Project 6: Randomization and Matching

Madeline Adee and Alagia Cirolia

07 April, 2023

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.

Setup: load packages.

```
knitr::opts_chunk$set(cache = TRUE, warning = FALSE,
                      message = FALSE, cache.lazy = FALSE)
# Load tidyverse and MatchIt
# Feel free to load other libraries as you wish

library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.1 --

## v ggplot2 3.4.2     v purrr   0.3.4
## v tibble  3.1.7     v dplyr   1.0.9
## v tidyr   1.2.0     v stringr 1.4.0
## v readr   2.1.2     v forcats 0.5.1

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

```
library(MatchIt)
library(broom)
library(gtsummary)
library(cobalt)
```

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

##
## Attaching package: 'cobalt'

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

```
# turn off scientific notation
options(scipen = 999)
```

Load data.

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

```
## # A tibble: 6 x 174
##   interviewid college student_vote student_meeting student_other student_button
##         <dbl>   <dbl>        <dbl>           <dbl>         <dbl>          <dbl>
## 1           1       1            1               0             0              0
## 2           2       1            1               1             1              1
## 3           3       1            1               0             0              1
## 4           4       0            0               0             0              0
## 5           5       1            1               1             0              0
## 6           6       1            1               0             0              0
## # ... with 168 more variables: student_money <dbl>, student_communicate <dbl>,
## #   student_demonstrate <dbl>, student_community <dbl>, student_ppnscal <dbl>,
## #   student_PubAff <dbl>, student_Newspaper <dbl>, student_Radio <dbl>,
## #   student_TV <dbl>, student_Magazine <dbl>, student_FamTalk <dbl>,
## #   student_FrTalk <dbl>, student_AdultTalk <dbl>, student_PID <dbl>,
## #   student_SPID <dbl>, student_GovtOpinion <dbl>, student_GovtCrook <dbl>,
## #   student_GovtWaste <dbl>, student_TrGovt <dbl>, student_GovtSmart <dbl>, ...
```

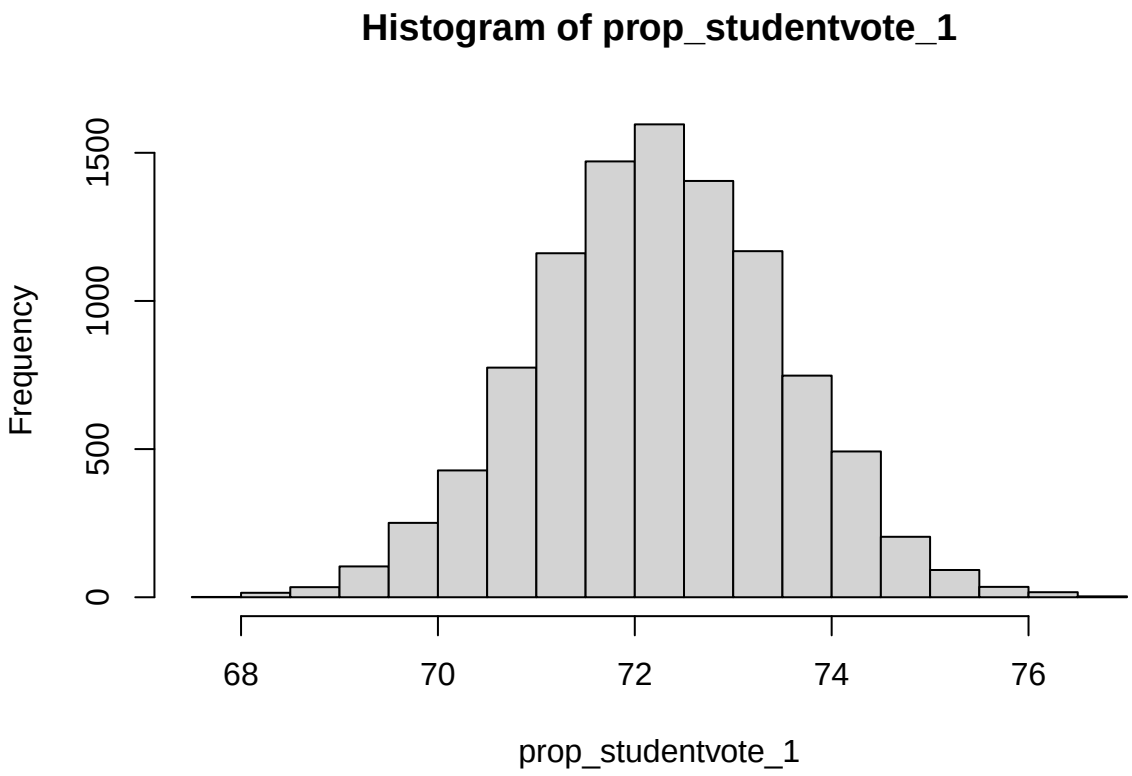
Randomization

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.

Your goal is to visualize what proportion of treated (or control) units have chosen covariate. You can use loop that runs 10,000 times. Initialize empty vector to store proportions outside loop i.e before starting the loop. Every time the loop runs, generate a vector that randomly assigns each unit to either treatment or control. Compute what proportion of the treated (or control) units have the chosen covariate and append the proportion to the vector that you initialized outside the loop.

```
# Generate a vector that randomly assigns each unit to treatment/control
prop_studentvote_1 <- c()

for (x in 1:10000) {
  grps <- 1:2
  ypsps$grps <- sample(grps, 1254, replace = TRUE)
  txt<- ypsps[ypsps$grps == 1,]
  prop <- (sum(txt$student_vote) / length(txt$grps)*100)
  prop_studentvote_1 <- append(prop_studentvote_1, prop)
}

hist(prop_studentvote_1)
```




```
###Normally distributed-- ggplot seems unnecessary for this
```

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: In the full sample, 72.25% of students voted. In the 10000 iterations of random assignment, the proportion of students who voted in the control sample also clusters around 72%. Random assignment did not balance the proportion of students who voted in the assigned groups because the baseline imbalance carries over, and there are likely to be more students who have voted in both control and treatment groups, without some other stratification.

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.

For this covariate selection, we have removed the variables that are not from baseline (from either 1973 or 1982 data collection). We also removed rows that had any missing data.

```
# Select covariates that represent the "true" model for selection, fit model

data_baseline <- ypsps %>%
  # remove all of the variables that are for 1973 and 1982
  # The appendix file they shared doesn't have variable names so I'm not sure how to limit this more
  select(-contains("1973"), -contains("1982"), -interviewid) %>%
  # remove all rows with any missing data
  na.omit()
```

Then, we calculated propensity scores (for likehood of attending college, the exposure in this analysis. We did this using a regression model.

```
# regress with treatment as the outcome, use all other variables to predict
model_ps <- glm(college ~ . - student_ppnscal, family = binomial(), data = data_baseline)
summary(model_ps)
```

```
##
## Call:
## glm(formula = college ~ . - student_ppnscal, family = binomial(),
##     data = data_baseline)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -3.6735  -0.3924   0.1301   0.5236   2.8779
##
## Coefficients:
##                                Estimate       Std. Error z value
## (Intercept)                   -2.659769         6.814212  -0.390
## student_vote                   0.639909         0.253750   2.522
## student_meeting                1.368349         0.391040   3.499
## student_other                 -0.362755         0.428862  -0.846
## student_button                 0.280137         0.290820   0.963
## student_money                 -0.123819         0.359414  -0.345
## student_communicate            0.249682         0.263813   0.946
## student_demonstrate            1.081162         0.429184   2.519
## student_community              0.106701         0.267049   0.400
## student_PubAff                -1.111068         0.629846  -1.764
## student_Newspaper              0.135858         0.095510   1.422
## student_Radio                 -0.053238         0.064931  -0.820
## student_TV                     0.112166         0.104530   1.073
## student_Magazine              -0.340601         0.128553  -2.649
## student_FamTalk                0.084168         0.135715   0.620
## student_FrTalk                -0.274001         0.120169  -2.280
## student_AdultTalk              0.094543         0.112421   0.841
## student_PID                    0.186901         0.075220   2.485
## student_SPID                  -0.135499         0.125805  -1.077
## student_GovtOpinion           -0.247896         0.187135  -1.325
## student_GovtCrook             -0.423000         0.196563  -2.152
## student_GovtWaste             -0.393257         0.199821  -1.968
## student_TrGovt                -0.115334         0.298566  -0.386
```

```
## student_GovtSmart                0.069421         0.196866   0.353
## student_Govt4All                 0.048704         0.193845   0.251
## student_Cynic                   -0.234458         0.182172  -1.287
## student_LifeWish                 0.197204         0.204473   0.964
## student_GLuck                    0.001647         0.202697   0.008
## student_FPlans                   0.008026         0.194942   0.041
## student_EgoA                     0.160471         0.287501   0.558
## student_WinArg                  -0.103411         0.165890  -0.623
## student_StrOpinion               0.514071         0.282405   1.820
## student_MChange                  0.190137         0.179035   1.062
## student_EgoB                     0.360345         0.344620   1.046
## student_TrOthers                -0.184100         0.522153  -0.353
## student_OthHelp                 -0.109435         0.509428  -0.215
## student_OthFair                 -0.321681         0.524121  -0.614
## student_Trust                   -0.287165         0.999435  -0.287
## student_Senate             1638548.014832   1662828.121747   0.985
## student_Tito               1638547.383034   1662828.068907   0.985
## student_Court              1638547.796086   1662828.113739   0.985
## student_Govern             1638547.190114   1662828.130238   0.985
## student_CCamp              1638548.113581   1662828.145476   0.985
## student_FDR                1638547.536945   1662828.121151   0.985
## student_Knowledge         -9831283.601853   9976968.890889  -0.985
## student_NextSch                  2.128844         0.347849   6.120
## student_GPA                     -0.573483         0.203880  -2.813
## student_SchOfficer              -0.290974         0.143924  -2.022
## student_SchPublish              -0.106880         0.107812  -0.991
## student_Hobby                   -0.279256         0.125899  -2.218
## student_SchClub                  0.049741         0.125350   0.397
## student_OccClub                 -0.208171         0.104008  -2.001
## student_NeighClub                0.219385         0.111902   1.960
## student_RelClub                 -0.015403         0.109665  -0.140
## student_YouthOrg                -0.171014         0.121715  -1.405
## student_MiscClub                 0.196530         0.750769   0.262
## student_ClubLev                 -0.062139         0.274578  -0.226
## student_Phone                    0.343857         0.458655   0.750
## student_Gen                      0.326500         0.253368   1.289
## student_Race                     1.565123         0.737547   2.122
## parent_Newspaper                 0.064532         0.087459   0.738
## parent_Radio                     0.034424         0.063498   0.542
## parent_TV                       -0.084346         0.101995  -0.827
## parent_Magazine                 -0.378104         0.254610  -1.485
## parent_LifeWish                  0.108988         0.119675   0.911
## parent_GLuck                     0.116028         0.191794   0.605
## parent_FPlans                    0.173079         0.133132   1.300
## parent_WinArg                   -0.073411         0.140237  -0.523
## parent_StrOpinion                0.180217         0.123756   1.456
## parent_MChange                  -0.021160         0.135683  -0.156
## parent_TrOthers                 -0.044626         0.145876  -0.306
## parent_OthHelp                   0.170616         0.163884   1.041
## parent_OthFair                  -0.033992         0.191434  -0.178
## parent_PID                      -0.116142         0.068576  -1.694
## parent_SPID                      0.299568         0.127440   2.351
## parent_Vote                     -0.207845         0.321649  -0.646
## parent_Persuade            3402656.467664   3325455.451043   1.023
```

```
## parent_Rally                  428373.876798   5842576.229364   0.073
## parent_OthAct                 428373.170550   5842576.210321   0.073
## parent_PolClub                428373.828338   5842575.966633   0.073
## parent_Button                 428373.914552   5842576.215148   0.073
## parent_Money                  428374.106532   5842576.260506   0.073
## parent_Participate1        -20415937.641787  19952732.346302  -1.023
## parent_Participate2         14871411.318296  31849857.281430   0.467
## parent_ActFrq                      0.074631         0.045978   1.623
## parent_GovtOpinion                -0.534520         0.193352  -2.764
## parent_GovtCrook                   0.423271         0.188936   2.240
## parent_GovtWaste                  -0.566150         0.217860  -2.599
## parent_TrGovt                      0.268926         0.203055   1.324
## parent_GovtSmart                   0.107138         0.143338   0.747
## parent_Govt4All                   -0.227234         0.163374  -1.391
## parent_Employ                      0.320708         0.308657   1.039
## parent_EducHH                      0.202728         0.120079   1.688
## parent_EducW                       0.057925         0.126118   0.459
## parent_ChurchOrg                   0.173835         0.103850   1.674
## parent_FratOrg                     0.013621         0.148043   0.092
## parent_ProOrg                      0.637748         0.197568   3.228
## parent_CivicOrg                    0.054675         0.239577   0.228
## parent_CLOrg                       0.302409         0.445412   0.679
## parent_NeighClub                  -0.049229         0.154073  -0.320
## parent_SportClub                   0.105139         0.138099   0.761
## parent_InfClub                    -0.108732         0.146034  -0.745
## parent_FarmGr                      0.113567         0.187608   0.605
## parent_WomenClub                  -0.165950         0.180888  -0.917
## parent_MiscClub                   -1.229913         0.611615  -2.011
## parent_ClubLev                     0.478668         0.279345   1.714
## parent_FInc                       -0.134673         0.125172  -1.076
## parent_HHInc                       0.178912         0.113832   1.572
## parent_OwnHome                     0.594267         0.319330   1.861
## parent_Senate                2111914.535950   1564044.980554   1.350
## parent_Tito                  2111915.021204   1564045.058834   1.350
## parent_Court                 2111913.713449   1564044.945251   1.350
## parent_Govern                2111915.370117   1564045.057469   1.350
## parent_CCamp                 2111914.437453   1564045.011374   1.350
## parent_FDR                   2111913.345301   1564045.026063   1.350
## parent_Knowledge           -12671485.799360   9384270.198278  -1.350
## parent_Gen                        -0.738630         0.345712  -2.137
## parent_Race                       -0.808848         0.714961  -1.131
## parent_GPHighSchoolPlacebo        -0.148187         0.307923  -0.481
## parent_HHCollegePlacebo            1.028287         0.651223   1.579
## grps                              -0.097627         0.221828  -0.440
##                                  Pr(>|z|)
## (Intercept)                      0.696295
## student_vote                     0.011675 *
## student_meeting                  0.000467 ***
## student_other                    0.397634
## student_button                   0.335413
## student_money                    0.730468
## student_communicate              0.343927
## student_demonstrate              0.011765 *
## student_community                0.689483
```

```
## student_PubAff                 0.077727 .
## student_Newspaper              0.154895
## student_Radio                  0.412262
## student_TV                     0.283248
## student_Magazine               0.008061 **
## student_FamTalk                0.535137
## student_FrTalk                 0.022600 *
## student_AdultTalk              0.400364
## student_PID                    0.012965 *
## student_SPID                   0.281458
## student_GovtOpinion            0.185275
## student_GovtCrook              0.031398 *
## student_GovtWaste              0.049063 *
## student_TrGovt                 0.699280
## student_GovtSmart              0.724365
## student_Govt4All               0.801620
## student_Cynic                  0.198091
## student_LifeWish               0.334821
## student_GLuck                  0.993516
## student_FPlans                 0.967159
## student_EgoA                   0.576737
## student_WinArg                 0.533038
## student_StrOpinion             0.068708 .
## student_MChange                0.288232
## student_EgoB                   0.295733
## student_TrOthers               0.724404
## student_OthHelp                0.829908
## student_OthFair                0.539379
## student_Trust                  0.773862
## student_Senate                 0.324428
## student_Tito                   0.324429
## student_Court                  0.324429
## student_Govern                 0.324429
## student_CCamp                  0.324428
## student_FDR                    0.324429
## student_Knowledge              0.324429
## student_NextSch          0.000000000936 ***
## student_GPA                    0.004910 **
## student_SchOfficer             0.043205 *
## student_SchPublish             0.321515
## student_Hobby                  0.026548 *
## student_SchClub                0.691504
## student_OccClub                0.045339 *
## student_NeighClub              0.049938 *
## student_RelClub                0.888299
## student_YouthOrg               0.160010
## student_MiscClub               0.793498
## student_ClubLev                0.820962
## student_Phone                  0.453430
## student_Gen                    0.197523
## student_Race                   0.033832 *
## parent_Newspaper               0.460603
## parent_Radio                   0.587726
## parent_TV                      0.408257
```

```
## parent_Magazine                0.137534
## parent_LifeWish                0.362451
## parent_GLuck                   0.545204
## parent_FPlans                  0.193582
## parent_WinArg                  0.600643
## parent_StrOpinion              0.145330
## parent_MChange                 0.876073
## parent_TrOthers                0.759668
## parent_OthHelp                 0.297837
## parent_OthFair                 0.859064
## parent_PID                     0.090335 .
## parent_SPID                    0.018740 *
## parent_Vote                    0.518159
## parent_Persuade                0.306206
## parent_Rally                   0.941552
## parent_OthAct                  0.941552
## parent_PolClub                 0.941552
## parent_Button                  0.941552
## parent_Money                   0.941552
## parent_Participate1            0.306206
## parent_Participate2            0.640555
## parent_ActFrq                  0.104549
## parent_GovtOpinion             0.005701 **
## parent_GovtCrook               0.025072 *
## parent_GovtWaste               0.009358 **
## parent_TrGovt                  0.185369
## parent_GovtSmart               0.454793
## parent_Govt4All                0.164261
## parent_Employ                  0.298785
## parent_EducHH                  0.091356 .
## parent_EducW                   0.646025
## parent_ChurchOrg               0.094148 .
## parent_FratOrg                 0.926693
## parent_ProOrg                  0.001247 **
## parent_CivicOrg                0.819480
## parent_CLOrg                   0.497175
## parent_NeighClub               0.749332
## parent_SportClub               0.446457
## parent_InfClub                 0.456532
## parent_FarmGr                  0.544950
## parent_WomenClub               0.358925
## parent_MiscClub                0.044333 *
## parent_ClubLev                 0.086614 .
## parent_FInc                    0.281973
## parent_HHInc                   0.116015
## parent_OwnHome                 0.062747 .
## parent_Senate                  0.176923
## parent_Tito                    0.176923
## parent_Court                   0.176923
## parent_Govern                  0.176923
## parent_CCamp                   0.176923
## parent_FDR                     0.176923
## parent_Knowledge               0.176923
## parent_Gen                     0.032635 *
```

```
## parent_Race                       0.257921
## parent_GPHighSchoolPlacebo        0.630341
## parent_HHCollegePlacebo           0.114334
## grps                              0.659863
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1223.94  on 953  degrees of freedom
## Residual deviance:  631.06  on 833  degrees of freedom
## AIC: 873.06
##
## Number of Fisher Scoring iterations: 6
```

```
# this gives us propensity scores
data_baseline <- data_baseline %>% mutate(prop_score = predict(model_ps))
```

Following the original assignment instructions, we looked at the balance of the top 10 variables that influenced the propensity score (based on the largest absolute value of the coefficients).

```
# balance for the top 10 covariates

# I asked Prashant about this in lab...he said its fine to ignore whether the coefficients are
# significant or not, and ignore that some of the coefficients are very large.

data_top10 <- tidy(model_ps) %>%
  # getting the absolute value since I think we just care about how much the variables influence
  # propensity score, not positive or negative.
  mutate(abs_estimate = abs(estimate)) %>%
  # arrange in descending order
  arrange(desc(abs_estimate)) %>%
  # select top 10
  slice(1:10)

# get variable names for the top 10
top10_list <- data_top10$term

# look at full data for these variables
data_top10_comp <- data_baseline %>%
  select(all_of(top10_list), college)
```

The balance between treatment (college) and control (no college) of the top 10 variables that affect propensity score are shown below.

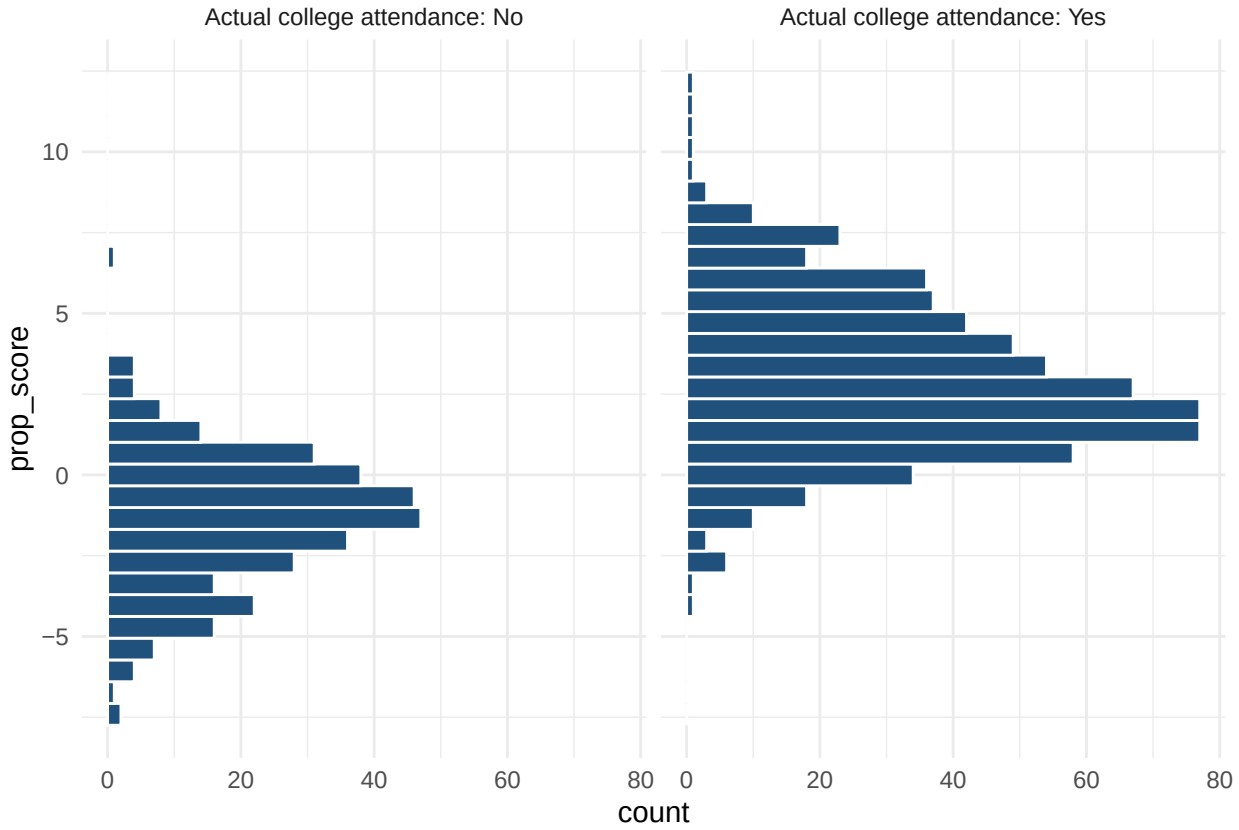
```
# I made a summary table instead of a figure because some of the variables have 5 categories and some a
tbl_summary(data_top10_comp, by = "college",
            type = list(parent_Participate1 ~ 'continuous',
                        parent_Participate2 ~ 'continuous',
                        parent_Knowledge ~ 'continuous',
                        student_Knowledge ~ 'continuous'),
            statistic = list(
            c("parent_Participate1", "parent_Participate2",
```

```
              "parent_Knowledge", "student_Knowledge") ~ "{mean} ({sd})")) %>%
  modify_spanning_header(all_stat_cols() ~ "**Attended College**")
```

Characteristic	**0**, N = 325	**1**, N = 629
parent_Participate1	0.16 (0.24)	0.29 (0.30)
parent_Participate2	0.13 (0.24)	0.25 (0.30)
parent_Knowledge	0.54 (0.22)	0.68 (0.22)
student_Knowledge	0.47 (0.23)	0.68 (0.23)
parent_Persuade	93 (29%)	305 (48%)
parent_Govern	298 (92%)	610 (97%)
parent_Tito	98 (30%)	346 (55%)
parent_Senate	58 (18%)	252 (40%)
parent_CCamp	251 (77%)	574 (91%)
parent_Court	59 (18%)	194 (31%)

Also, following the newer assignment instructions, we plotted the distribution/balance of the propensity scores for the treatment and control groups.

```
# Balance of propensity scores between treatment and control
data_ps <- data_baseline %>%
  select(prop_score, college)

# plot the propensity scores
labs <- paste("Actual college attendance:", c("No", "Yes"))

data_ps %>%
  mutate(college = ifelse(college == 0, labs[1], labs[2])) %>%
  ggplot(aes(x = prop_score)) +
  geom_histogram(color = "white", fill = "#20517D") +
  facet_wrap(~college) +
  coord_flip()+
  theme_minimal()
```



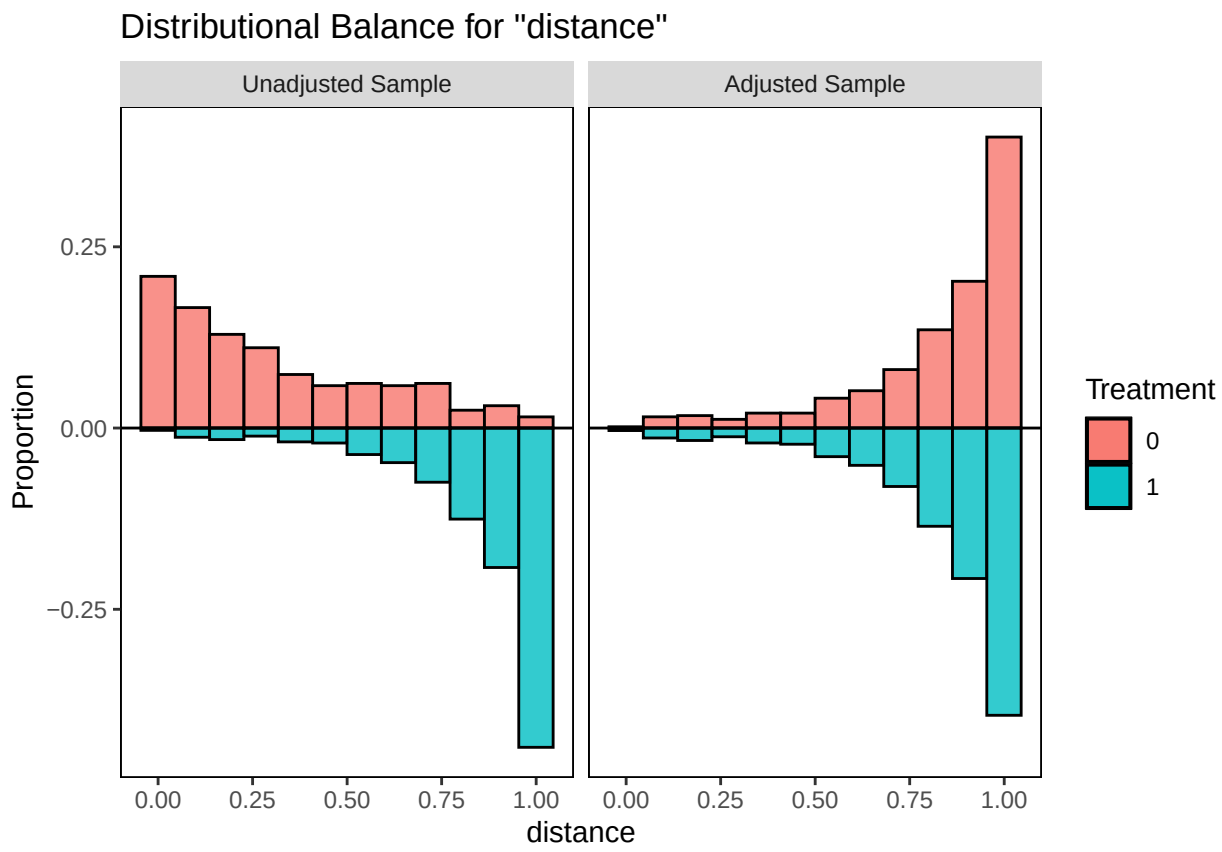
Then, we did the actual matching based on propensity score and calculated the ATT.

```
# match using nearest neighbor
match_nn_att <- matchit(college ~ . - student_ppnscal,
                  data = data_baseline,
                  method = "nearest", distance = "glm", discard = "both", replace = TRUE, estimand = "A
                  # discard: units whose propensity scores fall outside the corresponding region are di
                  # "both" indicates this is done for both treatments and controls

mdata <- get_matches(match_nn_att)

# from another class so the technique is a bit different, but should be the same answer (I was having a

# The simplest approach to get an estimate of the average treatment effects on the treated (ATT) after
# propensity matching is to take means of each group - matchit uses weights to represent individuals
# who have been selected multiple times into a comparison group, so need to take weighted means.
# This code below calculates a ATT marginal risk difference.

mdata_summary <- mdata  %>%
         group_by(college) %>%
         summarize(mean = weighted.mean(student_ppnscal, weights, na.rm=TRUE))
# extract weighted mean for exposed and unexposed and calculate risk difference
mean_exp <- mdata_summary[[2, 2]]
mean_unexp <- mdata_summary[[1, 2]]
ATT <- mean_exp - mean_unexp
```

```
# Prashant also said it is okay to ignore "algorithm did not converge" warning.
```

Then, using a threshold of standardized mean difference of p-score less than or equal to .1, we calculated and reported the number of covariates that meet that balance threshold.

```
m.sum <- summary(match_nn_att)
#plot(m.sum, var.order = "unmatched")

# get additional balance info
balance <- bal.tab(match_nn_att, un = TRUE, binary = "std", m.threshold = 0.1)

# put that balance info in a dataframe
df_balance <- as.data.frame(balance$Balance$M.Threshold)

# calculate percent that met the threshold
n_total <- nrow(df_balance)
n_balanced <- length(df_balance[which(balance$Balance$M.Threshold == "Balanced, <0.1"), ])

pct_balanced <- round((n_balanced/n_total)*100, 2)

# distributional balance before and after matching
bal.plot(match_nn_att, var.name = "distance", which = "both",
         type = "histogram", mirror = TRUE)
```

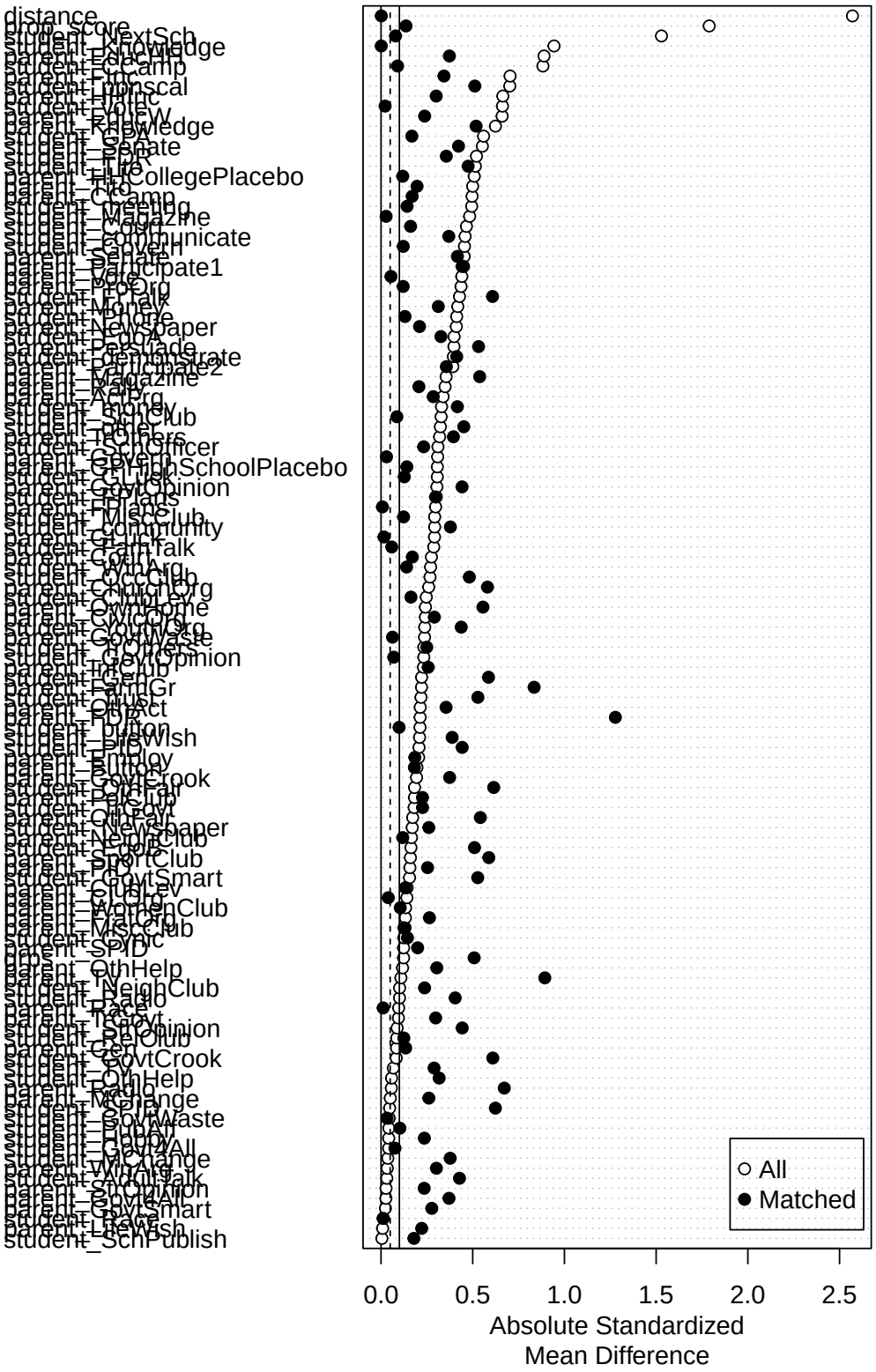


The ATT is -0.9966. This means that the the the average effect of exposure (college) for those who were exposed is to reduce the likelihood of `student_ppnscal` (political participation) by -99.66 percentage points.

The percent of covariates that are balances is 16.26 %.

Vialization for percent imprivement:

```
plot(m.sum, var.order = "unmatched")
```


distance
prop_score
All
Matched
0.0
0.5
1.0
1.5
2.0
2.5
Absolute Standardized
Mean Difference

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).

Next, we ran a simulation to randomly select the number of covariates and which covariates those were, adn then calculate the ATT and the proportion of covariates that meet the standardized mean difference (less than or equal to .1) threshold. NOTE: we had removed the post-treatment covariates using tidyverse in a previous step.

```
Y <- data_baseline$student_ppnscal
X <- data_baseline$college
# select all covariate options
C <- data_baseline %>% select(-college)

# create empty list
data_simulation <- data.frame(run_id = integer(),
                              n_covar = integer(),
                              ATT = double(),
                              pct_balanced = double())

# number of runs (smaller than 10,000 due to run time)
n <- 1000

# loop to run simulations
for (i in 1:n){

  run_id <- i
  # randomly select number of variables to include
  n <- floor(runif(1, min = 2, max = ncol(C)))
  # select n features randomly
  features <- sample(1:ncol(C), n, replace=FALSE)
```

```
  # dataframe of selected features
  tempC <- C[, features]
  # add back in X and Y
  tempdf <- cbind(Y, X, tempC)

  # conduct propensity score matching using nearest neighbor method
  match_nn_att <- matchit(X ~ . - Y,
                  data = tempdf,
                  method = "nearest", distance = "glm", discard = "both", replace = TRUE, estimand = "A

  mdata <- get_matches(match_nn_att)

  # extract weighted mean for exposed and unexposed and calculate ATT
  mdata_summary <- mdata  %>%
         group_by(X) %>%
         summarize(mean = weighted.mean(Y, weights, na.rm=TRUE))

  mean_unexp <- mdata_summary[[1, 2]]
  ATT <- mean_exp - mean_unexp

  # get the covariate balance info with standardized mean difference
  balance <- bal.tab(match_nn_att, un = TRUE, binary = "std", m.threshold = 0.1)

  df_balance <- as.data.frame(balance$Balance$M.Threshold)

  # calculate the percent of covariates balances based on our 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)

  # put all this into a dataframe
  data_temp <- data.frame(run_id = i,
                          n_covar = n,
                          ATT = ATT,
                          pct_balanced = pct_balanced)

  # append this dataframe to the simulation data frame
  data_simulation <- rbind(data_simulation, data_temp)

}
```

The simulation returned a dataframe with the ATT, number of covariates, and percent of covariates balanced, for each model.

```
head(data_simulation)
```

```
##   run_id n_covar          ATT pct_balanced
## 1      1      32 -1.010291595    0.1470588
## 2      2      58 -1.010291595    0.1355932
## 3      3      46  0.007051318    0.4375000
## 4      4     103 -1.010291595    0.1538462
## 5      5     103 -1.010291595    0.1730769
```

```
## 6      6      48  0.206181475    0.4600000
```

Summary of ATT and percent balanced.

```
summary(data_simulation$ATT)
```

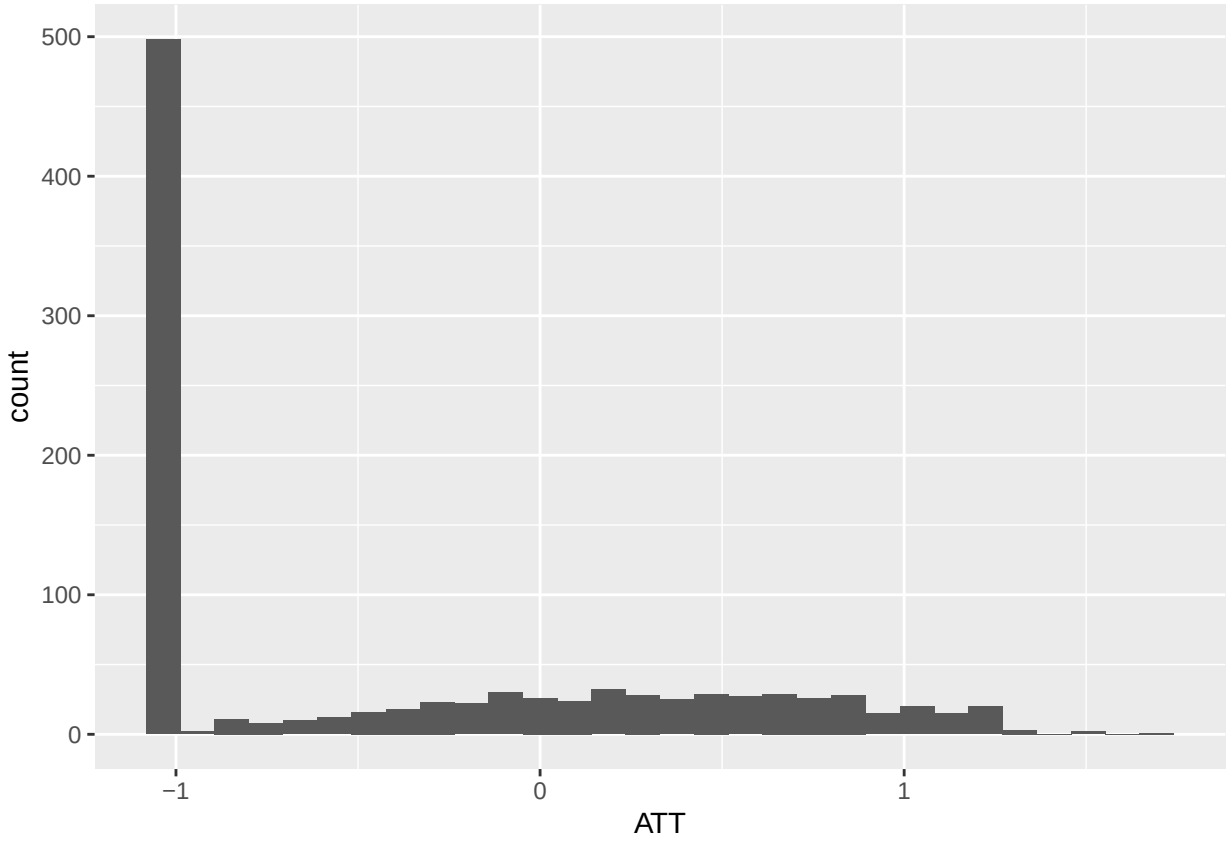
```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
## -1.0675 -1.0103 -0.8975 -0.3597  0.2930  1.6619
```

```
summary(data_simulation$pct_balanced)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
## 0.04255 0.16667 0.23607 0.37459 0.58373 1.00000
```

Distribution plot of ATT.

```
ggplot(data_simulation, aes(x = ATT))+
  geom_histogram()
```



Questions

1. **How many simulations resulted in models with a higher proportion of balanced covariates? Do you have any concerns about this?**

2. **Your Answer**: In our original matching analysis, we had 14.6

3. **Analyze the distribution of the ATTs. Do you have any concerns about this distribution?**

4. **Your Answer:** The distrubution of our ATTs, as seen in the above plot, is somewhat interesting. There are a large number of ATTs that are the same value (-1), regardless of having different covariates and different numbers of covariates selected. However, to the right of this, is a small somewhat normal distribution. We are assuming that our data likely includes covariates that are less helpful for calculating propensity scores, which could cause this discrepancy depending on when these variables vs. variables more predictive of going to college are selected.

Matching Algorithm of Your Choice

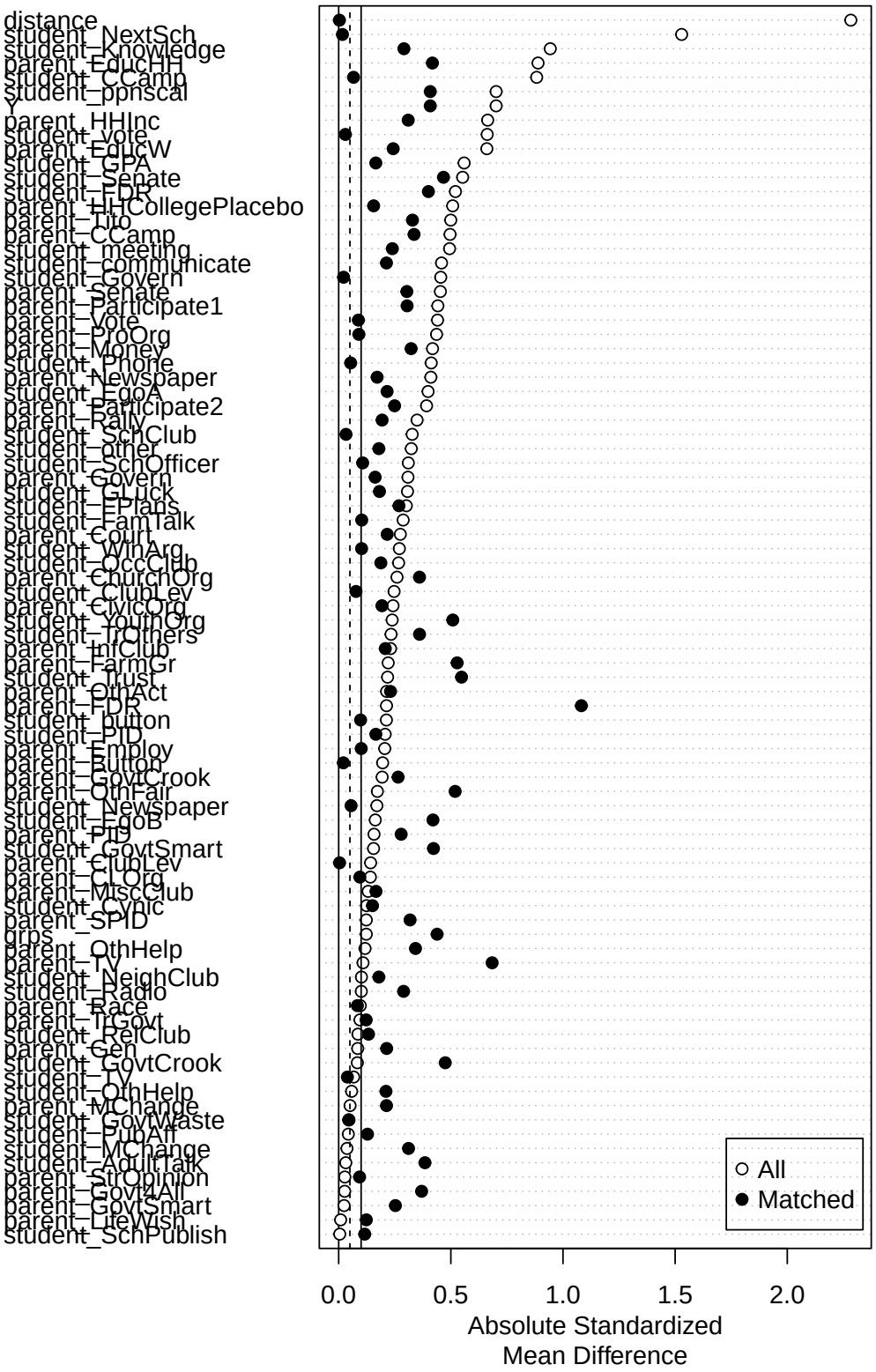
Simulate Alternative Model

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

```
# Remove post-treatment covariates

# Randomly select features

# Simulate random selection of features 10k+ times

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

# Plot ATT v. proportion

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


Y <- data_baseline$student_ppnscal
X <- data_baseline$college
# select all covariate options
C <- data_baseline %>% select(-college)

# create empty list
data_simulation_full <- data.frame(run_id = integer(),
                                   n_covar = integer(),
                                   ATT = double(),
                                   pct_balanced = double())

# number of runs
# limiting the number for now due to computational time
n <- 1000

# loop to run simulations
for (i in 1:n){

  run_id <- i
```

```
  # randomly select number of variables to include
  n <- floor(runif(1, min = 2, max = ncol(C)))
  # select n features randomly
  features <- sample(1:ncol(C), n, replace=FALSE)
  # dataframe of selected features
  tempC <- C[, features]
  # add back in X and Y
  tempdf <- cbind(Y, X, tempC)

  # conduct propensity score matching using optimal method
  match_full_att <- matchit(X ~ . - Y,
                  data = tempdf,
                  method = "full", distance = "mahalanobis")

  m_full_data <- match.data(match_full_att)

  # extract weighted mean for exposed and unexposed and calculate ATT
  m_full_data_summary <- m_full_data  %>%
         group_by(X) %>%
         summarize(mean = weighted.mean(Y, weights, na.rm=TRUE))
  mean_full_exp <- m_full_data_summary[[2, 2]]
  mean_full_unexp <- m_full_data_summary[[1, 2]]
  ATT_full <- mean_exp - mean_full_unexp

  # get the covariate balance info with standardized mean difference
  balance_full <- bal.tab(match_full_att, un = TRUE, binary = "std", m.threshold = 0.1)

  df_balance_full <- as.data.frame(balance_full$Balance$M.Threshold)

  # calculate the percent of covariates balances based on our threshold
  n_total_full <- nrow(df_balance_full)
  n_balanced_full <- length(df_balance_full[which(balance_full$Balance$M.Threshold == "Balanced, <0.1")

  pct_balanced_full <- (n_balanced_full/n_total_full)

  # put all this into a dataframe
  data_temp_full <- data.frame(run_id = i,
                           n_covar = n,
                           ATT_full = ATT_full,
                           pct_balanced_full = pct_balanced)

  # append this dataframe to the simulation data frame
  data_simulation_full <- rbind(data_simulation_full, data_temp_full)

}
```

```
# Visualization for distributions of percent improvement
match_nn_summary_plot <- summary(match_nn_att)
plot(match_nn_summary_plot, var.order = "unmatched")
```

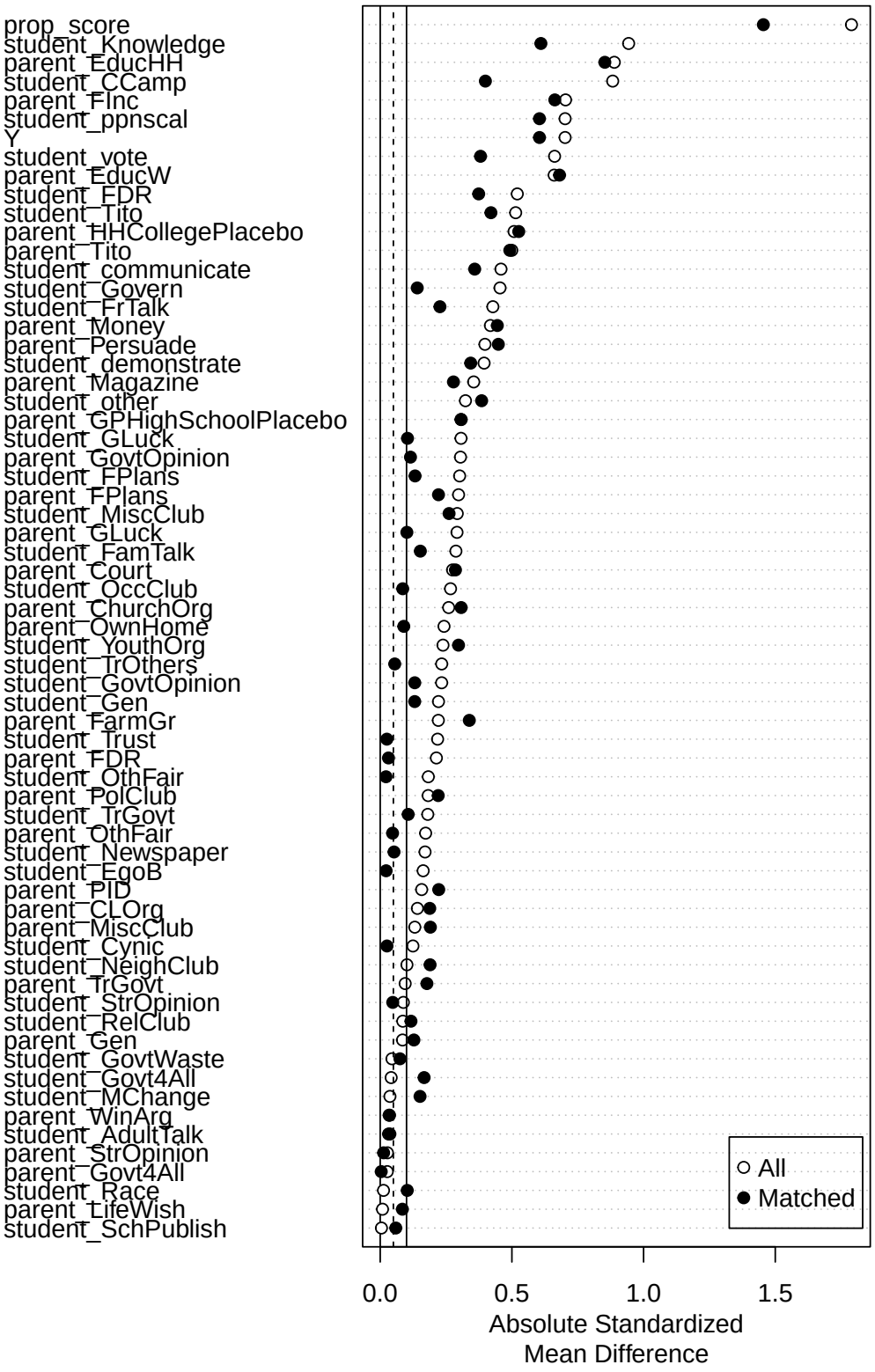

distance
student_NextSch
student_Knowledge
parent_EducHH
student_CCamp
student_ppnscal
Y
parent_HHInc
student_Vote
parent_EducW
student_GPA
student_Senate
student_FDR
parent_HHCollegePlacebo
parent_Info
parent_CCamp
student_meeting
student_communicate
student_Govern
parent_Senate
parent_Participate1
parent_Vote
parent_ProOrg
parent_Money
student_Phone
parent_Newspaper
student_EgoA
parent_Participate2
parent_Rally
student_SchClub
student_other
student_SchOfficer
parent_Govern
student_GLuck
student_FPlans
student_FamTalk
parent_Court
student_WthArg
student_OccClub
parent_ChurchOrg
student_ClubLev
parent_CivicOrg
student_YouthOrg
student_TrOthers
parent_InfClub
parent_FarmGr
student_Trust
parent_OthAct
parent_FDR
student_Button
student_PID
parent_Employ
parent_Button
parent_GovtCrook
parent_OthFair
student_Newspaper
student_EgoB
parent_PID
student_GovtSmart
parent_ClubLev
parent_CLOrg
parent_MiscClub
student_Cynic
parent_SPID
grps
parent_OthHelp
parent_TV
student_NeighClub
student_Radio
parent_Race
parent_TrGovt
student_RelClub
parent_Gen
student_GovtCrook
student_TV
student_OthHelp
parent_MChange
student_GovtWaste
student_PubAff
student_MChange
student_AdultTalk
parent_StrOpinion
parent_GovtAll
parent_GovtSmart
parent_LifeWish
student_SchPublish
All
Matched
0.0
0.5
1.0
1.5
2.0
Absolute Standardized
Mean Difference

```
df_balance
```

```
##    balance$Balance$M.Threshold
## 1                 Balanced, <0.1
## 2             Not Balanced, >0.1
## 3             Not Balanced, >0.1
## 4             Not Balanced, >0.1
## 5             Not Balanced, >0.1
## 6             Not Balanced, >0.1
## 7             Not Balanced, >0.1
## 8             Not Balanced, >0.1
## 9             Not Balanced, >0.1
## 10            Not Balanced, >0.1
## 11                Balanced, <0.1
## 12            Not Balanced, >0.1
## 13            Not Balanced, >0.1
## 14                Balanced, <0.1
## 15            Not Balanced, >0.1
## 16            Not Balanced, >0.1
## 17            Not Balanced, >0.1
## 18            Not Balanced, >0.1
## 19                Balanced, <0.1
## 20            Not Balanced, >0.1
## 21            Not Balanced, >0.1
## 22            Not Balanced, >0.1
## 23                Balanced, <0.1
## 24            Not Balanced, >0.1
## 25            Not Balanced, >0.1
## 26            Not Balanced, >0.1
## 27            Not Balanced, >0.1
## 28            Not Balanced, >0.1
## 29            Not Balanced, >0.1
## 30            Not Balanced, >0.1
## 31            Not Balanced, >0.1
## 32            Not Balanced, >0.1
## 33            Not Balanced, >0.1
## 34            Not Balanced, >0.1
## 35            Not Balanced, >0.1
## 36            Not Balanced, >0.1
## 37                Balanced, <0.1
## 38            Not Balanced, >0.1
## 39                Balanced, <0.1
## 40            Not Balanced, >0.1
## 41            Not Balanced, >0.1
## 42            Not Balanced, >0.1
## 43                Balanced, <0.1
## 44                Balanced, <0.1
## 45            Not Balanced, >0.1
## 46                Balanced, <0.1
## 47                Balanced, <0.1
## 48            Not Balanced, >0.1
## 49            Not Balanced, >0.1
## 50            Not Balanced, >0.1
```

```
## 51              Balanced, <0.1
## 52          Not Balanced, >0.1
## 53          Not Balanced, >0.1
## 54          Not Balanced, >0.1
## 55          Not Balanced, >0.1
## 56          Not Balanced, >0.1
## 57              Balanced, <0.1
## 58          Not Balanced, >0.1
## 59          Not Balanced, >0.1
## 60          Not Balanced, >0.1
## 61          Not Balanced, >0.1
## 62          Not Balanced, >0.1
## 63          Not Balanced, >0.1
## 64          Not Balanced, >0.1
## 65          Not Balanced, >0.1
## 66          Not Balanced, >0.1
## 67              Balanced, <0.1
## 68              Balanced, <0.1
## 69          Not Balanced, >0.1
## 70          Not Balanced, >0.1
## 71              Balanced, <0.1
## 72          Not Balanced, >0.1
## 73          Not Balanced, >0.1
## 74              Balanced, <0.1
## 75          Not Balanced, >0.1
## 76          Not Balanced, >0.1
## 77          Not Balanced, >0.1
## 78              Balanced, <0.1
## 79          Not Balanced, >0.1
## 80          Not Balanced, >0.1
## 81          Not Balanced, >0.1
## 82          Not Balanced, >0.1
## 83          Not Balanced, >0.1
## 84          Not Balanced, >0.1
## 85              Balanced, <0.1
```

```
match_full_summary_plot <- summary(match_full_att)
plot(match_full_summary_plot, var.order = "unmatched")
```


prop_score
student_Knowledge
parent_EducHH
student_CCamp
parent_FInc
student_ppnscal
Y
student_vote
parent_EducW
student_FDR
student_Tito
parent_HHCollegePlacebo
parent_Tito
student_communicate
student_Govern
student_FrTalk
parent_Money
parent_Persuade
student_demonstrate
parent_Magazine
student_other
parent_GPHighSchoolPlacebo
student_GLuck
parent_GovtOpinion
student_FPlans
parent_FPlans
student_MiscClub
parent_GLuck
student_FamTalk
parent_Court
student_OccClub
parent_ChurchOrg
parent_OwnHome
student_YouthOrg
student_TrOthers
student_GovtOpinion
student_Gen
parent_FarmGr
student_Trust
parent_FDR
student_OthFair
parent_PolClub
student_TrGovt
parent_OthFair
student_Newspaper
student_EgoB
parent_PID
parent_CLOrg
parent_MiscClub
student_Cynic
student_NeighClub
parent_TrGovt
student_StrOpinion
student_RelClub
parent_Gen
student_GovtWaste
student_Govt4All
student_MChange
parent_WinArg
student_AdultTalk
parent_StrOpinion
parent_Govt4All
student_Race
parent_LifeWish
student_SchPublish
All
Matched
0.0
0.5
1.0
1.5
Absolute Standardized
Mean Difference

```
#Number balanced for NN model = 23
#Number balanced for optimal model = 17
```

Questions

1. **Does your alternative matching method have more runs with higher proportions of balanced covariates?**
2. **Your Answer:** No, overall, the nearest neighbor models had more instances of higher proportions of balanced covariates than the optimal matching model.
3. **Use a visualization to examine the change in the distribution of the percent improvement in balance in propensity score matching vs. the distribution of the percent improvement in balance in your new method. Which did better? Analyze the results in 1-2 sentences.**
4. **Your Answer:** Both the nearest neighbor matching and the optimal matching are clustered between 0 and .5 absolute SMD.Overall, it looks like propensity score matching performed better, as more of the scores lie closer to zero across all vocariates. In comparison, the optimal matching somewhat follows the original distribution, and does not match as well for unmatched data with higher SMDs.

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

Discussion Questions

1. Why might it be a good idea to do matching even if we have a randomized or as-if-random design?
2. **Your Answer:** Matching is still a good idea because there may still be unmeasured baseline characteristics that skew a random sample. Randomization cannot account for the values of covariates, while matching can as an extra measure.
3. The standard way of estimating the propensity score is using a logistic regression to estimate probability of treatment. Given what we know about the curse of dimensionality, do you think there might be advantages to using other machine learning algorithms (decision trees, bagging/boosting forests, ensembles, etc.) to estimate propensity scores instead?
4. **Your Answer:** Yes, choosing "representative" covariates for a logistic model leaves us vulnerable to assumptions about what is representative. A robust, data-driven approach would be using a machine learning algorithm to either calculate propensity scores across high-dimensional data, or decide on the representative coviariates for us.