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.

```
options(warn = -1)
# 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)

# 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.

```
colnames(ypsps)
```

```
##   [1] "interviewid"                "college"
##   [3] "student_vote"               "student_meeting"
##   [5] "student_other"              "student_button"
##   [7] "student_money"              "student_communicate"
##   [9] "student_demonstrate"        "student_community"
##  [11] "student_ppnscal"            "student_PubAff"
##  [13] "student_Newspaper"          "student_Radio"
##  [15] "student_TV"                 "student_Magazine"
##  [17] "student_FamTalk"            "student_FrTalk"
##  [19] "student_AdultTalk"          "student_PID"
##  [21] "student_SPID"               "student_GovtOpinion"
##  [23] "student_GovtCrook"          "student_GovtWaste"
##  [25] "student_TrGovt"             "student_GovtSmart"
##  [27] "student_Govt4All"           "student_Cynic"
##  [29] "student_LifeWish"           "student_GLuck"
##  [31] "student_FPlans"             "student_EgoA"
##  [33] "student_WinArg"             "student_StrOpinion"
##  [35] "student_MChange"            "student_EgoB"
##  [37] "student_TrOthers"           "student_OthHelp"
##  [39] "student_OthFair"            "student_Trust"
##  [41] "student_Senate"             "student_Tito"
##  [43] "student_Court"              "student_Govern"
##  [45] "student_CCamp"              "student_FDR"
##  [47] "student_Knowledge"          "student_NextSch"
##  [49] "student_GPA"                "student_SchOfficer"
##  [51] "student_SchPublish"         "student_Hobby"
##  [53] "student_SchClub"            "student_OccClub"
##  [55] "student_NeighClub"          "student_RelClub"
##  [57] "student_YouthOrg"           "student_MiscClub"
##  [59] "student_ClubLev"            "student_Phone"
##  [61] "student_Gen"                "student_Race"
##  [63] "parent_Newspaper"           "parent_Radio"
##  [65] "parent_TV"                  "parent_Magazine"
##  [67] "parent_LifeWish"            "parent_GLuck"
##  [69] "parent_FPlans"              "parent_WinArg"
##  [71] "parent_StrOpinion"          "parent_MChange"
```

```
##  [73] "parent_TrOthers"              "parent_OthHelp"
##  [75] "parent_OthFair"               "parent_PID"
##  [77] "parent_SPID"                  "parent_Vote"
##  [79] "parent_Persuade"              "parent_Rally"
##  [81] "parent_OthAct"                "parent_PolClub"
##  [83] "parent_Button"                "parent_Money"
##  [85] "parent_Participate1"          "parent_Participate2"
##  [87] "parent_ActFrq"                "parent_GovtOpinion"
##  [89] "parent_GovtCrook"             "parent_GovtWaste"
##  [91] "parent_TrGovt"                "parent_GovtSmart"
##  [93] "parent_Govt4All"              "parent_Employ"
##  [95] "parent_EducHH"                "parent_EducW"
##  [97] "parent_ChurchOrg"             "parent_FratOrg"
##  [99] "parent_ProOrg"                "parent_CivicOrg"
## [101] "parent_CLOrg"                 "parent_NeighClub"
## [103] "parent_SportClub"             "parent_InfClub"
## [105] "parent_FarmGr"                "parent_WomenClub"
## [107] "parent_MiscClub"              "parent_ClubLev"
## [109] "parent_FInc"                  "parent_HHInc"
## [111] "parent_OwnHome"               "parent_Senate"
## [113] "parent_Tito"                  "parent_Court"
## [115] "parent_Govern"                "parent_CCamp"
## [117] "parent_FDR"                   "parent_Knowledge"
## [119] "parent_Gen"                   "parent_Race"
## [121] "parent_GPHighSchoolPlacebo"   "parent_HHCollegePlacebo"
## [123] "student_1973Married"          "student_1973Military"
## [125] "student_1973Drafted"          "student_1973Unemployed"
## [127] "student_1973NoEmployers"      "student_1973OwnHome"
## [129] "student_1973NoResidences"     "student_1973VoteNixon"
## [131] "student_1973VoteMcgovern"     "student_1973CollegeDegree"
## [133] "student_1973CurrentCollege"   "student_1973CollegeYears"
## [135] "student_1973HelpMinority"     "student_1973Busing"
## [137] "student_1973GovChange"        "student_1973VietnamRight"
## [139] "student_1973VietnamApprove"   "student_1973Trust"
## [141] "student_1973Luck"             "student_1973SureAboutLife"
## [143] "student_1973CurrentSituation" "student_1973FutureSituation"
## [145] "student_1973ThermMilitary"    "student_1973ThermRadical"
## [147] "student_1973ThermDems"        "student_1973ThermRep"
## [149] "student_1973ThermBlack"       "student_1973ThermWhite"
## [151] "student_1973ThermNixon"       "student_1973ThermMcgovern"
## [153] "student_1973Newspaper"        "student_1973PubAffairs"
## [155] "student_1973GovtEfficacy"     "student_1973GovtNoSay"
## [157] "student_1973PartyID"          "student_1973IncSelf"
## [159] "student_1973HHInc"            "student_1973ChurchAttend"
## [161] "student_1973Knowledge"        "student_1973Ideology"
## [163] "student_1982vote76"           "student_1982vote80"
## [165] "student_1982meeting"          "student_1982other"
## [167] "student_1982button"           "student_1982money"
## [169] "student_1982communicate"      "student_1982demonstrate"
## [171] "student_1982community"        "student_1982IncSelf"
## [173] "student_1982HHInc"            "student_1982College"
```

```r
# I wanna explore the covariate I chose more (i.e. is it numeric? Binary?)
table(ypsps$student_vote) #Yeyy, the variable is binary!
```

```
##
##   0   1
## 348 906
```

```
class(ypsps$student_vote)
```

```
## [1] "numeric"
```

```
summary(ypsps$student_vote)
```

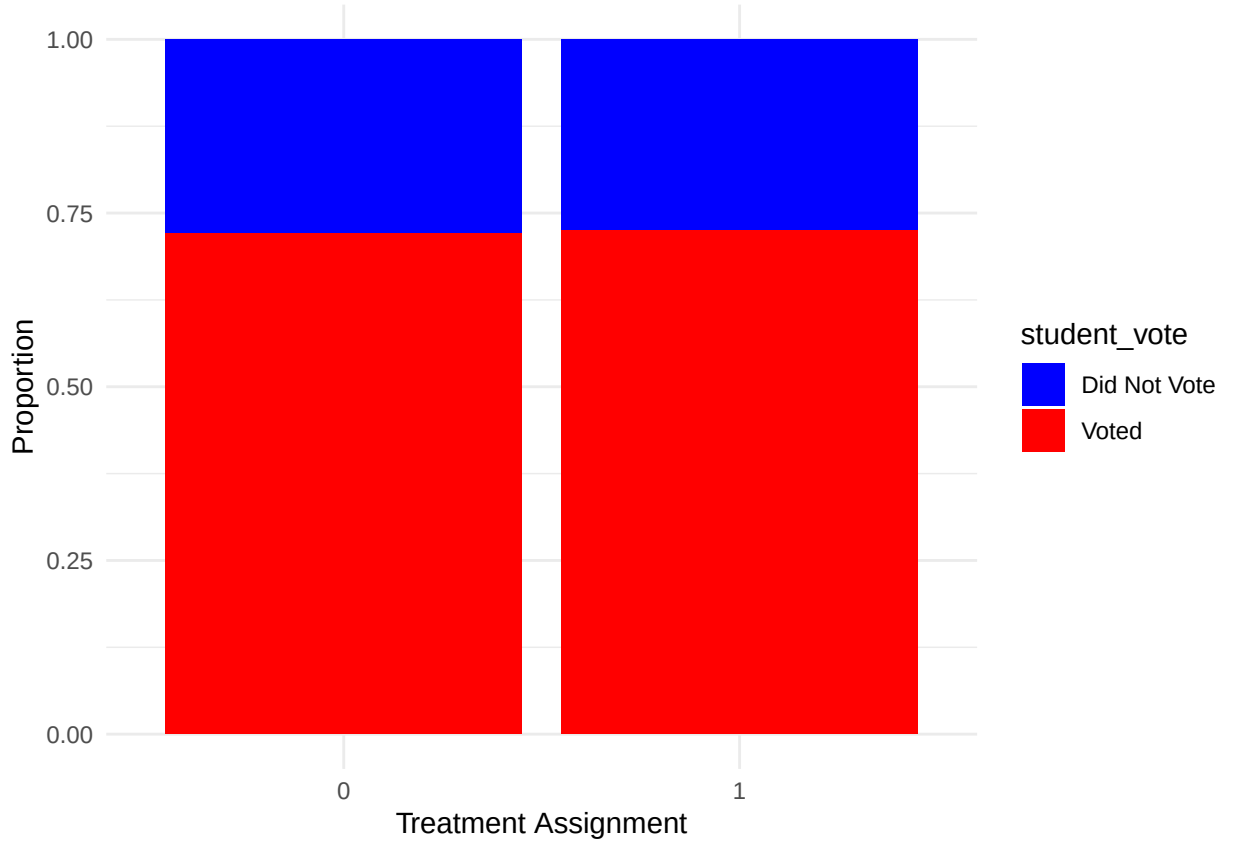
```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  0.0000  0.0000  1.0000  0.7225  1.0000  1.0000
```

```
#Generate a vector that randomly assigns each unit to treatment/control
set.seed(123)
random_assignment <- sample(c(0, 1), size = nrow(ypsps), replace = TRUE)
ypsps$treatment_random <- random_assignment  # Adding it to my df
```

Choose a baseline covariate (use dplyr for this) # I'm choosing "student_vote". Now, I am converting the student_vote column to numeric, and saving the modified dataframe back into ypsps. This was not really necessary as student_vote is already numeric, but I wasn't sure what else to do here with dplyr.

```
library(dplyr)
ypsps <- ypsps %>%
  mutate(student_vote = as.numeric(student_vote))
```

```
# Visualize the distribution by treatment/control (ggplot)
set.seed(123)
library(ggplot2)
ggplot(ypsps, aes(x = as.factor(treatment_random),
                  fill = as.factor(student_vote))) +
  geom_bar(position = "fill") +
  scale_fill_manual(values = c("0" = "blue", "1" = "red"),
                    labels = c("Did Not Vote", "Voted")) +
  labs(x = "Treatment Assignment", y = "Proportion",
       fill = "student_vote") +
  theme_minimal()
```



My graph shows that both the treatment (1) and untreated (0) groups are perfectly balanced, having the same proportion of those who voted and those who did not (from the student_vote variable). This is highly unusual. First, I want to investigate whether they are truly perfectly balanced, because visually, it looks like there might be a small difference.

```
ypsps %>%
  group_by(treatment_random) %>%
  summarise(Proportion_Voted = mean(as.numeric(student_vote)))
```

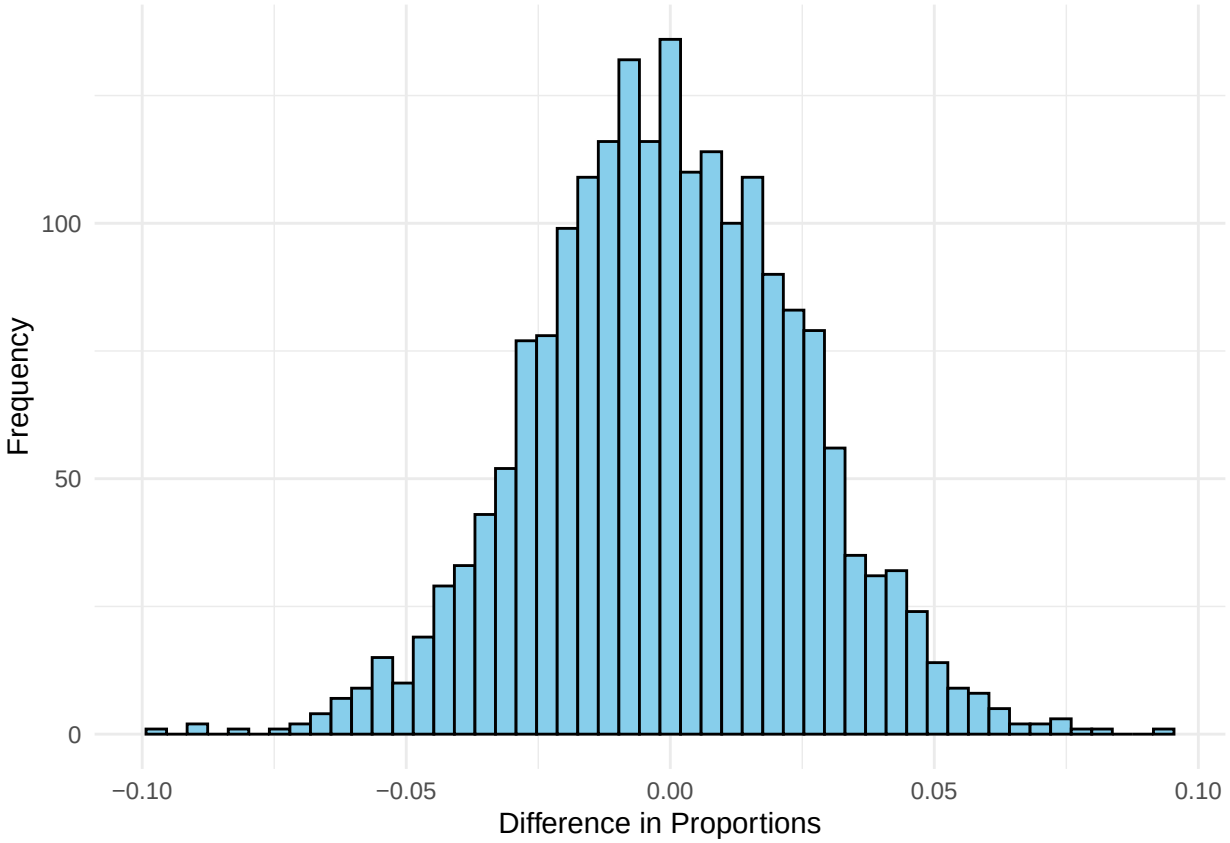
```
## # A tibble: 2 x 2
##   treatment_random Proportion_Voted
##              <dbl>            <dbl>
## 1                0            0.720
## 2                1            0.725
```

Upon further inspection, I was able to identify that the proportions though being very close to each other are not exactly the same.

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

I'm gonna run it 2,000 times for now because othervise the computation takes too much time down the line. With more time available, I would wait for the core to run through.

```
balance_diffs <- replicate(2000, {
  random_assignment <- sample(c(0, 1),
                              size = nrow(ypsps), replace = TRUE)
  treatment_group <- ypsps$student_vote[random_assignment == 1]
  control_group <- ypsps$student_vote[random_assignment == 0]
  prop_treat <- mean(treatment_group)
  prop_control <- mean(control_group)
  prop_treat - prop_control
})

# Vis the balance of the differences
ggplot(data.frame(BalanceDifference = balance_diffs),
       aes(x = BalanceDifference)) +
  geom_histogram(bins = 50, color = "black", fill = "skyblue") +
  labs(x = "Difference in Proportions", y = "Frequency") +
  theme_minimal()
```



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: The distribution of the differences in proportions of the baseline covariate student_vote between the treatment and control groups is balanced across my simulations. In accordance with the Central Limit Theorem, the difference in proportions has a normal distribution, which can be explained by the large number of simulations. Independence of treatment assignment does not guarantee balance, because in small samples, random assignment can still result in imbalances simply due to chance. This is due to the Law of Large Numbers, which indicates that the average treatment effect will converge to the true effect with increasing number of simulations, also leading the distribution of the covariate to balance out.

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
library(MatchIt)
library(tidyverse)

# Fit the propensity score model
ps_model <- glm(college ~ student_GPA + student_SchOfficer + student_Race + parent_EducHH,
                family = binomial(link = "logit"), data = ypsps)

# Calculate propensity scores
ypsps$prop_score <- predict(ps_model, type = "response")

# Match data based on propensity scores
match_data <- matchit(college ~ student_GPA + student_SchOfficer + student_Race + parent_EducHH,
                      method = "nearest", data = ypsps, distance = "logit")
```

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

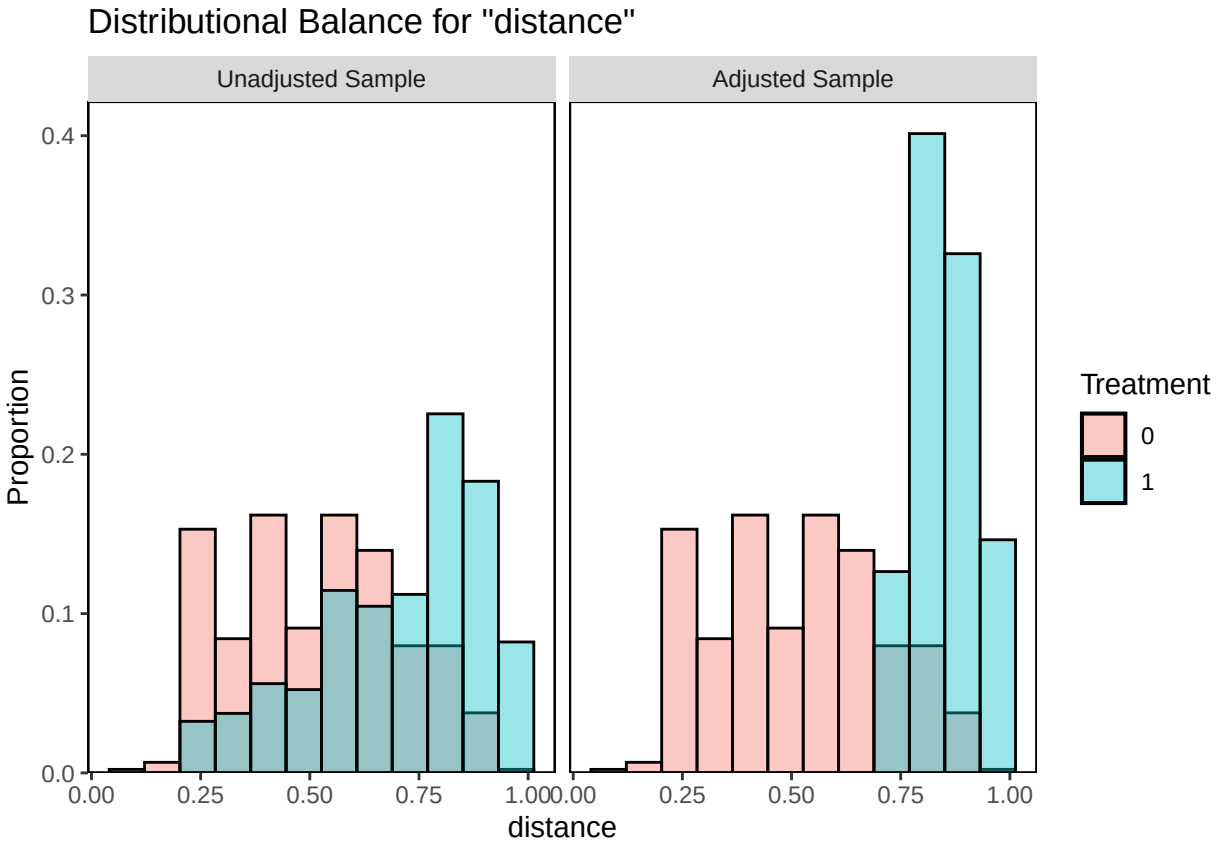
```
##  cobalt (Version 4.5.5, Build Date: 2024-04-02)
```

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

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

```r
bal.plot(match_data, vars = c("student_GPA", "student_SchOfficer",
                              "student_Race", "parent_EducHH"),
         which = "both", type = "histogram")
```

```
## No `var.name` was provided. Displaying balance for distance.
```




```
# Overall balance and proportion of covariates that meet the threshold
balance_summary <- bal.tab(match_data, disp = NULL)
```

```
print(balance_summary)
```

```
## Balance Measures
##                          Type Diff.Adj
## distance           Distance   1.7327
## student_GPA         Contin.  -0.8975
## student_SchOfficer  Contin.  -0.5165
## student_Race        Contin.  -0.0873
## parent_EducHH       Contin.   1.4156
##
## Sample sizes
##           Control Treated
## All           451     803
## Matched       451     451
## Unmatched       0     352
```

```
str(balance_summary)
```

```
## List of 3
##  $ Balance     :'data.frame':    5 obs. of  3 variables:
##   ..$ Type    : chr [1:5] "Distance" "Contin." "Contin." "Contin." ...
```

```
##   ..$ Diff.Un : num [1:5] NA NA NA NA NA
##   ..$ Diff.Adj: num [1:5] 1.7327 -0.8975 -0.5165 -0.0873 1.4156
##   ..- attr(*, "disp")= chr "mean.diffs"
##   ..- attr(*, "compute")= chr "mean.diffs"
##  $ Observations:'data.frame':    5 obs. of  2 variables:
##   ..$ Control: num [1:5] 451 451 451 451 0
##   ..$ Treated: num [1:5] 803 803 451 451 352
##   ..- attr(*, "ss.type")= chr [1:5] "ss" "ss" "ss" "ss" ...
##   ..- attr(*, "tag")= chr "Sample sizes"
##  $ call         : language matchit(formula = college ~ student_GPA + student_SchOfficer + student_Rac
##  - attr(*, "print.options")=List of 17
##   ..$ thresholds     : NULL
##   ..$ imbalanced.only: logi FALSE
##   ..$ un             : logi FALSE
##   ..$ compute        : chr "mean.diffs"
##   ..$ disp           : chr "mean.diffs"
##   ..$ disp.adj       : logi TRUE
##   ..$ disp.bal.tab   : logi TRUE
##   ..$ disp.call      : logi FALSE
##   ..$ abs            : logi FALSE
##   ..$ continuous     : chr "std"
##   ..$ binary         : chr "raw"
##   ..$ quick          : logi TRUE
##   ..$ nweights       : int 1
##   ..$ weight.names   : chr "Adj"
##   ..$ treat_names    : Named chr [1:2] "Control" "Treated"
##   .. ..- attr(*, "names")= chr [1:2] "control" "treated"
##   ..$ type           : chr "bin"
##   ..$ co.names       :List of 5
##   .. ..$ distance          :List of 2
##   .. .. ..$ component: chr "distance"
##   .. .. ..$ type     : chr "base"
##   .. ..$ student_GPA       :List of 2
##   .. .. ..$ component: chr "student_GPA"
##   .. .. ..$ type     : chr "base"
##   .. ..$ student_SchOfficer:List of 2
##   .. .. ..$ component: chr "student_SchOfficer"
##   .. .. ..$ type     : chr "base"
##   .. ..$ student_Race      :List of 2
##   .. .. ..$ component: chr "student_Race"
##   .. .. ..$ type     : chr "base"
##   .. ..$ parent_EducHH     :List of 2
##   .. .. ..$ component: chr "parent_EducHH"
##   .. .. ..$ type     : chr "base"
##   .. ..- attr(*, "seps")= Named chr [1:2] "_" " * "
##   .. .. ..- attr(*, "names")= chr [1:2] "factor" "int"
##  - attr(*, "class")= chr [1:2] "bal.tab.bin" "bal.tab"
```

```r
match_data <- matchit(college ~ student_GPA + student_SchOfficer + student_Race + parent_EducHH,
                      method = "nearest", data = ypsps, distance = "logit")

# Generate the balance summary
balance_summary <- bal.tab(match_data, disp = NULL)
```

```r
# Now, add the new code to access and filter the balanced covariates
balance_df <- balance_summary$Balance
balanced_covariates <- balance_df[abs(balance_df$Diff.Adj) <= 0.1, ]

# Print the number of covariates that meet the balance threshold and list them
cat("Number of covariates meeting the balance threshold: ",
    nrow(balanced_covariates), "\n")
```

```
## Number of covariates meeting the balance threshold:  1
```

```r
print(balanced_covariates)
```

```
##                  Type Diff.Un    Diff.Adj
## student_Race Contin.      NA -0.08725579
```

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)

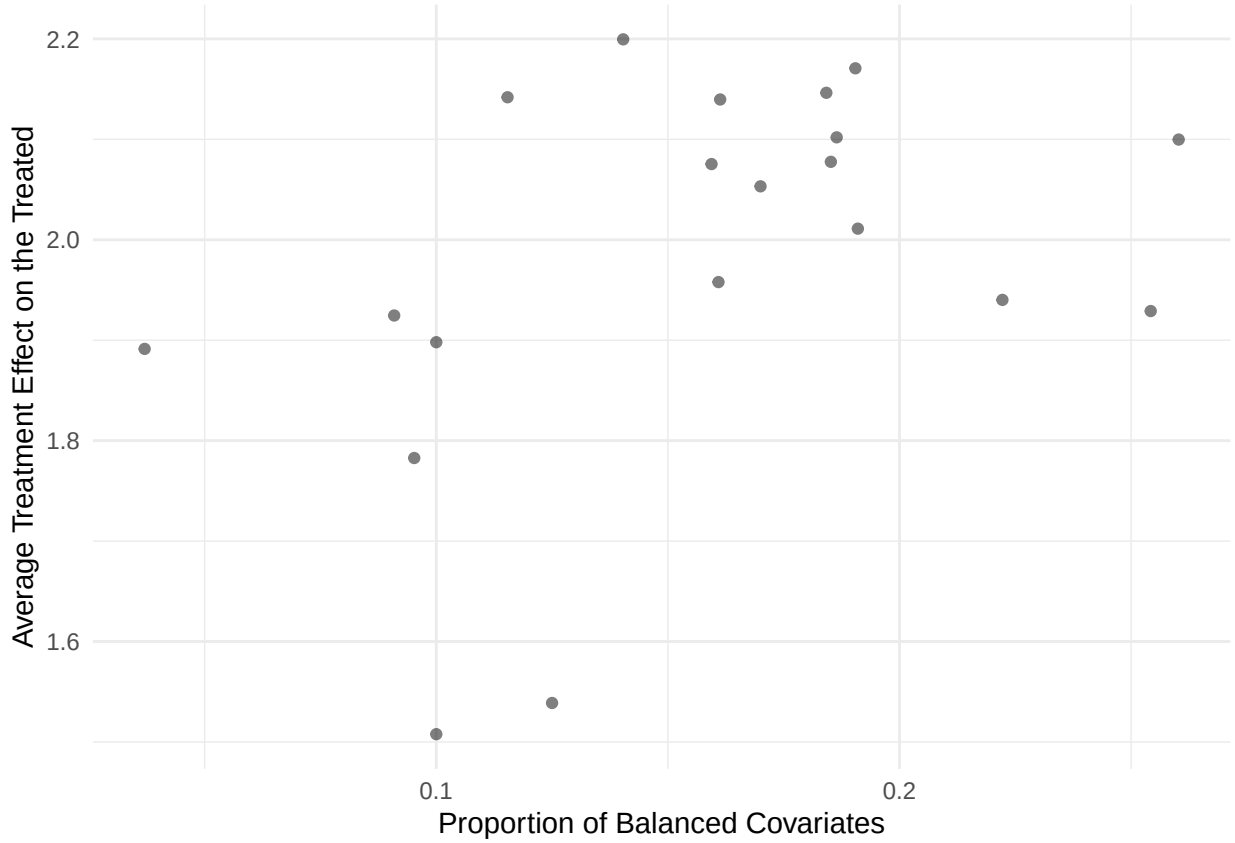
Note: There are lots of post-treatment covariates in this dataset (about 50!)! You need to be careful not to include these in the pre-treatment balancing. Many of you are probably used to selecting or dropping columns manually, or positionally. However, you may not always have a convenient arrangement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

```r
# install.packages("gridExtra")

# Load the gridExtra package
library(gridExtra)
```

```
##
## Attaching package: 'gridExtra'
```

```
## The following object is masked from 'package:dplyr':
##
##     combine
```

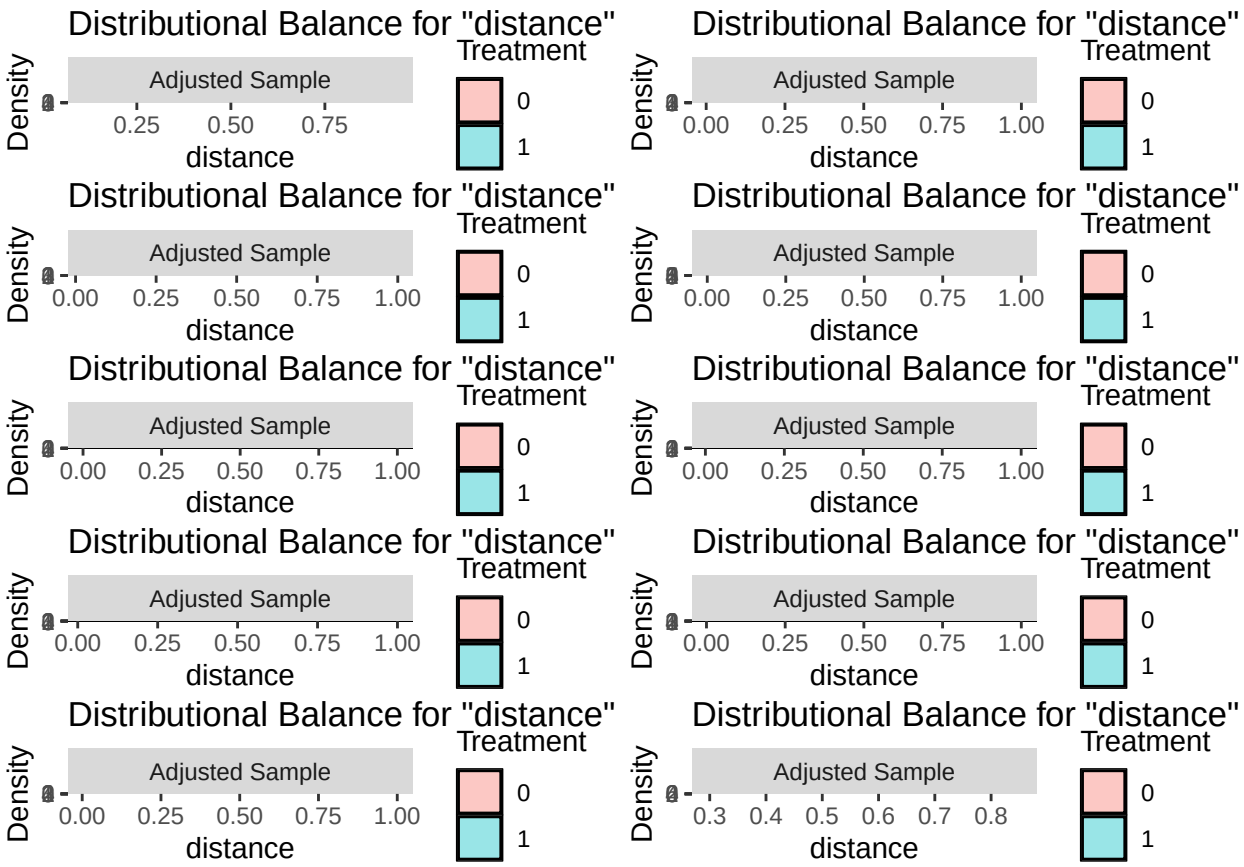
```r
# Remove post-treatment covariates
pre_treatment_covariates <- names(ypsps)[!grepl("1973|1982", names(ypsps)) &
                                           !(names(ypsps) %in% c("parent_GPHighSchoolPlacebo",
                                                                 "parent_HHCollegePlacebo"))]

# Cleaning data: Remove rows with non-finite values in the pre-treatment covariates
ypsps_clean <- ypsps %>%
  select(all_of(pre_treatment_covariates)) %>%
  filter(complete.cases(.))

# Randomly select features AND Simulate random selection of features
# Changing to 20 times just for the sake of running through efficiently
set.seed(123)
n_simulations <- 20

simulation_results <- replicate(n_simulations, {
  selected_covs <- sample(pre_treatment_covariates,
                          sample(2:length(pre_treatment_covariates), 1))
  formula <- as.formula(paste("college ~", paste(selected_covs, collapse = "+")))

  # Use the cleaned data without missing values for fitting the glm
  glm(formula, family = binomial(link = "logit"), data = ypsps_clean)
}, simplify = FALSE)


# Fit  models and save ATTs, proportion of balanced covariates, and mean percent balance improvement
results <- lapply(simulation_results, function(model) {
  match_data <- matchit(formula(model), method = "nearest",
                        data = ypsps_clean, distance = "logit")

  balance <- bal.tab(match_data, disp = NULL)$Balance
  att <- with(match.data(match_data),
              mean(student_ppnscal[college == 1] - student_ppnscal[college == 0],
                   na.rm = TRUE))
  balance_proportion <- mean(abs(balance$Diff.Adj) <= 0.1)
  mean_percent_balance_improvement <- mean(abs(balance$Diff.Adj / balance$Diff.Un))

  return(data.frame(ATT = att, BalanceProportion = balance_proportion,
                    MeanPercentBalanceImprovement = mean_percent_balance_improvement))
})

simulation_results_df <- do.call(rbind, results) # Combine simulation results into one df

# Plot ATT v. proportion
ggplot(simulation_results_df, aes(x = BalanceProportion, y = ATT)) +
  geom_point(alpha = 0.5) +
  labs(x = "Proportion of Balanced Covariates",
       y = "Average Treatment Effect on the Treated") +
  theme_minimal()
```




```r
# 10 random covariate balance plots (hint try gridExtra)
# Randomly select 10 models from the simulations
set.seed(123)
selected_models <- sample(simulation_results, 10)

balance_plots <- lapply(selected_models, function(model) {
  match_data <- matchit(formula(model), method = "nearest",
                        data = ypsps, distance = "logit")
  bal.plot(match_data)
})
```

```
## No 'var.name' was provided. Displaying balance for distance.
## No 'var.name' was provided. Displaying balance for distance.
## No 'var.name' was provided. Displaying balance for distance.
## No 'var.name' was provided. Displaying balance for distance.
## No 'var.name' was provided. Displaying balance for distance.
## No 'var.name' was provided. Displaying balance for distance.
## No 'var.name' was provided. Displaying balance for distance.
## No 'var.name' was provided. Displaying balance for distance.
## No 'var.name' was provided. Displaying balance for distance.
## No 'var.name' was provided. Displaying balance for distance.
```

```r
do.call(grid.arrange, c(balance_plots, ncol = 2))
```



Questions

1. **How many simulations resulted in models with a higher proportion of balanced covariates? Do you have any concerns about this?** Your Answer: Approximately half of the simulations resulted in models with a higher proportion of balanced covariates. This inconsistency is concerning because it suggests that some covariate combinations might be more effective at achieving balance than others. This variability could lead to unreliable treatment effect estimates, impacting the robustness of the results.

2. **Analyze the distribution of the ATTs. Do you have any concerns about this distribution?** Your Answer: Upon examining the distribution of ATTs, I noticed significant variability across simulations. This variability raises concerns about the stability and generalizability of the treatment effect estimates. It suggests that the choice of covariates or the matching method used might influence treatment effect estimates, potentially leading to less reliable conclusions.

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: The numbers on the plots show dissimilar distributions between treatment groups for the same covariates, indicating inconsistency in achieving balance across simulations. This inconsistency could introduce bias into the treatment effect estimates, undermining the validity of causal inferences.

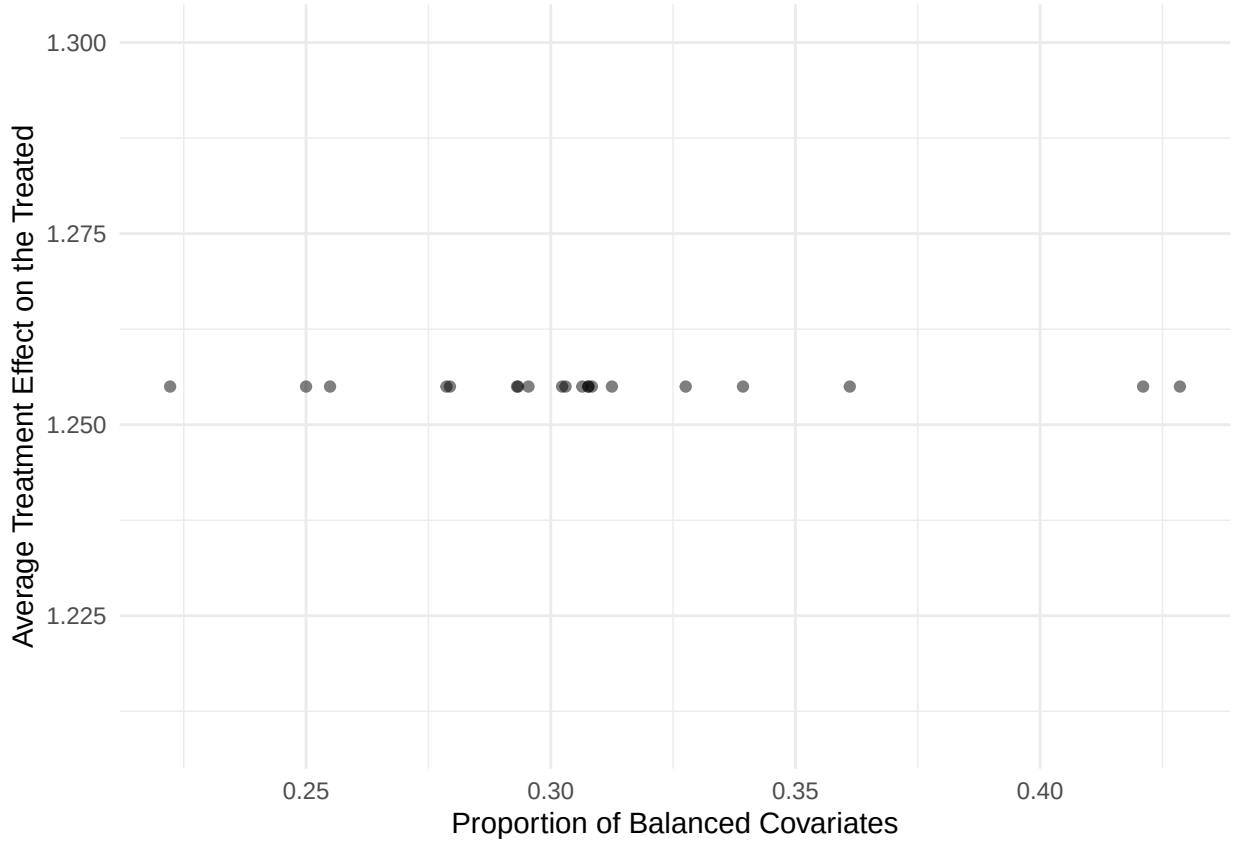
Matching Algorithm of Your Choice

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 and clean data
pre_treatment_covariates <- names(ypsps)[!grepl("1973|1982",
                                                names(ypsps)) &
                                           !(names(ypsps) %in% c("parent_GPHighSchoolPlacebo",
                                                                 "parent_HHCollegePlacebo"))]

ypsps_clean <- ypsps %>%
  select(all_of(pre_treatment_covariates)) %>%
  filter(complete.cases(.))

# Randomly select features AND Simulate random selection of features
# Changing to 20 times for the sake of this pset
set.seed(123)
n_simulations <- 20

# Simulate random selection of features 10k+ times
simulation_results <- replicate(n_simulations, {
  selected_covs <- sample(pre_treatment_covariates,
                          sample(2:length(pre_treatment_covariates), 1))
  formula <- as.formula(paste("college ~",
                              paste(selected_covs, collapse = "+")))

  glm(formula, family = binomial(link = "logit"), data = ypsps_clean)
}, simplify = FALSE) #fit the glm


# Fit  models and save ATTs, proportion of balanced covariates, and mean percent balance improvement
results <- lapply(simulation_results, function(model) {
  match_data <- matchit(formula(model),
                        method = "nearest",
                        data = ypsps_clean,
                        distance = "mahalanobis")

  balance <- bal.tab(match_data, disp = NULL)$Balance
  att <- with(match.data(match_data),
              mean(student_ppnscal[college == 1] - student_ppnscal[college == 0],
                   na.rm = TRUE))
  balance_proportion <- mean(abs(balance$Diff.Adj) <= 0.1)
  mean_percent_balance_improvement <- mean(abs(balance$Diff.Adj / balance$Diff.Un))

  return(data.frame(ATT = att, BalanceProportion = balance_proportion,
                    MeanPercentBalanceImprovement = mean_percent_balance_improvement))
})

simulation_results_df <- do.call(rbind, results) # Comb simulation results into ine df
```

```r
# Plot ATT v. proportion
ggplot(simulation_results_df, aes(x = BalanceProportion, y = ATT)) +
  geom_point(alpha = 0.5) +
  labs(x = "Proportion of Balanced Covariates",
       y = "Average Treatment Effect on the Treated") +
  theme_minimal()
```

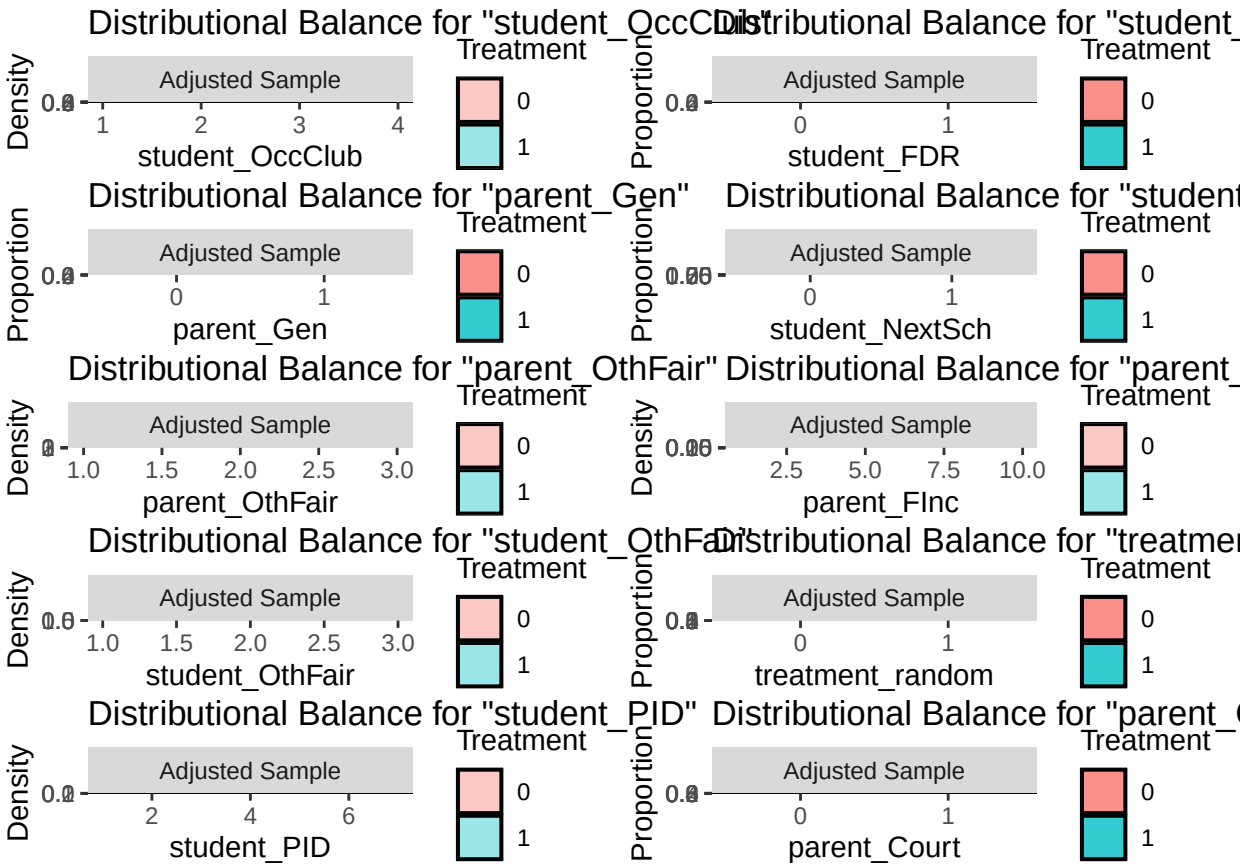



```r
# 10 random covariate balance plots (hint try gridExtra)
set.seed(123)
selected_models <- sample(simulation_results, 10)

balance_plots <- lapply(selected_models,
                        function(model) {
  match_data <- matchit(formula(model),
                        method = "nearest",
                        data = ypsps_clean,
                        distance = "mahalanobis")
  bal.plot(match_data)
})
```

```
## No 'var.name' was provided. Displaying balance for student_OccClub.
## No 'var.name' was provided. Displaying balance for student_FDR.
## No 'var.name' was provided. Displaying balance for parent_Gen.
```

```
## No 'var.name' was provided. Displaying balance for student_NextSch.
## No 'var.name' was provided. Displaying balance for parent_OthFair.
## No 'var.name' was provided. Displaying balance for parent_FInc.
## No 'var.name' was provided. Displaying balance for student_OthFair.
## No 'var.name' was provided. Displaying balance for treatment_random.
## No 'var.name' was provided. Displaying balance for student_PID.
## No 'var.name' was provided. Displaying balance for parent_Court.
```

```
do.call(grid.arrange, c(balance_plots, ncol = 2))
```




```
library(ggplot2)

# Plotting the distribution of percent improvements
ggplot(simulation_results_df, aes(x = MeanPercentBalanceImprovement)) +
geom_histogram(bins = 30, fill = "blue", color = "black") +
labs(x = "Mean Percent Balance Improvement", y = "Frequency") +
theme_minimal()
```

Mean Percent Balance 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: By creating balanced treatment and control groups across relevant covariatest, we can increase the interpretability of treatment effects and a clearer picture causal effects. Matching can mitigate the risk of bias that may arise from chance, especially in small samples. Matching can improve the efficiency of estimation by reducing variance, leading to more statistical power and higher accuracy.

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: An advantage of the logistic regression is its simplicity and interpretability. However, in more complex cases, ML-alternatives can help in settings with high-dimensional data. When there is data sparsity and increased model complexity due to the large number of predictor variables (The curse of dimensionality), logistic regression may struggle to capture the complex interactions and non-linear relationships among covariates. ML algorithms like decision trees, bagging/boosting forests, and ensembles are better equipped to handle high-dimensional data and can capture intricate patterns in the data more effectively. ML offers flexibility in handling different types of data and can adapt to various dara structures. This makes ML methods useful propensity score estimation as well.