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 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.

```
# libraries
xfun::pkg_attach2(c("tidyverse", # load all tidyverse packages
                    "here",      # set file path
                    "MatchIt",   # for matching
                    "optmatch",  # for matching

                    "cobalt"))   # for matching assessment


# chunk options ----------------------------------------------------------------
knitr::opts_chunk$set(
  warning = FALSE            # prevents warning from appearing after code chunk
)

# prevent scientific notation
# ----------
options(scipen = 999)
```

```
here::i_am('Project 6 Template.Rmd') # declare where you are -- "library:function"  allows you to run a
```

```
## here() starts at /Users/jama/Documents/GitHub/Computational-Social-Science-Projects/Project 6
```

```
library(here)                 # loading the library  - don't use require

# install libraries
# ----------
here()                        # setting working directory as the relative file path
```

```
## [1] "/Users/jama/Documents/GitHub/Computational-Social-Science-Projects/Project 6"
```

```
setwd(here())
```

```
# installing libraries
# ----------
library(readr)
```

```r
# load dataa
# ----------
df <- read_csv(here("data/ypsps.csv")) # here() essentially is the working directory
```

```
## 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.
```

```r
head(df)
```

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

```r
names(df)  # shows you the column names
```

```
##   [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"
```

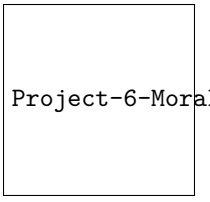
```
## [145] "student_1973ThermMilitary"    "student_1973ThermRadical"
## [147] "student_1973ThermDems"        "student_1973ThermRep"
## [149] "student_1973ThermBlack"       "student_1973ThermWhite"
## [151] "student_1973ThermNixon"       "student_1973ThermMcgovern"
## [153] "student_1973Newspaper"        "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"
```

```
dim(df)
```

```
## [1] 1254  174
```

```
#?read_csv  #check documentation
```

```
set.seed(1234)

# Generate a vector that randomly assigns each unit to treatment/control
n <- nrow(df)

# I will assing a variable 0 for control and 1 for treatment
df$random_treat <- sample(c(0, 1), n, replace = TRUE)
df$random_treat <- as.factor(df$random_treat)

# Choose a baseline covariate (use dplyr for this)
student_Phone_var <- df %>%
  group_by(random_treat) %>%
  summarise(student_Phone_mean = mean(student_Phone))

# Visualize the distribution by treatment/control (ggplot)

#   bar plot
ggplot(student_Phone_var, aes(x = random_treat, y = student_Phone_mean, fill = random_treat)) +
  geom_bar(stat = "identity") +
  theme_minimal() +
  labs(title = "Average Student has Phone by Treatment Status",
       x = "Treatment Status",
       y = "Average Student has Phone (Proportion of Yes)") +
  scale_fill_brewer(palette = "Set1", name = "Treatment Status")
```



```r
#Doing a Chi2 test
table_data <- table(df$student_Phone, df$random_treat)
chi2_result <- chisq.test(table_data)

# and a ttest
t.test( student_Phone~ random_treat, data = df)
```

```
##
##  Welch Two Sample t-test
##
## data:  student_Phone by random_treat
## t = -0.79899, df = 1250.6, p-value = 0.4244
## alternative hypothesis: true difference in means between group 0 and group 1 is not equal to 0
## 95 percent confidence interval:
##  -0.03915929  0.01649385
## sample estimates:
## mean in group 0 mean in group 1
##       0.9266771       0.9380098
```

```r
#No statistical difference, thus no difference in the distribution


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

n_iterations <- 10000

#we will store results
results <- data.frame(iteration = integer(n_iterations),
                      chi2_p_value = numeric(n_iterations),
                      ttest_p_value = numeric(n_iterations),
                      treatment_mean = numeric(n_iterations),
                      control_mean = numeric(n_iterations),
                      mean_differences <- numeric(n_iterations))

for (i in 1:n_iterations) {
# I will assing a variable 0 for control and 1 for treatment
df$random_treat <- sample(c(0, 1), n, replace = TRUE)
df$random_treat <- as.factor(df$random_treat)

# Choose a baseline covariate (use dplyr for this)
student_Phone_var <- df %>%
  group_by(random_treat) %>%
  summarise(student_Phone_mean = mean(student_Phone))

  # Store means
  results$treatment_mean[i] <- student_Phone_var$student_Phone_mean[df$random_treat == 1]
  results$control_mean[i] <- student_Phone_var$student_Phone_mean[df$random_treat == 0]

  # Chi-squared test
  table_data <- table(df$student_Phone, df$random_treat)
  chi2_result <- chisq.test(table_data)
  results$chi2_p_value[i] <- chi2_result$p.value
```

```r
  # T-test
  ttest_result <- t.test(student_Phone ~ random_treat, data = df)
  results$ttest_p_value[i] <- ttest_result$p.value

    # Calculate the difference in means
  treatment_mean <- student_Phone_var$student_Phone_mean[df$random_treat == 1]
  control_mean <- student_Phone_var$student_Phone_mean[df$random_treat == 0]
  mean_differences[i] <- treatment_mean - control_mean


  # iteration number
  results$iteration[i] <- i
}


mean(results$chi2_p_value < 0.05)
```

```
## [1] 0.0369
```

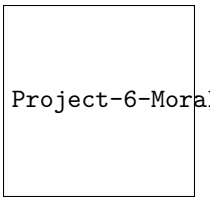
```r
mean(results$ttest_p_value < 0.05)
```

```
## [1] 0.0464
```

```r
# Analyze the balance of the variable
mean_difference_test <- t.test(mean_differences)
mean_difference_test
```

```
##
##  One Sample t-test
##
## data:  mean_differences
## t = 2.245, df = 5040, p-value = 0.02481
## alternative hypothesis: true mean is not equal to 0
## 95 percent confidence interval:
##  0.00005714814 0.00084451486
## sample estimates:
##    mean of x
## 0.0004508315
```

```r
# Visualize the balance of the variable
dist_studphone <- ggplot(data.frame(MeanDifference = mean_differences), aes(x = MeanDifference)) +
  geom_density(fill = "blue", alpha = 0.5) +
  geom_vline(xintercept = mean(mean_differences), linetype = "dashed", color = "red") +
  labs(title = "Balance of Student has Phone Across All Iterations",
       x = "Difference in Means (Treatment - Control)",
       y = "Density") +
  theme_minimal()
dist_studphone
```



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: My graph is showing that the difference of the mean of the variable student having a phone, is centered at 0, which means that it could be that by pure chance we can end up having no difference in most of the times. Also, it shows that there will be some cases where the mean will be higher or lower than zero, following a normal distribution. It also means that even with independence of treatment some cases will be unbalanced, although all the distribution is contained within the threshold of 0.1 It is important to recognize that depending on the underlying distribution of the data, even with randommization there could be inbalances on some covariates.

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.

```
# Select covariates that represent the "true" model for selection, fit model
#variable_names <- names(df)
#df_variable_names <- data.frame(VariableName = variable_names)

select_variables <- c("student_PubAff" ,
"student_Newspaper" ,
"student_Radio" ,
"student_TV" ,
"student_Magazine" ,
"student_FamTalk" ,
"student_FrTalk" ,
"student_AdultTalk" ,
"student_PID" ,
"student_SPID" ,
"student_GovtOpinion" ,
"student_Cynic" ,
"student_StrOpinion" ,
"student_TrOthers" ,
"student_Trust" ,
"student_Senate" ,
```

```
"student_Tito" ,
"student_Court" ,
"student_Govern" ,
"student_CCamp" ,
"student_FDR" ,
"student_Knowledge" ,
"student_NextSch" ,
"student_GPA" ,
"student_SchPublish" ,
"student_Phone" ,
"student_Gen" ,
"student_Race" ,
"parent_Newspaper" ,
"parent_Radio" ,
"parent_TV" ,
"parent_Magazine" ,
"parent_TrOthers" ,
"parent_OthHelp" ,
"parent_OthFair" ,
"parent_PolClub" ,
"parent_Button" ,
"parent_Money" ,
"parent_Participate1" ,
"parent_Participate2" ,
"parent_GovtOpinion" ,
"parent_Employ" ,
"parent_EducHH" ,
"parent_EducW" ,
"parent_CivicOrg" ,
"parent_HHInc" ,
"parent_OwnHome" ,
"parent_Senate" ,
"parent_Tito" ,
"parent_Court" ,
"parent_Govern" ,
"parent_CCamp" ,
"parent_Knowledge" ,
"parent_Gen" ,
"parent_Race"  )

ps_formula <- as.formula(paste("college ~", paste(select_variables, collapse = " + ")))

match_ps_att <-matchit(formula = ps_formula, data = df,  # formula
                         method = "nearest", # method
                         distance = "glm",
                         discard = "control" ,
                         replace = TRUE ,
                        ratio = 2) # estimand
# summary
summary(match_ps_att, un = FALSE )
```

```
##
## Call:
```

```
## matchit(formula = ps_formula, data = df, method = "nearest",
##     distance = "glm", discard = "control", replace = TRUE, ratio = 2)
##
## Summary of Balance for Matched Data:
##                       Means Treated Means Control Std. Mean Diff. Var. Ratio
## distance                     0.7884        0.7846          0.0182     1.0019
## student_PubAff               0.9465        0.9882         -0.1853          .
## student_Newspaper            1.9427        1.9626         -0.0152     1.1294
## student_Radio                2.6401        2.6096          0.0172     0.9461
## student_TV                   2.2167        2.0735          0.1121     1.1330
## student_Magazine             1.6139        1.4458          0.1927     1.1662
## student_FamTalk              1.8431        1.7503          0.1007     1.2845
## student_FrTalk               1.9938        1.9271          0.0704     0.8081
## student_AdultTalk            2.9614        3.1544         -0.1856     0.9614
## student_PID                  3.6314        2.9078          0.3705     1.1805
## student_SPID                 1.7360        1.8381         -0.1057     0.9537
## student_GovtOpinion          1.5965        1.4745          0.1916     0.9482
## student_Cynic                2.7011        2.4191          0.2283     1.1321
## student_StrOpinion           1.7796        1.5710          0.2157     1.1300
## student_TrOthers             1.6413        1.4832          0.1704     1.1420
## student_Trust                3.1283        3.2802         -0.1490     1.0100
## student_Senate               0.5965        0.5660          0.0622          .
## student_Tito                 0.3599        0.2858          0.1544          .
## student_Court                0.4819        0.3269          0.3103          .
## student_Govern               0.9377        0.9433         -0.0232          .
## student_CCamp                0.9278        0.9278          0.0000          .
## student_FDR                  0.7472        0.7098          0.0860          .
## student_Knowledge            0.6752        0.6266          0.2133     1.0010
## student_NextSch              0.9577        0.9477          0.0495          .
## student_GPA                  2.2740        2.1687          0.1607     1.0579
## student_SchPublish           1.7111        1.6955          0.0136     1.0807
## student_Phone                0.9614        0.9533          0.0420          .
## student_Gen                  0.5479        0.4122          0.2727          .
## student_Race                 1.0822        1.2005         -0.3880     0.5500
## parent_Newspaper             3.3674        3.4271         -0.0481     1.1011
## parent_Radio                 2.4770        2.4695          0.0042     1.0005
## parent_TV                    3.2130        3.2435         -0.0250     1.1359
## parent_Magazine              0.6600        0.5081          0.3207          .
## parent_TrOthers              1.4907        1.4078          0.0977     1.1296
## parent_OthHelp               1.5293        1.6476         -0.1375     0.8559
## parent_OthFair               1.3026        1.3456         -0.0621     0.8711
## parent_PolClub               0.0859        0.0729          0.0467          .
## parent_Button                0.3188        0.3263         -0.0160          .
## parent_Money                 0.2877        0.2646          0.0509          .
## parent_Participate1          0.2713        0.2527          0.0644     0.8140
## parent_Participate2          0.2339        0.2193          0.0502     0.8874
## parent_GovtOpinion           1.6887        1.5978          0.1335     1.0182
## parent_Employ                0.6961        0.6270          0.1503          .
## parent_EducHH                3.3238        2.8823          0.3011     1.5108
## parent_EducW                 3.1071        2.7323          0.3092     1.1344
## parent_CivicOrg              1.1968        1.1283          0.1093     1.8017
## parent_HHInc                 7.2864        6.8537          0.2019     1.7380
## parent_OwnHome               0.8443        0.8300          0.0395          .
## parent_Senate                0.3811        0.3188          0.1282          .
```

```
## parent_Tito                0.5455        0.5828         -0.0750          .
## parent_Court               0.2864        0.2391          0.1047          .
## parent_Govern              0.9651        0.9658         -0.0034          .
## parent_CCamp               0.8991        0.8699          0.0972          .
## parent_Knowledge           0.6721        0.6526          0.0891     0.9914
## parent_Gen                 0.4471        0.4458          0.0025          .
## parent_Race                1.0797        1.1993         -0.4021     0.5293
##                      eCDF Mean eCDF Max Std. Pair Dist.
## distance                0.0155   0.1905          0.0277
## student_PubAff          0.0417   0.0417          0.2904
## student_Newspaper       0.0289   0.0822          0.9188
## student_Radio           0.0116   0.0386          1.1015
## student_TV              0.0278   0.1139          0.9965
## student_Magazine        0.0560   0.0996          0.8237
## student_FamTalk         0.0251   0.0623          0.9080
## student_FrTalk          0.0452   0.1376          1.0805
## student_AdultTalk       0.0483   0.1196          1.1038
## student_PID             0.1034   0.1961          1.1714
## student_SPID            0.0255   0.0548          1.1231
## student_GovtOpinion     0.0411   0.1227          1.0403
## student_Cynic           0.0470   0.0978          1.0578
## student_StrOpinion      0.0695   0.1065          0.8969
## student_TrOthers        0.0527   0.0841          0.8749
## student_Trust           0.0380   0.1009          1.0040
## student_Senate          0.0305   0.0305          0.7958
## student_Tito            0.0741   0.0741          0.8108
## student_Court           0.1550   0.1550          0.9658
## student_Govern          0.0056   0.0056          0.4303
## student_CCamp           0.0000   0.0000          0.1283
## student_FDR             0.0374   0.0374          0.8309
## student_Knowledge       0.0416   0.0853          0.9389
## student_NextSch         0.0100   0.0100          0.2845
## student_GPA             0.0248   0.1139          0.9121
## student_SchPublish      0.0213   0.0504          0.9302
## student_Phone           0.0081   0.0081          0.4105
## student_Gen             0.1357   0.1357          1.0284
## student_Race            0.0440   0.1252          0.8535
## parent_Newspaper        0.0120   0.0243          0.7439
## parent_Radio            0.0167   0.0386          1.0483
## parent_TV               0.0168   0.0293          0.8910
## parent_Magazine         0.1519   0.1519          1.0936
## parent_TrOthers         0.0276   0.0461          0.7997
## parent_OthHelp          0.0394   0.0604          0.9521
## parent_OthFair          0.0143   0.0230          0.7771
## parent_PolClub          0.0131   0.0131          0.5088
## parent_Button           0.0075   0.0075          0.9353
## parent_Money            0.0230   0.0230          0.7497
## parent_Participate1     0.0401   0.1432          1.0599
## parent_Participate2     0.0314   0.0922          0.9876
## parent_GovtOpinion      0.0303   0.0672          0.9598
## parent_Employ           0.0691   0.0691          1.0005
## parent_EducHH           0.0808   0.1575          0.9713
## parent_EducW            0.0652   0.1445          1.0570
## parent_CivicOrg         0.0171   0.0361          0.4869
```

```
## parent_HHInc           0.0636   0.2011          0.9541
## parent_OwnHome         0.0143   0.0143          0.7128
## parent_Senate          0.0623   0.0623          0.8795
## parent_Tito            0.0374   0.0374          0.8178
## parent_Court           0.0473   0.0473          0.7906
## parent_Govern          0.0006   0.0006          0.3496
## parent_CCamp           0.0293   0.0293          0.6306
## parent_Knowledge       0.0242   0.0666          0.9796
## parent_Gen             0.0012   0.0012          1.0144
## parent_Race            0.0440   0.1258          0.8586
##
## Sample Sizes:
##               Control Treated
## All            451.       803
## Matched (ESS)   33.56     803
## Matched        257.       803
## Unmatched      160.         0
## Discarded       34.         0
```

```
match_ps_att_data <- match.data(match_ps_att)

att_formula <- as.formula(paste("student_ppnscal ~ college +", paste(select_variables, collapse = " + "))

lm_ps_att <- lm(att_formula, data = match_ps_att_data, weights = weights )

lm_ps_att_summ <- summary(lm_ps_att)
lm_ps_att_summ
```

```
##
## Call:
## lm(formula = att_formula, data = match_ps_att_data, weights = weights)
##
## Weighted Residuals:
##     Min      1Q  Median      3Q     Max
## -4.7659 -1.1380 -0.1298  0.8694  5.8407
##
## Coefficients: (1 not defined because of singularities)
##                     Estimate Std. Error t value         Pr(>|t|)
## (Intercept)         0.033420   0.976124   0.034         0.972694
## college             0.941429   0.131970   7.134 0.00000000000187 ***
## student_PubAff     -0.137572   0.270533  -0.509         0.611199
## student_Newspaper   0.027934   0.046083   0.606         0.544537
## student_Radio      -0.017596   0.032404  -0.543         0.587230
## student_TV         -0.088588   0.047125  -1.880         0.060421 .
## student_Magazine   -0.193223   0.067451  -2.865         0.004262 **
## student_FamTalk    -0.091391   0.067605  -1.352         0.176735
## student_FrTalk      0.013031   0.061496   0.212         0.832226
## student_AdultTalk  -0.158861   0.055168  -2.880         0.004067 **
## student_PID        -0.069609   0.030507  -2.282         0.022714 *
## student_SPID        0.039599   0.059352   0.667         0.504804
## student_GovtOpinion -0.063553   0.091190  -0.697         0.486009
## student_Cynic       0.032349   0.049348   0.656         0.512274
## student_StrOpinion -0.198450   0.060200  -3.297         0.001013 **
## student_TrOthers    0.028580   0.095143   0.300         0.763944
```

```
## student_Trust        0.080314   0.086725   0.926         0.354625
## student_Senate      -0.009795   0.119449  -0.082         0.934664
## student_Tito         0.438983   0.130278   3.370         0.000781 ***
## student_Court       -0.061407   0.118236  -0.519         0.603623
## student_Govern       0.356533   0.236369   1.508         0.131774
## student_CCamp        0.437227   0.218937   1.997         0.046090 *
## student_FDR          0.116801   0.132566   0.881         0.378487
## student_Knowledge          NA         NA      NA               NA
## student_NextSch      0.020597   0.268595   0.077         0.938889
## student_GPA          0.136628   0.093712   1.458         0.145164
## student_SchPublish   0.151124   0.050766   2.977         0.002982 **
## student_Phone       -0.277876   0.281627  -0.987         0.324036
## student_Gen          0.066086   0.120350   0.549         0.583046
## student_Race        -0.483179   0.452459  -1.068         0.285824
## parent_Newspaper     0.036587   0.048520   0.754         0.450986
## parent_Radio         0.014219   0.030923   0.460         0.645740
## parent_TV           -0.026579   0.048850  -0.544         0.586496
## parent_Magazine      0.029523   0.126407   0.234         0.815377
## parent_TrOthers     -0.102924   0.077075  -1.335         0.182055
## parent_OthHelp       0.011281   0.076278   0.148         0.882453
## parent_OthFair       0.078070   0.089567   0.872         0.383615
## parent_PolClub      -0.561543   0.272341  -2.062         0.039473 *
## parent_Button       -0.146721   0.187386  -0.783         0.433819
## parent_Money         0.051216   0.225195   0.227         0.820135
## parent_Participate1  0.473584   0.774408   0.612         0.540978
## parent_Participate2  0.489739   0.918960   0.533         0.594202
## parent_GovtOpinion   0.061920   0.085317   0.726         0.468150
## parent_Employ        0.336735   0.144300   2.334         0.019814 *
## parent_EducHH        0.152206   0.050707   3.002         0.002752 **
## parent_EducW        -0.022911   0.057231  -0.400         0.689008
## parent_CivicOrg      0.103565   0.093550   1.107         0.268538
## parent_HHInc         0.030291   0.032405   0.935         0.350124
## parent_OwnHome      -0.480219   0.153888  -3.121         0.001857 **
## parent_Senate       -0.925871   0.310177  -2.985         0.002905 **
## parent_Tito         -0.705957   0.306284  -2.305         0.021375 *
## parent_Court        -0.693641   0.291781  -2.377         0.017628 *
## parent_Govern       -1.090401   0.433846  -2.513         0.012115 *
## parent_CCamp        -0.549733   0.367592  -1.495         0.135099
## parent_Knowledge     4.543414   1.639504   2.771         0.005688 **
## parent_Gen          -0.287732   0.143938  -1.999         0.045877 *
## parent_Race          1.069616   0.463337   2.309         0.021173 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.71 on 1004 degrees of freedom
## Multiple R-squared:  0.207,  Adjusted R-squared:  0.1636
## F-statistic: 4.766 on 55 and 1004 DF,  p-value: < 0.00000000000000022
```

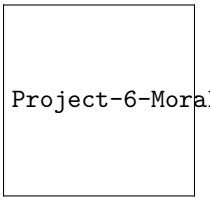
```
ATT_ps <- lm_ps_att_summ$coefficients["college", "Estimate"]
ATT_ps
```

```
## [1] 0.9414291
```

```r
# BALANCE
balance_table <- bal.tab(match_ps_att, un = FALSE)
# Ensure we are accessing the standardized mean differences correctly
effect_sizes <- abs(balance_table$Balance$Diff.Adj)
names(effect_sizes) <- rownames(balance_table$Balance)

# Step 2: Identify the Top 10 Covariates Based on Effect Size
top_10_covariates <- names(sort(effect_sizes, decreasing = TRUE)[1:10])
print(top_10_covariates)
```

```
##  [1] "parent_Race"        "student_Race"       "student_PID"
##  [4] "parent_EducW"       "parent_EducHH"      "student_Cynic"
##  [7] "student_StrOpinion" "student_Knowledge"  "parent_HHInc"
## [10] "student_Magazine"
```

```r
# Step 3: Create a Love Plot for Top 10 Covariates
love.plot(match_ps_att, thresholds = c(m = 0.1), stars="raw", drop.distance = TRUE, var.order="adjusted
```




```r
# Assess the Balance
# Recheck the balance table for the selected top 10 covariates
balance_table_selected <- bal.tab(match_ps_att, vars = top_10_covariates, un = FALSE)
# Checking the number of covariates with SMD  0.1 among the top 10
balanced_covariates <- sum(abs(balance_table_selected$Balance$Diff.Adj) <= 0.1)
balanced_covariates
```

```
## [1] 32
```

Simulations

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

- Using as many simulations as is feasible (at least 10,000 should be ok, more is better!), randomly select the number of and the choice of covariates for the propensity score model.
- For each run, store the ATT, the proportion of covariates that meet the standardized mean difference $\leq .1$ threshold, and the mean percent improvement in the standardized mean difference. You may also wish to store the entire models in a list and extract the relevant attributes as necessary.
- Plot all of the ATTs against all of the balanced covariate proportions. You may randomly sample or use other techniques like transparency if you run into overplotting problems. Alternatively, you may use plots other than scatterplots, so long as you explore the relationship between ATT and the proportion of covariates that meet the balance threshold.

- Finally choose 10 random models and plot their covariate balance plots (you may want to use a library like gridExtra to arrange these)

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

```
# Remove post-treatment covariates
# I will also remove interview id and rows that have missing variables
baseline_vars <- colnames(ypsps)[!grepl("1973", colnames(ypsps)) &
                                 !grepl("1982", colnames(ypsps)) &
                                 colnames(ypsps) != "interviewid" &
                                 colnames(ypsps) != "student_ppnscal"]

# Remove rows with missing values in the selected columns
ypsps_clean <- ypsps[complete.cases(ypsps[, baseline_vars]), ]


# Print the filtered list of variable names
print(baseline_vars)
```

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

```
##  [59] "student_Gen"                "student_Race"
##  [61] "parent_Newspaper"           "parent_Radio"
##  [63] "parent_TV"                  "parent_Magazine"
##  [65] "parent_LifeWish"            "parent_GLuck"
##  [67] "parent_FPlans"              "parent_WinArg"
##  [69] "parent_StrOpinion"          "parent_MChange"
##  [71] "parent_TrOthers"            "parent_OthHelp"
##  [73] "parent_OthFair"             "parent_PID"
##  [75] "parent_SPID"                "parent_Vote"
##  [77] "parent_Persuade"            "parent_Rally"
##  [79] "parent_OthAct"              "parent_PolClub"
##  [81] "parent_Button"              "parent_Money"
##  [83] "parent_Participate1"        "parent_Participate2"
##  [85] "parent_ActFrq"              "parent_GovtOpinion"
##  [87] "parent_GovtCrook"           "parent_GovtWaste"
##  [89] "parent_TrGovt"              "parent_GovtSmart"
##  [91] "parent_Govt4All"            "parent_Employ"
##  [93] "parent_EducHH"              "parent_EducW"
##  [95] "parent_ChurchOrg"           "parent_FratOrg"
##  [97] "parent_ProOrg"              "parent_CivicOrg"
##  [99] "parent_CLOrg"               "parent_NeighClub"
## [101] "parent_SportClub"           "parent_InfClub"
## [103] "parent_FarmGr"              "parent_WomenClub"
## [105] "parent_MiscClub"            "parent_ClubLev"
## [107] "parent_FInc"                "parent_HHInc"
## [109] "parent_OwnHome"             "parent_Senate"
## [111] "parent_Tito"                "parent_Court"
## [113] "parent_Govern"              "parent_CCamp"
## [115] "parent_FDR"                 "parent_Knowledge"
## [117] "parent_Gen"                 "parent_Race"
## [119] "parent_GPHighSchoolPlacebo" "parent_HHCollegePlacebo"
```

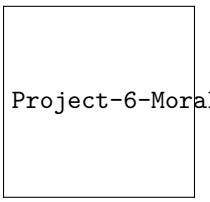
```r
# Randomly select features
# I tried with more iterations, but after a while having to rerun the code as I was debugging I had to
set.seed(123)
n_iterations <- 100
results <- vector("list", n_iterations)

data_simulation <- data.frame(run_id = integer(),
                              n_covar = integer(),
                              ATT = double(),
                              pct_balanced = double(),
                              mean_pct_impr=double())



for (i in 1:n_iterations) {
  # Randomly select a number of variables from baseline_vars
  n_vars <- sample(1:length(baseline_vars), 1)
  selected_vars <- sample(baseline_vars, n_vars)

  # Create and perform matching using selected_vars
  formula_str <- paste("college ~", paste(selected_vars, collapse = " + "))
  formula <- as.formula(formula_str)
```

```
  matchit_res <- matchit(formula, data = ypsps_clean, method = "nearest" ,
                         distance = "glm",# use glm, which by default is logistic regression
                         link = "logit",# specify we want a logit model, default when distance is speci
                         estimand="ATT",
                         discard = "control",# obs to be discarded that are outside region of common su
                         replace = TRUE,# whether matching should be done with replacement
                         ratio = 2)

  # Estimate ATT using linear regression on the matched data
  matched_data <- match.data(matchit_res)
  lm_formula <- as.formula(paste("student_ppnscal ~ college +", paste(selected_vars, collapse = " + ")))
  att_model <- lm(lm_formula, data = matched_data, weights=weights)
  att <- coef(summary(att_model))["college", "Estimate"]


  # Calculate the proportion of covariates with SMD  0.1 and mean percent improvement in SMD
  balance<- bal.tab(matchit_res , binary="std", m.threshold=0.1 )
  df_balance <- as.data.frame(balance$Balance$M.Threshold)

  n_total <- nrow(df_balance)
  n_balanced <- length(df_balance[which(balance$Balance$M.Threshold == "Balanced, <0.1"),])
  pct_balanced <- (n_balanced/n_total)

   #Calculate the mean of the improvement
matchit_summary <- summary(matchit_res, improvement = TRUE)

# Extract the reduction in standardized mean differences
improvement_data <- matchit_summary$reduction

smd_reduction <- improvement_data[,"Std. Mean Diff."]

# Calculate mean percent
mean_percent_improvement <- mean(smd_reduction, na.rm = TRUE)



  # put all this into a dataframe
  data_sim <- data.frame(run_id = i,
                         n_covar = n_vars,
                         ATT = att,
                         pct_balanced = pct_balanced,
                        mean_pct_impr=mean_percent_improvement)

  data_simulation <- rbind(data_simulation, data_sim)

 results[[i]] <- list(
    formula = formula_str,
    ATT = att,
    mean_improvement_smd = mean_percent_improvement,
    model = matchit_res
  )
}
```

```r
pct_balanced_sim_mean <- mean(data_simulation$pct_balanced)
pct_balanced_sim_mean
```

```
## [1] 0.4017319
```

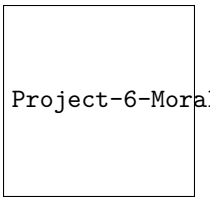
```r
# Simulate random selection of features 10k+ times

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

# Plot ATT v. proportion

ggplot(data_simulation, aes(x =pct_balanced, y = ATT)) +
  geom_point() +  # Add points
  theme_minimal() +  # Use a minimal theme for aesthetics
  labs(x = "Percentage Balanced", y = "ATT",
       title = "Scatter Plot of ATT vs. Percentage Balanced")
```




```r
# 10 random covariate balance plots (hint try gridExtra)


set.seed(456)
selected_indices <- sample(length(results), 10)
selected_models <- results[selected_indices]

library(gridExtra)
```

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

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

```r
plot_list <- lapply(selected_models, function(model) {
  # Extract the matchit object
  matchit_obj <- model$model

  # Create the Love plot
  p <- love.plot(matchit_obj, print = FALSE, drop.distance = TRUE, thresholds = c(m = 0.1))
  return(p)
})

# Arrange the plots in a grid
grid.arrange(grobs = plot_list, ncol = 2)
```




```
# Note: ggplot objects are finnicky so ask for help if you're struggling to automatically create them;
```

Questions

1. **How many simulations resulted in models with a higher proportion of balanced covariates? Do you have any concerns about this?** Your Answer: In the first model we had $32/55 = 58\%$ of balanced covariates, in the simulations it was 40
2. **Analyze the distribution of the ATTs. Do you have any concerns about this distribution?** Your Answer:Yes, the ATT increases as the percentage of balanced covariates increases, which can make the model results very weak to specifications of the matching that lead to different covariate balances.
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:It is not easy to see, but honestly I expect that this might change a lot depending of the inclusion of the variables, and specifications. Again this aligns with the idea that psm is prone to mispecification.

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:

```
#alternative model

# I will also remove interview id and rows that have missing variables
baseline_vars <- colnames(ypsps)[!grepl("1973", colnames(ypsps)) &
                                 !grepl("1982", colnames(ypsps)) &
                                 colnames(ypsps) != "interviewid" &
                                 colnames(ypsps) != "student_ppnscal"]

# Remove rows with missing values in the selected columns
ypsps_clean <- ypsps[complete.cases(ypsps[, baseline_vars]), ]


# Print the filtered list of variable names
#print(baseline_vars)

# I will comment all to avoid running all again .
# Randomly select features
```

```
#set.seed(546)

# It tried running a 1000, it was too much. I stopped the code at 198. Ill do the analysis with that.
#n_iterations <- 198
#data_simulation2 <- data.frame(run_id = integer(),
                        #      n_covar = integer(),
                        #      ATT = double(),
                        #      pct_balanced = double(),
                         #     mean_pct_impr = double())



#for (i in 1:n_iterations) {
  # Randomly select a number of variables from baseline_vars
 # n_vars <- sample(1:length(baseline_vars), 1)
 # selected_vars <- sample(baseline_vars, n_vars)

  # Create and perform matching using selected_vars
  #formula_str <- paste("college ~", paste(selected_vars, collapse = " + "))
  #formula <- as.formula(formula_str)
  #matchit_res2 <- matchit(formula, data = ypsps_clean, method = "genetic" ,
                  #       distance = "glm",# use glm, which by default is logistic regression
                    #     link = "logit",# specify we want a logit model, default when distance is spec
                    #     estimand="ATT",
                    #     discard = "none",# obs to be discarded that are outside region of common supp
                         # whether matching should be done with replacement
                     #    ratio = 1)

  # Estimate ATT using linear regression on the matched data
 # matched_data2 <- match.data(matchit_res2)
 # lm_formula <- as.formula(paste("student_ppnscal ~ college +", paste(selected_vars, collapse = " + "))
  #att_model2 <- lm(lm_formula, data = matched_data2, weights=weights)
  #att2 <- coef(summary(att_model2))["college", "Estimate"]


  # Calculate the proportion of covariates with SMD  0.1 and mean percent improvement in SMD
  #balance2<- bal.tab(matchit_res2 , binary="std", m.threshold=0.1 )
 # df_balance2 <- as.data.frame(balance2$Balance$M.Threshold)

#  n_total2 <- nrow(df_balance2)
 # n_balanced2 <- length(df_balance2[which(balance2$Balance$M.Threshold == "Balanced, <0.1"),])
 # pct_balanced2 <- (n_balanced2/n_total2)


#matchit_summary2 <- summary(matchit_res2, improvement = TRUE)

# Extract the reduction in standardized mean differences
#improvement_data2 <- matchit_summary2$reduction

# Specifically, if you're interested in the reduction of SMDs
#smd_reduction2 <- improvement_data2[,"Std. Mean Diff."]

# You can calculate the mean percent improvement if needed
```

```
#mean_percent_improvement2 <- mean(smd_reduction2, na.rm = TRUE)



  # put all this into a dataframe
#  data_sim2 <- data.frame(run_id = i,
 #                          n_covar = n_vars,
 #                          ATT = att2,
 #                          pct_balanced2 = pct_balanced2,
 #                          mean_pct_impr2=mean_percent_improvement2)

#  data_simulation2 <- rbind(data_simulation2, data_sim2)

#}
```

```
# Plot ATT v. proportion

#graph_genetic <- ggplot(data_simulation2, aes(x =pct_balanced2, y = ATT)) +
 # geom_point() +  # Add points
  #theme_minimal() +  # Use a minimal theme for aesthetics
  #labs(x = "Percentage Balanced", y = "ATT",
   #    title = "Scatter Plot of ATT vs. Percentage Balanced Genetic Matching")
#graph_genetic
#ggsave("graph_genetic.png", graph_genetic, width = 6, height = 4)
```

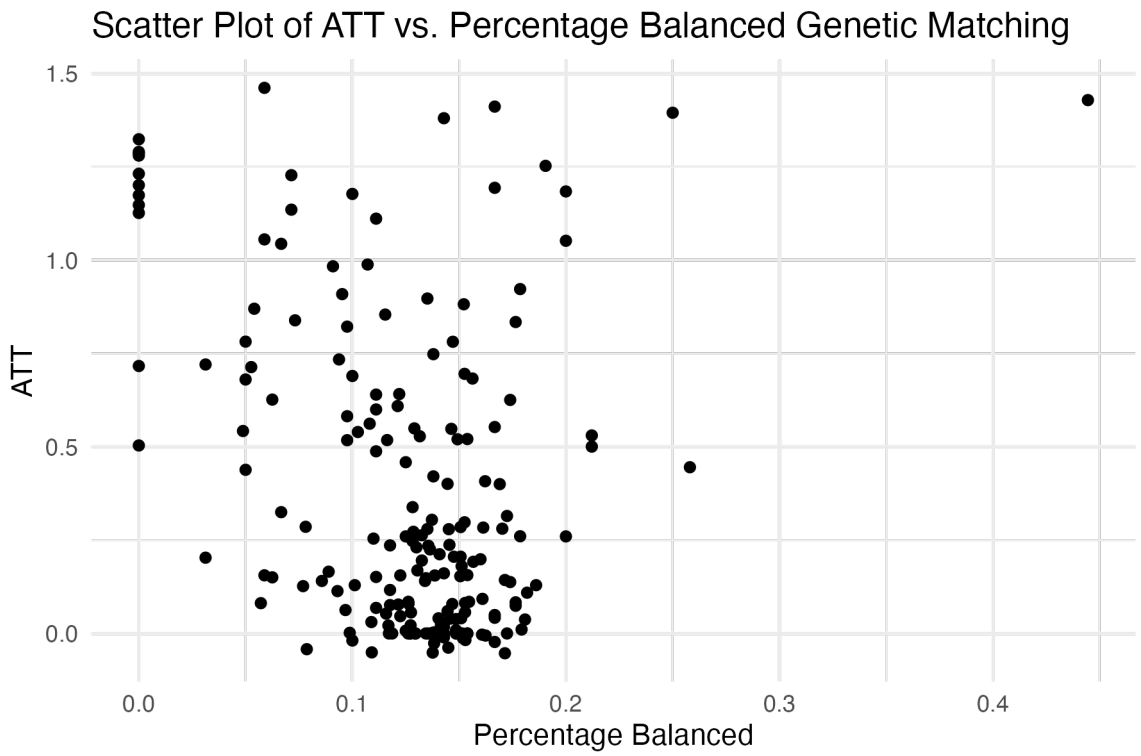


Figure 1: GraphGen

```
# Percentage balanced
#mean_pct_bal_genetic <- mean(data_simulation2$pct_balanced2)
#mean_pct_bal_nn <- mean(data_simulation$pct_balanced)
#mean_pct_bal_genetic
#mean_pct_bal_nn
```

```
# Visualization for distributions of percent improvement

#graph_denisities <- ggplot() +
 # geom_density(data = data_simulation, aes(x = mean_pct_impr, fill = "Propensity Score Model"), alpha
  #geom_density(data = data_simulation2, aes(x = mean_pct_impr2, fill = "Genetic"), alpha = 0.5) +
  #labs(title = "Density Plot of Mean Percentage Improvement by Models",
   #    x = "Mean Percentage Improvement",
    # y = "Density") +
  #scale_fill_manual(values = c("Propensity Score Model" = "blue", "Genetic" = "red")) +
  #theme_minimal()

#ggsave("graph_denisities.png", graph_denisities, width = 6, height = 4)
```

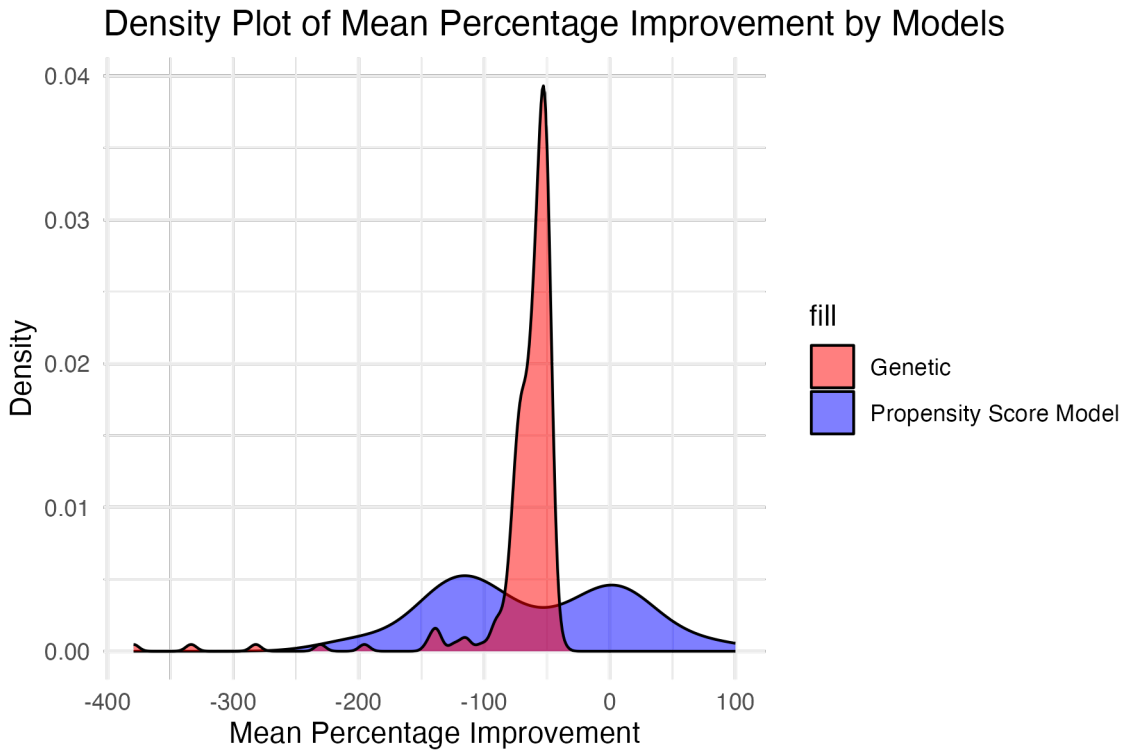


Figure 2: GraphDensities

Questions

1. **Does your alternative matching method have more runs with higher proportions of balanced covariates?** Your Answer:No, when I compared the averages of percentage covars balanced in each case, I found that the percentage is higher (40%) than the genetic case (13%)

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: In the graph above we can see how the genetic one had a very narrowed distribution around -60 more or less, and then the original model, nearest neighbors is sparse with two modes, one in zero and one below -100. This is quite curious as I dont know exactly why. My intepretation is that at least the genetic one was making more improvements than the nn, because nn has many cases in 0 improvement and genetic doesnt.

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: There will be some cases in which the distribution of the covariables in each group might make the group substantially different, maybe in the randomization something happened, or by mere chance the groups were unbalanced. In that case Matching can help to solve any potential bias arising from substantially different distributions of covariables that might have an effect on the outcome. Importantly, this correction uses observable characteristics so it is also prone to biases, and could end up adding more bias.

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:Yes, yes, and yes. That is what is behind TMLE for example. Using ensamble models we can find the best function to approximate the distribution or function of the treatment function, thus improving our chances to get the first stage of a doubly robust estimator correctly estimated. I think models that can reduce dimensions will improve matching.