.g. in RStudio by clicking on the **Knit** button). Submit both the Rmd file and the knitted Pdf via Canvas.*

Question 1: Calculating Variance Estimators by Hand

Setup

Consider the following joint PMF, also used in HW7 question 1:

$$f_{X,Y}(x, y) = \begin{cases} 1/3 & x = 0, y = 0 \\ 1/6 & x = 0, y = 1 \\ 1/6 & x = 1, y = 1 \\ 1/3 & x = 1, y = 2 \\ 0 & \text{otherwise} \end{cases}$$

(1a) Start with your tibble from HW7's question 1c. Regress `y` on `x` in this data set using `lm()`, and save the resulting linear model an an object, `lm1`.

```
# your code here
```

(1b) Create a vector `Y` that is the `y` column from the original data set. Create a matrix `X` that is the full regressor matrix $\mathbf{X}$ from the regression model (Definition 4.1.3), i.e., a matrix with a column of 1's, and a column that is the `x` column in our original dataset. A shortcut is that you can use the output of `model.matrix(lm1)`. *(We often use tibbles when we are interested in evaluating an object as a data set, but vectors and matrices are designed for the matrix manipulation we are about to do.)*

```
# your code here
```

(1c) Check your coefficient estimates from `lm1` against the model produced from the OLS solution in Matrix Algebra, Theorem 4.1.4. (You will have seen the R code for this formula in the class slides over the last two weeks.) Save this solution as an object `beta`.

```
# your code here
```

(1d)

Extract the residuals from your `lm1` object, and save them as a vector `e`.

```
# your code here
```

Robust sampling variance estimator

(1e) Calculate the robust sampling variance estimator for OLS, using the matrix algebra formula in Definition 4.2.1.

- between each element, use the matrix multiplication operator, `%*%`
- $(\cdot)^{-1}$ implies a matrix inversion, so for this operator, use the function `solve()`
- $\cdot^T$ implies the transpose of a matrix, and for this operator, use the function `t()`.
- for `diag()`, use the function `diag()`.

Save this as an object, `var_robust`. Take the square root of the diagonal of this object, again using the `diag()` function.

(Note: Don't spin your wheels too much on this one. If you hit a wall, check in on Stack Overflow or just skip to the next question.)

```
# your code here
```

(1f) Check your answer against the standard errors produced from regressing `y` on `x` using the `lm_robust()` function from the `estimatr` package, setting the standard error type to 'HC0'.

```
# your code here
```

Classical sampling variance estimator

(1g)

Calculate an object `sigma2` that is our estimate of the variance of the regression error,

$$\hat{\sigma}^2 = \frac{1}{n-(K+1)} \sum_{i=1}^{n} \mathbf{e}^2$$

where K is the number of covariates in our model. (This is not a trick question, we only have one covariate in our model.)

```
# your code here
```

(1h)

Calculate the classical sampling variance estimator for OLS using the matrix algebra formula in Definition 4.2.3, using the object `sigma2` from above, and the instructions from part 1e.

Save this as an object, `var_classical`. Take the square root of the diagonal of this object, again using the `diag()` function.

```
# your code here
```

(1i) Check your answer against the standard errors produced from `summary(lm1)`.

```
# your code here
```

Question 2: Robust standard errors with real data

We now return to data from an experiment that measured the effect of constituent names in emails on legislator replies. The published paper is:

Butler, D. M., & Broockman, D. E. (2011). *Do politicians racially discriminate against constituents? A field experiment on state legislators.* AJPS.

The data file is `Butler_Broockman_AJPS_2011_public.csv` and it is found in the `data` directory of the course github repository.

To load the data you can either download and read in the local file, or you can read in the url from github. Note that reading in by the url will only work when you have an internet connection:

```
file <- 'https://raw.githubusercontent.com/UChicago-pol-methods/IntroQSS-F22/main/data/Butler_Broockman
bb <- read_csv(url(file))
```

(2a) Using `lm_robust`, regress `reply_atall` on `treat_deshawn` interacted with `leg_republican`. Print the model object.

```
# your code here
```

(2b) From the model object above, report and interpret the standard errors and 95% confidence intervals on `treat_deshawn` and `treat_deshawn:leg_republican`. Do the confidence intervals include zero? If so/if not, what does that imply?

(2c) Using `map()` (or a `for` loop) and `slice_sample(, replace = TRUE)`, take 1000 bootstrap re-samples with replacement of the same size as the original data from the `bb` dataset. Save your bootstrapped samples as an object. (Don't print them here!)

```
# your code here
```

(2d) Using `map_dfr`, (or a `for` loop) run the same regression as in part 2a on *each* of your bootstrapped samples, and extract just the coefficient estimates. This creates a tibble where each row represents estimates from one of your bootstrap samples, and each column is one of the coefficients. (Again, don't print it here!)

```
# your code here
```

(2e) Report the bootstrapped estimates of the standard errors of each of the coefficients. To do this, get the standard deviations of each of the columns from the object you created in 2d.

```
# your code here
```

(2f) Produce percentile confidence intervals for each of the coefficients, by reporting the 2.5th and 97.5th quantiles of each of your columns from the object you created in 2d. **(`summarise_all` may be helpful, but there are different ways to achieve this solution.)**

```
# your code here
```