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 1.3.2 --
## v ggplot2 3.4.1     v purrr   1.0.1
## v tibble  3.1.8     v dplyr   1.1.0
## v tidyr   1.3.0     v stringr 1.5.0
## v readr   2.1.4     v forcats 1.0.0
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(MatchIt)

# 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_~1 stude~2 stude~3 stude~4 stude~5 stude~6 stude~7
##         <dbl>   <dbl>      <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>
## 1           1       1          1       0       0       0       0       0       1
## 2           2       1          1       1       1       1       0       1       1
## 3           3       1          1       0       0       1       0       0       0
## 4           4       0          0       0       0       0       0       1       0
## 5           5       1          1       1       0       0       0       1       0
## 6           6       1          1       0       0       0       1       0       0
## # ... with 165 more variables: 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>,
## #   student_Govt4All <dbl>, student_Cynic <dbl>, student_LifeWish <dbl>, ...
```

Randomization

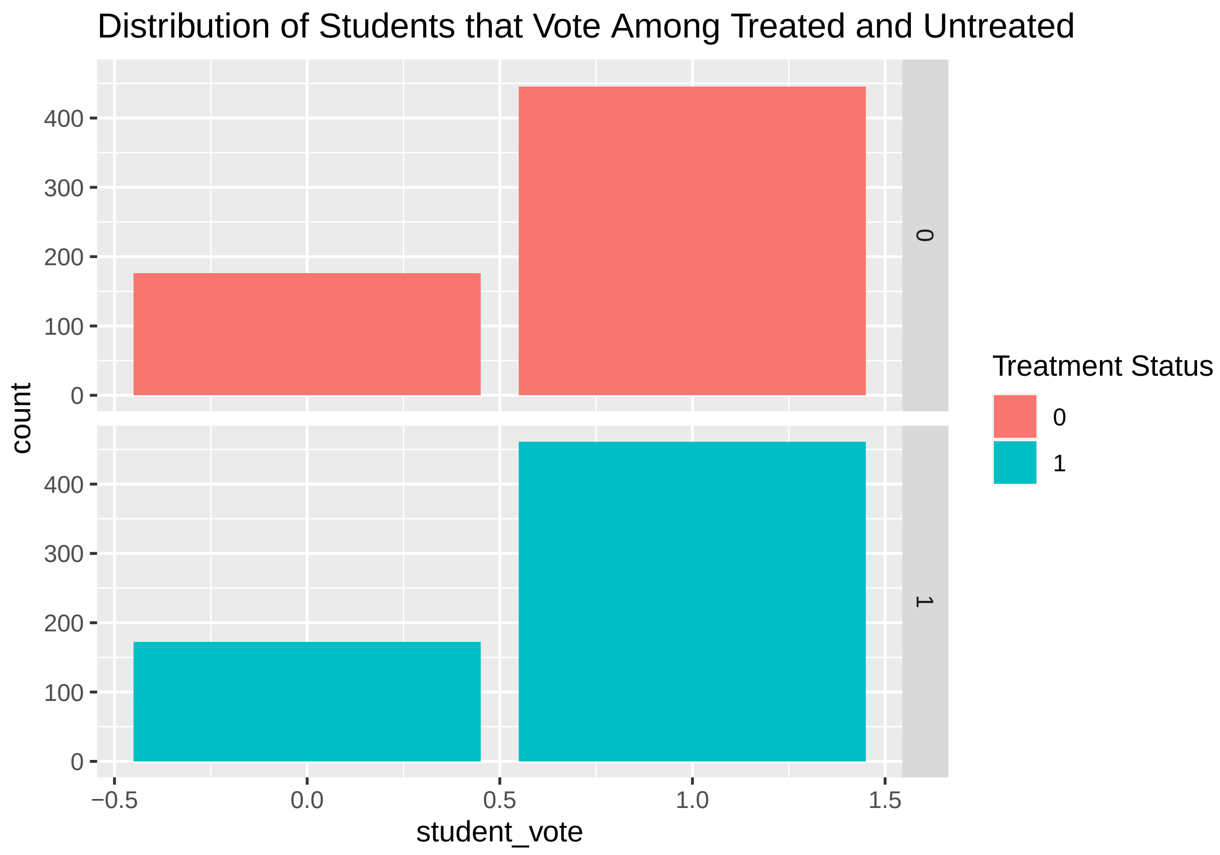
Matching is usually used in observational studies to to approximate random assignment to treatment. But could it be useful even in randomized studies? To explore the question do the following:

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
ypsps_random <- ypsps %>%
  mutate(A_comp = as.numeric(rbernoulli(nrow(ypsps), p=0.5)),
         Y_comp = as.numeric((A_comp & college==1) | (!A_comp & college==0)))
```

```
## Warning: There was 1 warning in ‘mutate()‘.
## i In argument: ‘A_comp = as.numeric(rbernoulli(nrow(ypsps), p = 0.5))‘.
## Caused by warning:
## ! ‘rbernoulli()‘ was deprecated in purrr 1.0.0.
```

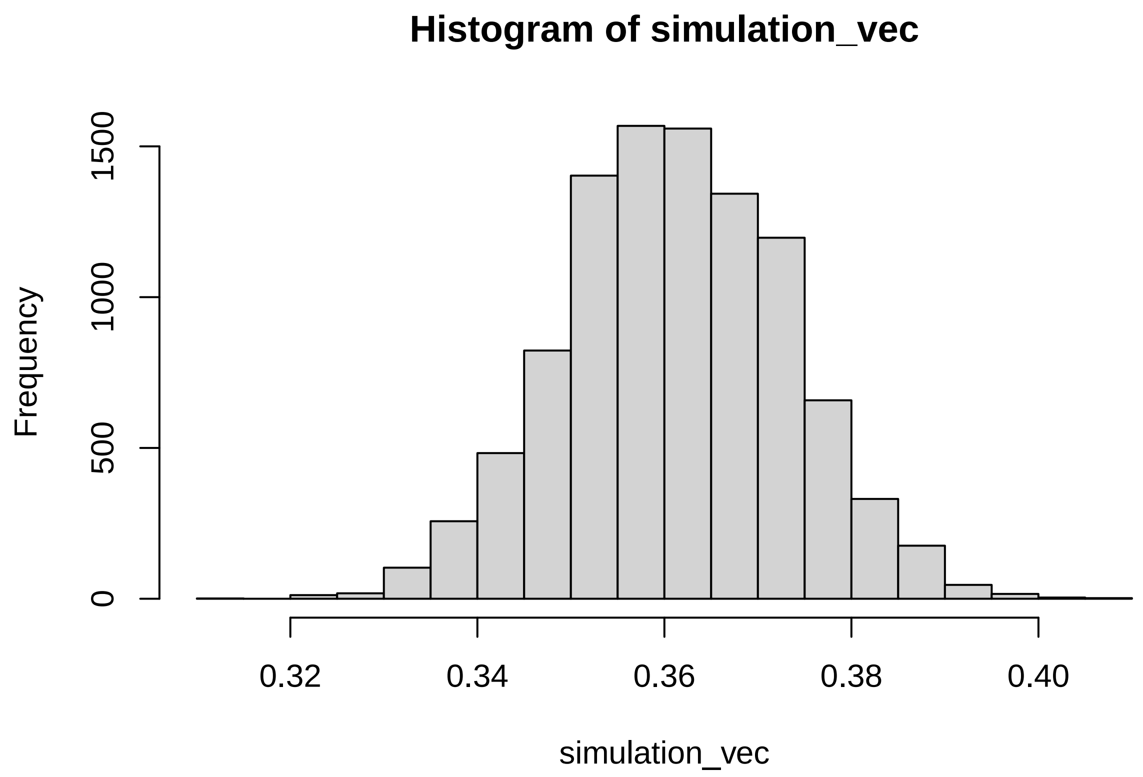
```
# Choose a baseline covariate (use dplyr for this)
ypsps_random <- ypsps_random %>% dplyr::select(Y_comp, student_vote)
# Visualize the distribution by treatment/control (ggplot)
ypsps_random %>% ggplot(aes(x=student_vote, fill=factor(Y_comp))) +
  geom_bar() + facet_grid(Y_comp~.) +
  labs(title = "Distribution of Students that Vote Among Treated and Untreated",
       fill = "Treatment Status")
```



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

#### Saving this as an rds file because it took ~5 minutes to run ####
# simulation_n <- 1:10000
# return_prop <- function(n) {
#   ypsps_simulate <- ypsps %>%
#   mutate(A_comp = as.numeric(rbernoulli(nrow(ypsps), p=0.5)),
#          Y_comp = as.numeric((A_comp & college==1) | (!A_comp & college==0))) %>%
#   dplyr::select(Y_comp, student_vote) %>%
#   count(Y_comp, student_vote) %>%
#   mutate(prop = n/sum(n))
#
#   return(ypsps_simulate$prop[4])
# }
# simulation_vec <- sapply(simulation_n, return_prop)
# saveRDS(simulation_vec, file = "simulation_vec.Rds")
```

Treatment and control are balanced on this covariate.

```
simulation_vec <- readRDS("data/simulation_vec.Rds")
hist(simulation_vec)
```



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?**

Here I only show the proportion of the control units that have the chosen covariate of whether students voted. It definitely varies and does not remain consistent. Independence of treatment assignment and baseline covariates does not guarantee balance of treatment assignment and baseline covariates because the units selected for treatment or control have attached covariate relationships that need to be matched to maintain balance in the covariate distribution.:

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 less than or equal 0.1, report the number of covariates that meet that balance threshold.

```r
# Select covariates that represent the "true" model for selection, fit model
match_exact_att <- matchit(formula = college ~ student_community + student_GPA +
                             student_AdultTalk + student_GovtOpinion +
                             student_SchOfficer + student_YouthOrg + parent_HHInc +
                             parent_Employ,
                           data = ypsps, method = "exact", estimand = "ATT")
summary(match_exact_att, un = FALSE)
```

```
##
## Call:
## matchit(formula = college ~ student_community + student_GPA +
##     student_AdultTalk + student_GovtOpinion + student_SchOfficer +
##     student_YouthOrg + parent_HHInc + parent_Employ, data = ypsps,
##     method = "exact", estimand = "ATT")
##
## Summary of Balance for Matched Data:
##                     Means Treated Means Control Std. Mean Diff. Var. Ratio
## student_community          0.0781        0.0781               0          .
## student_GPA                2.3750        2.3750               0     0.9954
## student_AdultTalk          3.3984        3.3984               0     0.9954
## student_GovtOpinion        1.5703        1.5703               0     0.9954
## student_SchOfficer         2.2266        2.2266               0     0.9954
## student_YouthOrg           1.0391        1.0391               0     0.9954
## parent_HHInc               6.8984        6.8984               0     0.9954
## parent_Employ              0.8047        0.8047               0          .
##                     eCDF Mean eCDF Max Std. Pair Dist.
## student_community           0        0               0
## student_GPA                 0        0               0
## student_AdultTalk           0        0               0
## student_GovtOpinion         0        0               0
## student_SchOfficer          0        0               0
## student_YouthOrg            0        0               0
## parent_HHInc                0        0               0
## parent_Employ               0        0               0
##
## Sample Sizes:
##               Control Treated
## All             451.      803
## Matched (ESS)    80.5     128
## Matched         120.      128
## Unmatched       331.      675
## Discarded         0.        0
```

```r
match_exact_att_data <- match.data(match_exact_att)
lm_exact_att <- lm(student_ppnscal ~ college + student_community + student_GPA +
                             student_AdultTalk + student_GovtOpinion +
                             student_SchOfficer + student_YouthOrg + parent_HHInc +
                             parent_Employ,
                   data = match_exact_att_data, weights = weights)
lm_exact_att_summ <- summary(lm_exact_att)
lm_exact_att_summ
```

```
##
```

```
## Call:
## lm(formula = student_ppnscal ~ college + student_community +
##     student_GPA + student_AdultTalk + student_GovtOpinion + student_SchOfficer +
##     student_YouthOrg + parent_HHInc + parent_Employ, data = match_exact_att_data,
##     weights = weights)
##
## Weighted Residuals:
##     Min      1Q  Median      3Q     Max
## -2.3733 -0.8989 -0.1118  0.7409  4.5179
##
## Coefficients:
##                      Estimate Std. Error t value Pr(>|t|)
## (Intercept)          2.328261   0.832690   2.796   0.0056 **
## college              0.973958   0.174979   5.566 7.00e-08 ***
## student_community    1.588282   0.352626   4.504 1.04e-05 ***
## student_GPA         -0.302455   0.179196  -1.688   0.0927 .
## student_AdultTalk   -0.075369   0.099087  -0.761   0.4476
## student_GovtOpinion -0.136180   0.150598  -0.904   0.3668
## student_SchOfficer  -0.029419   0.107567  -0.273   0.7847
## student_YouthOrg     0.150361   0.343166   0.438   0.6617
## parent_HHInc        -0.005639   0.060750  -0.093   0.9261
## parent_Employ       -0.224240   0.234188  -0.958   0.3393
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.377 on 238 degrees of freedom
## Multiple R-squared:  0.2119, Adjusted R-squared:  0.1821
## F-statistic: 7.112 on 9 and 238 DF,  p-value: 4.047e-09
```

```
ATT_exact <- lm_exact_att_summ$coefficients["college", "Estimate"]
ATT_exact
```

```
## [1] 0.9739583
```

\textbf{The ATT is ~0.97, which means that among those actually treated, going to college increases the number of political activities you participate in later in life by about 97% on average.}

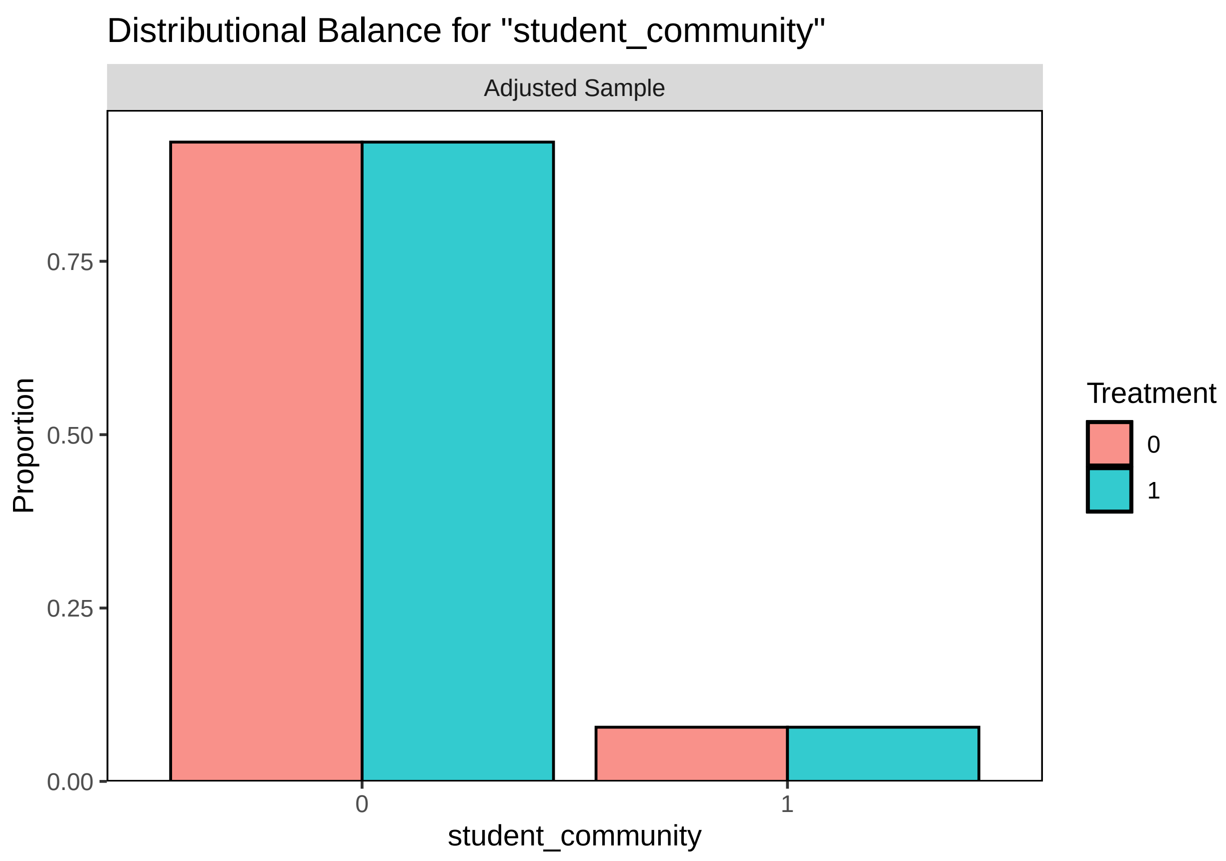
```
# Plot the balance for the top 10 covariates
library("cobalt")
```

```
##  cobalt (Version 4.5.0, Build Date: 2023-03-21)
```

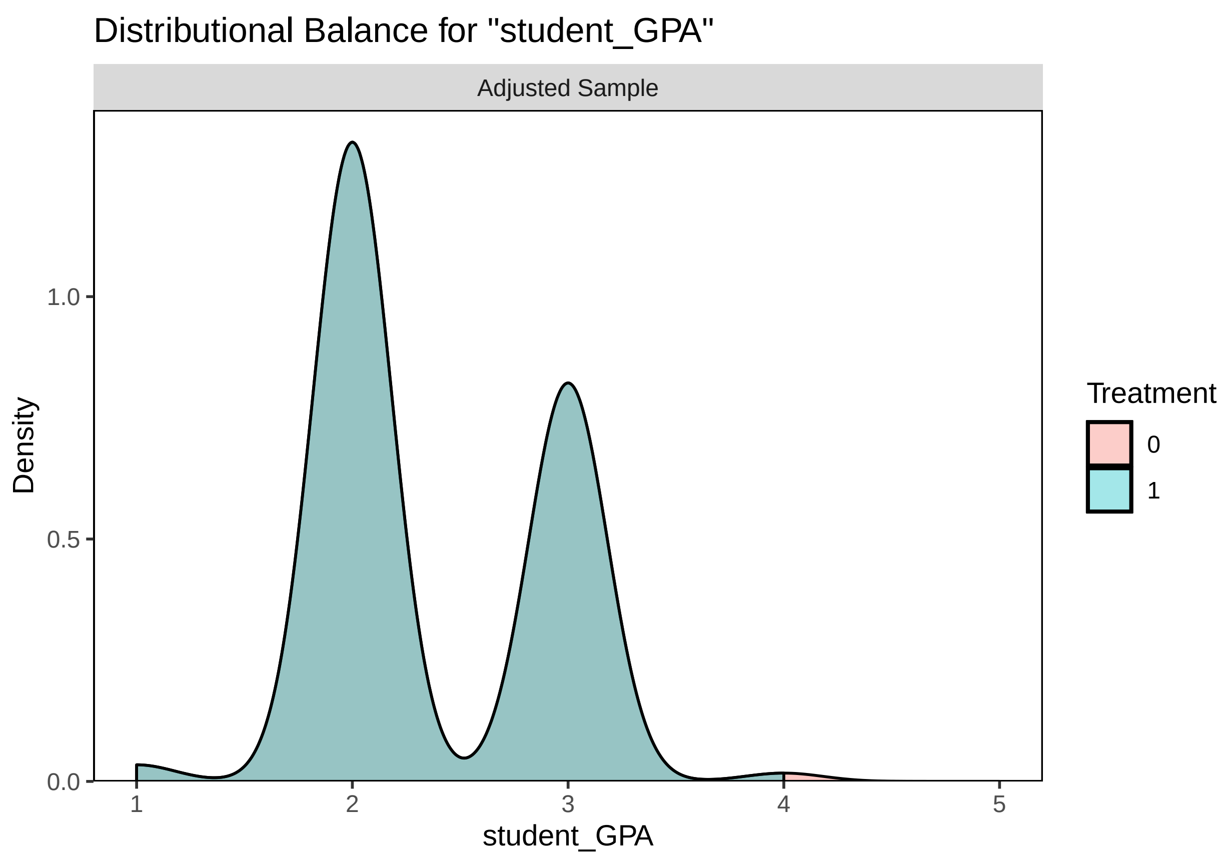
```
##
## Attaching package: 'cobalt'
```

```
## The following object is masked from 'package:MatchIt':
##
##     lalonde
```

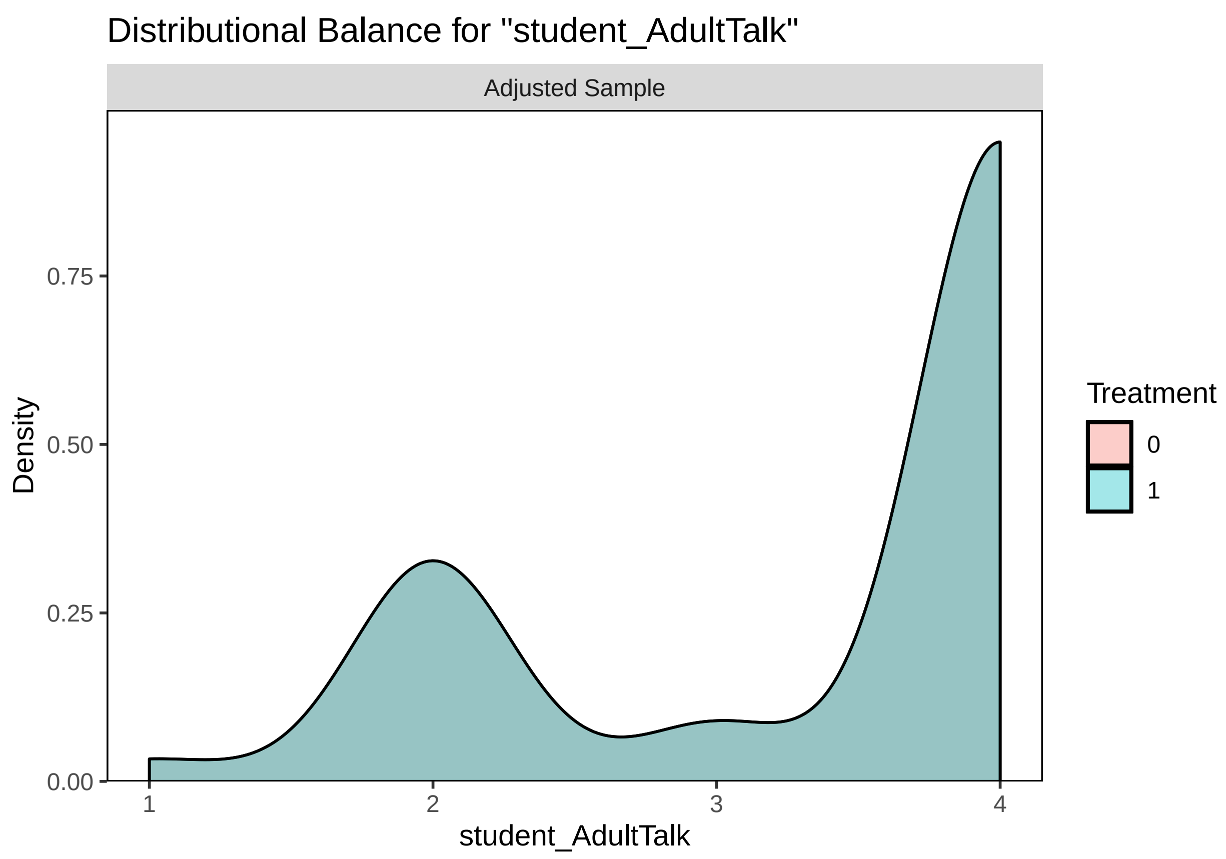
```
# Report the overall balance and the proportion of covariates that meet the balance threshold
bal.plot(match_exact_att, var.name = "student_community")
```



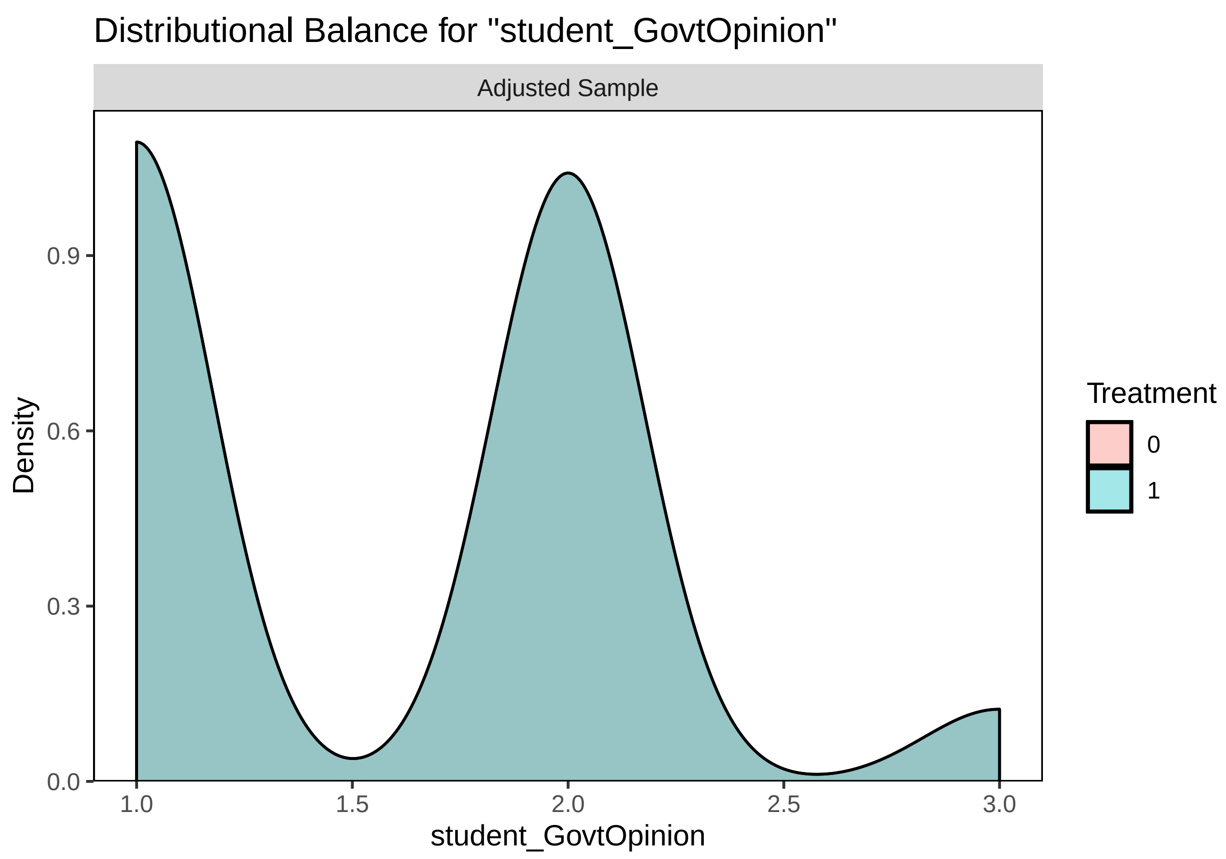
```r
bal.plot(match_exact_att, var.name = "student_GPA")
```



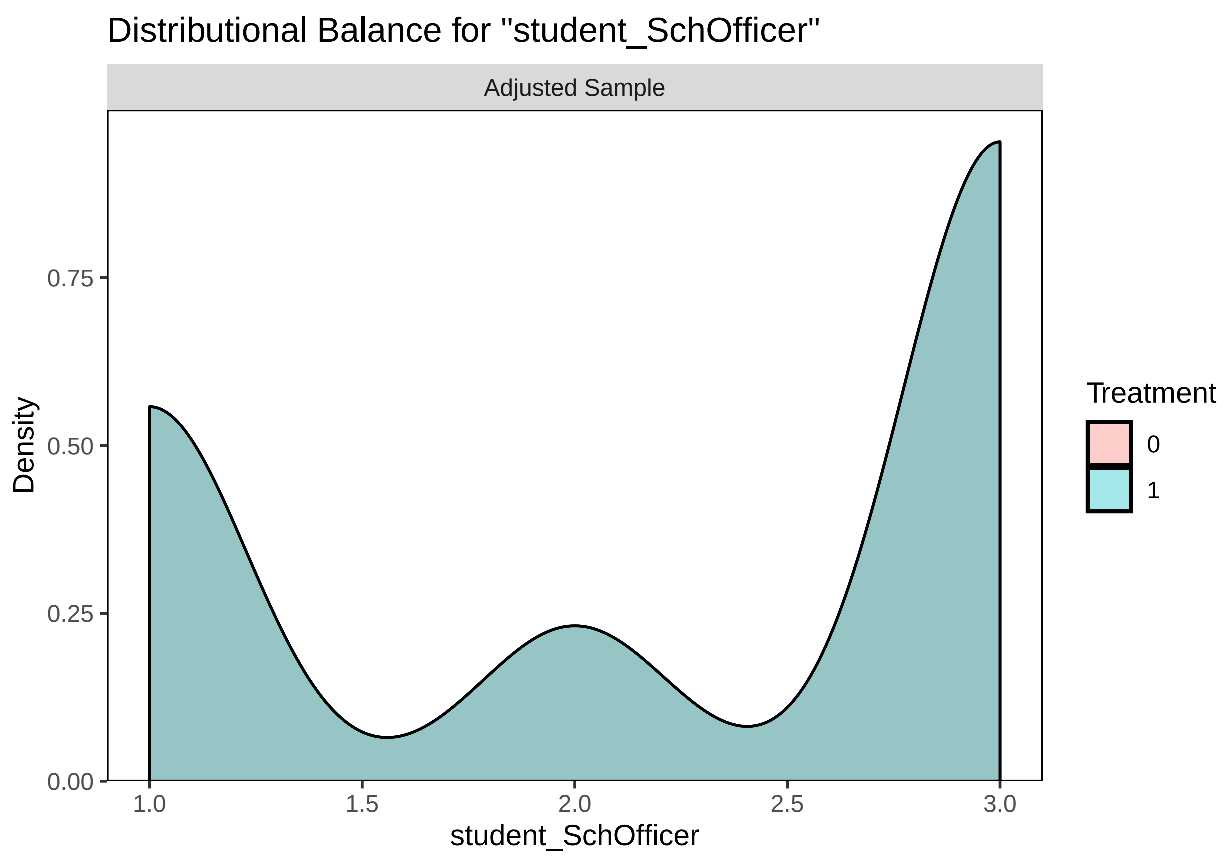
```r
bal.plot(match_exact_att, var.name = "student_AdultTalk")
```



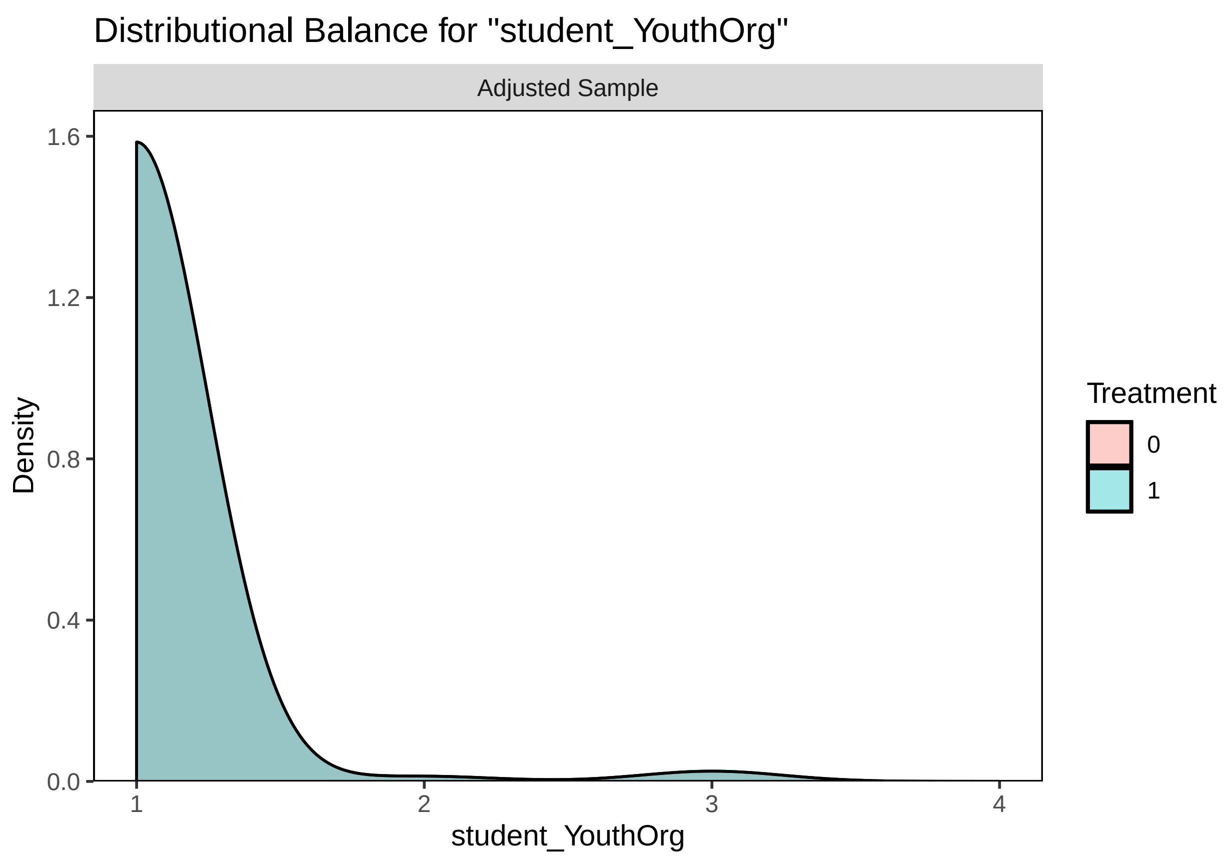
```
bal.plot(match_exact_att, var.name = "student_GovtOpinion")
```



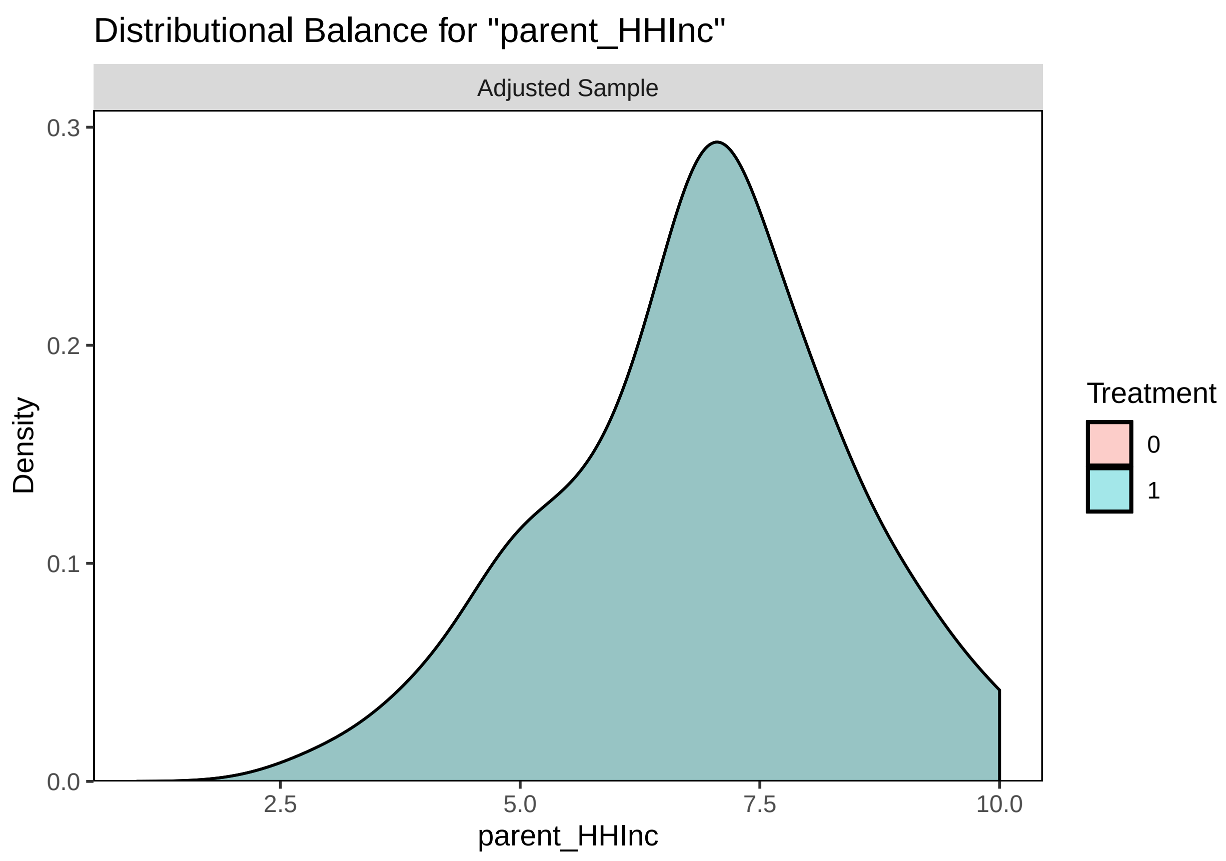
```r
bal.plot(match_exact_att, var.name = "student_SchOfficer")
```



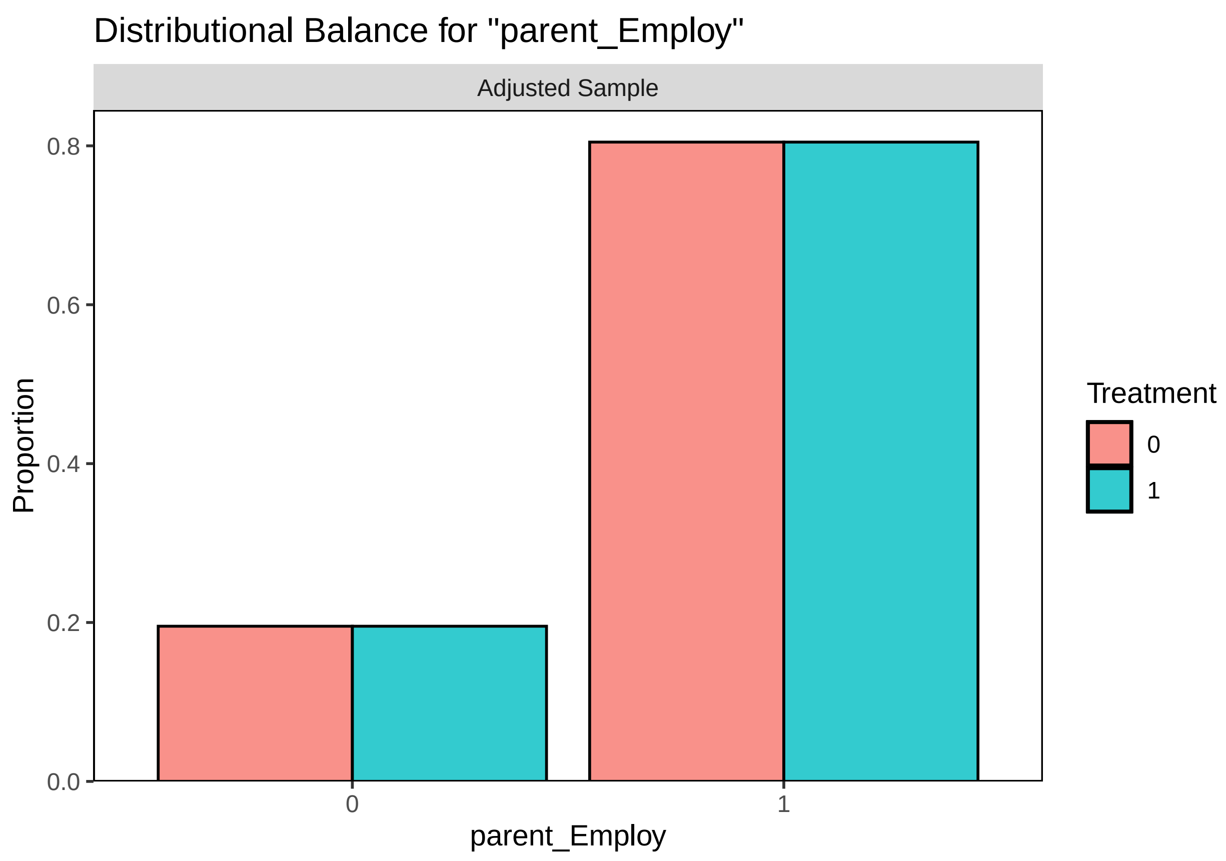
```
bal.plot(match_exact_att, var.name = "student_YouthOrg")
```



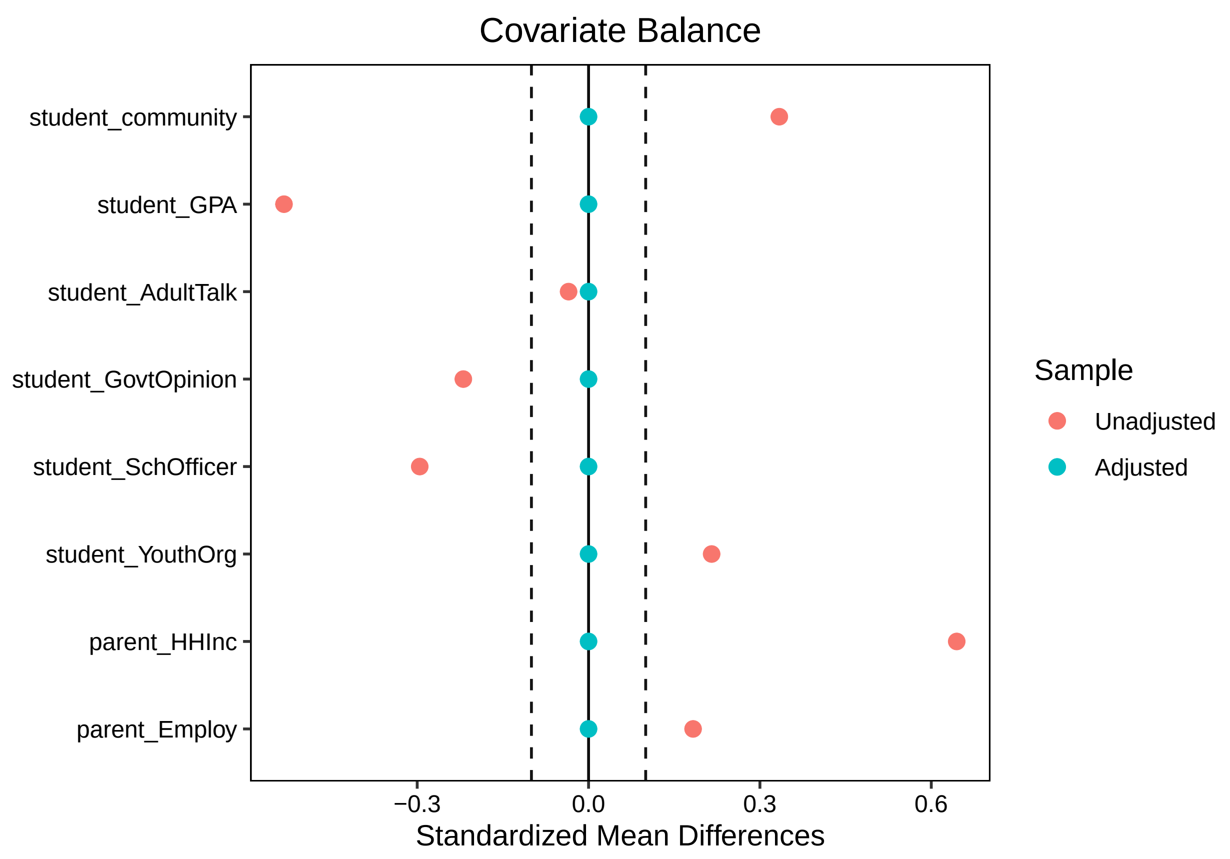
```
bal.plot(match_exact_att, var.name = "parent_HHInc")
```



```
bal.plot(match_exact_att, var.name = "parent_Employ")
```



```
love.plot(match_exact_att, binary = "std", thresholds = c(m = .1))
```



The propensity scores between the treated and controlled groups are very well balanced. Student GPA is the only case where we can't say the groups are exchangeable because of the unmatched units at the right-end of the distribution. All adjusted covariates are balanced when using a threshold of standardized mean difference of propensity scores with less than 0.1.

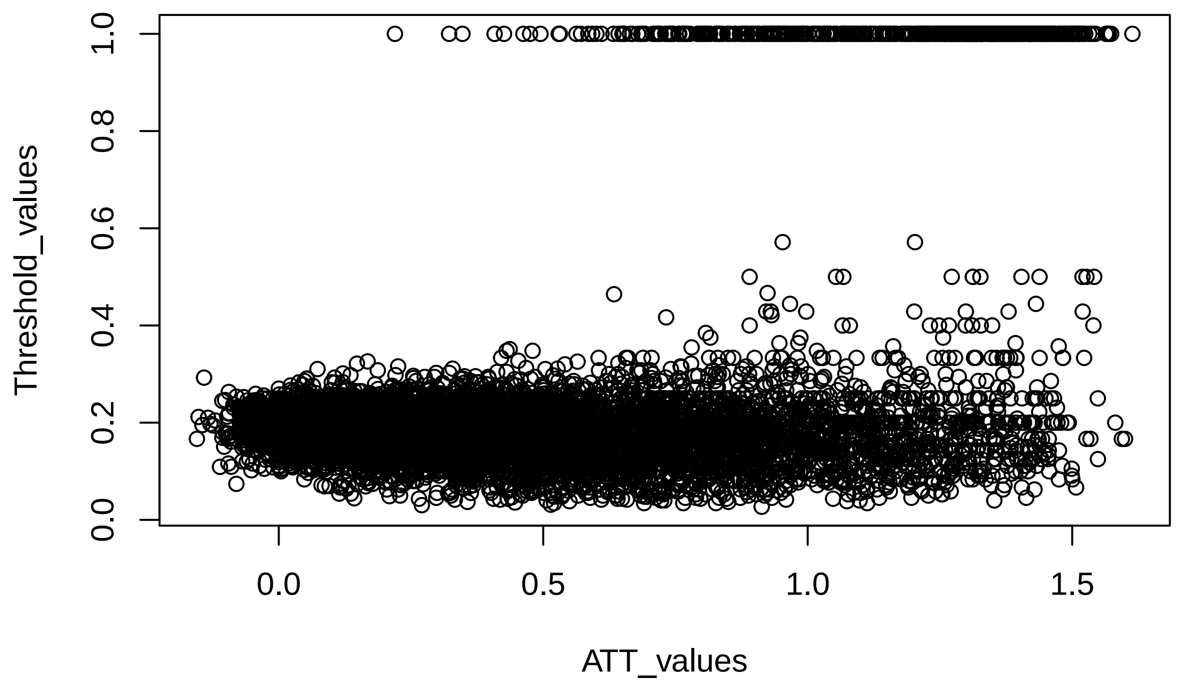
Simulations

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 less than or equal to .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.

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).

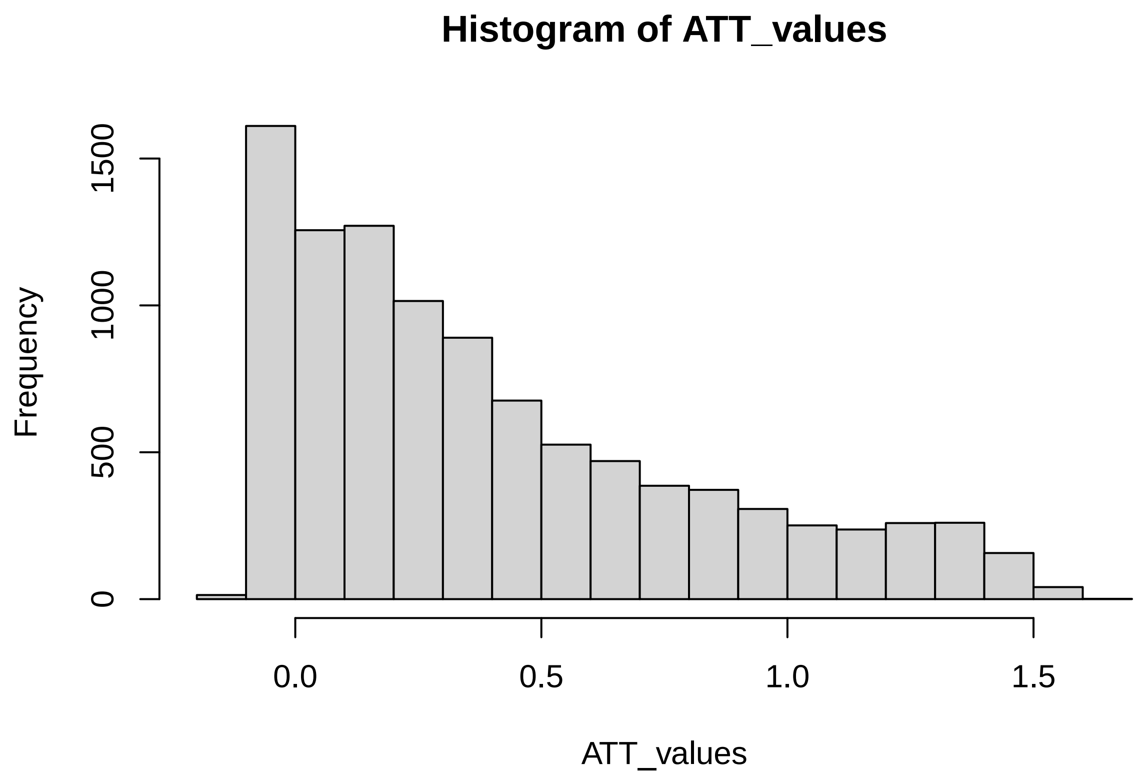
```
# Remove post-treatment covariates
ypsps_pre <- ypsps %>% dplyr::select(-starts_with("student_1973")) %>%
  dplyr::select(-starts_with("student_1982"), -interviewid, -college, -student_ppnscal)

#To account for the following error message below, I will remove the variables as I test them in the fo
#Error: Missing and non-finite values are not allowed in the covariates. Covariates with missingness or
ypsps_pre <- ypsps_pre %>% dplyr::select(-parent_HHCollegePlacebo, -parent_GPHighSchoolPlacebo)

# Simulate random selection of features 10k+ times
# ATT_values <- c()
# Threshold_values <- c()
# Improvement <- c()
# for (i in 1:10000) {
#   # Randomly select features
#   vector_locate <- sample(ncol(ypsps_pre), size=sample(ncol(ypsps_pre), size=1))
#   ypspsp_pre_new <- ypsps_pre[,vector_locate]
#   ypspsp_pre_new$student_ppnscal <- ypsps$student_ppnscal
#   ypspsp_pre_new$college <- ypsps$college
#   formula_c <- paste("college ~ ", paste(colnames(ypspsp_pre_new)[1:(length(ypspsp_pre_new)-2)], coll
#   match_model <- matchit(formula = as.formula(formula_c), data = ypspsp_pre_new, method = "nearest",
#                              link = "logit", discard = "control", replace = FALSE, ratio = 2, estimand =
#   match_model_data <- match.data(match_model)
#   formula_b <- paste("student_ppnscal ~ ", paste(colnames(match_model_data)[1:(length(match_model_dat
#   lm_att <- lm(as.formula(paste(formula_b, "+ college")), data = match_model_data, weights = weights)
#   lm_summary <- summary(lm_att)
#   lm_ATT <- lm_summary$coefficients["college", "Estimate"]
#   bal_summary <- bal.tab(match_model, thresholds = c(m = .1))
#   bal_summary$Balance$M.Threshold
#   table_vals <- table(bal_summary$Balance$M.Threshold[2:length(bal_summary$Balance$M.Threshold)])
#   threshold_value <- as.vector(prop.table(table_vals))[1]
#
#   # Fit p-score models and save ATTs, proportion of balanced covariates, and mean percent balance imp
#   ATT_values <- c(ATT_values, lm_ATT)
#   Threshold_values <- c(Threshold_values, threshold_value)
#   if (i==1) {
#     Improvement <- c(Improvement, threshold_value)
#   }
#   else {
#     new_improvement <- (threshold_value - mean(Threshold_values[1:(i-1)])) / abs(mean(Threshold_value
#     Improvement <- c(Improvement, new_improvement)
#     }
# }
# saveRDS(ATT_values, file = "ATT_values.Rds")
# saveRDS(Threshold_values, file = "Threshold_values.Rds")
# saveRDS(Improvement, file = "Improvement.Rds")
ATT_values <- readRDS("data/ATT_values.Rds")
Threshold_values <- readRDS("data/Threshold_values.Rds")
Improvement <- readRDS("data/Improvement.Rds")

# Plot ATT v. proportion
plot(ATT_values, Threshold_values)
```



```
# Number of models with threshold proportion values that were 1.0
as.data.frame(Threshold_values) %>% dplyr::filter(Threshold_values==1) %>% nrow()
```

```
## [1] 595
```

```
hist(ATT_values)
```



Questions

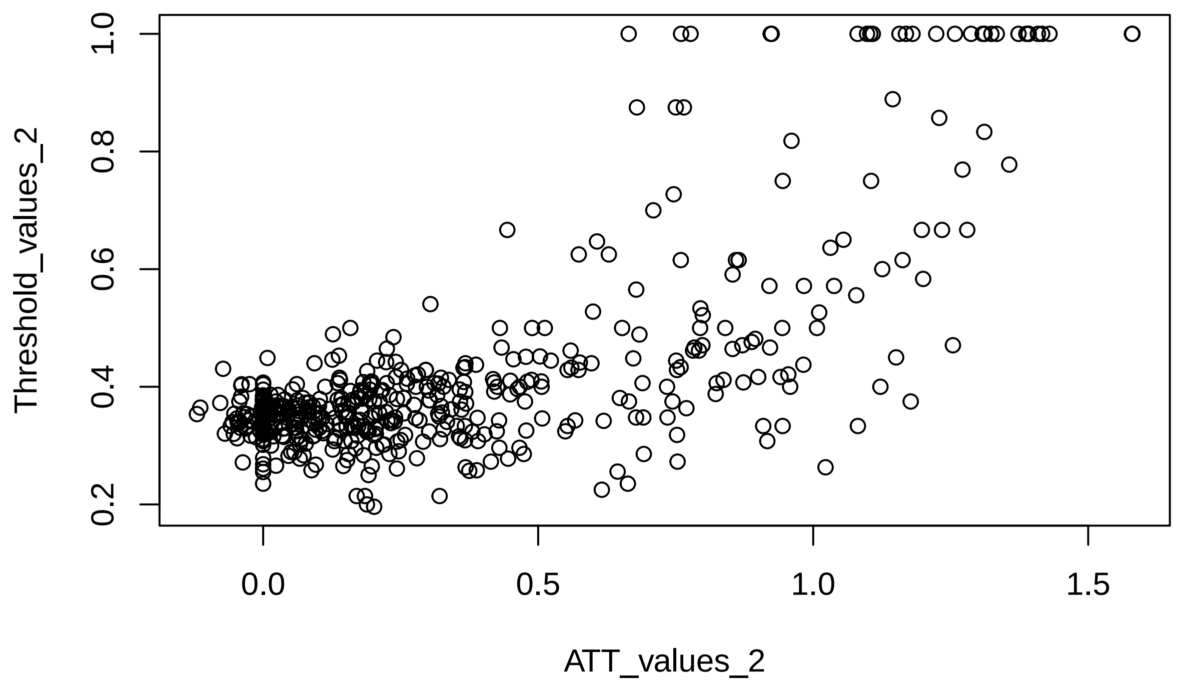
1. **How many simulations resulted in models with a higher proportion of balanced covariates? Do you have any concerns about this?**
2. **Out of 10,000 simulations, 595 models had a perfection total proportion of balanced covariates. This is also seen by the plot. I do have concerns about this because most of the models did not have balanced covariates, hence the moment when I did have exact matching could have just been by luck of which covariates I selected.**:
3. **Analyze the distribution of the ATTs. Do you have any concerns about this distribution?**
4. **The distribution of ATTS is right skewed. I am not concerned about this distribution because the result remains clear that college does increase the increased political activities of people at older ages, even with a mix of different covariates. This is a good sign.**

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:

```
#Full Optimal Mahalanobis distance Matching

# Remove post-treatment covariates
ypsps_pre <- ypsps %>% dplyr::select(-starts_with("student_1973")) %>%
  dplyr::select(-starts_with("student_1982"), -interviewid, -college, -student_ppnscal)

#To account for the following error message below, I will remove the variables as I test them in the fo
#Error: Missing and non-finite values are not allowed in the covariates. Covariates with missingness or
ypsps_pre <- ypsps_pre %>% dplyr::select(-parent_HHCollegePlacebo, -parent_GPHighSchoolPlacebo)

# Simulate random selection of features 507 times
# ATT_values_2 <- c()
# Threshold_values_2 <- c()
# Improvement_2 <- c()
# for (i in 1:507) {
#   # Randomly select features
#   vector_locate <- sample(ncol(ypsps_pre), size=sample(ncol(ypsps_pre), size=1))
#   ypspsp_pre_new <- ypsps_pre[,vector_locate]
#   ypspsp_pre_new$student_ppnscal <- ypsps$student_ppnscal
#   ypspsp_pre_new$college <- ypsps$college
#   formula_c <- paste("college ~ ", paste(colnames(ypspsp_pre_new)[1:(length(ypspsp_pre_new)-2)], coll
#   match_model <- matchit(formula = as.formula(formula_c), data = ypspsp_pre_new, method = "full", dis
#   match_model_data <- match.data(match_model)
#   formula_b <- paste("student_ppnscal ~ ", paste(colnames(match_model_data)[1:(length(match_model_dat
#   lm_att <- lm(as.formula(paste(formula_b, "+ college")), data = match_model_data, weights = weights)
#   lm_summary <- summary(lm_att)
#   lm_ATT <- lm_summary$coefficients["college", "Estimate"]
#   bal_summary <- bal.tab(match_model, thresholds = c(m = .1))
#   bal_summary$Balance$M.Threshold
#   table_vals <- table(bal_summary$Balance$M.Threshold[2:length(bal_summary$Balance$M.Threshold)])
#   threshold_value <- as.vector(prop.table(table_vals))[1]
#
#   # Fit p-score models and save ATTs, proportion of balanced covariates, and mean percent balance imp
#   ATT_values_2 <- c(ATT_values_2, lm_ATT)
#   Threshold_values_2 <- c(Threshold_values_2, threshold_value)
#   if (i==1) {
#     Improvement_2 <- c(Improvement_2, threshold_value)
#   }
#   else {
#     new_improvement <- (threshold_value - mean(Threshold_values_2[1:(i-1)])) / abs(mean(Threshold_val
#     Improvement_2 <- c(Improvement_2, new_improvement)
#     }
# }
# saveRDS(ATT_values_2, file = "ATT_values_2.Rds")
# saveRDS(Threshold_values_2, file = "Threshold_values_2.Rds")
# saveRDS(Improvement_2, file = "Improvement_2.Rds")
ATT_values_2 <- readRDS("data/ATT_values_2.Rds")
Threshold_values_2 <- readRDS("data/Threshold_values_2.Rds")
Improvement_2 <- readRDS("data/Improvement_2.Rds")

# Plot ATT v. proportion
plot(ATT_values_2, Threshold_values_2)
```



Questions

1. **Does your alternative matching method have more runs with higher proportions of balanced covariates?**
2. **Because of time issues, I decided to stop the for-loop after 10 minutes waiting and obtained only 507 observations that were saved in my vectors of interest. Here is the same distribution from the earlier analysis. There seems to be a more linear relationship this time, compared to before. I cannot definitively say, however, that there are more runs with higher proportions of balanced covariates.**

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. **Matching is necessary when working with observational data to ensure independence between the exposure variable and the covariates. A randomized design cannot be sufficient here since it is likely randomization occurred only conditional on observable characteristics.**

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. **I do not advocate for or disagree on the use of other machine learning algorithms. The curse of dimensionality is definitely real when working with these type of data, however I personally prefer using simpler language over complexity to explain my results. That being said, as the popularization of these tools become more mainstream in the social sciences, my opinion might change.**