Measuring Impact with A/B Testing (Krea University, 2025)
Lab 2 Exercises
Quantifying Uncertainty

Estimating the Impact of Tip Recommendation (Alexander et al. 2021)

1. In case you want to look at the article, you can find it here. The tip recommendation system was designed with multiple conditions.
   A = \$2, \$4,and \$6; B = \$3, \$4, and \$5; C = \$3.95, \$4, and \$4.05; D = \$3.99, \$5.99, and \$7.99; E = \$4, \$6, and \$8; F = \$5, \$8,and \$10; G = 5%, 10%, and 15%; H = 10%, 15%, and 20%; I = 12%, 15%, and 18%; J = 15%, 18%, and 20%; or K = 15%, 20%, and 25%.
2. Since there are many tip recommendation options, we will narrow down on a few pairs for comparison.
   a. Those which differed only by recommended tip size (E vs A; H v G; K v H; K v G)
   b. Those which differed only by the range in the recommended tip size (A v B; H v I)
   c. Tip sizes with different left and right-most digits (E v D)
   d. % vs absolute tip size for bills above \$40 (**G v A**; H v E).

3. Download and read in the data. It's the csv called "washio" in the data subfolder of the course folder.
   a. *Hint 1: use read_csv('data/filename.csv').*
   b. *Hint 2: Please load the .Rproj file from the root folder.*
4. Explore the data. Look at the names of the variables and create histograms of the variables we're interested in using *names()* and *hist()*. For the first exercise, we'll focus on ***tip_amount***. Using *summary()* might also be appropriate to get a sense of what we are working with.
5. Create a treatment dummy where recommendation system G is seen as treatment and A as control.
6. Calculate the estimated ATE of G vs A on ***tip_amount*** and store this estimate in a variable called ***ate_tip_gva***.
7. A skeptic would tell us that we would see differences this big by chance -- after all, don't we know that experiments sometimes give us slight overestimates or underestimates because we've randomly assigned the groups? If this skeptic were right, let's see how often random assignment in this data would actually get us estimates as big as we saw. This is called calculating the sampling distribution of the ATE under the null hypothesis.
   a. First, write a line of code that takes the treatment variable in the dataset as it was randomly assigned, re-randomizes, and saves this in a new variable called ***fake.treatment***.
   To do this, just take the existing random assignment and shuffle the order of the existing

variable randomly.
Hint: use sample(). You only need to give sample one argument -- you don't need to give it a number or specify whether you're sampling with replacement or not.
Verify this works by running your line of code a few times.

b. Now, write a function that a) creates a new random assignment using the line of code from part a. above, and b) computes the estimate that you would get back on tip_amount if that were the random assignment and the outcomes were the same. That is, the function should compute a randomization using the first line of code you wrote and then compute the "ATE estimate" one would get under that randomization. This gives you a draw from the sampling distribution you would see if the skeptic were correct and the experiment was just randomly shuffling individuals between groups but there was no actual effect of being assigned to the treatment group.

   This is your first time writing functions, so here is some code you can start with:

```
name.of.your.function <- function() {
        # write your code here
        return(variable.to.return)
}
```

   In order to define your function for R, highlight the entire thing (including the { and }) and run the code once. Now your function has been defined for R and you can call it.

c. Test that your function works by running it a few times. E.g., run the line:
   name.of.your.function()
   This should compute one possible ATE estimate you could get if the skeptic were correct.

d. Now, replicate that function 10,000 times using *replicate(10000, name.of.your.function())* and store the results in a new variable. This is the sampling distribution under the null hypothesis.

8. Now that we have the sampling distribution under the null hypothesis, let's compare the ATE estimate we received to the estimates we would have seen by chance if the true effect were zero.
   a. Graph the sampling distribution from the previous part in a histogram. Insert a vertical line that represents the observed ATE by adding this second line after you make the histogram: *abline(v=ate_tip_gva)*.
   b. This seems to be an example where our ATE estimate is very unlikely under the null hypothesis. But just how unlikely? Compute the *p*-value by comparing the distribution to

the observed ATE. What percent of randomizations would give you an estimate as large as the one you saw by chance?

i. Hint: use the mean() function to do this.

ii. Hint 2: You'll want to pass the mean() function a list of 10,000 entries, each of which is a TRUE or a FALSE that records whether each of the "fake ATEs" under the null hypothesis is as large or larger than the ATE you actually observed. When you take the mean of a TRUE or FALSE value, R will make the TRUEs 1 and the FALSE's 0.

c. What do you conclude about the impact of the recommendation system G vs A from this data?

d. Let's do this with a canned function from R. Use with this command: *t.test(outcomevar ~ treatmentvar)*. This command gives you the treatment and control means and the p-value on the difference being equal to zero. (Do this using the actual data, not your fake treatment.) Do we see similar results?

9. Let's estimate the uncertainty around these estimates in a more nuanced way, moving beyond simply concluding whether the effect is zero or not.
    a. First, calculate the standard deviation of the sampling distribution under the null that you computed above. Use the sd() function to do this. This is a measure of how noisy the experiment is. From experiment to experiment, we'd expect to see differences between the estimates about sd() in size.
    b. Using the formula we saw in class, calculate the interval in which the estimate will be within 95% of the time. Hint: the number 1.96 is involved.
    c. We can also get this estimate for the confidence interval from the canned regression function in R. Use this code to run the regression:
    summary(lm(tip_amount~treatment,data=name.of.your.dataset)).
    Then, compute the confidence interval by combining the estimate and the standard errors.
10. Complete steps 6-9 again, this time looking at how satisfaction rating changed (data$satisfaction_ratings). Since we're working with a new variable, take a look at its distribution before starting to get a sense of what we are working with. Remember that you've already generated the necessary code, so this time you can simply copy and paste these functions with minor changes to create the sampling distribution with satisfaction_rating.

Estimating the Effects of Applicant Race on Job Call-backs

Let's go through this exercise again (same as steps 2-7 above), but this time without training wheels. We'll be looking at data that looks at the effects of race on job call-back rates. Remember that you have already written functions for each of these parts, so you can re-copy them and only make minor changes.

1. You can briefly review the set-up for this experiment here.
2. Download and read in the data, which is called bertrand_clean.csv.

3. Explore the data. Your treatment variable, race, is a binary variable called *black* (1=black, 0=white). Your outcome variable is *call* (1=called back, 0=not called back).
4. Calculate the ATE for the effects on getting a call back and name it ate.call.
5. Calculate the sampling distribution of the ATE under the null hypothesis. *Hint: You'll probably need to change the rand.est.ate function because you're working with a different data set with different variables.*
6. Estimate the confidence intervals around the observed ATE.
7. Do the same, but for the effect of attending college on callback rates *just among the subset of observations where black is equal to 1.* You can use the subset() function to save a new dataset with just the relevant observations like this:
   bertrand.black <- subset(bertrand.data, black==1)