Project 6: Randomization and Matching

Sofia Guo & Stacy Chen

Introduction

In this project, you will explore the question of whether college education causally affects political participation. Specifically, you will use replication data from Who Matches? Propensity Scores and Bias in the Causal Effects of Education on Participation by former Berkeley PhD students John Henderson and Sara Chatfield. Their paper is itself a replication study of Reconsidering the Effects of Education on Political Participation by Cindy Kam and Carl Palmer. In their original 2008 study, Kam and Palmer argue that college education has no effect on later political participation, and use the propensity score matching to show that pre-college political activity drives selection into college and later political participation. Henderson and Chatfield in their 2011 paper argue that the use of the propensity score matching in this context is inappropriate because of the bias that arises from small changes in the choice of variables used to model the propensity score. They use genetic matching (at that point a new method), which uses an approach similar to optimal matching to optimize Mahalanobis distance weights. Even with genetic matching, they find that balance remains elusive however, thus leaving open the question of whether education causes political participation.

You will use these data and debates to investigate the benefits and pitfalls associated with matching methods. Replication code for these papers is available online, but as you'll see, a lot has changed in the last decade or so of data science! Throughout the assignment, use tools we introduced in lab from the tidyverse and the MatchIt packages. Specifically, try to use dplyr, tidyr, purrr, stringr, and ggplot instead of base R functions. While there are other matching software libraries available, MatchIt tends to be the most up to date and allows for consistent syntax.

Data

The data is drawn from the Youth-Parent Socialization Panel Study which asked students and parents a variety of questions about their political participation. This survey was conducted in several waves. The first wave was in 1965 and established the baseline pre-treatment covariates. The treatment is whether the student attended college between 1965 and 1973 (the time when the next survey wave was administered). The outcome is an index that calculates the number of political activities the student engaged in after 1965. Specifically, the key variables in this study are:

- **college**: Treatment of whether the student attended college or not. 1 if the student attended college between 1965 and 1973, 0 otherwise.
- **ppnscal**: Outcome variable measuring the number of political activities the student participated in. Additive combination of whether the student voted in 1972 or 1980 (student_vote), attended a campaign rally or meeting (student_meeting), wore a campaign button (student_button), donated money to a campaign (student_money), communicated with an elected official (student_communicate), attended a demonstration or protest (student_demonstrate), was involved with a local community event (student_community), or some other political participation (student_other)

Otherwise, we also have covariates measured for survey responses to various questions about political attitudes. We have covariates measured for the students in the baseline year, covariates for their parents in the baseline year, and covariates from follow-up surveys. **Be careful here**. In general, post-treatment covariates will be clear from the name (i.e. student_1973Married indicates whether the student was married in the 1973 survey).

Be mindful that the baseline covariates were all measured in 1965, the treatment occurred between 1965 and 1973, and the outcomes are from 1973 and beyond. We will distribute the Appendix from Henderson and Chatfield that describes the covariates they used, but please reach out with any questions if you have questions about what a particular variable means.

```
# Load tidyverse and MatchIt
# Feel free to load other libraries as you wish
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.4     v readr     2.1.5
## v forcats   1.0.0     v stringr   1.5.1
## v ggplot2   3.5.0     v tibble    3.2.1
## v lubridate 1.9.3     v tidyr     1.3.1
## v purrr     1.0.2
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

```
library(MatchIt)
library(ggplot2)
library(cobalt)
```

```
##  cobalt (Version 4.5.4, Build Date: 2024-02-26)
##
## Attaching package: 'cobalt'
##
## The following object is masked from 'package:MatchIt':
##
##     lalonde
```

```
# Load ypsps data
ypsps <- read_csv('data/ypsps.csv')
```

```
## Rows: 1254 Columns: 174
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## dbl (174): interviewid, college, student_vote, student_meeting, student_othe...
##
## 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.
```

```
head(ypsps)
```

```
## # A tibble: 6 x 174
##   interviewid college student_vote student_meeting student_other student_button
##         <dbl>   <dbl>        <dbl>           <dbl>         <dbl>          <dbl>
## 1           1       1            1               0             0              0
## 2           2       1            1               1             1              1
## 3           3       1            1               0             0              1
## 4           4       0            0               0             0              0
## 5           5       1            1               1             0              0
## 6           6       1            1               0             0              0
## # i 168 more variables: student_money <dbl>, student_communicate <dbl>,
## #   student_demonstrate <dbl>, student_community <dbl>, student_ppnscal <dbl>,
## #   student_PubAff <dbl>, student_Newspaper <dbl>, student_Radio <dbl>,
```

```
## #   student_TV <dbl>, student_Magazine <dbl>, student_FamTalk <dbl>,
## #   student_FrTalk <dbl>, student_AdultTalk <dbl>, student_PID <dbl>,
## #   student_SPID <dbl>, student_GovtOpinion <dbl>, student_GovtCrook <dbl>,
## #   student_GovtWaste <dbl>, student_TrGovt <dbl>, student_GovtSmart <dbl>, ...
```

Randomization

Matching is usually used in observational studies to to approximate random assignment to treatment. But could it be useful even in randomized studies? To explore the question do the following:

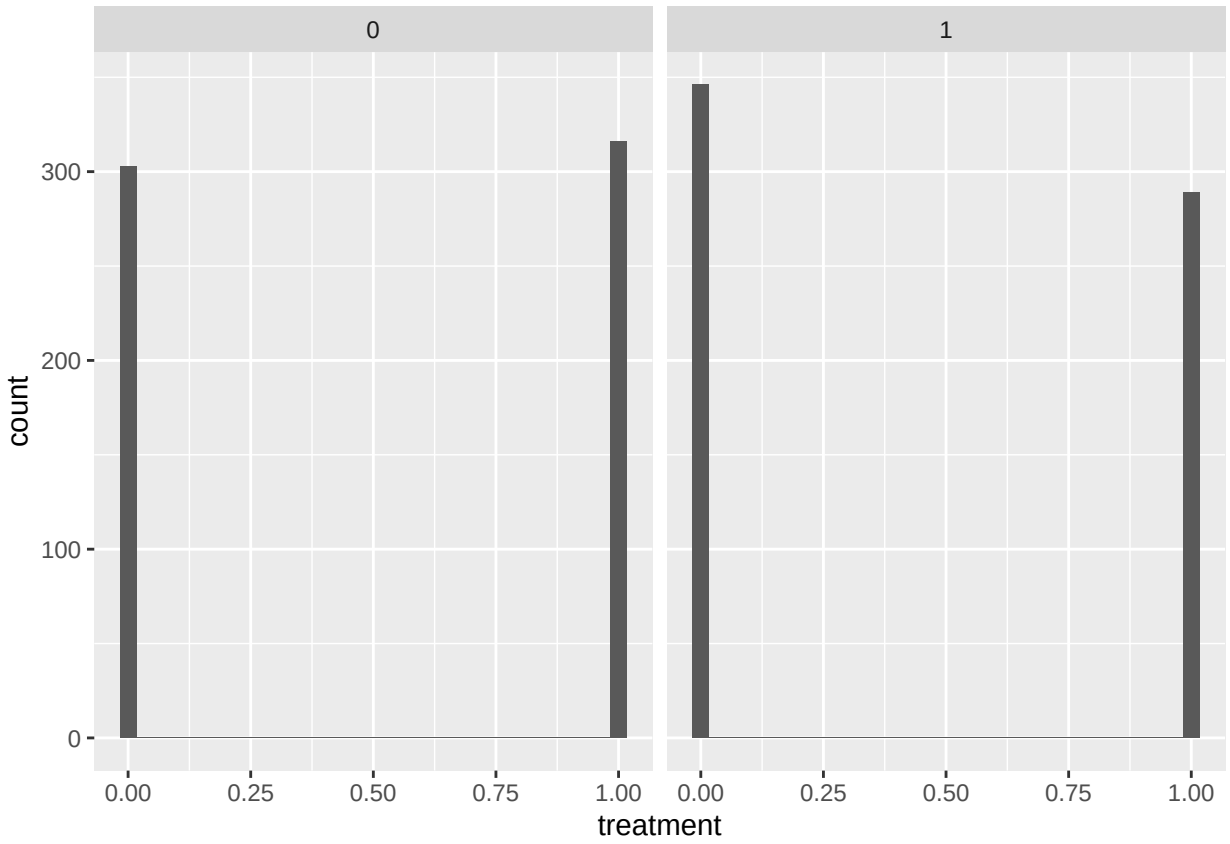
1. Generate a vector that randomly assigns each unit to either treatment or control
2. Choose a baseline covariate (for either the student or parent). A binary covariate is probably best for this exercise.
3. Visualize the distribution of the covariate by treatment/control condition. Are treatment and control balanced on this covariate?
4. Simulate the first 3 steps 10,000 times and visualize the distribution of treatment/control balance across the simulations.

```
# Generate a vector that randomly assigns each unit to treatment/control
# completely randomize treatment
# ----------
# set seed - unfortunately seed needs to be set within cell to be reproducible in .Rmd but just once in
set.seed(14)

df <-                                                    # save object
  ypsps %>%                                                 # pass data
  mutate(treatment = as.numeric(rbernoulli(length(unique(interviewid)), p=0.5)))            # create c
```

```
## Warning: There was 1 warning in `mutate()`.
## i In argument: `treatment = as.numeric(rbernoulli(length(unique(interviewid)),
##   p = 0.5))`.
## Caused by warning:
## ! `rbernoulli()` was deprecated in purrr 1.0.0.
```

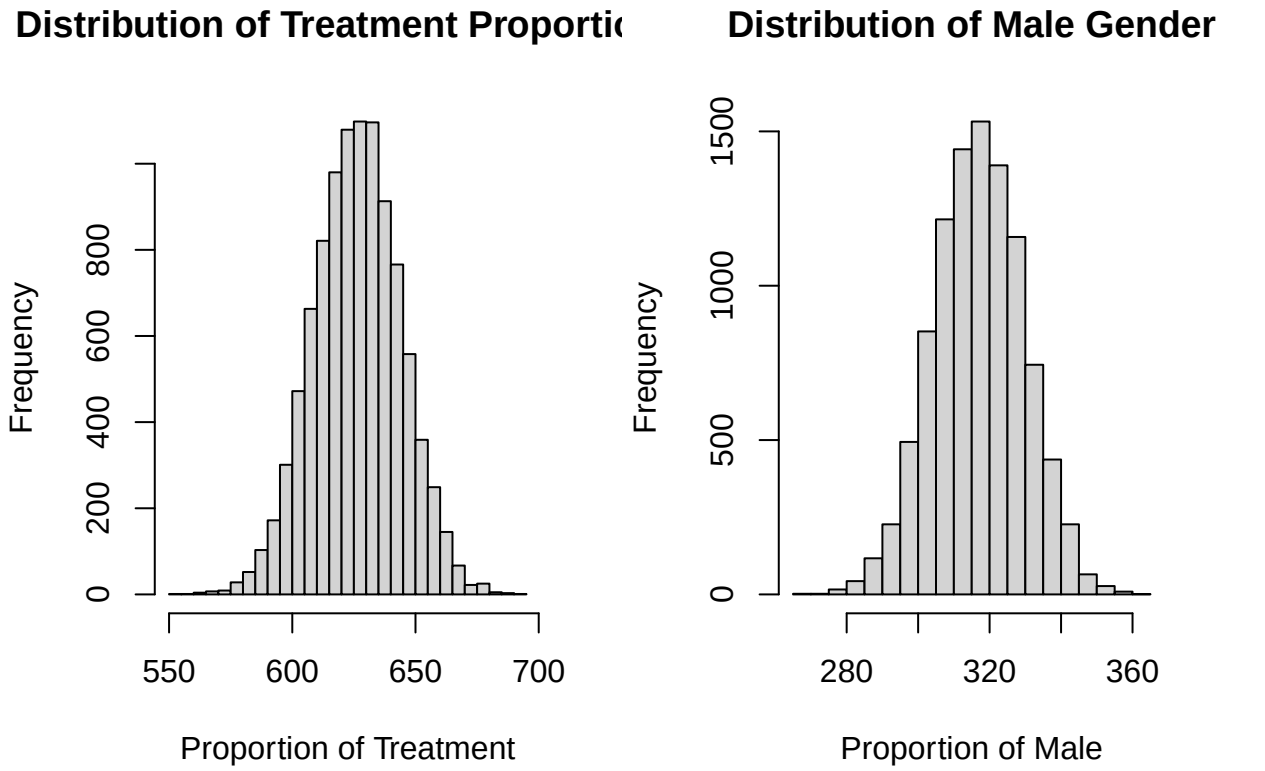
```
       # Y_comp = as.numeric((A_comp & student_ppnscal) | (!A_comp & student_ppnscal))) # create comple

# Choose a baseline covariate (use dplyr for this)
baselinecov <- df %>%
  select(student_Gen,treatment)

# Visualize the distribution by treatment/control (ggplot)
ggplot(baselinecov) +
  geom_histogram(aes(treatment)) +
  facet_wrap(.~student_Gen)
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```




```r
# Simulate this 10,000 times (monte carlo simulation - see R Refresher for a hint)
n_simulations <- 10000

# Perform Monte Carlo simulation
# Empty matrix to store simulation results
sim_results <- matrix(nrow = n_simulations, ncol = 2)

# Perform Monte Carlo simulation
for (i in 1:n_simulations) {
  # Generate treatment assignment vector
  df <- ypsps %>%
    select(interviewid,student_Gen) %>%
    mutate(treatment = as.numeric(rbernoulli(length(unique(interviewid)), p=0.5)))

  # Calculate the proportion of treatment units
  proportion_treatment <- sum(df$treatment)

  # Calculate the proportion of Male gender
  proportion_male <- sum(df$student_Gen[df$treatment == 1])

  # Store the results
  sim_results[i, 1] <- proportion_treatment
  sim_results[i, 2] <- proportion_male
}

# Visualize the distribution of treatment proportions
par(mfrow = c(1, 2)) # Arrange plots in one row and two columns
```

```r
hist(sim_results[, 1], breaks = 30, main = "Distribution of Treatment Proportions",
     xlab = "Proportion of Treatment", ylab = "Frequency")
hist(sim_results[, 2], breaks = 30, main = "Distribution of Male Gender",
     xlab = "Proportion of Male", ylab = "Frequency")
```



Questions

1. **What do you see across your simulations? Why does independence of treatment assignment and baseline covariates not guarantee balance of treatment assignment and baseline covariates?**

Your Answer: Across our simulations, we see that although the proportion of treatment is almost exactly balanced (half of 1254 is 627, which according to the histogram on the left is right where the mean sits), slightly more than half of the student genders are equal to 1 (since the documentation is unclear on whether student_gender ==1 is male or female, I just assume that it means male for the purpose of this exercise). I make this observation because half of 627 is 314, but the mean of the right side histogram of the gender distribution appears to be closer to 320 than 314. This imbalance can occur because of random chance and when our simulation/sampling size is not large enough towards infinity.

Propensity Score Matching

One Model

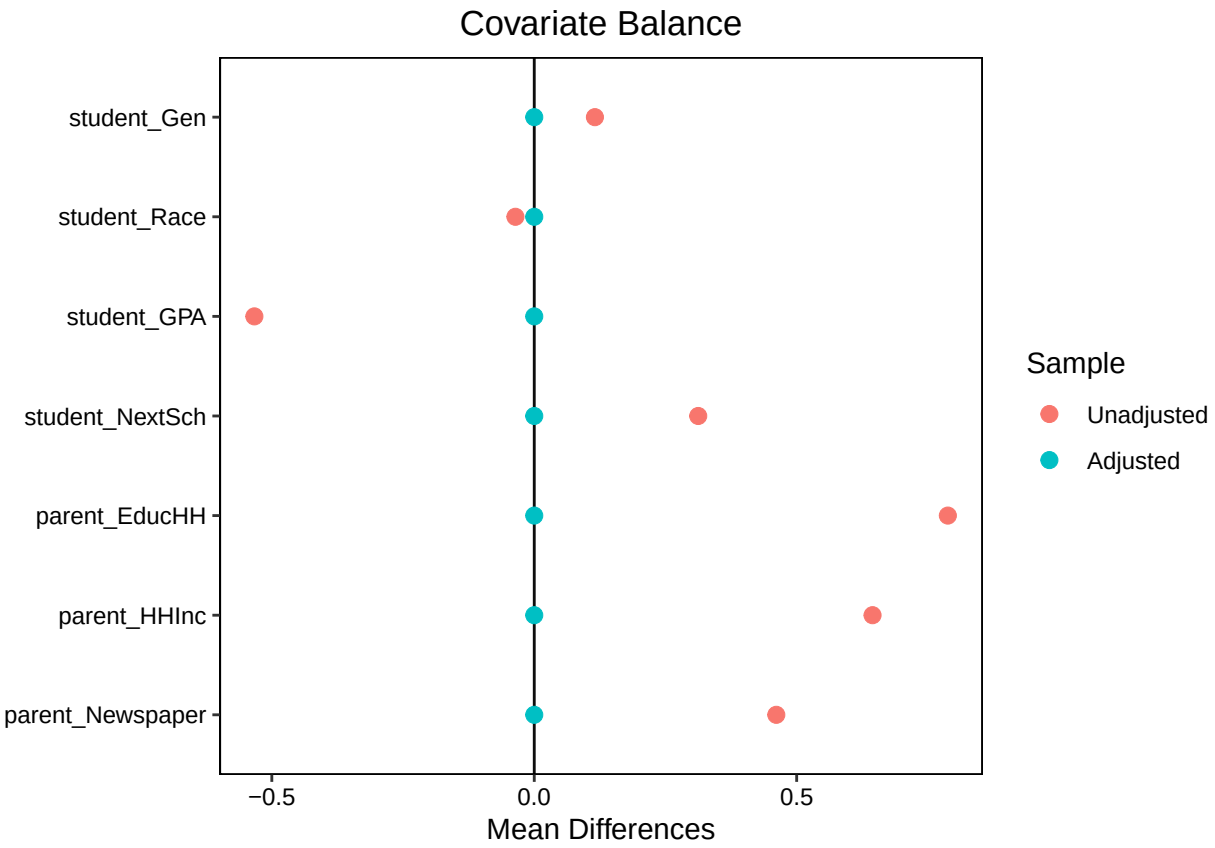
Select covariates that you think best represent the "true" model predicting whether a student chooses to attend college, and estimate a propensity score model to calculate the Average Treatment Effect on the Treated (ATT). Plot the balance of the top 10 (or fewer if you select fewer covariates). Report the balance of the p-scores across both the treatment and control groups, and using a threshold of standardized mean difference of p-score $\leq .1$, report the number of covariates that meet that balance threshold.

```r
# Select covariates that represent the "true" model for selection, fit model
df <- ypsps %>%
  select(interviewid, college, student_ppnscal, student_Gen, student_Race, student_GPA, student_NextSch

# fit model
M1_student_college <- glm(college ~ 0 + student_Gen + student_Race + student_GPA + student_NextSch + pa

# Step 2: Predict propensity scores
df$propensity_score <- predict(M1_student_college, type = "response")

# Step 3: Calculate ATT
# Assuming 'df' is your dataset containing treatment, covariates, and outcome
# Match treated and control units
match_exact_att <- matchit(college ~ student_Gen + student_Race + student_GPA + student_NextSch + paren

# Report the overall balance and the proportion of covariates that meet the balance threshold
summary(match_exact_att, un=F)
```

```
##
## Call:
## matchit(formula = college ~ student_Gen + student_Race + student_GPA +
##     student_NextSch + parent_EducHH + parent_HHInc + parent_Newspaper,
##     data = df, method = "exact", estimand = "ATT")
##
## Summary of Balance for Matched Data:
##                  Means Treated Means Control Std. Mean Diff. Var. Ratio
## student_Gen             0.5723        0.5723              -0          .
## student_Race            1.0120        1.0120              -0     0.9899
## student_GPA             2.4488        2.4488              -0     0.9899
## student_NextSch         0.9608        0.9608              -0          .
## parent_EducHH           2.9217        2.9217              -0     0.9899
## parent_HHInc            7.2410        7.2410              -0     0.9899
## parent_Newspaper        3.7229        3.7229              -0     0.9899
##                  eCDF Mean eCDF Max Std. Pair Dist.
## student_Gen              0        0               0
## student_Race             0        0               0
## student_GPA              0        0               0
## student_NextSch          0        0               0
## parent_EducHH            0        0               0
## parent_HHInc             0        0               0
## parent_Newspaper         0        0               0
##
## Sample Sizes:
##               Control Treated
## All            451.       803
## Matched (ESS)   76.65     332
## Matched        178.       332
## Unmatched      273.       471
## Discarded        0.         0
```

```r
#make covariate plot
love.plot(match_exact_att)
```

```
## Warning: Standardized mean differences and raw mean differences are present in the same plot.
## Use the `stars` argument to distinguish between them and appropriately label the x-axis.
```



It looks like all the covariates I chose met the threshold, so I went ahead and estimated the ATT:

```
#estimate the ATT using linear regression
match_exact_att_data <- match.data(match_exact_att)

#specify model
lm_full_att <- lm(student_ppnscal ~ college + student_Gen + student_Race + student_GPA + student_NextSc

#summarize results
lm_full_att_summ <- summary(lm_full_att)

#calculate ATT
ATT_full <- lm_full_att_summ$coefficients["college","Estimate"]
ATT_full
```

```
## [1] 0.9228414
```

Simulations

Henderson/Chatfield argue that an improperly specified propensity score model can actually *increase* the bias of the estimate. To demonstrate this, they simulate 800,000 different propensity score models by choosing different permutations of covariates. To investigate their claim, do the following:

- Using as many simulations as is feasible (at least 10,000 should be ok, more is better!), randomly select the number of and the choice of covariates for the propensity score model.
- For each run, store the ATT, the proportion of covariates that meet the standardized mean difference $\leq .1$ threshold, and the mean percent improvement in the standardized mean difference. You may also wish to store the entire models in a list and extract the relevant attributes as necessary.

- Plot all of the ATTs against all of the balanced covariate proportions. You may randomly sample or use other techniques like transparency if you run into overplotting problems. Alternatively, you may use plots other than scatterplots, so long as you explore the relationship between ATT and the proportion of covariates that meet the balance threshold.
- Finally choose 10 random models and plot their covariate balance plots (you may want to use a library like gridExtra to arrange these)

Note: There are lots of post-treatment covariates in this dataset (about 50!)! You need to be careful not to include these in the pre-treatment balancing. Many of you are probably used to selecting or dropping columns manually, or positionally. However, you may not always have a convenient arrangement of columns, nor is it fun to type out 50 different column names. Instead see if you can use dplyr 1.0.0 functions to programatically drop post-treatment variables (here is a useful tutorial).

```
# Remove post-treatment covariates

# Randomly select features

# Simulate random selection of features 10k+ times

# Fit p-score models and save ATTs, proportion of balanced covariates, and mean percent balance improve

# Plot ATT v. proportion

# 10 random covariate balance plots (hint try gridExtra)
# Note: ggplot objects are finnicky so ask for help if you're struggling to automatically create them;
```

Questions

1. **How many simulations resulted in models with a higher proportion of balanced covariates? Do you have any concerns about this?** Your Answer:...
2. **Analyze the distribution of the ATTs. Do you have any concerns about this distribution?** Your Answer:...
3. **Do your 10 randomly chosen covariate balance plots produce similar numbers on the same covariates? Is it a concern if they do not?** Your Answer:...

Matching Algorithm of Your Choice

Simulate Alternative Model

Henderson/Chatfield propose using genetic matching to learn the best weights for Mahalanobis distance matching. Choose a matching algorithm other than the propensity score (you may use genetic matching if you wish, but it is also fine to use the greedy or optimal algorithms we covered in lab instead). Repeat the same steps as specified in Section 4.2 and answer the following questions:

```
# Remove post-treatment covariates

# Randomly select features

# Simulate random selection of features 10k+ times

# Fit  models and save ATTs, proportion of balanced covariates, and mean percent balance improvement

# Plot ATT v. proportion
```

```
# 10 random covariate balance plots (hint try gridExtra)
# Note: ggplot objects are finnicky so ask for help if you're struggling to automatically create them;
```

```
# Visualization for distributions of percent improvement
```

Questions

1. **Does your alternative matching method have more runs with higher proportions of balanced covariates?** Your Answer:...
2. **Use a visualization to examine the change in the distribution of the percent improvement in balance in propensity score matching vs. the distribution of the percent improvement in balance in your new method. Which did better? Analyze the results in 1-2 sentences.** Your Answer:...

Optional: Looking ahead to the discussion questions, you may choose to model the propensity score using an algorithm other than logistic regression and perform these simulations again, if you wish to explore the second discussion question further.

Discussion Questions

1. **Why might it be a good idea to do matching even if we have a randomized or as-if-random design?** Your Answer:...
2. **The standard way of estimating the propensity score is using a logistic regression to estimate probability of treatment. Given what we know about the curse of dimensionality, do you think there might be advantages to using other machine learning algorithms (decision trees, bagging/boosting forests, ensembles, etc.) to estimate propensity scores instead?** Your Answer:...