Project 6: Randomization and Matching

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 packages --------------------------------------- tidyverse 1.3.1 --
```

```
## v ggplot2 3.3.5     v purrr   0.3.4
## v tibble  3.1.6     v dplyr   1.0.8
## v tidyr   1.2.0     v stringr 1.4.0
## v readr   2.1.2     v forcats 0.5.1
```

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

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

```
##  cobalt (Version 4.3.2, Build Date: 2022-01-19)
```

```
##
## 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
## # ... with 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>, ...
```

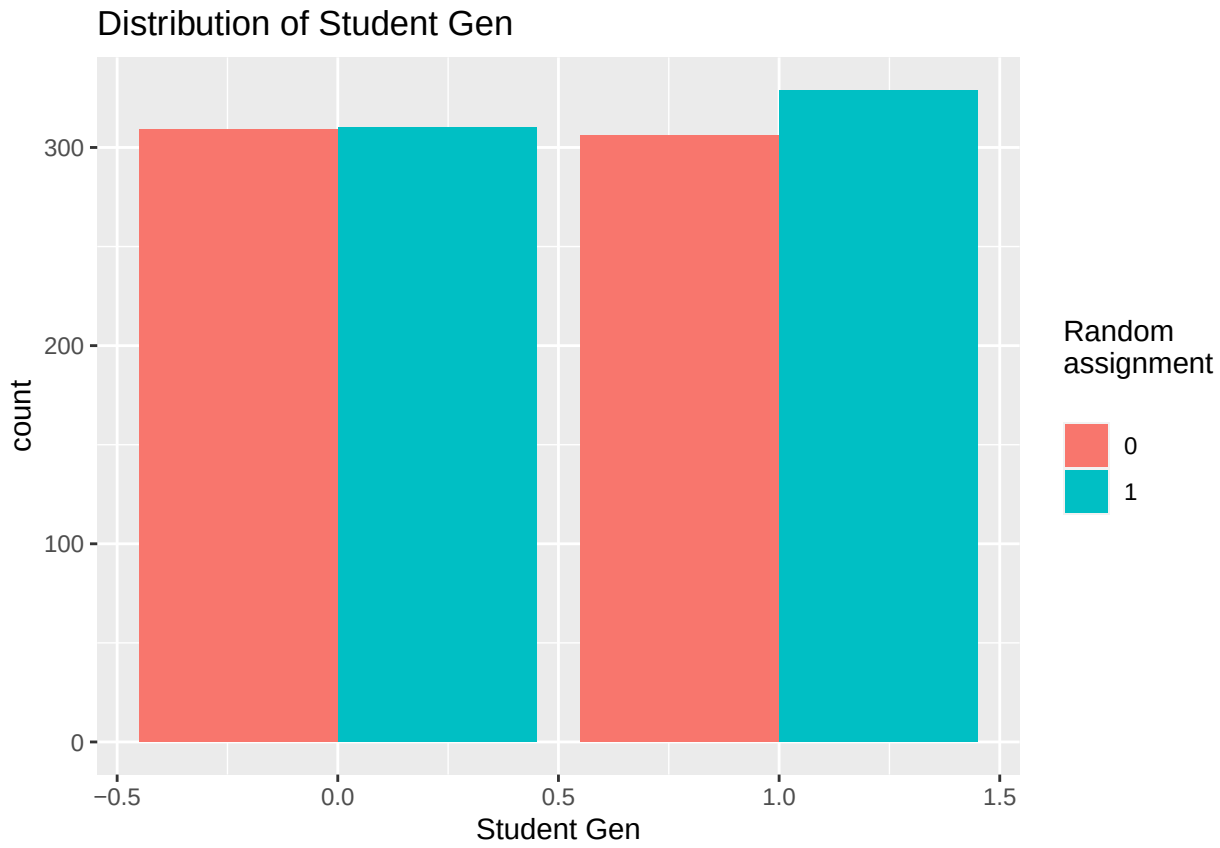
```
#Remove variables used to create the outcome
ypsps <- ypsps %>% select(!c(student_vote, student_button, student_money, student_meeting, student_commu
```

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.

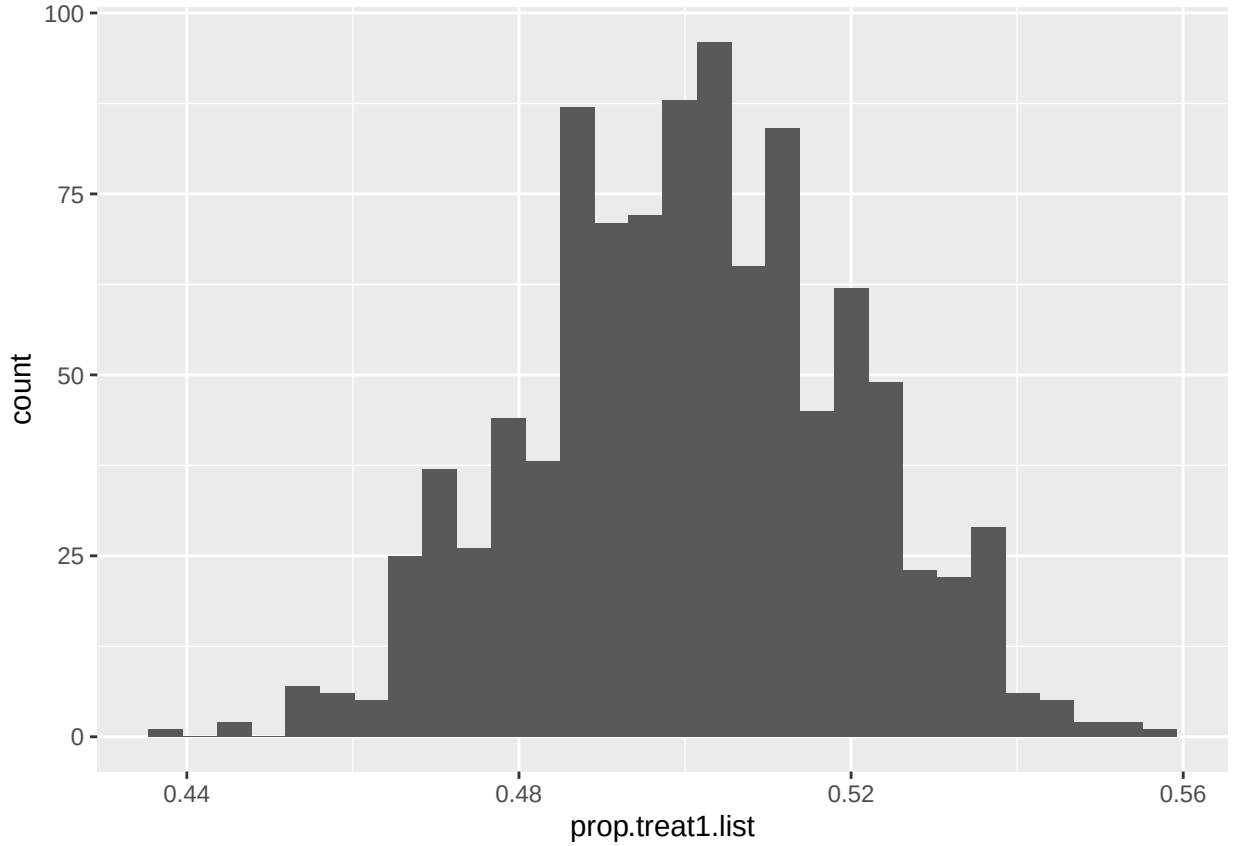
```
# Set seed
set.seed(2459)

# Generate a vector that randomly assigns each unit to treatment/control
rdm_vtr <- sample(0:1, size = nrow(ypsps), replace = TRUE, prob = c(0.5,0.5))
## 0 = treatment and 1 = control group

# Choose a baseline covariate
temp_df <-  ypsps %>%
  select(student_Gen) %>% ## 1 = college and 0 = no college
  mutate(random_assignment = rdm_vtr)

# Visualize the distribution by treatment/control (ggplot)
ggplot(temp_df) +
  geom_bar(aes(x=student_Gen,
                 fill = factor(random_assignment)),
       position = 'dodge') +
     labs(title = "Distribution of Student Gen", fill = "Random \nassignment\n") +
    xlab("Student Gen")
```




```
# Simulate this 10,000 times (monte carlo simulation - see R Refresher for a hint)
nsim = 1000

prop.treat1.list <- rep(NA, nsim)  # create empty vector to append distributions across simulations - c
prop.treat0.list <- rep(NA, nsim)  # create empty vector to append distributions across simulations  -

for (iter_num in 1:nsim) {  # create for loop to repeat the three steps above

  # Generate a vector that randomly assigns each unit to treatment/control
   rdm_vtr <- sample(0:1, size = nrow(ypsps), replace = TRUE, prob = c(0.5,0.5))

  # Create temp df that keeps covariate (student_money) and adds random vector from above
   temp_df <-  ypsps %>%
     select(student_Gen) %>% ## 1 = college and 0 = no college
     mutate(random_assignment = rdm_vtr)

   # Calculate proportion treated when covariate = 1 and when covariate = 0
   prop.treat1 <- (temp_df %>% filter(student_Gen==1 & random_assignment==1) %>% nrow())/(temp_df %>% f

   prop.treat0 <- (temp_df %>% filter(student_Gen==0 & random_assignment==1) %>% nrow())/(temp_df %>% f

  # Save proportions in list
  prop.treat1.list[iter_num] <- prop.treat1
  prop.treat0.list[iter_num] <- prop.treat0
}

ggplot() +
```

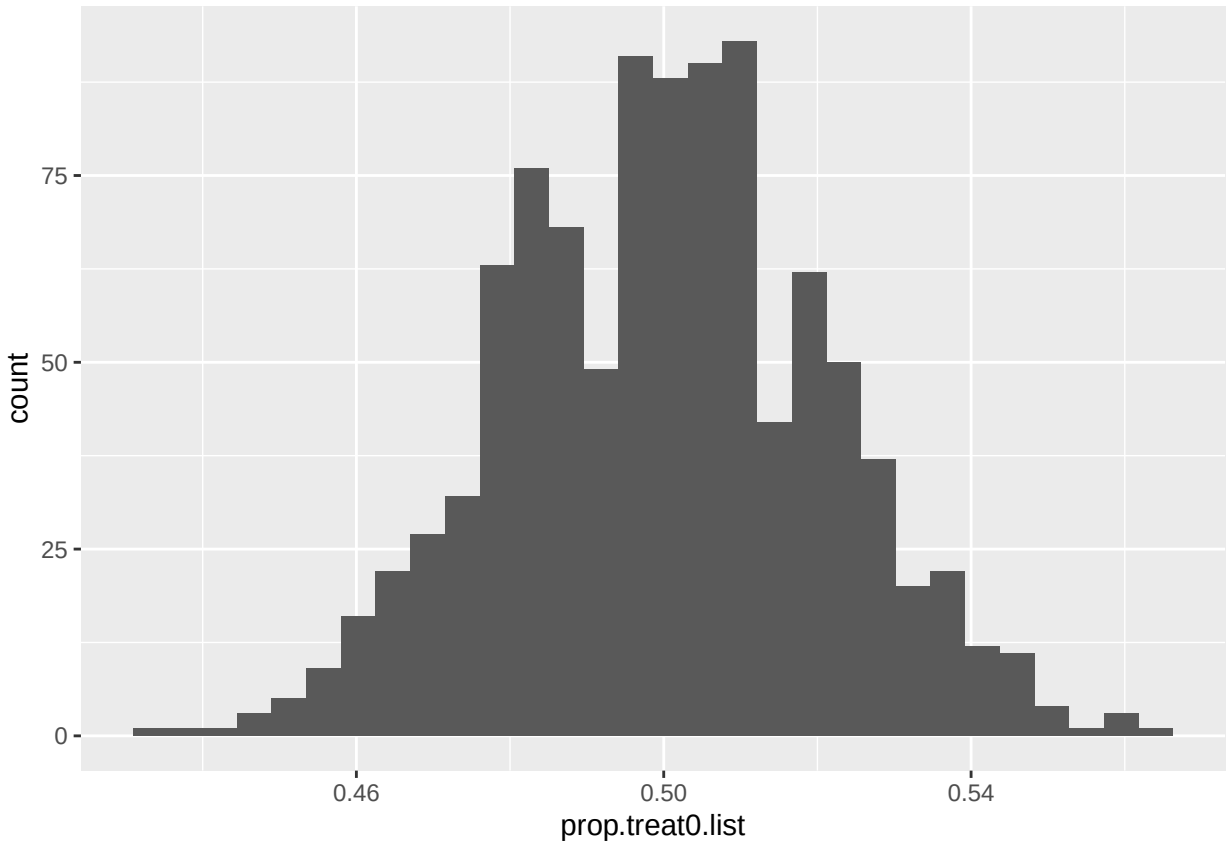
```
  geom_histogram(aes(prop.treat1.list))
```

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




```
ggplot() +
  geom_histogram(aes(prop.treat0.list))
```

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



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: As the histogram above shows, there is a normal distribution (or something close to normal distribution) of student's Gen (we assume this means gender, although an answer is not provided in any documentation of these data) provided by individuals grouped in the "treatment" class during the random assignment. This normal distribution is precisely what we would expect to find if there is independence between the exposure variable and other covariates. However, this is an observational study, not an experimental one. As such, we cannot expect independence of treatment assignment, no matter the sample size. Put simply, there may be one or more covariates that may lead units (ie, individuals) to select the treatment (ie, attend college) or the control group (ie, not attend college), for example, financial standing, parents' education level, views towards the school system, etc.

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.

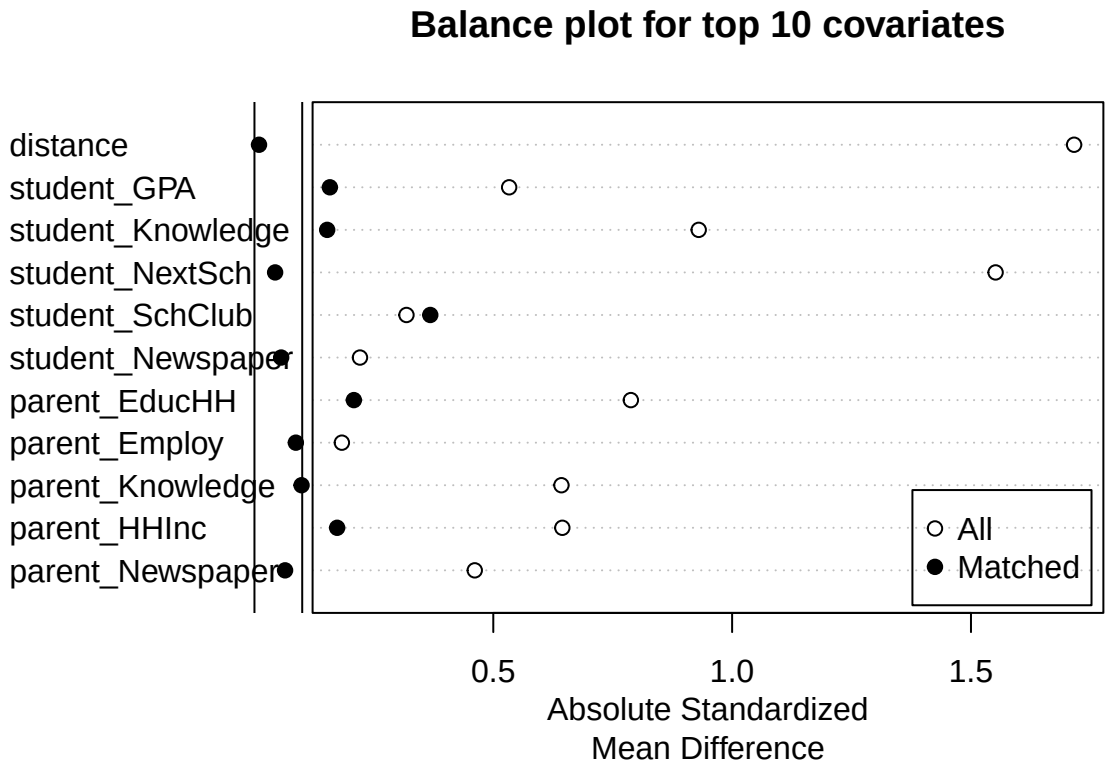
```r
# Select covariates that represent the "true" model for selection, fit model
df <-  ypsps %>%
  select(student_ppnscal, college, student_GPA, student_Knowledge, student_NextSch, student_SchClub, stu

model_ps <- glm(college ~ student_GPA + student_Knowledge + student_NextSch + student_SchClub + student_
summary(model_ps)
```

```
##
## Call:
## glm(formula = college ~ student_GPA + student_Knowledge + student_NextSch +
##     student_SchClub + student_Newspaper + parent_EducHH + parent_Employ +
##     parent_Knowledge + parent_HHInc + parent_Newspaper, family = binomial(),
##     data = df)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.4667  -0.7081   0.3878   0.7116   2.8237
##
## Coefficients:
##                   Estimate Std. Error z value Pr(>|z|)
## (Intercept)       -4.46512    0.54902  -8.133 4.19e-16 ***
## student_GPA       -0.32795    0.11422  -2.871 0.004090 **
## student_Knowledge  2.69866    0.34606   7.798 6.28e-15 ***
## student_NextSch    1.92276    0.22111   8.696  < 2e-16 ***
## student_SchClub    0.18760    0.07790   2.408 0.016031 *
## student_Newspaper  0.02151    0.05431   0.396 0.692104
## parent_EducHH      0.34322    0.06323   5.428 5.69e-08 ***
## parent_Employ      0.21765    0.15519   1.402 0.160775
## parent_Knowledge   0.49225    0.38166   1.290 0.197138
## parent_HHInc       0.12978    0.03586   3.619 0.000296 ***
## parent_Newspaper   0.03495    0.05262   0.664 0.506535
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1638.3  on 1253  degrees of freedom
## Residual deviance: 1158.5  on 1243  degrees of freedom
## AIC: 1180.5
##
## Number of Fisher Scoring iterations: 5
```

```r
# PSM
model_matchit <- matchit(formula(model_ps),
                         data = df,
                         method="nearest",
                         distance="glm",
                         link="logit",
                         estimand = "ATT",
                         replace = TRUE)

# ATT
outcome.model <- glm(student_ppnscal ~ ., data = df)
summary(outcome.model)$coefficients["college", "Estimate"]
```

```
## [1] 0.8388787
```

```
# Plot the balance for the top 10 covariates
plot(summary(model_matchit), threshold = 0.1, main = "Balance plot for top 10 covariates")
```




```
# Report the balance across treatment and controls
balance_table_full <- bal.tab(model_matchit, thresholds = .1)
balance_table_full
```

```
## Call
##  matchit(formula = formula(model_ps), data = df, method = "nearest",
##     distance = "glm", link = "logit", estimand = "ATT", replace = TRUE)
##
## Balance Measures
##                        Type Diff.Adj        M.Threshold
## distance           Distance   0.0095     Balanced, <0.1
## student_GPA         Contin.  -0.1579 Not Balanced, >0.1
## student_Knowledge   Contin.  -0.1522 Not Balanced, >0.1
## student_NextSch      Binary   0.0087     Balanced, <0.1
## student_SchClub     Contin.  -0.3681 Not Balanced, >0.1
## student_Newspaper   Contin.   0.0559     Balanced, <0.1
## parent_EducHH       Contin.   0.2081 Not Balanced, >0.1
## parent_Employ        Binary   0.0399     Balanced, <0.1
## parent_Knowledge    Contin.   0.0986     Balanced, <0.1
## parent_HHInc        Contin.   0.1731 Not Balanced, >0.1
## parent_Newspaper    Contin.   0.0641     Balanced, <0.1
##
## Balance tally for mean differences
##                     count
## Balanced, <0.1          6
## Not Balanced, >0.1      5
```

```
##
## Variable with the greatest mean difference
##          Variable Diff.Adj        M.Threshold
##  student_SchClub  -0.3681 Not Balanced, >0.1
##
## Sample sizes
##                       Control Treated
## All                    451.       803
## Matched (ESS)           39.11     803
## Matched (Unweighted)   201.       803
## Unmatched              250.         0
```

```
table(balance_table_full$Balance[-1, "M.Threshold"])["Balanced, <0.1"]  # 5 out of 10
```

```
## Balanced, <0.1
##              5
```

Simulations

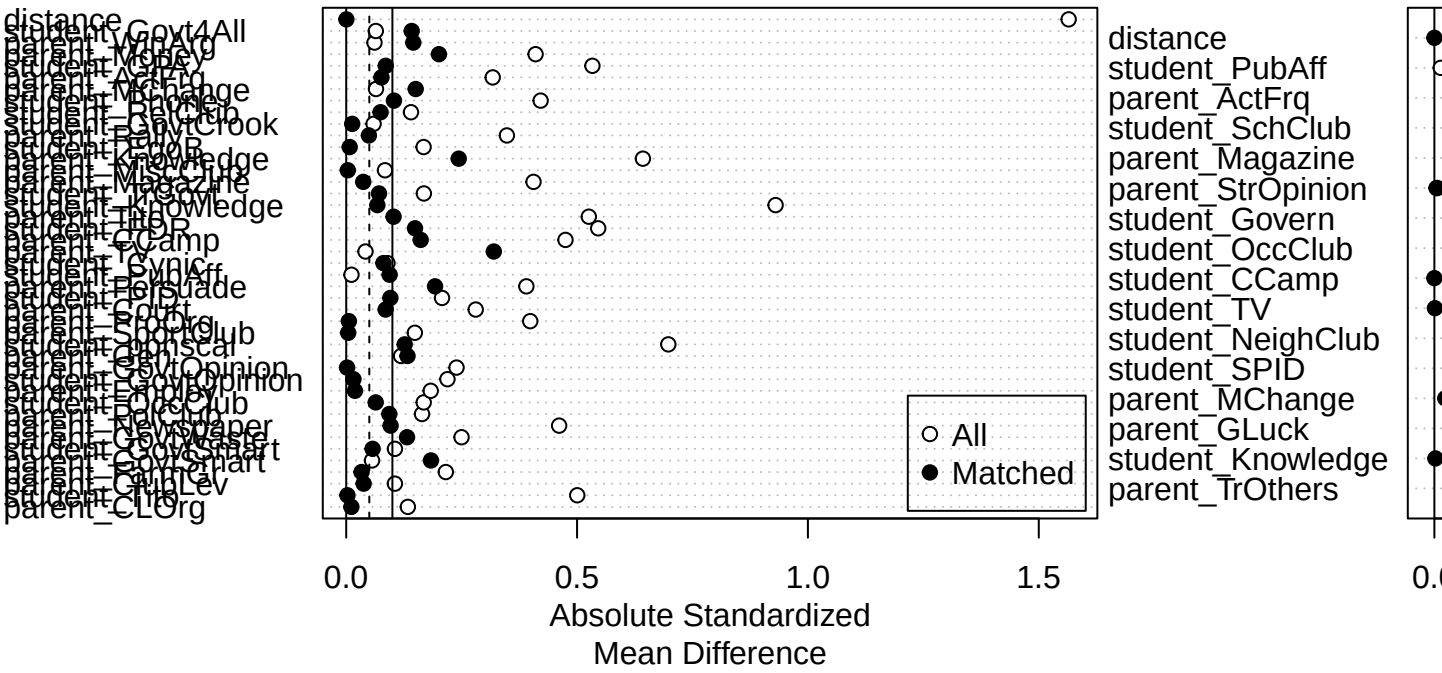
Henderson/Chatfield argue that an improperly specified propensity score model can actually *increase* the bias of the estimate. To demonstrate this, they simulate 800,000 different propensity score models by choosing different permutations of covariates. To investigate their claim, do the following:

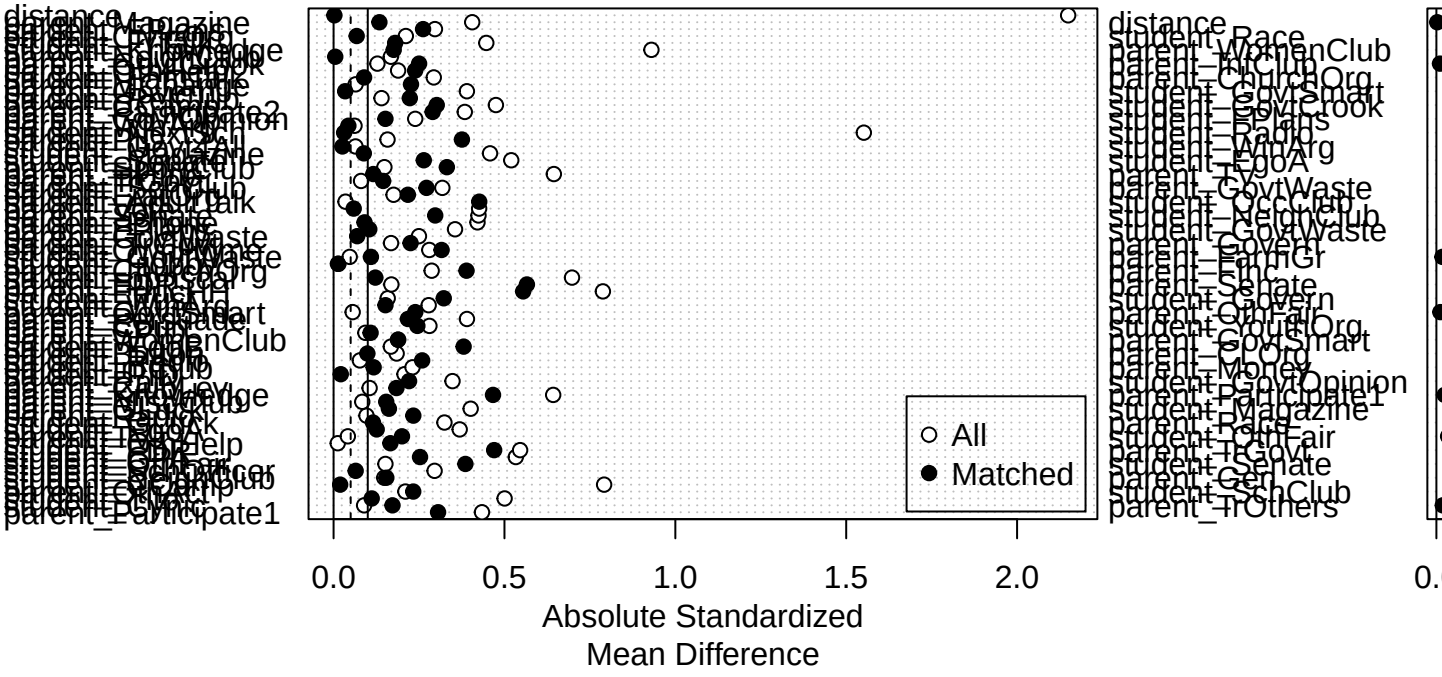
- Using as many simulations as is feasible (at least 10,000 should be ok, more is better!), randomly select the number of and the choice of covariates for the propensity score model.
- For each run, store the ATT, the proportion of covariates that meet the standardized mean difference $\leq .1$ threshold, 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)

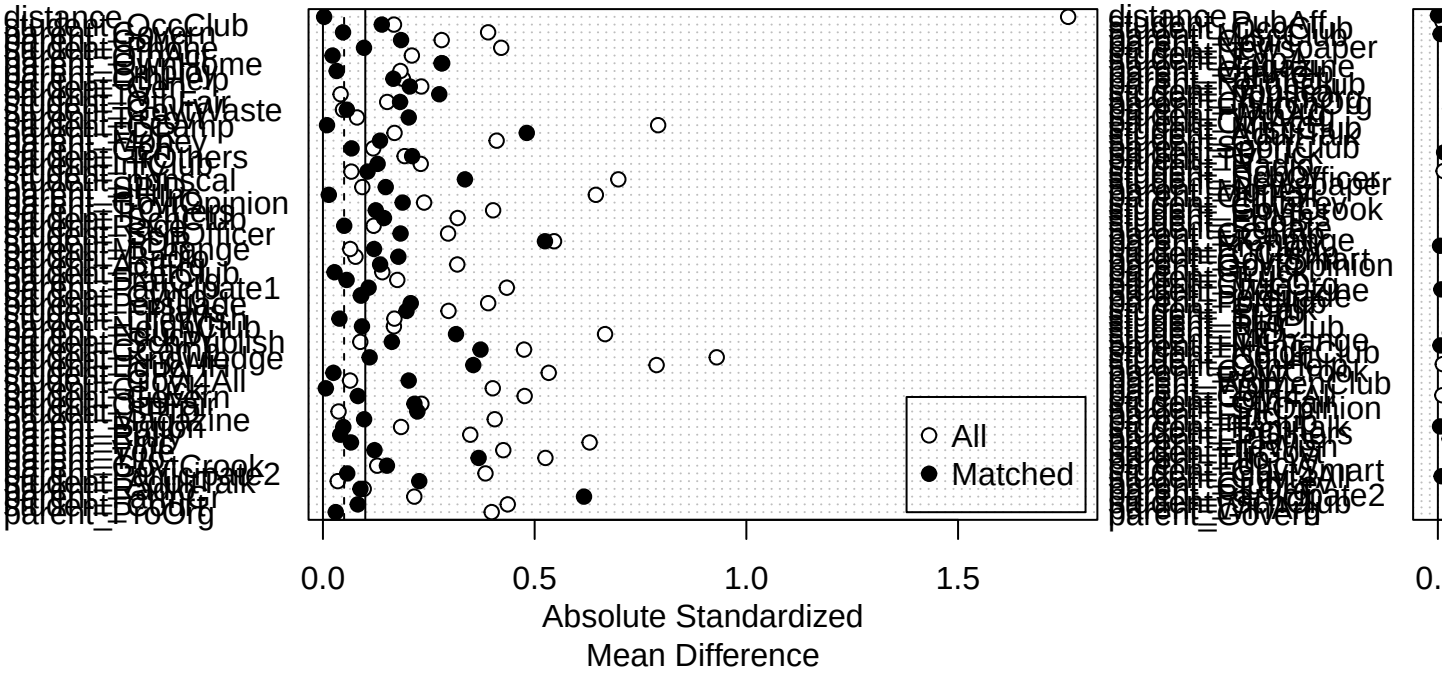
Note: There are lots of post-treatment covariates in this dataset (about 50!)! You need to be careful not to include these in the pre-treatment balancing. Many of you are probably used to selecting or dropping columns manually, or positionally. However, you may not always have a convenient arrangement 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).

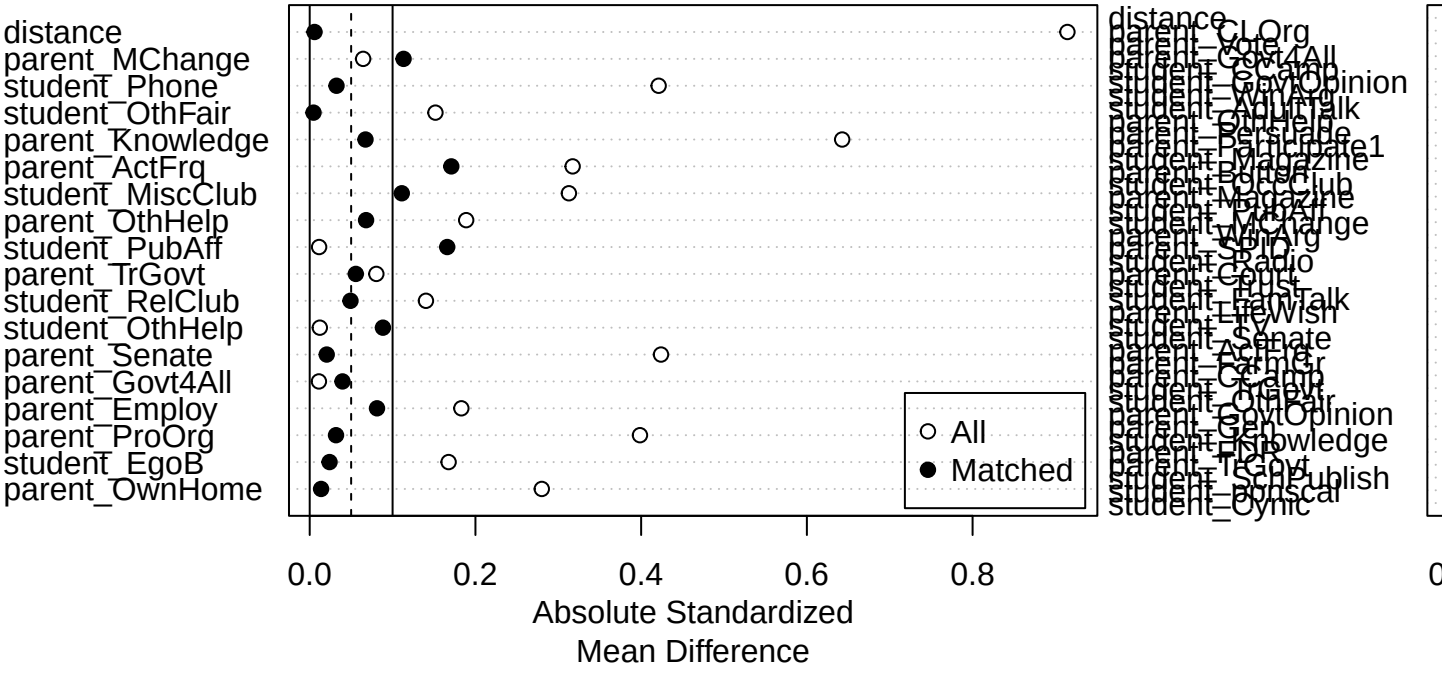
```
# Simulate random selection of features 10k+ times

# Drop columns with missing values
ypsps <- ypsps %>% select(-colnames(ypsps)[colSums(is.na(ypsps))>0])

# Remove post-treatment covariates
cov <- ypsps %>% select(!contains("1973") & !contains("1982") & !"interviewid" & !"college")

# Create empty vectors to populate ATTs, proportion of balanced covariates, and mean percent balance im
ATT = rep(NA, nsim)
balanced.cov = rep(NA, nsim)
balance.improvement = rep(NA, nsim)
```

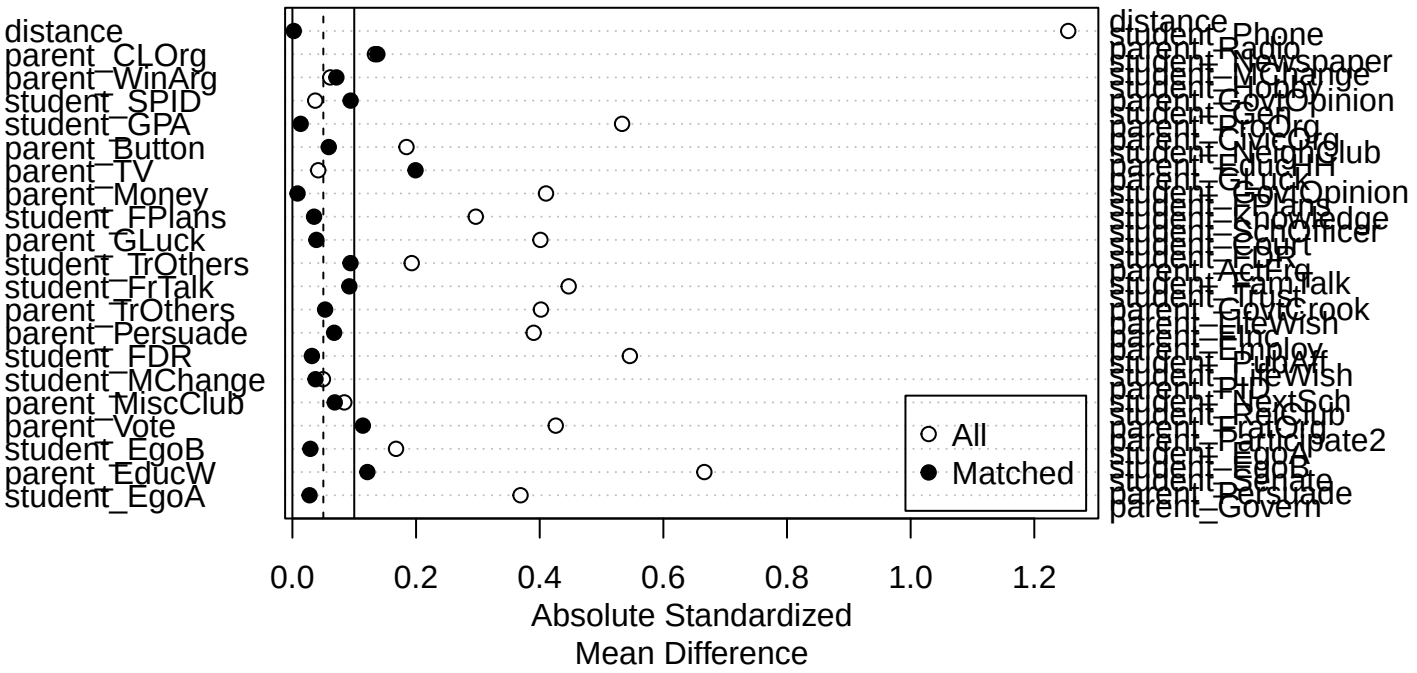
```r
# create vector of 10 random numbers from 1 to 10000
rand.num <- sample(1:nsim, 10)

for (i in 1:nsim) {

  # Randomly select features
    cov.rand <- sample(cov, size = sample(1:ncol(cov)), replace=FALSE)

  # Fit p-score models
    ps.model <- paste("college ~", paste(names(cov.rand), collapse=" + "))
    psm_matchit <- matchit(formula(ps.model),
                           data = ypsps,
                           method="nearest",
                           distance="glm",
                           link="logit",
                           estimand = "ATT",
                           replace = TRUE)

    psm_matchit.sum <- summary(psm_matchit)

  # Save ATTs, proportion of balanced covariates, and mean percent balance improvement
    psm_att_data <- match.data(psm_matchit)
    outcome.model <- paste("student_ppnscal ~ college +", paste(names(cov.rand), collapse=" + "))
    ps_att <- lm(formula(outcome.model), data = psm_att_data, weights = weights)
    ps_att_summ <- summary(ps_att)

    balance_table_full <- bal.tab(psm_matchit, thresholds = .1)
    balance_table_full$Balanced.mean.diffs # not including distance, 5 of the 10 meet the balanced thre

    ATT[i] <- ps_att_summ$coefficients["college", "Estimate"]
    balanced.cov[i] <- table(balance_table_full$Balance[-1, "M.Threshold"])["Balanced, <0.1"]/ncol(cov.
    balance.improvement[i] <- mean(psm_matchit.sum$reduction[-1,1]) # remove distance

  # 10 random covariate balance plots
    if(i %in% rand.num){
      model_name <- paste0("model_", i)
        n <-  match(i, rand.num) # get new index for where model # is in list of 10 models to save to l
        plot(summary(psm_matchit, main= paste("Balance plot for model ", i)))
  }
}
```


distance
student_PubAff
parent_ActFrq
student_SchClub
parent_Magazine
parent_StrOpinion
student_Govern
student_OccClub
student_CCamp
student_TV
student_NeighClub
student_SPID
parent_MChange
parent_GLuck
student_Knowledge
parent_TrOthers
All
Matched
0.0
0.5
1.0
1.5
Absolute Standardized
Mean Difference


distance
All
Matched
0.0
0.5
1.0
1.5
2.0
Absolute Standardized
Mean Difference

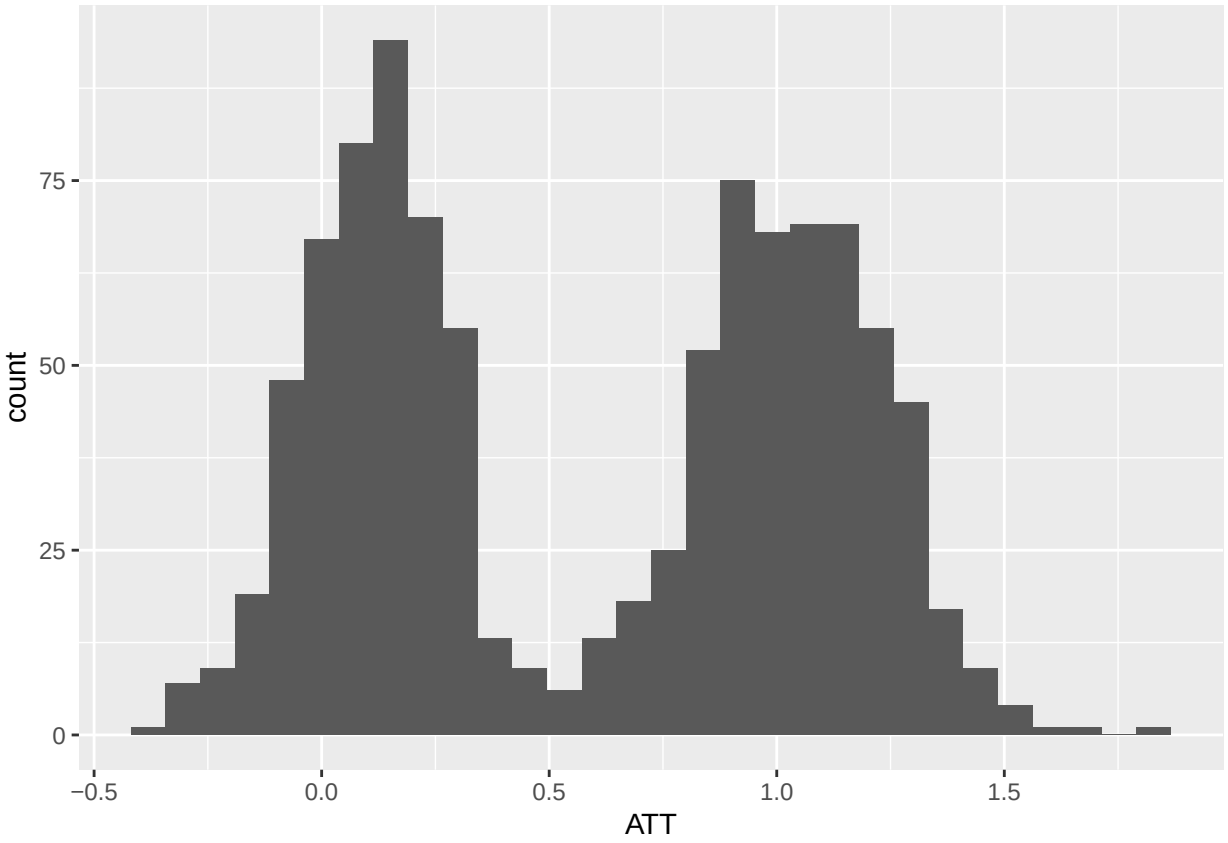

All
Matched
0.0
0.5
1.0
1.5
Absolute Standardized
Mean Difference


distance
parent_MChange
student_Phone
student_OthFair
parent_Knowledge
parent_ActFrq
student_MiscClub
parent_OthHelp
student_PubAff
parent_TrGovt
student_RelClub
student_OthHelp
parent_Senate
parent_Govt4All
parent_Employ
parent_ProOrg
student_EgoB
parent_OwnHome
All
Matched
0.0
0.2
0.4
0.6
0.8
Absolute Standardized
Mean Difference

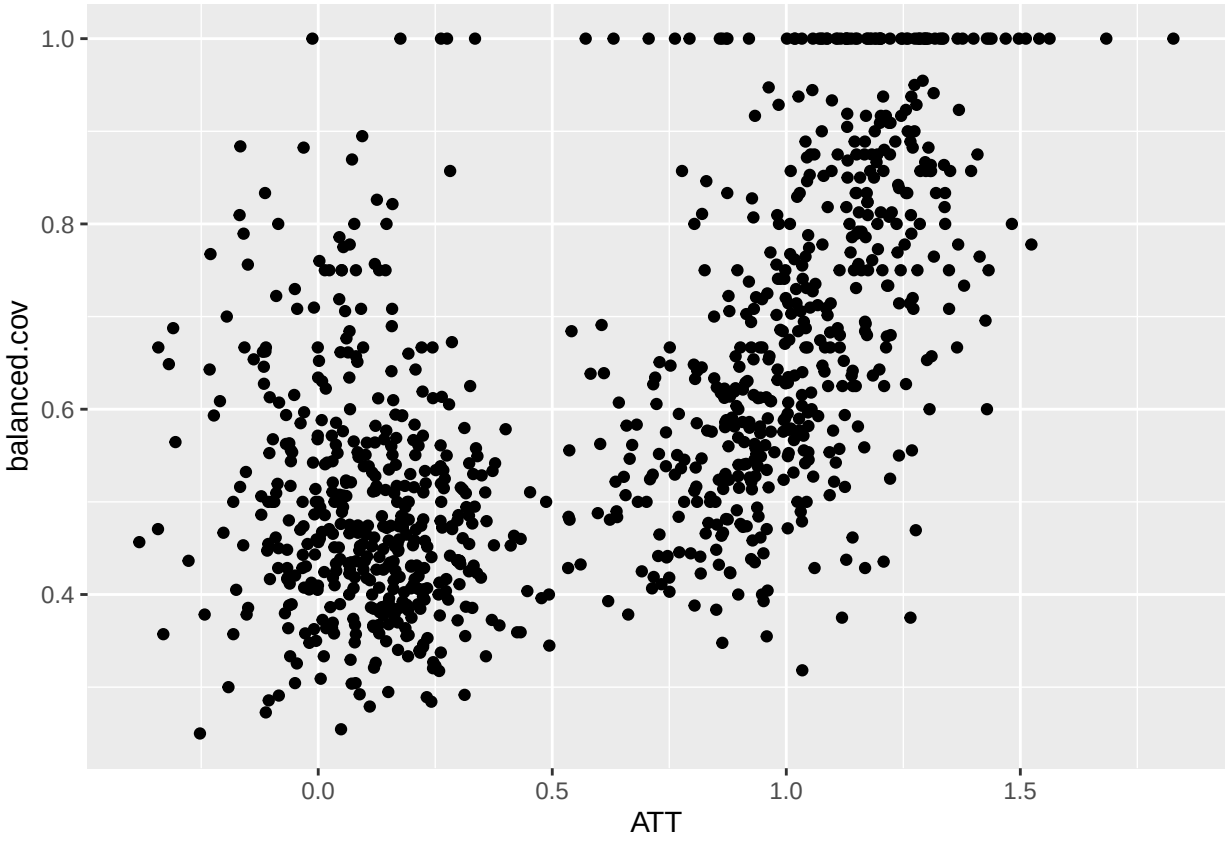



```
# Histogram of ATTs from simulation
ggplot() +
  geom_histogram(aes(x=ATT))
```

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



```
# Plot ATT v. proportion
ggplot() +
geom_point(aes(x=ATT, y=balanced.cov)) +
geom_smooth(method = "lm", se = FALSE)
```



Questions

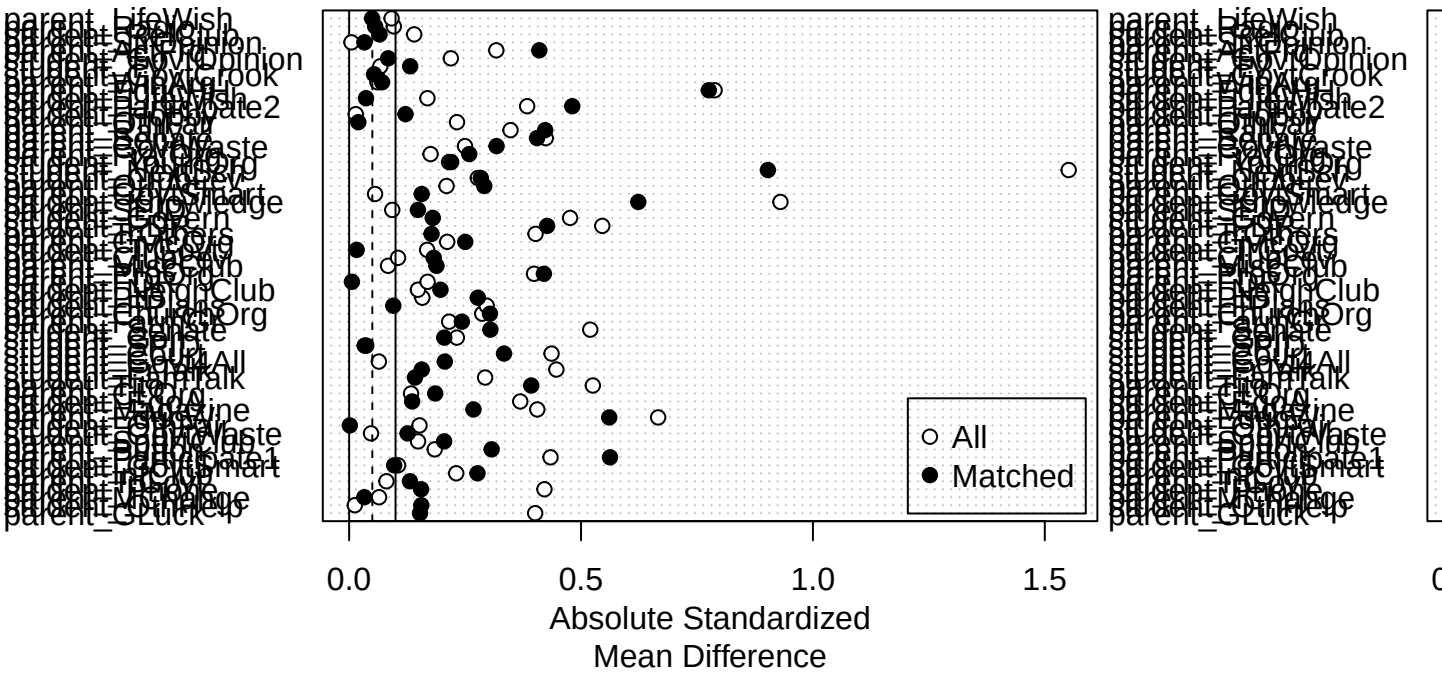
1. **How many simulations resulted in models with a higher proportion of balanced covariates? Do you have any concerns about this?**
2. **Your Answer**:
3. **Analyze the distribution of the ATTs. Do you have any concerns about this distribution?**
4. **Your Answer:**
5. **Do your 10 randomly chosen covariate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

Matching Algorithm of Your Choice

Simulate Alternative Model

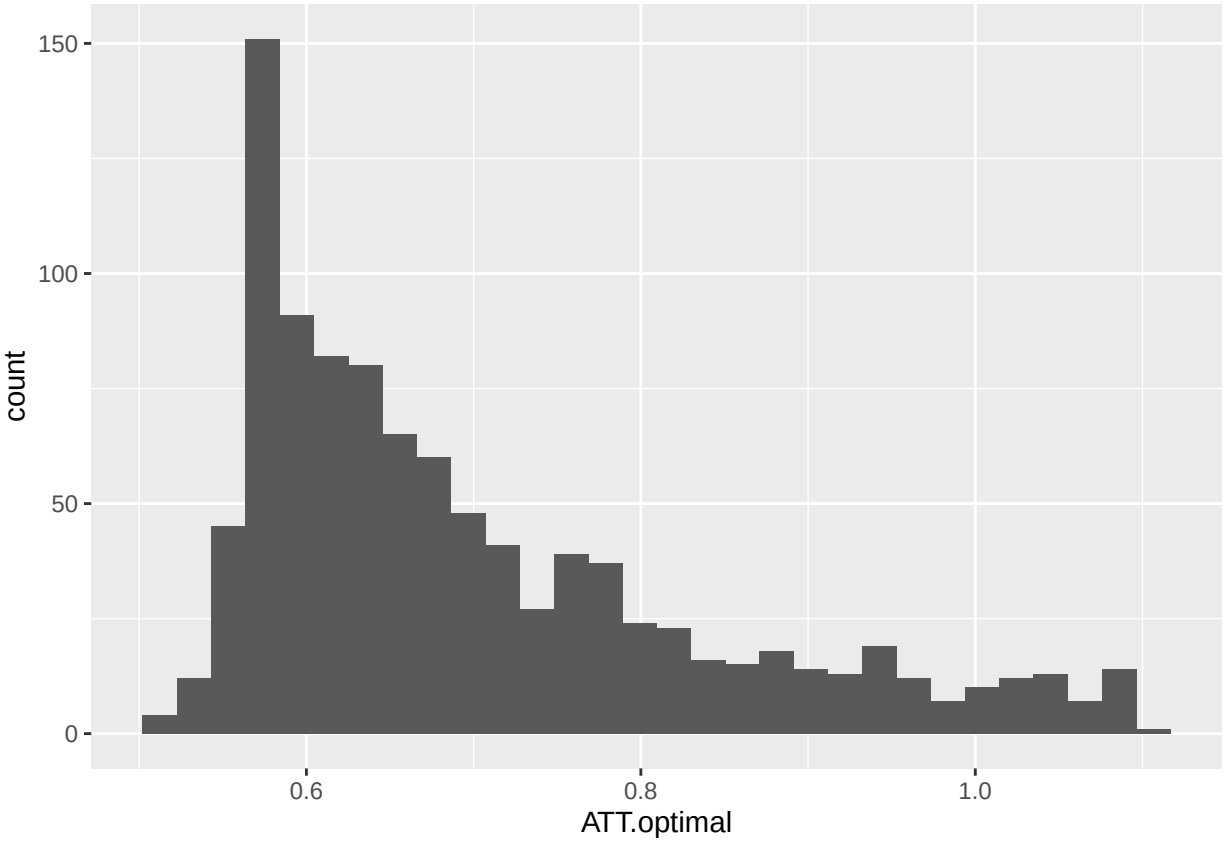
Henderson/Chatfield propose using genetic matching to learn the best weights for Mahalanobis distance matching. Choose a matching algorithm other than the propensity score (you may use genetic matching if you wish, but it is also fine to use the greedy or optimal algorithms we covered in lab instead).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

```r
## Full Optimal Mahalanobis Matching

# Simulate random selection of features 10k+ times

# Create empty vectors to populate ATTs, proportion of balanced covariates, and mean percent balance im
ATT.optimal = rep(NA, nsim)
balanced.cov.optimal = rep(NA, nsim)
balance.improvement.optimal = rep(NA, nsim)

for (i in 1:nsim) {

  # Randomly select features
    cov.rand <- sample(cov, size = sample(1:ncol(cov)), replace=FALSE)

  # Fit p-score models
    optimal.model <- paste("college ~", paste(names(cov.rand), collapse=" + "))
    optimal_matchit <- matchit(formula(ps.model),
                         data = ypsps,
                         method="full",
                         distance="mahalanobis",
                         estimand = "ATT",
                         replace = TRUE)

    optimal_matchit.sum <- summary(optimal_matchit)

  # Save ATTs, proportion of balanced covariates, and mean percent balance improvement
    optimal_att_data <- match.data(optimal_matchit)
    outcome.model <- paste("student_ppnscal ~ college +", paste(names(cov.rand), collapse=" + "))
  optimal_att <- lm(formula(outcome.model), data = optimal_att_data, weights = weights)
    optimal_att_summ <- summary(optimal_att)

    balance_table_full <- bal.tab(optimal_matchit, thresholds = .1)
    balance_table_full$Balanced.mean.diffs # not including distance, 5 of the 10 meet the balanced thre

    ATT.optimal[i] <- optimal_att_summ$coefficients["college", "Estimate"]
    balanced.cov.optimal[i] <- table(balance_table_full$Balance[-1, "M.Threshold"])["Balanced, <0.1"]/n
    balance.improvement.optimal[i] <- mean(optimal_matchit.sum$reduction[-1,1]) # remove distance

  # 10 random covariate balance plots
    if(i %in% rand.num){
      model_name <- paste0("model_", i)
        n <-  match(i, rand.num) # get new index for where model # is in list of 10 models to save to l
        plot(summary(optimal_matchit, main= paste("Balance plot for model ", i)))
  }
}
```

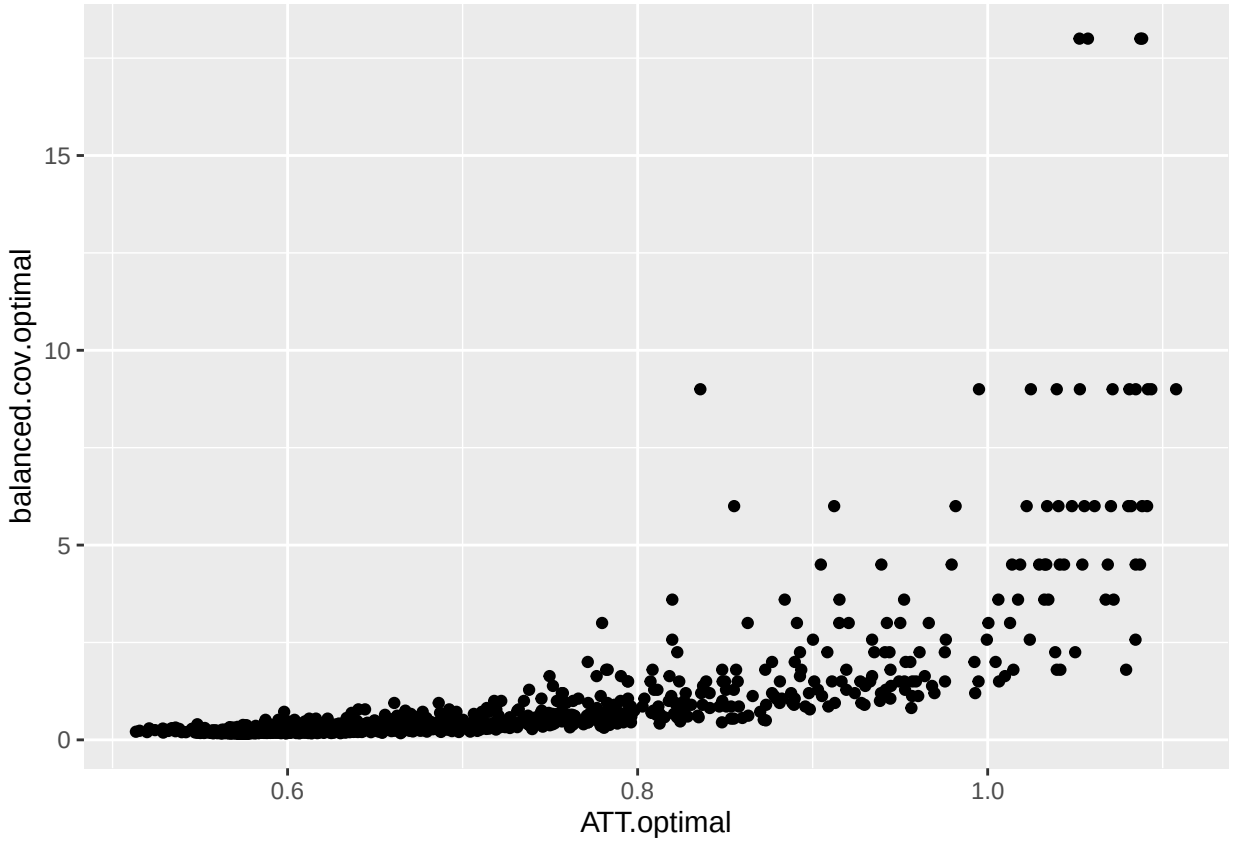



```
# Histogram of ATTs from simulation
ggplot() +
  geom_histogram(aes(x=ATT.optimal))
```

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

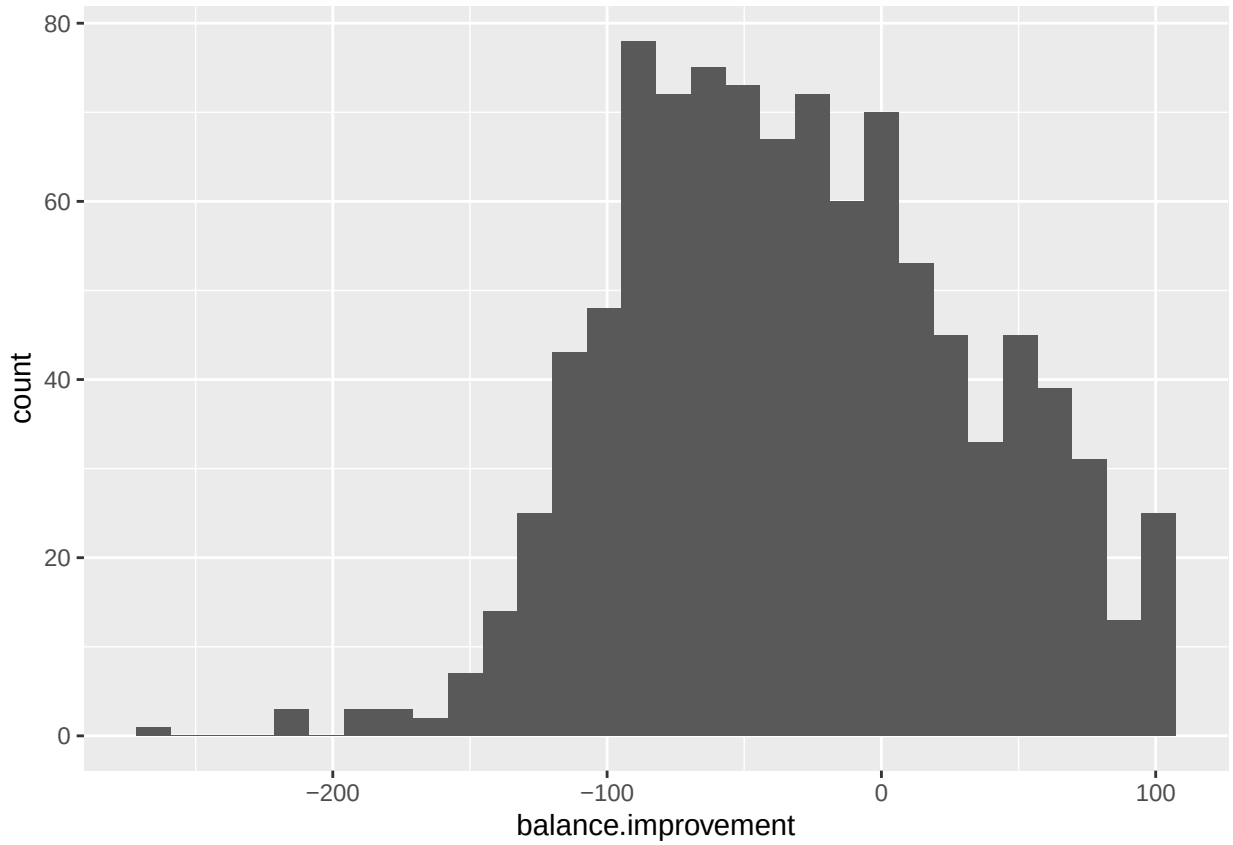


```
# Plot ATT v. proportion
ggplot() +
geom_point(aes(x=ATT.optimal, y=balanced.cov.optimal))+
geom_smooth(method = "lm", se = FALSE)
```



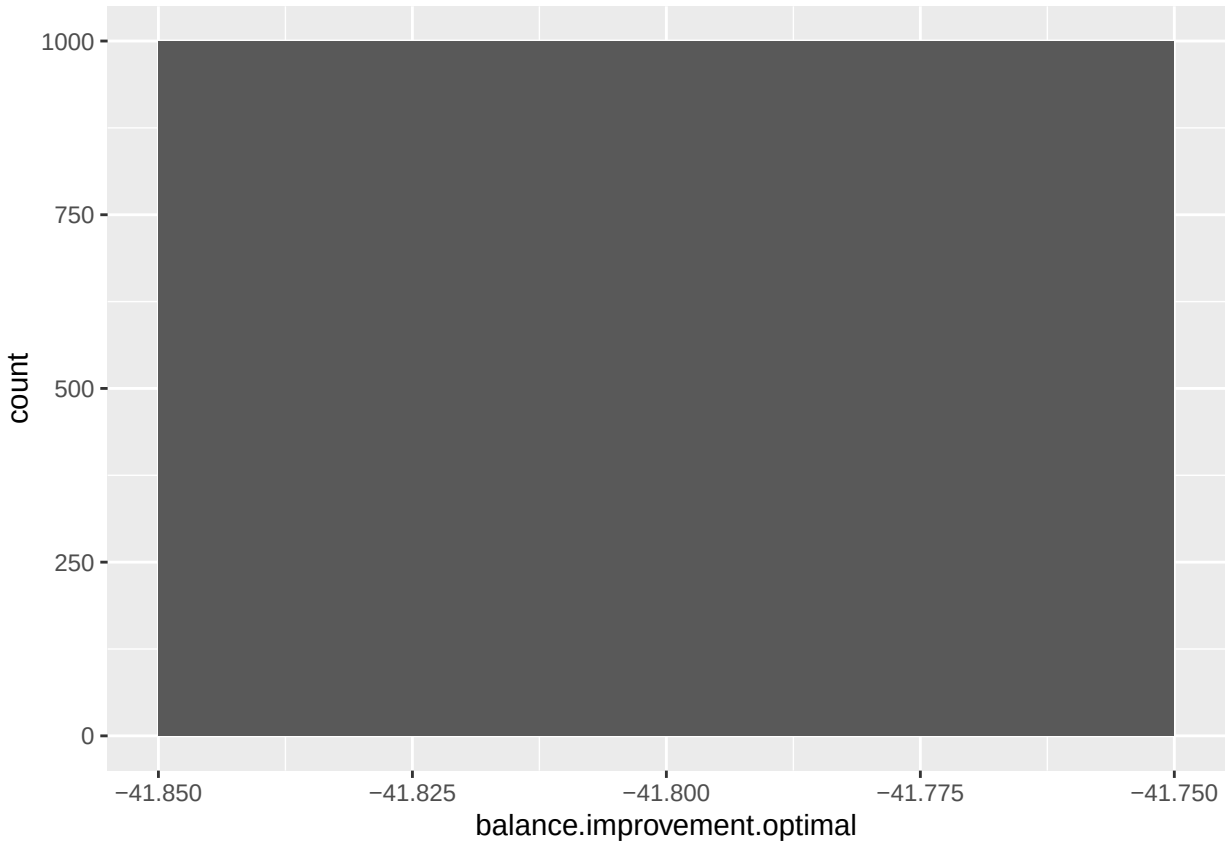

```
# Plot Distribution of Balance Improvement for the PS and Alternative Models
ggplot() +
  geom_histogram(aes(x=balance.improvement))
```

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




```
ggplot() +
  geom_histogram(aes(x=balance.improvement.optimal))
```

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



Questions

1. **Does your alternative matching method have more runs with higher proportions of balanced covariates?**
2. **Your Answer:**
3. **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.**
4. **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?
2. **Your Answer:**
3. 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?
4. **Your Answer:**