Propensity Score Analysis in R

Peter Sun and Shenyang Guo

March 17-19, 2022

Contents

1 How to Setup R and RStudio

1.1 Download R, RStudio, and PSA-R

1. Download the latest version of R: https://www.r-project.org/
2. Download the latest version of RStudio Desktop: https://www.rstudio.com/products/rstudio/download/
3. Download the two PSA-R zip files under "R Syntax": https://ssw.unc.edu/psa/

1.2 Run the Code

1. To view the code output without running it, extract the "PSA-R_Output.zip" file and open "index.html"
2. To run an individual section:
   - Extract "PSA-R_Code_Data.zip"
   - Open "PSA-R.Rproj"
   - Open the desired section code in the file browser (e.g., "01_Section4.4.1.Rmd")
   - Install packages if necessary
   - Click on "Run All"
3. To knit the entire book into HTML output, click on "Build Book"

1.3 Troubleshoot Package Errors

If a line of code using a certain package is not working, try installing an older version of that package. See the output `sessionInfo()` below for package versions that are known to be compatible with the PSA-R code.

As of March 16, 2022, the latest version of `PSweight` (1.1.6) will only work if line 142 in "08_Section6.5.2.Rmd" is changed from:

```
data = sur_subclass1,
```

to

```
data = as.data.frame(sur_subclass1),
```

An alternative solution is to install an older version of `PSweight` that works (1.1.2):

```
packageVersion("PSweight")
detach("package:PSweight", unload = T)
remove.packages("PSweight")
library(devtools)
devtools::install_version("PSweight", version = "1.1.2",
  repos = "http://cran.us.r-project.org")
```

1.4 PSA-R Session Info

The following output for `sessionInfo()` lists package versions that are known to be compatible with the PSA-R code (if the fix for `PSweight` above is implemented).

```
R version 4.1.3 (2022-03-10)
Platform: x86_64-w64-mingw32/x64 (64-bit)
Running under: Windows 10 x64 (build 19043)

Matrix products: default

locale:
[1] LC_COLLATE=English_United States.1252  LC_CTYPE=English_United States.1252
[3] LC_MONETARY=English_United States.1252 LC_NUMERIC=C
```

```
[5] LC_TIME=English_United States.1252

attached base packages:
[1] stats     graphics  grDevices utils     datasets  methods   base

loaded via a namespace (and not attached):
 [1] tidyr_1.2.0            VGAM_1.1-6             splines_4.1.3          foreach_1.5.2
 [5] carData_3.0-5          gam_1.20.1             gtools_3.9.2           Formula_1.2-4
 [9] assertthat_0.2.1       stats4_4.1.3           coin_1.4-2             yaml_2.3.5
[13] numDeriv_2016.8-1.1    pillar_1.7.0           backports_1.4.1        lattice_0.20-45
[17] glue_1.6.2             digest_0.6.29          colorspace_2.0-3       sandwich_3.0-1
[21] gbm_2.1.8              htmltools_0.5.2        Matrix_1.4-0           pkgconfig_2.0.3
[25] broom_0.7.12           haven_2.4.3            gmodels_2.18.1         bookdown_0.25
[29] purrr_0.3.4            mvtnorm_1.1-3          scales_1.1.1           gdata_2.18.0
[33] tibble_3.1.6           generics_0.1.2         car_3.0-12             ggplot2_3.3.5
[37] sjlabelled_1.1.8       ellipsis_0.3.2         cobalt_4.3.2           TH.data_1.1-0
[41] nnet_7.3-17            maxLik_1.5-2           cli_3.2.0              survival_3.2-13
[45] magrittr_2.0.2         crayon_1.5.0           MatchIt_4.3.4          evaluate_0.15
[49] fansi_1.0.2            MASS_7.3-55            SuperLearner_2.0-28    forcats_0.5.1
[53] WeightIt_0.12.0        rsconnect_0.8.25       tools_4.1.3            hms_1.1.1
[57] mitools_2.4            multcomp_1.4-18        matrixStats_0.61.0     lifecycle_1.0.1
[61] munsell_0.5.0          systemfit_1.1-24       compiler_4.1.3         rlang_1.0.2
[65] grid_4.1.3             Matching_4.9-11        iterators_1.0.14       miscTools_0.6-26
[69] rbounds_2.1            rmarkdown_2.13         gtable_0.3.0           codetools_0.2-18
[73] abind_1.4-5            DBI_1.1.2              R6_2.5.1               nnls_1.4
[77] zoo_1.8-9              knitr_1.37             dplyr_1.0.8            fastmap_1.1.0
[81] utf8_1.2.2             libcoin_1.0-9          insight_0.16.0         sampleSelection_1.2-12
[85] modeltools_0.2-23      parallel_4.1.3         Rcpp_1.0.8.2           vctrs_0.3.8
[89] tidyselect_1.1.2       xfun_0.30              PSweight_1.1.6         lmtest_0.9-39
```

2 Greedy Nearest Neighbor Matching

2.1 Load Packages

The `haven` and `sjlabelled` packages are used to load and clean Stata data files (.dta); the `MatchIt` package contains functions for greedy matching; the `cobalt` package contains functions for balance checking; and the `tidyverse` package is loaded for its data manipulation functions.

```
library(haven)
library(sjlabelled)
library(cobalt)
library(MatchIt)
library(tidyverse)
library(kableExtra)
```

2.2 Description of Dataset

This dataset is a sample of 2,758 children from the National Survey of Child and Adolescent Well-Being (NSCAW). The treatment condition is `aodserv` or caregivers who received (aodserv = 1) or did not receive (aodserv = 0) substance abuse services. Two matching procedures are illustrated here. The full code contains 12 matching schemes and can be found in Section 5.8.1 of the PSA-R code.

2.3 Load Data and Sort

```
set.seed(1000)
gm_df <- haven::read_dta("data/chpt5_1_original.dta") %>%
  haven::zap_formats() %>%
  sjlabelled::remove_all_labels() %>%
  as_tibble() %>%
  add_column(runif = runif(nrow(.))) %>%
  arrange(runif) %>%
  select(-runif)
```

2.4 Check Balance Before Matching

Use `chisq.test` to check balance before matching:

```
gm_df %>%
  select(married, educ, pov, employ, open, race, chdage, cgage, CRA47A, mental,
         arrest, PSH17A, maltx, ra, cidi, cgneed, cwwrep, aodserv) %>%
  pivot_longer(-aodserv, names_to = "variable") %>%
  group_by(variable) %>%
  nest() %>%
  mutate(bivariate.test = map(data, ~chisq.test(.$aodserv, .$value, correct = F))) %>%
  mutate(statistic = map(bivariate.test, ~round(.$statistic, 3))) %>%
  mutate(p.value = map(bivariate.test, ~round(.$p.value, 3))) %>%
  unnest(cols = c(statistic, p.value)) %>%
  select(variable, statistic, p.value)
```

```
## # A tibble: 17 x 3
## # Groups:   variable [17]
##    variable statistic p.value
##    <chr>        <dbl>   <dbl>
##  1 married       2.97   0.085
##  2 educ         10.5    0.005
```

```
##  3 pov          11.3    0.023
##  4 employ       23.1    0
##  5 open         58.4    0
##  6 race         11.5    0.009
##  7 chdage       55.4    0
##  8 cgage         3.56   0.313
##  9 CRA47A       17.5    0
## 10 mental       92.5    0
## 11 arrest      127.     0
## 12 PSH17A      179.     0
## 13 maltx        49.7    0
## 14 ra          585.     0
## 15 cidi        157.     0
## 16 cgneed      139.     0
## 17 cwwrep     1240.     0
```

Alternatively, the `cobalt` package provides several convenient functions for assessing balance.

The standardized mean difference (SMD) is a commonly used balance measure. It is calculated as the difference in means of a covariate across the treatment groups, divided by the standard deviation in the treated group (ATT), the control group (ATC), or the pooled standard deviation (ATE). Stuart et al. (2013) recommend 0.1 or 0.25 as reasonable cut-offs for acceptable standardized biases.[1]

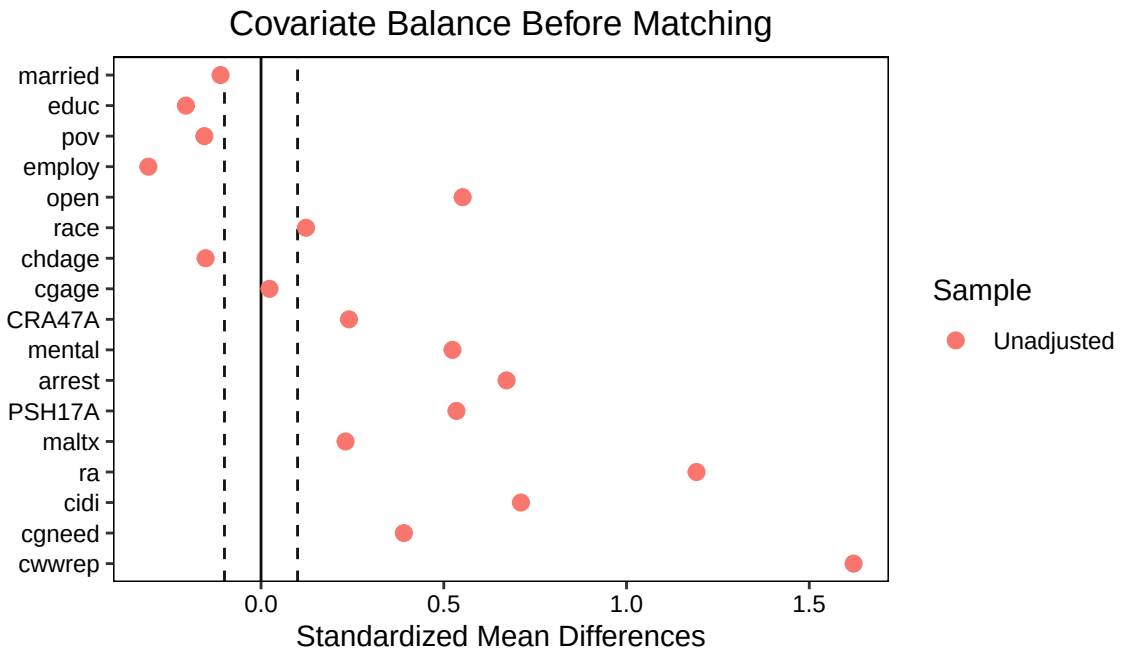
```
# Balance table
bal.tab(select(
  gm_df, married, educ, pov, employ, open, race, chdage, cgage, CRA47A,
  mental, arrest, PSH17A, maltx, ra, cidi, cgneed, cwwrep
),
treat = gm_df$aodserv,
s.d.denom = "treated",
threshold = .1
)
```

```
## Balance Measures
##            Type Diff.Un     M.Threshold.Un
## married  Binary -0.0485     Balanced, <0.1
## educ    Contin. -0.2055 Not Balanced, >0.1
## pov     Contin. -0.1550 Not Balanced, >0.1
## employ   Binary -0.1472 Not Balanced, >0.1
## open     Binary  0.2327 Not Balanced, >0.1
## race    Contin.  0.1231 Not Balanced, >0.1
## chdage  Contin. -0.1517 Not Balanced, >0.1
## cgage   Contin.  0.0232     Balanced, <0.1
## CRA47A   Binary  0.1184 Not Balanced, >0.1
## mental   Binary  0.2619 Not Balanced, >0.1
## arrest   Binary  0.3254 Not Balanced, >0.1
## PSH17A   Binary  0.2512 Not Balanced, >0.1
## maltx   Contin.  0.2312 Not Balanced, >0.1
## ra       Binary  0.5598 Not Balanced, >0.1
## cidi     Binary  0.3482 Not Balanced, >0.1
## cgneed   Binary  0.1494 Not Balanced, >0.1
## cwwrep   Binary  0.7036 Not Balanced, >0.1
##
```

[1] Stuart, E. A., Lee, B. K., & Leacy, F. P. (2013). Prognostic score–based balance measures for propensity score methods in comparative effectiveness research. *Journal of Clinical Epidemiology*, 66(8 0), S84-S90.e1. https://doi.org/10.1016/j.jclinepi.2013.01.013

```
## Balance tally for mean differences
##                     count
## Balanced, <0.1          2
## Not Balanced, >0.1     15
##
## Variable with the greatest mean difference
##  Variable Diff.Un     M.Threshold.Un
##     cwwrep  0.7036 Not Balanced, >0.1
##
## Sample sizes
##     Control Treated
## All    2460     298
```

```
# Love plot
love.plot(select(
  gm_df, married, educ, pov, employ, open, race, chdage, cgage, CRA47A,
  mental, arrest, PSH17A, maltx, ra, cidi, cgneed, cwwrep
),
treat = gm_df$aodserv,
binary = "std",
s.d.denom = "treated",
threshold = .1
) +
  labs(title = "Covariate Balance Before Matching")
```



2.5 Greedy Nearest Neighbor Matching Without Replacement

By default, the `MatchIt::matchit()` function performs greedy nearest neighbor matching without replacement, therefore the `method = "nearest"` and `replace = F` arguments do not need to be specified.

To avoid dissimilar matches, we can constrain matches so that the absolute distance of propensity scores between two participants is less than a specified tolerance for matching or a caliper. The width of the caliper is by default in standard deviation units and can be specified using the `caliper` argument. A wide caliper may result in more matches and a larger sample, but inexact matching may occur as indicated by large distances on the propensity score between the treated and nontreated cases. Using varying caliper sizes can test the sensitivity of the findings. Here we use a caliper size of a quarter of a standard deviation, which is suggested by Rosenbaum and Rubin (1985).

The order of the matching can be specified using the `m.order` argument. If this argument is set to `largest`, then matching begins with the treated subject with the highest propensity score; if set to `smallest`, then matching takes places in ascending order of the distance measures; and if `random`, matching takes place in a random order.

Finally, the logit of the predicted probability from a logistic regression model can be supplied to the `distance` argument. The logit of the predicted probability is used, because the logit is approximately normally distributed.

```
# Logistic regression specification
(gm_f <- cobalt::f.build("aodserv", select(gm_df, PSH17A:other, -aodserv)))
```

```
## aodserv ~ PSH17A + CRA47A + married + high + bahigh + poverty2 +
##     poverty3 + poverty4 + poverty5 + employ + open + black +
##     hispanic + natam + cgrage1 + cgrage2 + cgrage3 + chdage1 +
##     chdage2 + chdage3 + mental + arrest + sexual + provide +
##     supervis + other
## <environment: 0x0000000021c94f68>
```

```
# Calculate the logit of the predicted probability as the propensity score
gm_psm <- glm(gm_f, data = gm_df, family = binomial)
gm_ps <- predict(gm_psm, newdata = gm_df, type = "response")
gm_ps_logit <- log((1 - gm_ps) / gm_ps)

# Greedy nearest neighbor matching without replacement
set.seed(1000)
(gm_out <- MatchIt::matchit(
  gm_f,
  data = gm_df,
  distance = gm_ps_logit,
  m.order = "largest", # descending order
  caliper = .25
))
```

```
## A matchit object
##  - method: 1:1 nearest neighbor matching without replacement
##  - distance: User-defined [caliper]
##  - caliper: <distance> (0.311)
##  - number of obs.: 2758 (original), 566 (matched)
##  - target estimand: ATT
##  - covariates: PSH17A, CRA47A, married, high, bahigh, poverty2, poverty3, poverty4, poverty5, employ
```

Notice that a limitation of this matching scheme is that it reduces the sample size from 2758 to 574—287 cases in the control group and 287 cases in the treated group.

The `matchit` object will return a `match.matrix`, which contains the treated units as the rownames and the

values in each row the names or indices of the control units matched to the treated units:

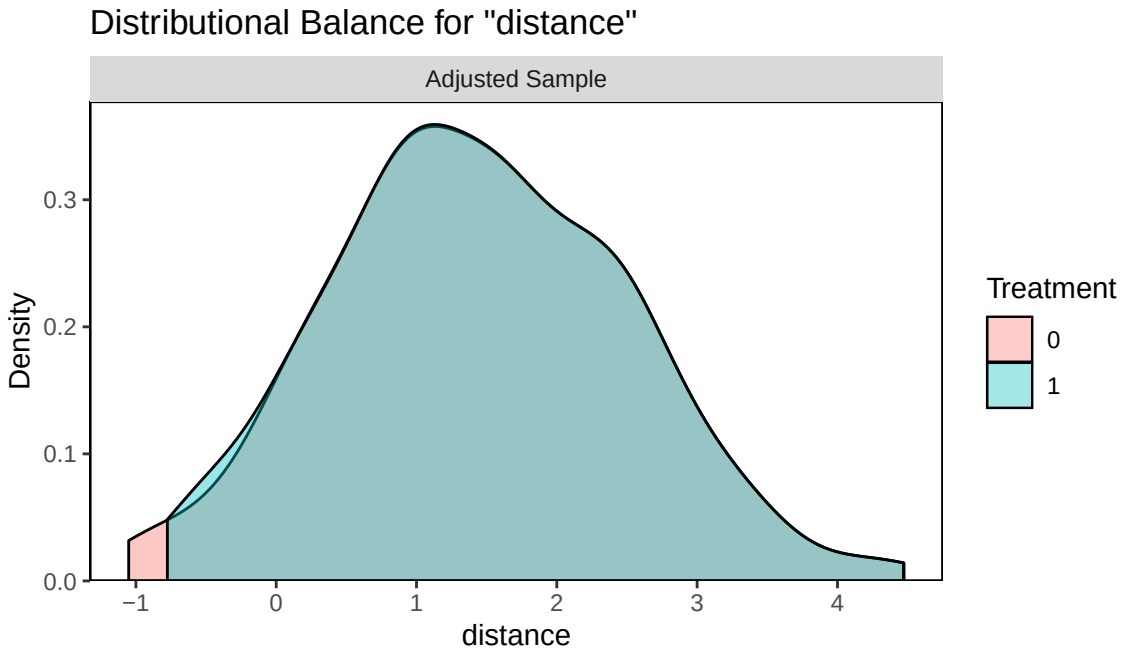
```
head(gm_out$match.matrix)
```

```
##    [,1]
## 2  "420"
## 3  "1852"
## 10 "2130"
## 14 NA
## 41 "782"
## 49 "1162"
```

2.5.1 Check Common Support

Greedy matching is criticized, because it requires a sizeable common-support region to work. The common support region is defined as the region bounded by the maximum value of estimated propensity scores for the treated participants and by the minimum value of the estimated propensity scores for the nontreated participants. In this example, a sizeable common-support region exists. The `discard` argument in `matchit()` can be used to discard units outside a region of common support.

```
cobalt::bal.plot(gm_out, var.name = "distance")
```



2.5.2 Check Balance

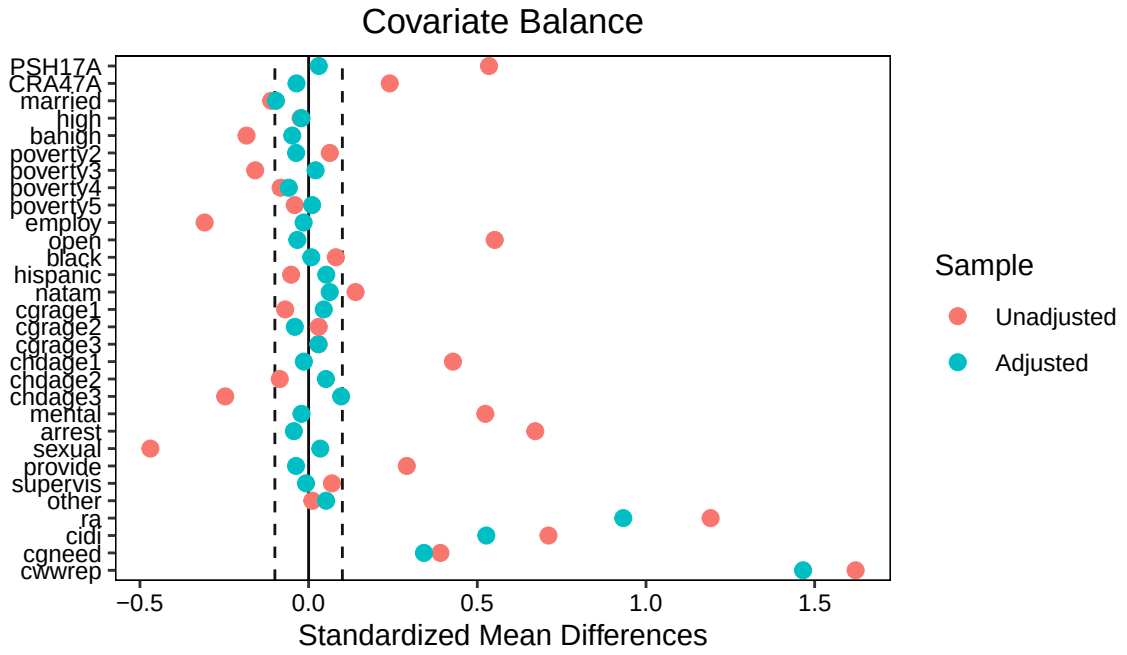
Covariate balance can be assessed using hypothesis tests, such as `chisq.test`:

```
gm_out_data <- MatchIt::match.data(gm_out)
chisq.test(gm_out_data$ra, gm_out_data$aodserv)
```

```
##
##  Pearson's Chi-squared test with Yates' continuity correction
##
## data:  gm_out_data$ra and gm_out_data$aodserv
## X-squared = 108.05, df = 1, p-value < 2.2e-16
```

The object from `matchit()` can be directly used in `cobalt` functions to produce balance tables and plots. To specify additional variables for which to display balance, use the argument `addl` in conjunction with `data`.

```
cobalt::love.plot(gm_out, binary = "std", threshold = c(m = .1), drop.distance = T,
                  addl = c("ra", "cidi", "cgneed", "cwwrep"), data = gm_df)
```




```
cobalt::bal.tab(gm_out, binary = "std", threshold = c(m = .1), un = T,
                addl = c("ra", "cidi", "cgneed", "cwwrep"), data = gm_df)
```

```
## Call
##  MatchIt::matchit(formula = gm_f, data = gm_df, distance = gm_ps_logit,
##     m.order = "largest", caliper = 0.25)
##
## Balance Measures
##              Type Diff.Un Diff.Adj        M.Threshold
## distance Distance -1.3021   0.0068     Balanced, <0.1
## PSH17A     Binary  0.5347   0.0301     Balanced, <0.1
## CRA47A     Binary  0.2406  -0.0359     Balanced, <0.1
## married    Binary -0.1108  -0.0969     Balanced, <0.1
## high       Binary -0.0236  -0.0214     Balanced, <0.1
## bahigh     Binary -0.1840  -0.0485     Balanced, <0.1
## poverty2   Binary  0.0628  -0.0370     Balanced, <0.1
## poverty3   Binary -0.1584   0.0207     Balanced, <0.1
## poverty4   Binary -0.0829  -0.0587     Balanced, <0.1
## poverty5   Binary -0.0410   0.0105     Balanced, <0.1
## employ     Binary -0.3082  -0.0148     Balanced, <0.1
## open       Binary  0.5516  -0.0335     Balanced, <0.1
## black      Binary  0.0807   0.0078     Balanced, <0.1
## hispanic   Binary -0.0519   0.0524     Balanced, <0.1
## natam      Binary  0.1392   0.0626     Balanced, <0.1
## cgrage1    Binary -0.0694   0.0448     Balanced, <0.1
## cgrage2    Binary  0.0300  -0.0409     Balanced, <0.1
## cgrage3    Binary  0.0297   0.0289     Balanced, <0.1
## chdage1    Binary  0.4279  -0.0142     Balanced, <0.1
```

```
## chdage2    Binary -0.0856   0.0513     Balanced, <0.1
## chdage3    Binary -0.2473   0.0963     Balanced, <0.1
## mental     Binary  0.5238  -0.0212     Balanced, <0.1
## arrest     Binary  0.6718  -0.0438     Balanced, <0.1
## sexual     Binary -0.4690   0.0346     Balanced, <0.1
## provide    Binary  0.2909  -0.0373     Balanced, <0.1
## supervis   Binary  0.0690  -0.0077     Balanced, <0.1
## other      Binary  0.0110   0.0518     Balanced, <0.1
## ra         Binary  1.1915   0.9327 Not Balanced, >0.1
## cidi       Binary  0.7110   0.5267 Not Balanced, >0.1
## cgneed     Binary  0.3907   0.3419 Not Balanced, >0.1
## cwwrep     Binary  1.6213   1.4656 Not Balanced, >0.1
##
## Balance tally for mean differences
##                     count
## Balanced, <0.1         27
## Not Balanced, >0.1      4
##
## Variable with the greatest mean difference
##  Variable Diff.Adj        M.Threshold
##    cwwrep   1.4656 Not Balanced, >0.1
##
## Sample sizes
##           Control Treated
## All          2460     298
## Matched       283     283
## Unmatched    2177      15
```

2.6 Greedy Nearest Neighbor Mahalanobis Distance Matching Without Replacement

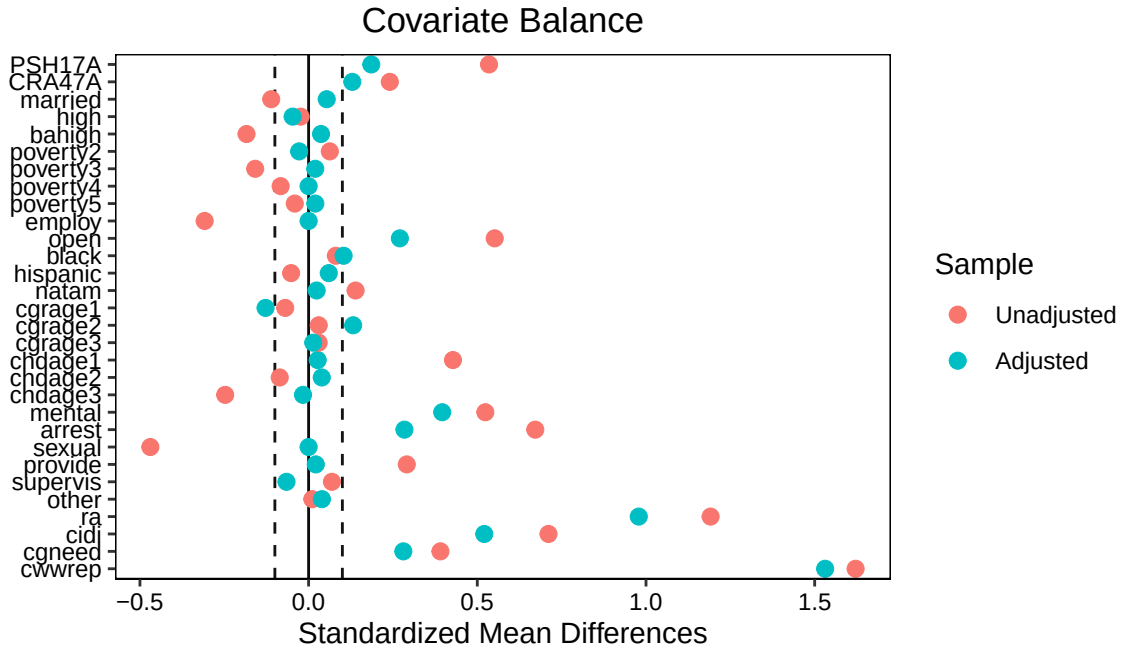
Here we perform Mahalanobis distance matching without replacement and without including an estimated propensity score.

```
set.seed(1000)
(gm_out2 <- MatchIt::matchit(gm_f, data = gm_df,
  method = "nearest", distance = "mahalanobis"))
```

```
## A matchit object
##  - method: 1:1 nearest neighbor matching without replacement
##  - distance: Mahalanobis
##  - number of obs.: 2758 (original), 596 (matched)
##  - target estimand: ATT
##  - covariates: PSH17A, CRA47A, married, high, bahigh, poverty2, poverty3, poverty4, poverty5, employ
```

2.6.1 Check Balance

```
cobalt::love.plot(gm_out2, binary = "std", threshold = c(m = .1),
                  addl = c("ra", "cidi", "cgneed", "cwwrep"), data = gm_df)
```




```
cobalt::bal.tab(gm_out2, binary = "std", threshold = c(m = .1),
                addl = c("ra", "cidi", "cgneed", "cwwrep"), data = gm_df)
```

```
## Call
##  MatchIt::matchit(formula = gm_f, data = gm_df, method = "nearest",
##     distance = "mahalanobis")
##
## Balance Measures
##             Type Diff.Adj        M.Threshold
## PSH17A    Binary   0.1857 Not Balanced, >0.1
## CRA47A    Binary   0.1295 Not Balanced, >0.1
## married   Binary   0.0537     Balanced, <0.1
```

```
## high     Binary  -0.0474     Balanced, <0.1
## bahigh   Binary   0.0368     Balanced, <0.1
## poverty2 Binary  -0.0281     Balanced, <0.1
## poverty3 Binary   0.0197     Balanced, <0.1
## poverty4 Binary   0.0000     Balanced, <0.1
## poverty5 Binary   0.0199     Balanced, <0.1
## employ   Binary   0.0000     Balanced, <0.1
## open     Binary   0.2705 Not Balanced, >0.1
## black    Binary   0.1037 Not Balanced, >0.1
## hispanic Binary   0.0597     Balanced, <0.1
## natam    Binary   0.0238     Balanced, <0.1
## cgrage1  Binary  -0.1276 Not Balanced, >0.1
## cgrage2  Binary   0.1320 Not Balanced, >0.1
## cgrage3  Binary   0.0137     Balanced, <0.1
## chdage1  Binary   0.0269     Balanced, <0.1
## chdage2  Binary   0.0390     Balanced, <0.1
## chdage3  Binary  -0.0166     Balanced, <0.1
## mental   Binary   0.3960 Not Balanced, >0.1
## arrest   Binary   0.2841 Not Balanced, >0.1
## sexual   Binary   0.0000     Balanced, <0.1
## provide  Binary   0.0213     Balanced, <0.1
## supervis Binary  -0.0656     Balanced, <0.1
## other    Binary   0.0394     Balanced, <0.1
## ra       Binary   0.9786 Not Balanced, >0.1
## cidi     Binary   0.5207 Not Balanced, >0.1
## cgneed   Binary   0.2808 Not Balanced, >0.1
## cwwrep   Binary   1.5310 Not Balanced, >0.1
##
## Balance tally for mean differences
##                     count
## Balanced, <0.1         18
## Not Balanced, >0.1     12
##
## Variable with the greatest mean difference
##  Variable Diff.Adj        M.Threshold
##    cwwrep    1.531 Not Balanced, >0.1
##
## Sample sizes
##           Control Treated
## All          2460     298
## Matched       298     298
## Unmatched    2162       0
```

As seen above, balance has not been achieved in multiple covariates. According to Stuart (2010), "the Mahalanobis distance can work quite well when there are relatively few covariates (fewer than 8), but it does not perform as well when the covariates are not normally distributed or there are many covariates."[2]

[2]Stuart, E. A. (2010). Matching Methods for Causal Inference: A Review and a Look Forward. *Statistical Science*, *25*(1), 1–21. https://doi.org/10.1214/09-STS313

3 Propensity Score Weighting

3.1 Load Packages

Propensity score weighting can be accomplished with base R. However, we need the `lmtest` and `sandwich` packages to estimate clustered covariance matrices in this example. Using these packages, we can obtain estimates and standard errors that are identical to Stata's `regress` program.

```
library(lmtest)
library(sandwich)
```

3.2 Description of Dataset

This dataset is from a study that investigates intergenerational dependence on welfare and its relation to child academic achievement.[3]

The dependent variable is `lwss97` or "letter-word identification" score, and the treatment condition is `kuse` or children who used Aid to Families With Dependent Children (AFDC). The covariates are:

- `male`: Child's Gender: Male (Reference: Female)
- `black`: Child's Race: African American (Reference: Other)
- `age97`: Child's Age in 1997
- `pcged97`: Caregiver's Education in 1997 (Years of Schooling)
- `mratio96`: Ratio of Family Income to Poverty Line in 1996

Additionally, `pcg_id` is a cluster variable that identifies children nested within families.

3.3 Estimate ATE and ATT Weights

Separate weights need to be calculated for estimating the average treatment effect (ATE) and the average treatment effect for the treated (ATT).

For ATE, the weight estimates are calculated as follows for the treatment group:

$$\omega = \frac{1}{\hat{e}(x)}$$

And for the control group:

$$\omega = \frac{1}{1 - \hat{e}(x)}$$

For ATT, the weight is 1 for a treated case. The weight for a comparison case is:

$$\omega = \frac{\hat{e}(x)}{1 - \hat{e}(x)}$$

[3]Hofferth, S., Stafford, F. P., Yeung, W. J., Duncan, G. J., Hill, M. S., Lepkowski, J., et al. (2001). *Panel study of income dynamics, 1968–1999: Supplemental files (computer file), ICPSR version.* Ann Arbor: University of Michigan Survey Research Center.

3.4 Load Data with Propensity Scores and Calculate Weights

```
psw_df <- read_dta("data/chpt5_2_original.dta") %>%
  haven::zap_formats() %>%
  sjlabelled::remove_all_labels() %>%
  as_tibble() %>%
  mutate(ate_w = ifelse(kuse == 0, 1/(1 - ps), 1 / ps),
         att_w = ifelse(kuse == 0, ps/(1 - ps), 1))
```

3.5 Calculate Weights with the WeightIt Package

```
# Load Package
library(WeightIt)

# Estimate ATE and ATT weights and Compare with Previous Results
ate_w2 <- WeightIt::get_w_from_ps(ps = psw_df$ps, treat = psw_df$kuse, estimand = "ATE")
table(ate_w2 == psw_df$ate_w)
```

```
##
## TRUE
## 1003
```

```
att_w2 <- WeightIt::get_w_from_ps(ps = psw_df$ps, treat = psw_df$kuse, estimand = "ATT")
table(att_w2 == (psw_df$ate_w * psw_df$ps))
```

```
##
## TRUE
## 1003
```

3.6 Outcome Analysis

3.6.1 Weighted Regression with ATE Weights

After creating the weights, use the `weights` argument in `lm()` to run a weighted outcome analysis and `lmtest::coeftest()` to control for clustering effects.

This analysis showed that children who used Aid to Families With Dependent Children (AFDC) had an average letter-word identification score that was 5.16 points lower than children who never used AFDC, $p < .01$.

```
psw_ate <- lm(lwss97 ~ kuse + male + black + age97 + pcged97 + mratio96,
  data = psw_df, weights = ate_w)
lmtest::coeftest(psw_ate, vcov. = vcovCL(psw_ate, cluster = psw_df$pcg_id))
```

```
##
## t test of coefficients:
##
##             Estimate Std. Error t value  Pr(>|t|)
## (Intercept) 84.19992    4.83032 17.4315 < 2.2e-16 ***
## kuse        -5.16399    1.42438 -3.6254 0.0003031 ***
## male        -1.62201    1.09186 -1.4855 0.1377180
## black       -2.49898    1.34670 -1.8556 0.0638009 .
## age97        0.73868    0.18075  4.0867 4.727e-05 ***
## pcged97      0.99264    0.35596  2.7886 0.0053938 **
## mratio96     1.13856    0.32220  3.5337 0.0004286 ***
## ---
```

```
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

3.6.2 Weighted Regression with ATT Weights

When considering only individuals assigned to the treatment condition, children who used AFDC had an average letter-word identification score that was 4.62 points lower than children who never used AFDC, $p < .01$.

```
psw_att <- lm(lwss97 ~ kuse + male + black + age97 + pcged97 + mratio96,
  data = psw_df, weights = att_w)
lmtest::coeftest(psw_att, vcov. = vcovCL(psw_att, cluster = psw_df$pcg_id))
```

```
##
## t test of coefficients:
##
##             Estimate Std. Error t value  Pr(>|t|)
## (Intercept) 85.29467    5.05331 16.8790 < 2.2e-16 ***
## kuse        -4.62058    1.41182 -3.2728  0.001102 **
## male        -1.58995    1.14705 -1.3861  0.166020
## black       -2.74605    1.41605 -1.9392  0.052756 .
## age97        0.61577    0.20001  3.0787  0.002136 **
## pcged97      0.92698    0.36718  2.5246  0.011738 *
## mratio96     1.26018    0.33556  3.7555  0.000183 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

3.7 Check Balance

To assess balance before and after propensity score weighting, use weighted logistic regression for dummy covariates and weighted simple regression for continuous covariates. Some examples are included below, and the full code can be found in Section 7.3.1 of the PSA-R code.

In model `psw_c3` below, the treatment dummy variable is significant, meaning that there is no sufficient balance after the propensity score weighting.

To assess balance before propensity score weighting, remove the `weights` argument.

```
psw_c1 <- glm(male ~ kuse, family = quasibinomial, data = psw_df, weights = ate_w)
lmtest::coeftest(psw_c1, vcov. = vcovCL(psw_c1, cluster = psw_df$pcg_id))
```

```
##
## z test of coefficients:
##
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)  0.142911   0.075538  1.8919   0.0585 .
## kuse        -0.060271   0.150143 -0.4014   0.6881
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
psw_c2 <- glm(male ~ kuse, family = quasibinomial, data = psw_df, weights = att_w)
lmtest::coeftest(psw_c2, vcov. = vcovCL(psw_c2, cluster = psw_df$pcg_id))
```

```
##
## z test of coefficients:
##
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)  0.152522   0.077410  1.9703   0.0488 *
## kuse        -0.079497   0.147480 -0.5390   0.5899
```

```
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
psw_c3 <- lm(age97 ~ kuse, weights = ate_w, data = psw_df)
lmtest::coeftest(psw_c3, vcov. = vcovCL(psw_c3, cluster = psw_df$pcg_id))
```

```
##
## t test of coefficients:
##
##             Estimate Std. Error t value  Pr(>|t|)
## (Intercept)  6.51389    0.10900 59.7615 < 2.2e-16 ***
## kuse         0.61064    0.21883  2.7905  0.005362 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
psw_c4 <- lm(age97 ~ kuse, weights = att_w, data = psw_df)
lmtest::coeftest(psw_c4, vcov. = vcovCL(psw_c4, cluster = psw_df$pcg_id))
```

```
##
## t test of coefficients:
##
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  6.55501    0.11499 57.0057  < 2e-16 ***
## kuse         0.56178    0.21939  2.5606  0.01059 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Alternatively, balance can be assessed using standardized mean differences:

```
# ATE
cobalt::bal.tab(
  x = select(psw_df, male, black, age97, pcged97, mratio96),
  treat = psw_df$kuse,
  weights = psw_df$ate_w,
  binary = "std",
  continuous = "std",
  s.d.denom = "pooled",
  un = T,
  stats = c("mean.diffs"),
  thresholds = c(m = .1)
)
```

```
## Balance Measures
##             Type Diff.Un Diff.Adj        M.Threshold
## male      Binary -0.0335  -0.0301     Balanced, <0.1
## black     Binary  0.9343   0.8006 Not Balanced, >0.1
## age97    Contin.  0.2196   0.2181 Not Balanced, >0.1
## pcged97  Contin. -0.9067  -0.8244 Not Balanced, >0.1
## mratio96 Contin. -1.0158  -0.8775 Not Balanced, >0.1
##
## Balance tally for mean differences
##                     count
## Balanced, <0.1          1
## Not Balanced, >0.1      4
##
## Variable with the greatest mean difference
##  Variable Diff.Adj        M.Threshold
```

```
##  mratio96  -0.8775 Not Balanced, >0.1
##
## Effective sample sizes
##            Control Treated
## Unadjusted  729.    274.
## Adjusted    725.43  257.71
```

```
# ATT
cobalt::bal.tab(
  x = select(psw_df, male, black, age97, pcged97, mratio96),
  treat = psw_df$kuse,
  weights = psw_df$ate_w,
  binary = "std",
  continuous = "std",
  s.d.denom = "treated",
  un = T,
  stats = c("mean.diffs"),
  thresholds = c(m = .1)
)
```

```
## Balance Measures
##             Type Diff.Un Diff.Adj        M.Threshold
## male      Binary -0.0335  -0.0301     Balanced, <0.1
## black     Binary  1.0129   0.8680 Not Balanced, >0.1
## age97    Contin.  0.2181   0.2167 Not Balanced, >0.1
## pcged97  Contin. -1.0082  -0.9167 Not Balanced, >0.1
## mratio96 Contin. -2.1090  -1.8217 Not Balanced, >0.1
##
## Balance tally for mean differences
##                     count
## Balanced, <0.1          1
## Not Balanced, >0.1      4
##
## Variable with the greatest mean difference
##  Variable Diff.Adj        M.Threshold
##  mratio96  -1.8217 Not Balanced, >0.1
##
## Effective sample sizes
##            Control Treated
## Unadjusted  729.    274.
## Adjusted    725.43  257.71
```

4 Propensity Score Estimation Using Generalized Boosted Regression

4.1 Load Package

Generalized boosted regression (GBR) requires the `gbm` package.

```
library(gbm)
```

4.2 Load Data and Sort

Generalized boosted regression is an iterative method for creating propensity scores. Therefore, to create reproducible results, we need to use the `set.seed()` function.

After importing the data, missing data is deleted listwise, and the data is sorted randomly. According to the `gbm` package vignette, if the data is sorted in a systematic way, then the data should be shuffled before running `gbm`.

```
set.seed(1000)
gbr_df <- read_dta("data/g3aca1.dta") %>%
  haven::zap_formats() %>%
  sjlabelled::remove_all_labels() %>%
  as_tibble() %>%
  select(intbl, ageyc, fmale, blck, whit, hisp, pcedu, ipovl, pcemft, fthr,
         dicsagg2, dicsint2, dccereg2, dccscom2, dccpros2, draggr2) %>%
  drop_na() %>%
  add_column(runif = runif(nrow(.))) %>%
  arrange(runif) %>%
  select(-runif)
```

4.3 Fit Generalized Boosted Regression Model

The `gbm::gbm()` function has many arguments that can be fine-tuned. See `?gbm` for a detailed description of each argument.

A summary of the fitted model provides us with *relative influence*, which is the percentage of log likelihood explained by each input variable. The percentages of influence for all predictor variables sum to 100%.

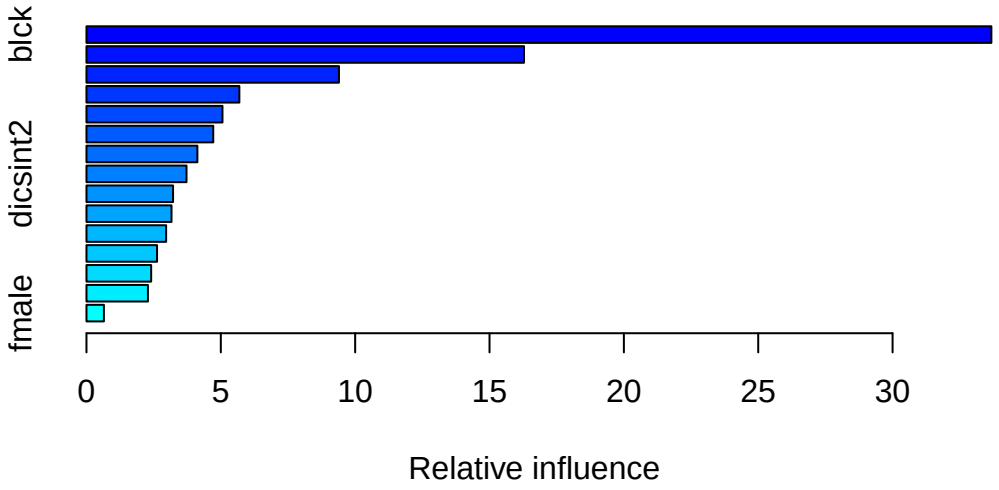
The GBM showed that `blck` had the strongest influence on the likelihood function (33.7%), followed by `ageyc` (16.3%) and `draggr2` (9.4%).

```
(gbr_f <- cobalt::f.build("intbl", select(gbr_df, -intbl)))
```

```
## intbl ~ ageyc + fmale + blck + whit + hisp + pcedu + ipovl +
##     pcemft + fthr + dicsagg2 + dicsint2 + dccereg2 + dccscom2 +
##     dccpros2 + draggr2
## <environment: 0x000000002e1c96e0>
```

```
set.seed(1000)
gbr_m1 <- gbm::gbm(
  formula = gbr_f,
  data = gbr_df,
  distribution = "bernoulli",
  n.trees = 1000, # number of trees to fit
  train.fraction = 0.8, # a random 80% subsample for estimation
  interaction.depth = 4, # allow all four-way interactions
  shrinkage = 0.0005 # small shrinkage to ensure smooth fit
```

```
)
summary(gbr_m1)
```




```
##                var    rel.inf
## blck         blck 33.6768868
## ageyc       ageyc 16.2834103
## draggr2   draggr2  9.3962151
## whit         whit  5.6895119
## ipovl       ipovl  5.0614691
## pcemft     pcemft  4.7179839
## pcedu       pcedu  4.1280063
## dicsint2 dicsint2  3.7223185
## dicsagg2 dicsagg2  3.2228567
## dccscom2 dccscom2  3.1648839
## dccereg2 dccereg2  2.9647178
## dccpros2 dccpros2  2.6269129
## hisp         hisp  2.4079157
## fthr         fthr  2.2874508
## fmale       fmale  0.6494603
```

4.4 Estimate Propensity Scores

After fitting the model, estimate propensity scores using the `predict.gbm()` function.

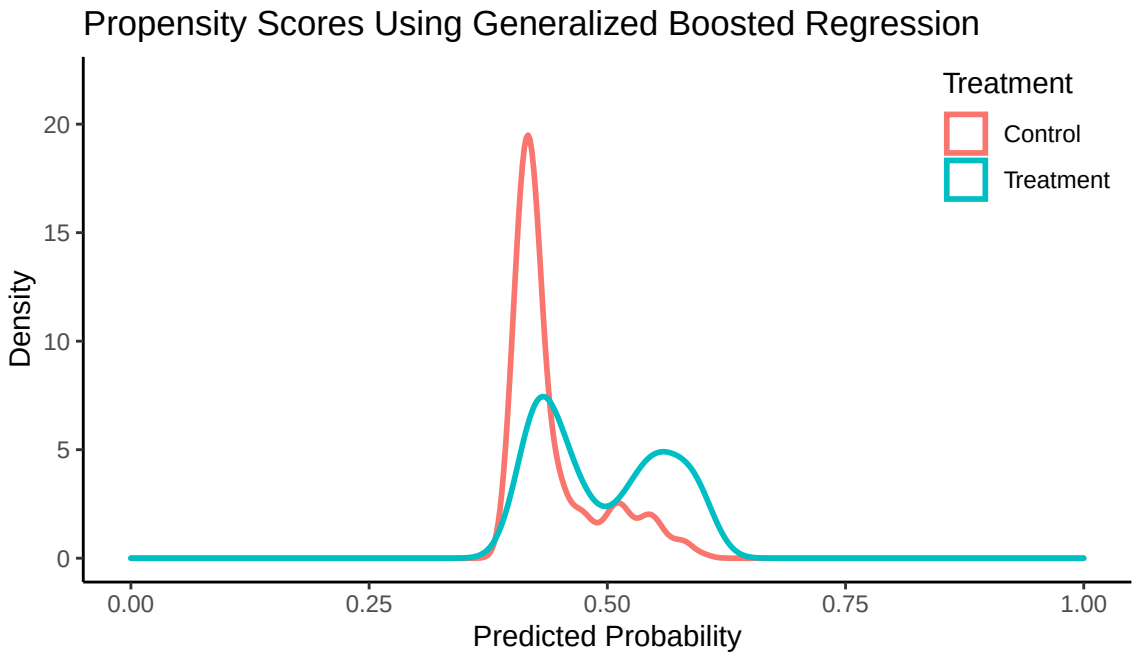
```
psb <- gbm::predict.gbm(gbr_m1, data = gbr_df, type = "response")
head(psb)
```

```
## [1] 0.4441531 0.4405950 0.5288983 0.5958331 0.3985049 0.5562076
```

4.5 Plot Propensity Score Distributions

As seen in the figure below, the propensity scores estimated by GBM has sufficient overlap between the control and treatment groups (i.e., "common support"), and the distributions are similar.

```
gbr_df %>%
  mutate(psb = psb, intbl = factor(intbl, labels = c("Control", "Treatment"))) %>%
  ggplot(aes(x = psb, color = intbl)) + theme_classic() +
  geom_density(size = 1) + xlim(0, 1) + ylim(0, 22) +
  labs(x = "Predicted Probability", y = "Density",
       title = "Propensity Scores Using Generalized Boosted Regression",
       color = "Treatment") +
  theme(legend.position = c(0.9, 0.85))
```



4.6 Summary Statistics of Propensity Scores

```
summary(psb)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  0.3913  0.4189  0.4383  0.4674  0.5202  0.6082
```

4.7 GBR Using the WeightIt Package

As an alternative to the `gbm` package, the `WeightIt` package can also fit GBR models.

```
set.seed(1000)
(gbr_m2 <- WeightIt::weightit(
  formula = gbr_f,
  data = gbr_df,
  method = "gbm",
  estimand = "ATE",
  distribution = "bernoulli",
  stop.method = "es.mean",
  n.trees = 10000, # different
  nTrain = 0.8 * nrow(gbr_df), # different
  interaction.depth = 4,
  shrinkage = 0.0005
))
```

```
## A weightit object
##  - method: "gbm" (propensity score weighting with GBM)
##  - number of obs.: 603
##  - sampling weights: none
##  - treatment: 2-category
##  - estimand: ATE
##  - covariates: ageyc, fmale, blck, whit, hisp, pcedu, ipovl, pcemft, fthr, dicsagg2, dicsint2, dccer
```

```
# Propensity Scores
head(gbr_m2$ps)
```

```
## [1] 0.3781517 0.6075811 0.6596871 0.7766516 0.2138153 0.7927817
```

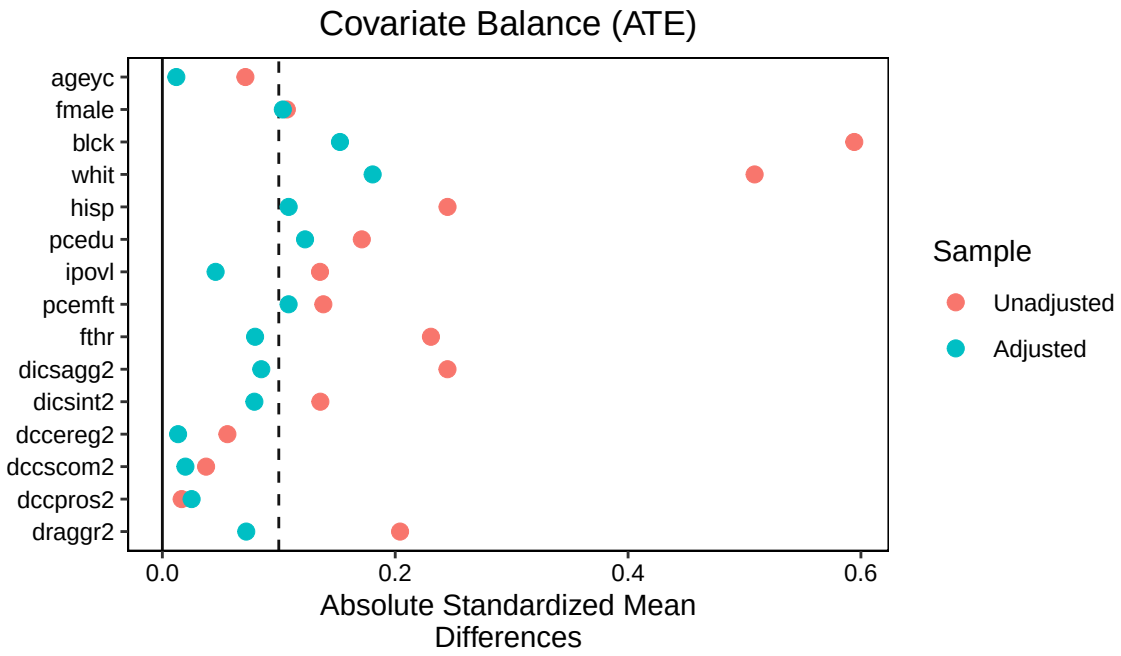
```
# ATE Weights
head(gbr_m2$weights)
```

```
## [1] 2.644442 1.645871 1.515870 1.287579 1.271966 1.261381
```

4.7.1 Check Balance

Using a standardized mean difference cut-off point of 0.1, it can be seen below that balance has been achieved in most, but not all, of the covariates:

```
cobalt::love.plot(gbr_m2,
  thresholds = c(m = .1),
  binary = "std", abs = T, drop.distance = T
) +
  labs(title = "Covariate Balance (ATE)")
```



5 Matching Estimators

5.1 Load Packages

A variety of matching estimators are implemented in the `Matching` package. Because the assumptions about constant treatment effect and homoskedasticity may not be valid for certain types of data, we will also import the `lmtest` package for the Breusch-Pagan Test to check this assumption.

```
library(Matching)
library(lmtest)
library(broom)
select <- dplyr::select
```

Note that the `Matching::Match()` function is intended to be used in conjunction with the `Matching::MatchBalance()` function. However, functions from the `cobalt` package also work well and tend to be cleaner in presentation:[4]

```
data(lalonde)
ex_f <- as.formula(treat ~ age + I(age^2) + educ + I(educ^2) + black +
  hisp + married + nodegr + re74 + I(re74^2) + re75 + I(re75^2) +
  u74 + u75)
ex_m1 <- glm(ex_f, family = binomial, data = lalonde)
ex_p <- ex_m1$fitted.values
ex_X <- ex_m1$fitted
ex_Y <- lalonde$re78
ex_Tr <- lalonde$treat
ex_rr <- Match(Y = ex_Y, Tr = ex_Tr, X = ex_X, M = 1)
summary(ex_rr)
ex_mb <- MatchBalance(treat ~ age + I(age^2) + educ + I(educ^2) + black +
  hisp + married + nodegr + re74 + I(re74^2) + re75 + I(re75^2) +
  u74 + u75, data = lalonde, match.out = ex_rr, nboots = 10)
(ex_bal <- cobalt::bal.tab(ex_rr, ex_f,
  data = lalonde, distance = ~ex_p, un = T,
  binary = "std", threshold = .1
))
cobalt::love.plot(ex_bal)
```

5.2 Description of Dataset

This example uses the 1997 Child Development Supplement (CDS) to the Panel Study of Income Dynamics (PSID) and the core PSID annual data from 1968 to 1997.

The dependent variable in this dataset is `pcss97`, a passage comprehension score. Higher scores on this measure indicate higher academic achievement. The treatment variable is `kuse` or children who ever used Aid to Families With Dependent Children (AFDC). The covariates or matching variables are:

- `pcg_adc`: Caregiver's History of Using Welfare (Number of Years; range: 0-7)
- `age97`: Child's Age in 1997
- `mratio96`: Ratio of Family Income to Poverty Line in 1996
- `pcged97`: Caregiver's Education in 1997 (Years of Schooling)
- `male`: Child's Gender: Male (1 = Male; 0 = Female)
- `black`: Child's Race: African American (1 = African American; 0 = Other)

[4] https://cran.r-project.org/web/packages/cobalt/vignettes/cobalt_A1_other_packages.html#using-bal.tab-with-matching

5.3 Load Data

```
me_df <- read_dta("data/cds_pcss97.dta") %>%
  haven::zap_formats() %>%
  sjlabelled::remove_all_labels() %>%
  as_tibble()
head(me_df) %>%
  kbl(booktabs = T, linesep = "", digits = 2) %>%
  kable_styling(position = "center") %>%
  kable_styling(latex_options = c("striped", "hold_position"))
```

kid	pcg_id	age97	pcss97	k_adc	pcged97	pcg_adc	mratio96	black	male	kuse
4180	4179	12	93	7.69	9	0	0.41	0	0	1
5032	5170	12	122	0.00	12	0	5.00	0	0	0
7041	7030	6	93	0.00	11	2	1.07	0	1	0
10033	10177	7	101	8.33	12	0	2.40	1	0	1
10034	10006	6	127	0.00	14	0	1.00	1	0	0
14030	14172	11	103	0.00	12	0	3.77	0	1	0

5.4 Breusch-Pagan Test for Heteroskedasticity

The homoskedastic variance estimator assumes that the unit-level treatment effect is constant and that the conditional variance of $Y_i(w)$ given X_i does not vary with either covariates or the treatment.

To carry out the Breusch-Pagan Test, first we regress the outcome variable on the matching variables using OLS:

```
# Regress outcome on treatment and matching variables using OLS
me_m1 <- lm(pcss97 ~ kuse + male + black + age97 + pcged97 + mratio96 + pcg_adc,
  data = me_df
)
```

Next, we can run the Breusch-Pagan test for each matching variable:

```
lmtest::bptest(me_m1, ~kuse, data = me_df, studentize = F)
lmtest::bptest(me_m1, ~male, data = me_df, studentize = F)
lmtest::bptest(me_m1, ~black, data = me_df, studentize = F)
lmtest::bptest(me_m1, ~age97, data = me_df, studentize = F) # significant
lmtest::bptest(me_m1, ~pcged97, data = me_df, studentize = F)
lmtest::bptest(me_m1, ~mratio96, data = me_df, studentize = F)
lmtest::bptest(me_m1, ~pcg_adc, data = me_df, studentize = F)
```

Or use a function to test every variable:

```
bp <- function(var, df, md) {
  lmtest::bptest(md, as.formula(paste0("~", var)), data = df, studentize = F) %>%
    broom::tidy() %>%
    mutate(variable = var) %>%
    select(variable, statistic, p.value)
}
map_dfr(c("kuse", "male", "black", "age97", "pcged97", "mratio96", "pcg_adc"), bp,
  df = me_df, md = me_m1
) %>%
  kbl(
    booktabs = T, linesep = "", digits = 2,
```

```
  caption = "Results of Breusch-Pagan Tests for Heteroskedasticity"
) %>%
kable_styling(position = "center") %>%
kable_styling(latex_options = c("striped", "hold_position"))
```

Table 1: Results of Breusch-Pagan Tests for Heteroskedasticity

variable	statistic	p.value
kuse	0.01	0.92
male	3.07	0.08
black	1.43	0.23
age97	25.27	0.00
pcged97	0.19	0.66
mratio96	0.62	0.43
pcg_adc	0.29	0.59

Results from the Breusch-Pagan tests showed that the homoskedasticity assumption is not valid for child's age (`age97`) ($p < .05$) and indicated that the conditional variance of the outcome variable was not constant across levels of child's age. Based on this finding, we should use the robust variance estimator that allows for heteroskedasticity (i.e., the `Var.calc` argument in the `Matching:Match()` function).

5.5 Matching Estimators

Of the six matching variables in this example, four are continuous and two are categorical, therefore bias-corrected matching estimator is necessary to correct for bias corresponding to the matching discrepancies between matched units and their matches on the four continuous covariates. Regression adjustment can be used with the `BiasAdjust = T` argument. (Tip: When you have one or more continuous covariate in your matching, always use the bias-corrected matching estimator.)

By default, the `Matching::Match()` function uses the matching variables to make bias adjustments. However, these covariates can be specified using the `Z` argument (example shown below).

The `M` argument specifies the number of matches which should be found. The default is one-to-one matching (i.e., M = 1). Abadie and Imbens suggest that M be small, and `M = 4` typically performs well in terms of mean-squared error.

If `Var.calc = 0`, then homoskedasticity is assumed. Use `Var.calc = 4` to request the robust variance estimator using four matches. This algorithm developed by Abadie and Imbens (2002) includes a second matching procedure such that it matches treated units to treated units and control units to controls.

The `estimand` argument is by default "ATT," but can be set to "ATE" or "ATC". Typically, we are interested in the ATE or ATT.

The `sample` argument is a logical flag for whether the population or sample variance is returned. An example may help illustrate the difference between PATE and SATE: "While the SATE is useful for judging how a job-training program has affected a particular group of participants, the PATE can be used to evaluate whether another group of participants drawn from the same population is likely to benefit from the program."[5] In other words, the sample effect shows whether the program is successful in the sample at hand, while the population effect shows whether the same program would be successful in a second sample from the population.

Results from the `Matching::Match()` function are identical to Stata's `nnmatch` program.

[5] https://journals.sagepub.com/doi/pdf/10.1177/1536867X0400400307

5.6 Define Outcome (Y), Treatment Index (Tr), and Variables to Match On (X)

Use the `Y`, `Tr`, and `X`, arguments in the `Match()` function to specify the outcome, treatment index, and variables to match on, respectively.

```
me_Y <- me_df$pcss97
me_Tr <- me_df$kuse
me_X <- select(me_df, male, black, age97, pcged97, mratio96, pcg_adc)
```

5.7 Get Estimators Individually

Note that by default matching is done with replacement. However, this can be changed with the `replace` argument.

```
# Sample Average Treatment Effect (SATE)
me1 <- Match(Y = me_Y, Tr = me_Tr, X = me_X, M = 4, BiasAdjust = T, Var.calc = 4,
             estimand = "ATE", sample = T)
summary(me1)

# Population Average Treatment Effect (PATE)
summary(Match(Y = me_Y, Tr = me_Tr, X = me_X, M = 4, BiasAdjust = T, Var.calc = 4,
             estimand = "ATE", sample = F))

# Sample average treatment effect for the treated (SATT)
summary(Match(Y = me_Y, Tr = me_Tr, X = me_X, M = 4, BiasAdjust = T, Var.calc = 4,
             estimand = "ATT", sample = T))

# Population average treatment effect for the treated (PATT)
summary(Match(Y = me_Y, Tr = me_Tr, X = me_X, M = 4, BiasAdjust = T, Var.calc = 4,
             estimand = "ATT", sample = F))

# Sample average treatment effect for the controls (SATC)
summary(Match(Y = me_Y, Tr = me_Tr, X = me_X, M = 4, BiasAdjust = T, Var.calc = 4,
             estimand = "ATC", sample = T))

# Population average treatment effect for the controls (PATC)
summary(Match(Y = me_Y, Tr = me_Tr, X = me_X, M = 4, BiasAdjust = T, Var.calc = 4,
             estimand = "ATC", sample = F))
```

5.8 Get All Estimators

```
# Function for extracting estimate, SE, t-stat, and p-value from Match()
get_match <- function(estimand, sample, Y, Tr, X) {
  m <- Matching::Match(
    Y = Y, Tr = Tr, X = X, M = 4, BiasAdjust = T, Var.calc = 4,
    estimand = estimand, sample = sample
  )
  return(list(
    est = m$est[, 1],
    se = m$se,
    t.stat = m$est[, 1] / m$se,
    p = (1 - pnorm(abs(m$est[, 1] / m$se))) * 2
  ))
}
```

```r
# Estimate different matching estimators
tribble(
  ~estimator, ~estimand, ~sample,
  "SATE", "ATE", T,
  "PATE", "ATE", F,
  "SATT", "ATT", T,
  "PATT", "ATT", F,
  "SATC", "ATC", T,
  "PATC", "ATC", F
) %>%
  rowwise() %>%
  mutate(match = list(get_match(estimand, sample, me_Y, me_Tr, me_X))) %>%
  unnest_wider(match) %>%
  select(-estimand, -sample) %>%
  kbl(
    booktabs = T, linesep = "",
    caption = "Bias-Corrected Matching Estimators with Robust Standard Errors"
  ) %>%
  kable_styling(position = "center") %>%
  kable_styling(latex_options = c("striped", "hold_position"))
```

Table 2: Bias-Corrected Matching Estimators with Robust Standard Errors

estimator	est	se	t.stat	p
SATE	-4.703773	1.769696	-2.657956	0.0078616
PATE	-4.703773	1.765187	-2.664746	0.0077047
SATT	-5.229651	1.781217	-2.935999	0.0033248
PATT	-5.229651	1.720590	-3.039451	0.0023701
SATC	-4.467254	2.133536	-2.093827	0.0362754
PATC	-4.467254	2.135647	-2.091757	0.0364602

The results suggest that childhood poverty strongly affected children's academic achievement.

ATE: On average, children who used AFDC in childhood had a passage comprehension score 4.7 units lower than that of children who had never used AFDC in childhood. This effect is statistically significant in the sample at hand ($p < .05$) as well as in a second sample drawn from the same population ($p < .05$).

ATT: With regard to the subpopulation of treated participants, on average, children who used AFDC in childhood had a passage comprehension score 5.2 units lower than that of children who had never used AFDC in childhood. This effect is statistically significant in the sample at hand ($p < .05$) as well as in a second sample drawn from the same population ($p < .05$).

ATC: Had all controls (i.e., children who never used AFDC) used AFDC and all treated children had not used AFDC, then on average, the control children would have a passage comprehension score 4.5 units lower than their counterparts ($p < .05$ for SATC and $p < .05$ for PATC).

Additional observations:

1. A population effect indicates whether the tested intervention will be effective in a second sample taken from the same population. Taking SATT and PATT as examples, the study indicated that the treatment effect for the treated group was statistically significant in the sample at a level of .01. If we take a second sample from the population, we are likely to observe the same level of treatment effect for the treated, and the effect should remain statistically significant at a level of .01. The point estimate of the population effect is identical to the point estimate of its corresponding sample effect. A population

effect differs from its corresponding sample effect on variance, and thus significance test on a population effect may have a different conclusion than that on its corresponding sample effect.

2. Note that in this study, SATT = -5.23 and SATC = -4.47, or a difference of 0.76 units. This difference is attributable either to additional selection bias that was not accounted for in the study or to study data that violated assumptions of matching estimators, which suggests the need for further scrutiny.

3. All six treatment effects were statistically significant ($p < .05$). Thus, we can conclude that the study data could not reject a null hypothesis of a zero treatment effect, and childhood poverty appears to be an important factor causing children's poor achievement in passage comprehension.

5.9 Specify Variables in the Bias-Corrected Matching

The `Z` argument can be used to specify the covariates for which we wish to make bias adjustments.

```
# Sample Average Treatment Effect (SATE)
me_Z <- select(me_df, age97, pcged97, mratio96, pcg_adc)
summary(Matching::Match(
  Y = me_Y, Tr = me_Tr, X = me_X, Z = me_Z, M = 4,
  BiasAdjust = T, Var.calc = 4, estimand = "ATE", sample = T
))
```

```
##
## Estimate...  -4.4867
## AI SE......  1.7697
## T-stat.....  -2.5353
## p.val......  0.011235
##
## Original number of observations..............  606
## Original number of treated obs...............  188
## Matched number of observations...............  606
## Matched number of observations  (unweighted).  2441
```

5.10 Check Balance

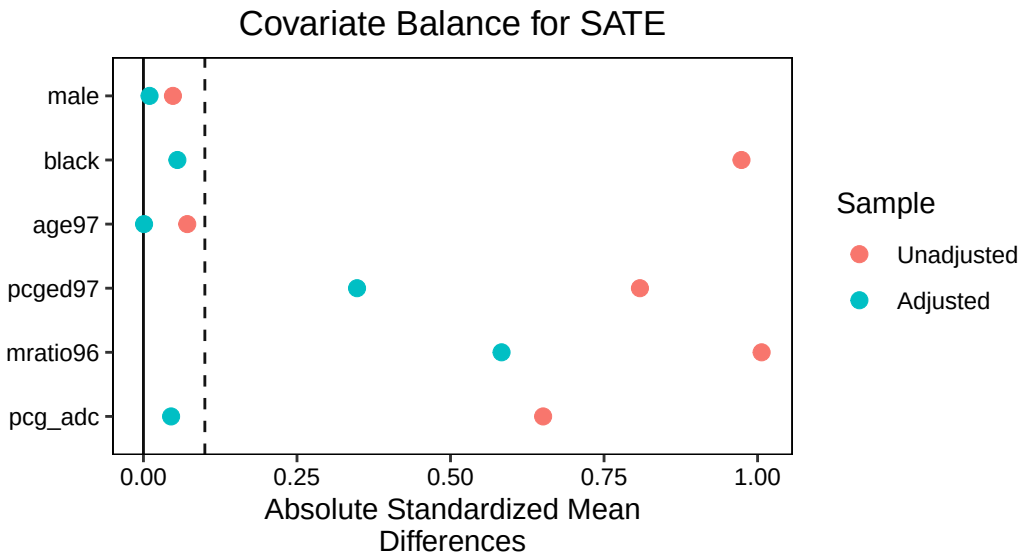
The `Matching::Match()` function works well in conjunction with the `cobalt::bal.tab()` function for checking covariate balance.

By default, the denominator for standardized mean differences uses a pooled estimate (square root of the average of the group variances) for ATE and the standard deviation of the treated group for ATT, and both standard deviations are computed using the sample before matching. This option can also be manually set with the `s.d.denom` option.

```
# Example of Checking Balance for SATE
me_SATE <- Match(
  Y = me_Y, Tr = me_Tr, X = me_X, M = 4, BiasAdjust = T, Var.calc = 4,
  estimand = "ATE", sample = T
)
(me_SATE_bal <- bal.tab(
  me_SATE, kuse ~ male + black + age97 + pcged97 + mratio96 + pcg_adc,
  data = me_df,
  abs = T,
  un = T,
  binary = "std",
  thresholds = c(m = .1),
  s.d.denom = "pooled"
))
```

```
## Balance Measures
##             Type Diff.Un Diff.Adj        M.Threshold
## male      Binary  0.0480   0.0099     Balanced, <0.1
## black     Binary  0.9738   0.0553     Balanced, <0.1
## age97    Contin.  0.0713   0.0010     Balanced, <0.1
## pcged97  Contin.  0.8084   0.3478 Not Balanced, >0.1
## mratio96 Contin.  1.0066   0.5831 Not Balanced, >0.1
## pcg_adc  Contin.  0.6508   0.0450     Balanced, <0.1
##
## Balance tally for mean differences
##                     count
## Balanced, <0.1          4
## Not Balanced, >0.1      2
##
## Variable with the greatest mean difference
##  Variable Diff.Adj        M.Threshold
##  mratio96   0.5831 Not Balanced, >0.1
##
## Sample sizes
##                       Control Treated
## All                    418.    188.
## Matched (ESS)          318.46   85.38
## Matched (Unweighted)   418.    188.
```

```
love.plot(me_SATE_bal) + labs(title = "Covariate Balance for SATE")
```


Covariate Balance for SATE
male
black
age97
pcged97
mratio96
pcg_adc
0.00
0.25
0.50
0.75
1.00
Absolute Standardized Mean Differences
Sample
Unadjusted
Adjusted

6 Practice Problems

6.1 Practice 1, Problem 1: Generalized Boosted Regression

6.1.1 Description of Dataset

This dataset is a subset of the experimental dataset used by LaLonde (1986).[6] The LaLonde study is very famous among observational researchers. LaLonde is one of few pioneering researchers who used experimental data to cross-validate estimates of treatment effects generated by nonexperimental approaches. In this study, LaLonde's original dataset was created by a randomized experiment—a study examining trainee earnings of an employment program where participants were randomly assigned to treatment and control conditions. LaLonde's study compares the "true" treatment effect from this randomized experiment to a set of estimates generated by nonexperimental approaches. His study shows that "many of the nonexperimental procedures do not replicate the experimentally determined results, and suggests that researchers should be aware of the potential for specification errors in other nonexperimental evaluations" (LaLonde, 1986, p.604).

In this example, we are interested in the effect of participation in a job-training program on individuals' earnings in 1978. Thus, the dependent variable is `re78` or earnings in 1978 (in thousands of 1978 dollars). The binary treatment variable is `t` or participation in a job-training program (1 = treated; 0 = control). The observable covariates are:

- `age`: Age (in years)
- `educ`: Years of education
- `black`: African-American
- `hisp`: Hispanic
- `married`: Married
- `u74`: Unemployed in 1974
- `u75`: Unemployed in 1975
- `re74`: Earnings in 1974 (in thousands of 1978 dollars)
- `re75`: Earnings in 1975 (in thousands of 1978 dollars)

6.1.2 Load and Sort Data

```
set.seed(1000)
p11_df <- read_dta("data/ldw_exper.dta") %>%
  haven::zap_formats() %>%
  sjlabelled::remove_all_labels() %>%
  as_tibble() %>%
  add_column(runif = runif(nrow(.))) %>%
  arrange(runif) %>%
  select(-runif)
```

6.1.3 Estimate Propensity Scores

```
set.seed(1000)
p11_m1 <- gbm::gbm(
  formula = t ~ age + educ + black + hisp + married + re74 + re75 + u74 + u75,
  data = p11_df,
  distribution = "bernoulli",
  n.trees = 1000,
  train.fraction = 0.8,
  interaction.depth = 4,
  shrinkage = 0.0005
```

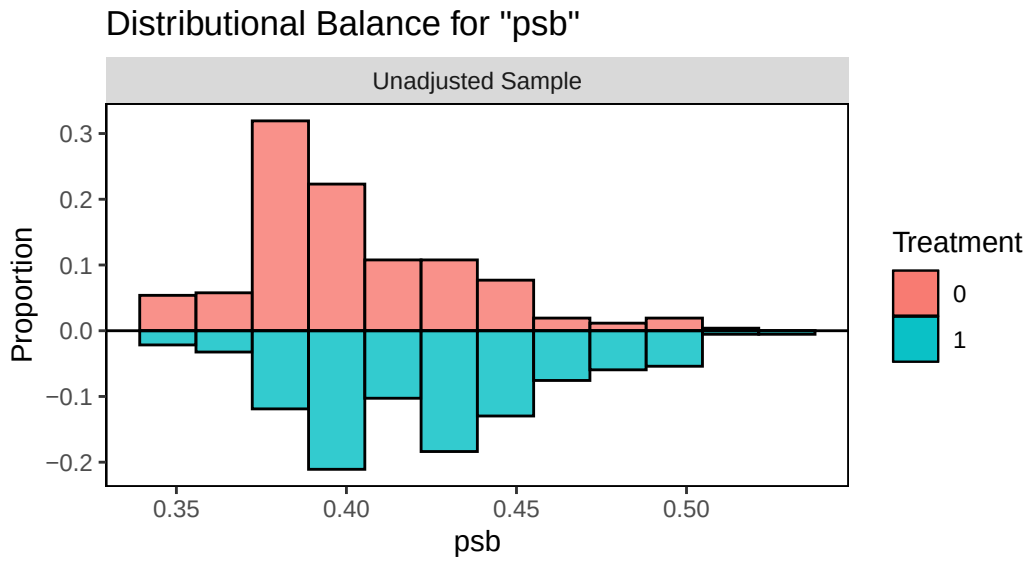
[6] LaLonde, R. J. (1986). Evaluating the Econometric Evaluations of Training Programs with Experimental Data. *The American Economic Review*, *76*(4), 604–620. JSTOR.

```
)

# Estimate Propensity Scores and Obtain Summary Statistics
p11_df$psb <- gbm::predict.gbm(p11_m1, data = p11_df, type = "response")
summary(p11_df$psb)
```

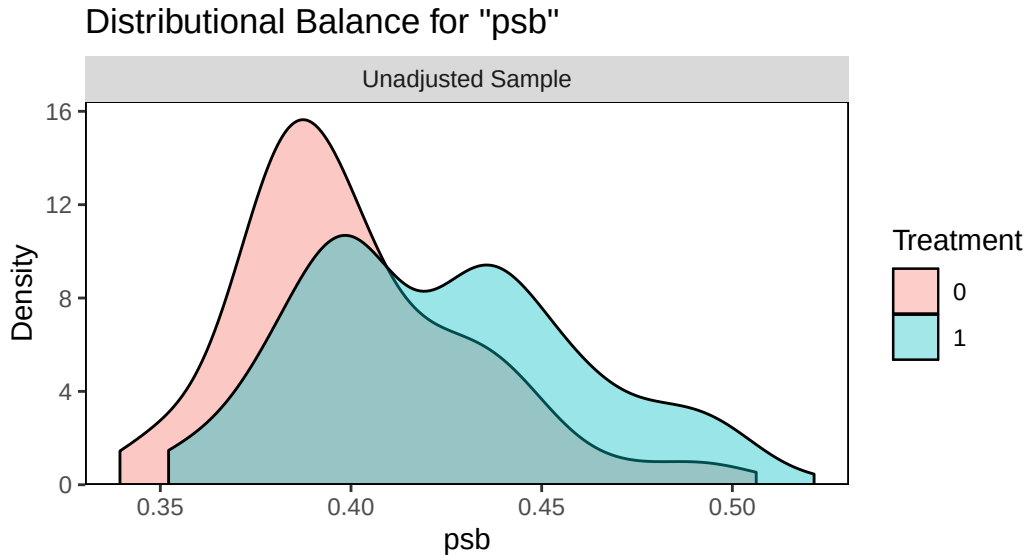
```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  0.3394  0.3855  0.4006  0.4108  0.4346  0.5214
```

6.1.4 Histogram and Density Plots of Propensity Scores

```
cobalt::bal.plot(
  t ~ psb,
  data = p11_df,
  var.name = "psb",
  type = "histogram",
  mirror = T
)
```

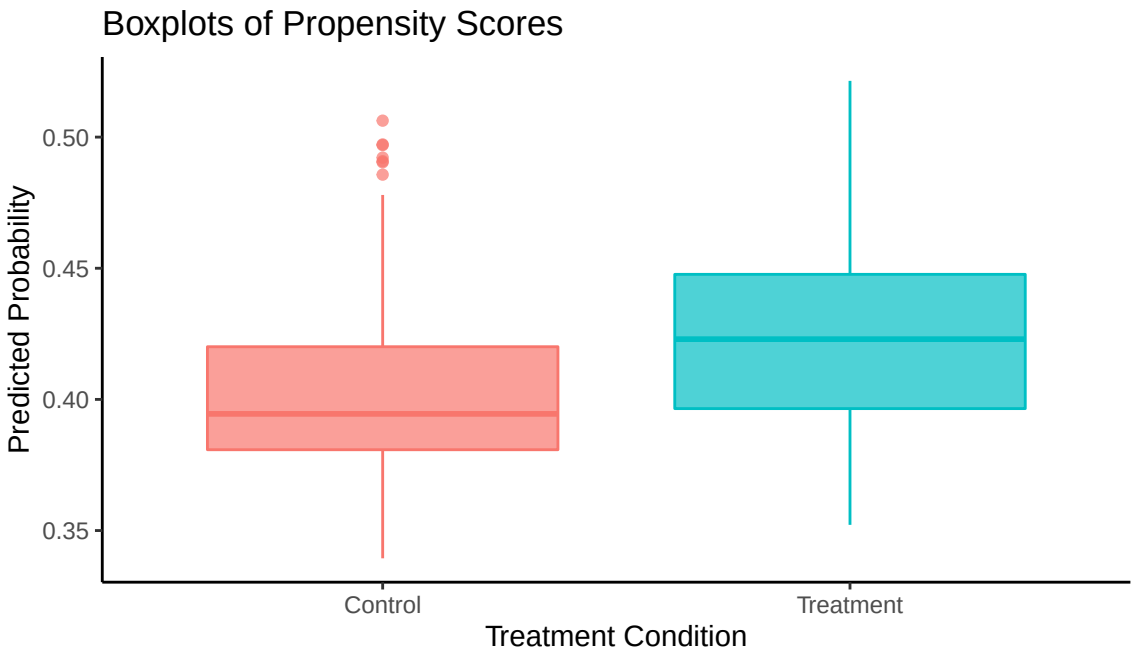



```
cobalt::bal.plot(
  t ~ psb,
  data = p11_df,
  var.name = "psb"
)
```



6.1.5 Boxplot of Propensity Scores

```
p11_df %>%
  mutate(t = factor(t, labels = c("Control", "Treatment"))) %>%
  ggplot(aes(x = t, y = psb, color = t, fill = t)) +
  theme_classic() +
  geom_boxplot(alpha = 0.7) +
  labs(x = "Treatment Condition",
       y = "Predicted Probability",
       title = "Boxplots of Propensity Scores") +
  theme(legend.position = "none")
```



6.2 Practice 1, Problem 2: Propensity Score Weighting

6.2.1 Import Stata-Generated Weights for Comparison

```
stata_weights <- read_dta("data/ldw1.dta") %>%
  haven::zap_formats() %>%
  sjlabelled::remove_all_labels() %>%
  as_tibble() %>%
  mutate(stata_ate_w = ifelse(t == 0, 1 / (1 - psb), 1 / psb),
         stata_att_w = ifelse(t == 0, psb / (1 - psb), 1)) %>%
  select(id, stata_ate_w, stata_att_w)
```

6.2.2 Estimate ATE and ATT Weights

```
p12_df <- p11_df %>%
  mutate(ate_w = ifelse(t == 0, 1 / (1 - psb), 1 / psb),
         att_w = ifelse(t == 0, psb / (1 - psb), 1)) %>%
  left_join(stata_weights, by = "id") # merge with Stata weights
```

6.2.3 Outcome Analysis with ATE and ATT Weights

```
# Define outcome formula
p12_f <- as.formula(re78 ~ t + age + educ + black + hisp +
  married + re74 + re75 + u74 + u75)

# Weighted OLS with R-Generated Propensity Scores
p12_m1 <- lm(p12_f, data = p12_df, weights = ate_w)
tidy(lmtest::coeftest(p12_m1, vcov. = vcovHC(p12_m1, "HC1"))) %>% filter(term == "t")
```

```
## # A tibble: 1 x 5
##   term  estimate std.error statistic p.value
##   <chr>    <dbl>     <dbl>     <dbl>   <dbl>
## 1 t         1.65     0.655      2.52  0.0122
```

```
p12_m2 <- lm(p12_f, data = p12_df, weights = att_w)
tidy(lmtest::coeftest(p12_m2, vcov. = vcovHC(p12_m2, "HC1"))) %>% filter(term == "t")
```

```
## # A tibble: 1 x 5
##   term  estimate std.error statistic p.value
##   <chr>    <dbl>     <dbl>     <dbl>   <dbl>
## 1 t         1.72     0.663      2.60 0.00965
```

```
# Weighted OLS with Stata-Generated Propensity Scores
p12_m1_stata <- lm(p12_f, data = p12_df, weights = stata_ate_w)
tidy(lmtest::coeftest(p12_m1_stata, vcov. = vcovHC(p12_m1_stata, "HC1"))) %>%
  filter(term == "t")
```

```
## # A tibble: 1 x 5
##   term  estimate std.error statistic p.value
##   <chr>    <dbl>     <dbl>     <dbl>   <dbl>
## 1 t         1.63     0.656      2.48  0.0135
```

```
p12_m2_stata <- lm(p12_f, data = p12_df, weights = stata_att_w)
tidy(lmtest::coeftest(p12_m2_stata, vcov. = vcovHC(p12_m2_stata, "HC1"))) %>%
  filter(term == "t")
```

```
## # A tibble: 1 x 5
##   term  estimate std.error statistic p.value
##   <chr>    <dbl>     <dbl>     <dbl>   <dbl>
## 1 t         1.70     0.665      2.55  0.0111
```

6.2.4 Check Balance

6.2.4.1 Hypothesis Tests We can use logistic regression and OLS regression to check the balance of categorical and continuous covariates, respectively:[7]

```
# Categorical Covariates
i1 <- glm(black ~ t, family = quasibinomial, data = p12_df, weights = stata_ate_w)
i2 <- glm(hisp ~ t, family = quasibinomial, data = p12_df, weights = stata_ate_w)
i3 <- glm(married ~ t, family = quasibinomial, data = p12_df, weights = stata_ate_w)
i4 <- glm(u74 ~ t, family = quasibinomial, data = p12_df, weights = stata_ate_w)
i5 <- glm(u75 ~ t, family = quasibinomial, data = p12_df, weights = stata_ate_w)
robustse(i1, coef = "odd.ratio")
robustse(i2, coef = "odd.ratio")
robustse(i3, coef = "odd.ratio")
robustse(i4, coef = "odd.ratio")
robustse(i5, coef = "odd.ratio")

# Continuous Covariates
i6 <- lm(age ~ t, data = p12_df, weights = stata_ate_w)
i7 <- lm(educ ~ t, data = p12_df, weights = stata_ate_w)
i8 <- lm(re74 ~ t, data = p12_df, weights = stata_ate_w)
i9 <- lm(re75 ~ t, data = p12_df, weights = stata_ate_w)
lmtest::coeftest(i6, vcov. = vcovHC(i6, "HC1"))
lmtest::coeftest(i7, vcov. = vcovHC(i7, "HC1"))
lmtest::coeftest(i8, vcov. = vcovHC(i8, "HC1"))
lmtest::coeftest(i9, vcov. = vcovHC(i9, "HC1"))
```

[7] See Appendix A for the custom function `robustse()` that is used to replicate the robust standard errors in Stata.

```r
# Hypothesis testing for categorical covariates
p12_df %>%
  select(black, hisp, married, u74, u75, ate_w:stata_att_w, t) %>%
  pivot_longer(black:u75, names_to = "cat_covs", values_to = "cat_val") %>%
  pivot_longer(ate_w:stata_att_w, names_to = "estimand", values_to = "weight") %>%
  group_by(estimand, cat_covs) %>%
  nest() %>%
  mutate(p.value = map(data, ~robustse(
    glm(.$cat_val ~ .$t, family = quasibinomial, weights = .$weight),
    coef = "odd.ratio")[2,4])) %>%
  unnest(p.value) %>%
  ungroup() %>%
  select(-data)
```

```
## # A tibble: 20 x 3
##    cat_covs estimand    p.value
##    <chr>    <chr>         <dbl>
##  1 black    ate_w         0.896
##  2 black    att_w         0.977
##  3 black    stata_ate_w   0.716
##  4 black    stata_att_w   0.772
##  5 hisp     ate_w         0.222
##  6 hisp     att_w         0.215
##  7 hisp     stata_ate_w   0.127
##  8 hisp     stata_att_w   0.128
##  9 married  ate_w         0.452
## 10 married  att_w         0.458
## 11 married  stata_ate_w   0.402
## 12 married  stata_att_w   0.383
## 13 u74      ate_w         0.534
## 14 u74      att_w         0.492
## 15 u74      stata_ate_w   0.593
## 16 u74      stata_att_w   0.492
## 17 u75      ate_w         0.260
## 18 u75      att_w         0.236
## 19 u75      stata_ate_w   0.241
## 20 u75      stata_att_w   0.181
```

```r
# Hypothesis testing for continuous covariates
p12_df %>%
  select(age, educ, re74, re75, ate_w:stata_att_w, t) %>%
  pivot_longer(age:re75, names_to = "cont_covs", values_to = "cont_val") %>%
  pivot_longer(ate_w:stata_att_w, names_to = "estimand", values_to = "weight") %>%
  group_by(estimand, cont_covs) %>%
  nest() %>%
  mutate(lm = map(data, ~lm(.$cont_val ~ .$t, weights = .$weight)),
         p.value = map(lm, ~lmtest::coeftest(., vcov. = vcovHC(., "HC1"))[2,4])) %>%
  unnest(p.value) %>%
  ungroup() %>%
  select(-data, -lm)
```

```
## # A tibble: 16 x 3
##    cont_covs estimand    p.value
##    <chr>     <chr>         <dbl>
##  1 age       ate_w         0.520
```

```
##  2 age       att_w         0.512
##  3 age       stata_ate_w   0.455
##  4 age       stata_att_w   0.426
##  5 educ      ate_w         0.299
##  6 educ      att_w         0.315
##  7 educ      stata_ate_w   0.356
##  8 educ      stata_att_w   0.302
##  9 re74      ate_w         0.863
## 10 re74      att_w         0.908
## 11 re74      stata_ate_w   0.722
## 12 re74      stata_att_w   0.850
## 13 re75      ate_w         0.612
## 14 re75      att_w         0.585
## 15 re75      stata_ate_w   0.666
## 16 re75      stata_att_w   0.553
```

The `survey` package can also be used to incorporate weights:

```
library(survey)
i.svy <- survey::svydesign(~1, weights = p12_df$stata_ate_w, data = p12_df)
survey::svychisq(~black + t, design = i.svy)
survey::svychisq(~hisp + t, design = i.svy)
survey::svychisq(~married + t, design = i.svy)
survey::svychisq(~u74 + t, design = i.svy)
survey::svychisq(~u75 + t, design = i.svy)
survey::svyttest(age ~ t, design = i.svy)
survey::svyttest(educ ~ t, design = i.svy)
survey::svyttest(re74 ~ t, design = i.svy)
survey::svyttest(re75 ~ t, design = i.svy)
```

6.2.4.2 Standardized Mean Differences

Finally, the `cobalt` package can be used to check balance using standardized mean differences.[8]

```
cobalt::bal.tab(
  p12_df %>% select(black, hisp, married, u74, u75, age, educ, re74, re75),
  treat = p12_df$t,
  weights = p12_df$ate_w,
  abs = T,
  threshold = .1,
  binary = "std",
  s.d.denom = "pooled",
  un = T
)
```

```
## Balance Measures
##            Type Diff.Un Diff.Adj        M.Threshold
## black    Binary  0.0440   0.0126     Balanced, <0.1
## hisp     Binary  0.1749   0.1245 Not Balanced, >0.1
## married  Binary  0.0939   0.0721     Balanced, <0.1
## u74      Binary  0.0944   0.0595     Balanced, <0.1
## u75      Binary  0.1772   0.1084 Not Balanced, >0.1
## age     Contin.  0.1073   0.0615     Balanced, <0.1
## educ    Contin.  0.1412   0.1009 Not Balanced, >0.1
## re74    Contin.  0.0022   0.0162     Balanced, <0.1
## re75    Contin.  0.0839   0.0490     Balanced, <0.1
##
## Balance tally for mean differences
##                     count
## Balanced, <0.1          6
## Not Balanced, >0.1      3
##
## Variable with the greatest mean difference
##  Variable Diff.Adj        M.Threshold
##      hisp   0.1245 Not Balanced, >0.1
##
## Effective sample sizes
##            Control Treated
## Unadjusted  260.    185.
## Adjusted    259.23  183.68
```

[8] "The "effective sample size" (ESS) is a measure of the sample size a non-weighted sample would have to have to achieve the same level of precision as the weighted sample (Ridgeway 2006)" (https://cran.r-project.org/web/packages/cobalt/vignettes/cobalt.html#effective-sample-size-for-weighting)

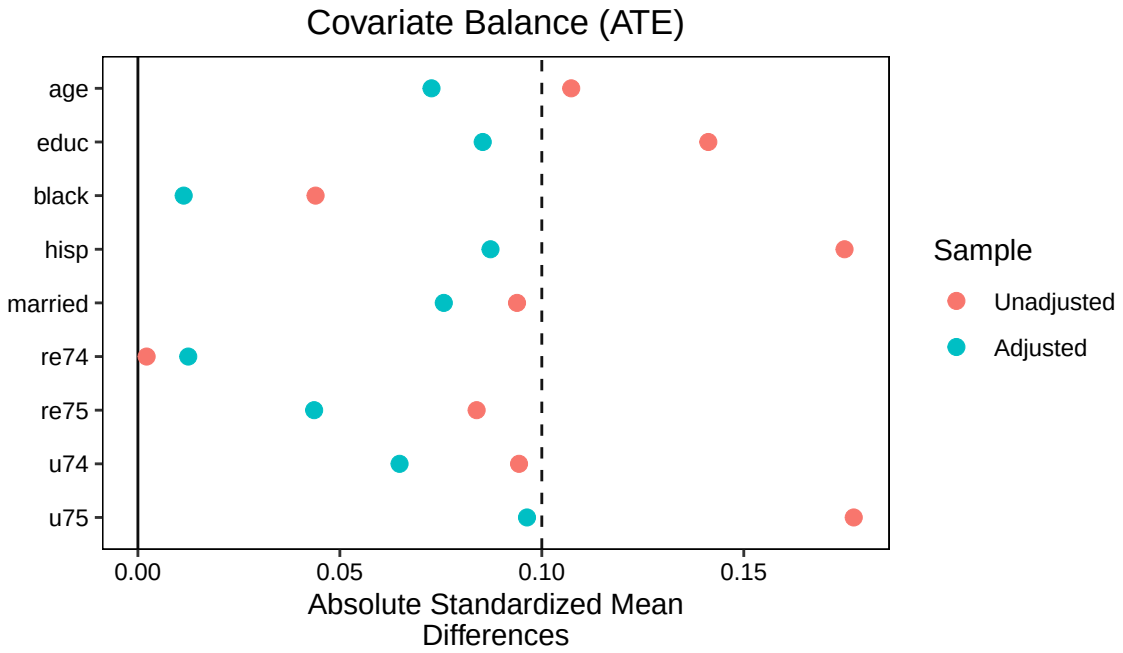
6.3 Practice 1: Alternative Solution with the WeightIt Package

Use the GBM implementation in `WeightIt::weightit()` to estimate ATE and ATT:

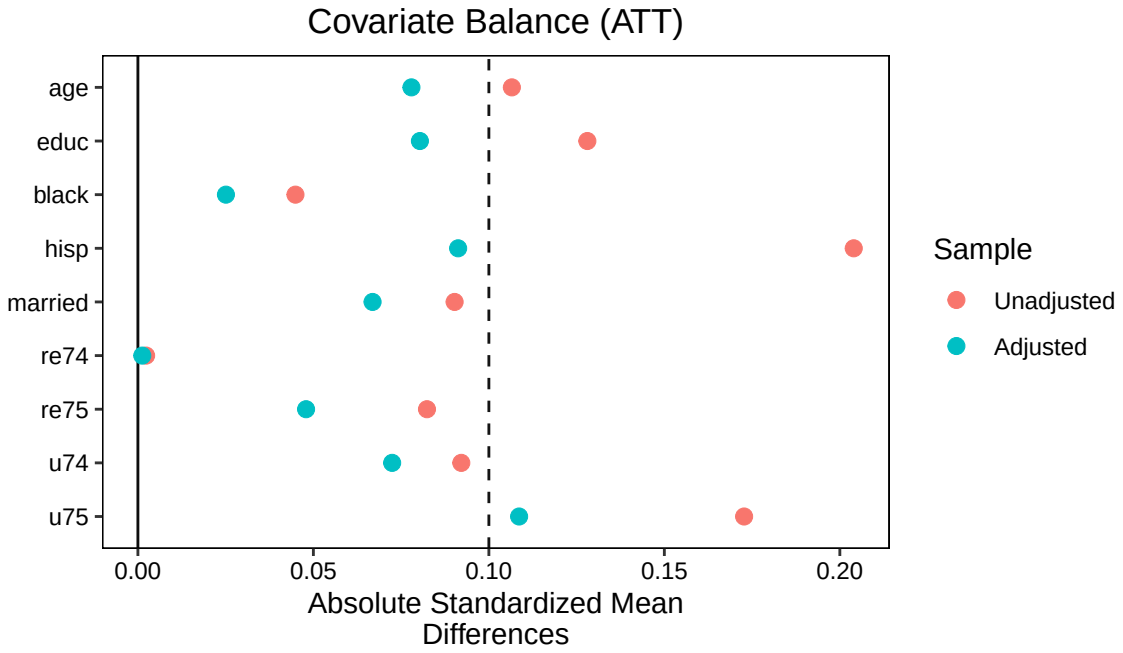
```
set.seed(1000)
p1_ate <- WeightIt::weightit(
  formula = t ~ age + educ + black + hisp + married + re74 +
    re75 + u74 + u75,
  data = p11_df,
  method = "gbm",
  distribution = "bernoulli",
  stop.method = "es.mean",
  n.trees = 1000,
  nTrain = 0.8 * nrow(p11_df),
  interaction.depth = 4,
  shrinkage = 0.0005,
  estimand = "ATE"
)

set.seed(1000)
p1_att <- WeightIt::weightit(
  formula = t ~ age + educ + black + hisp + married + re74 +
    re75 + u74 + u75,
  data = p11_df,
  method = "gbm",
  distribution = "bernoulli",
  stop.method = "es.mean",
  n.trees = 1000,
  nTrain = 0.8 * nrow(p11_df),
  interaction.depth = 4,
  shrinkage = 0.0005,
  estimand = "ATT"
)
```

6.3.1 Check Balance

```
cobalt::love.plot(p1_ate, thresholds = .1, binary = "std", abs = T, drop.distance = T) +
  labs(title = "Covariate Balance (ATE)")
```




```
cobalt::love.plot(p1_att, thresholds = .1, binary = "std", abs = T, drop.distance = T) +
  labs(title = "Covariate Balance (ATT)")
```



6.3.2 Outcome Analysis

For the outcome analysis, the ATE and ATT weights can be obtained with `p1_ate$weights` (ATE) and `p1_att$weights` (ATT):

```
# ATE
p1_m1 <- lm(p12_f, data = p11_df, weights = p1_ate$weights)
tidy(lmtest::coeftest(p1_m1, vcov. = vcovHC(p1_m1, "HC1"))) %>% filter(term == "t")
```

```
## # A tibble: 1 x 5
##   term  estimate std.error statistic p.value
##   <chr>    <dbl>     <dbl>     <dbl>   <dbl>
## 1 t         1.57     0.646      2.43  0.0155
```

```
# ATT
p1_m2 <- lm(p12_f, data = p11_df, weights = p1_att$weightit)
tidy(lmtest::coeftest(p1_m2, vcov. = vcovHC(p1_m2, "HC1"))) %>% filter(term == "t")
```

```
## # A tibble: 1 x 5
##   term  estimate std.error statistic p.value
##   <chr>    <dbl>     <dbl>     <dbl>   <dbl>
## 1 t         1.67     0.662      2.53  0.0119
```

6.4 Practice 2: Matching Estimators

6.4.1 Description of Dataset

This example uses the 1997 Child Development Supplement (CDS) to the Panel Study of Income Dynamics (PSID) and the core PSID annual data from 1968 to 1997.

The dependent variable in this dataset is `lwss97`, a standardized letter-word identification score in 1997 measuring academic achievement. Higher scores on this measure indicate higher academic achievement. The treatment variable is `kuse` or children who ever used Aid to Families With Dependent Children (AFDC). The covariates or matching variables are:

- `pcg_adc`: Caregiver's History of Using Welfare (Number of Years; range: 0-7)
- `age97`: Child's Age in 1997
- `mratio96`: Ratio of Family Income to Poverty Line in 1996
- `pcged97`: Caregiver's Education in 1997 (Years of Schooling)
- `male`: Child's Gender: Male (1 = Male; 0 = Female)
- `black`: Child's Race: African American (1 = African American; 0 = Other)

The research question is: What are the sample and population average impacts of being poor on academic achievement, after correcting for observed selection effects?

6.4.2 Load Data

```
p2_df <- haven::read_dta("data/prac2.dta") %>%
  haven::zap_formats() %>%
  sjlabelled::remove_all_labels() %>%
  as_tibble()
```

6.4.3 Breusch-Pagan Test for Heteroskedasticity

The Breusch-Pagan test fits a linear regression model to the residuals of a linear regression model. A statistically significant result indicates that too much of the variance is explained by the additional explanatory variables.

The homoscedasticity assumption is not valid (e.g., p-value of the test for `age97` is $< .05$), indicating that the conditional variance of the outcome variable was not constant across levels of child's age, therefore a robust estimation of variance is warranted.

```
# Fit a linear regression model, regressing the outcome on the covariates
p2_m1 <- lm(lwss97 ~ kuse + male + black + age97 + pcged97 + mratio96 + pcg_adc,
  data = p2_df
)

# Breusch-Pagan test (one by one)
lmtest::bptest(p2_m1, ~kuse, data = p2_df, studentize = F)
```

```
##
##  Breusch-Pagan test
##
## data:  p2_m1
## BP = 1.7835, df = 1, p-value = 0.1817
```

```
lmtest::bptest(p2_m1, ~male, data = p2_df, studentize = F)
```

```
##
##  Breusch-Pagan test
##
## data:  p2_m1
```

```
## BP = 0.85601, df = 1, p-value = 0.3549
```

```
lmtest::bptest(p2_m1, ~black, data = p2_df, studentize = F)
```

```
##
##  Breusch-Pagan test
##
## data:  p2_m1
## BP = 1.1485, df = 1, p-value = 0.2839
```

```
lmtest::bptest(p2_m1, ~age97, data = p2_df, studentize = F) # significant
```

```
##
##  Breusch-Pagan test
##
## data:  p2_m1
## BP = 8.5467, df = 1, p-value = 0.003462
```

```
lmtest::bptest(p2_m1, ~pcged97, data = p2_df, studentize = F) # significant
```

```
##
##  Breusch-Pagan test
##
## data:  p2_m1
## BP = 4.4312, df = 1, p-value = 0.03529
```

```
lmtest::bptest(p2_m1, ~mratio96, data = p2_df, studentize = F) # significant
```

```
##
##  Breusch-Pagan test
##
## data:  p2_m1
## BP = 6.8509, df = 1, p-value = 0.00886
```

```
lmtest::bptest(p2_m1, ~pcg_adc, data = p2_df, studentize = F)
```

```
##
##  Breusch-Pagan test
##
## data:  p2_m1
## BP = 0.59758, df = 1, p-value = 0.4395
```

```
# Breusch-Pagan test ()
bp <- function(var) {
  lmtest::bptest(p2_m1, as.formula(paste0("~", var)), data = p2_df, studentize = F) %>%
    broom::tidy() %>%
    mutate(variable = var) %>%
    dplyr::select(variable, statistic, p.value)
}
map_dfr(c("kuse", "male", "black", "age97", "pcged97", "mratio96", "pcg_adc"), bp) %>%
  kbl(
    booktabs = T, linesep = "", digits = 2,
    caption = "Results of Breusch-Pagan Tests for Heteroskedasticity"
  ) %>%
  kable_styling(position = "center") %>%
  kable_styling(latex_options = c("striped", "hold_position"))
```

Results from the Breusch-Pagan tests showed that child's age, caregiver's education, and income/poverty ratio

Table 3: Results of Breusch-Pagan Tests for Heteroskedasticity

variable	statistic	p.value
kuse	1.78	0.18
male	0.86	0.35
black	1.15	0.28
age97	8.55	0.00
pcged97	4.43	0.04
mratio96	6.85	0.01
pcg_adc	0.60	0.44

were statistically significant ($p < .05$) and indicated that the conditional variance of the outcome variable was not constant across levels of each variable. Based on this finding, we should use the robust variance estimator that allows for heteroskedasticity (i.e., the `Var.calc` argument in the `Matching:Match()` function).

6.4.4 Define Outcome (Y), Treatment Index (Tr), and Variables to Match On (X)

```
p2_Y <- p2_df$lwss97
p2_Tr <- p2_df$kuse
p2_X <- select(p2_df, male, black, age97, pcged97, mratio96, pcg_adc)
```

6.4.5 Define Function for Matching

By default, the `Matching::Match()` function uses the matching variables to make bias adjustments. However, these covariates can be specified using the `Z` argument.

```
get_match <- function(estimand, sample, Y, Tr, X) {
  m <- Matching::Match(
    Y = Y, Tr = Tr, X = X, M = 4, BiasAdjust = T, Var.calc = 4,
    estimand = estimand, sample = sample
  )
  return(list(
    est = m$est[, 1],
    se = m$se,
    t.stat = m$est[, 1] / m$se,
    p = (1 - pnorm(abs(m$est[, 1] / m$se))) * 2
  ))
}
```

6.4.6 Get All Estimators

The outcome analysis of matching estimators is the difference in means between the treated and control group after matching. No covariates are controlled for in the estimation of the treatment effect.

```
tribble(
  ~estimator, ~estimand, ~sample,
  "SATE", "ATE", T,
  "PATE", "ATE", F,
  "SATT", "ATT", T,
  "PATT", "ATT", F,
  "SATC", "ATC", T,
  "PATC", "ATC", F
) %>%
  rowwise() %>%
  mutate(match = list(get_match(estimand, sample, p2_Y, p2_Tr, p2_X))) %>%
  tidyr::unnest_wider(match) %>%
  select(-estimand, -sample) %>%
  kbl(booktabs = T, linesep = "") %>%
  kable_styling(position = "center") %>%
  kable_styling(latex_options = c("striped", "hold_position"))
```

estimator	est	se	t.stat	p
SATE	-5.448863	1.646936	-3.3084850	0.0009380
PATE	-5.448863	1.652232	-3.2978811	0.0009742
SATT	-1.277287	1.683284	-0.7588067	0.4479682
PATT	-1.277287	1.695820	-0.7531973	0.4513314
SATC	-7.016781	1.965677	-3.5696503	0.0003575
PATC	-7.016781	1.969424	-3.5628594	0.0003668

The findings suggest that childhood poverty affected children's academic achievement. On average, children who used AFDC in childhood had a letter-word identification score 5.4 units lower than that of children who had never used AFDC in childhood. This effect is statistically significant in the sample at hand ($p < .05$) as well as in a second sample drawn from the same population ($p < .05$). By design, the point estimates of SATE and PATE are identical, but the standard errors are different.

With regard to the subpopulation of treated participants, the treatment effect was not statistically significant ($p = .448$ for SATT and $p = .451$ for PATT). Despite the non-significant finding, the direction of the estimates are consistent with ATE and ATC.

Had all controls (i.e., children who never used AFDC) used AFDC and all treated children had not used AFDC, then on average, the control children would have a letter-word identification score 7.02 units lower than their counterparts ($p < .05$ for SATC and $p < .05$ for PATC).

6.4.7 Compare with Propensity Score Weighting Results

The conclusions reached above are consistent with the results reached via propensity score weighting.

```
# ATE
psw_ate <- lm(lwss97 ~ kuse + male + black + age97 + pcged97 + mratio96,
  data = psw_df, weights = ate_w)
lmtest::coeftest(psw_ate, vcov. = vcovCL(psw_ate, cluster = psw_df$pcg_id))
```

```
##
## t test of coefficients:
##
```

```
##             Estimate Std. Error t value  Pr(>|t|)
## (Intercept) 84.19992    4.83032 17.4315 < 2.2e-16 ***
## kuse        -5.16399    1.42438 -3.6254 0.0003031 ***
## male        -1.62201    1.09186 -1.4855 0.1377180
## black       -2.49898    1.34670 -1.8556 0.0638009 .
## age97        0.73868    0.18075  4.0867 4.727e-05 ***
## pcged97      0.99264    0.35596  2.7886 0.0053938 **
## mratio96     1.13856    0.32220  3.5337 0.0004286 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
# ATT
psw_att <- lm(lwss97 ~ kuse + male + black + age97 + pcged97 + mratio96,
  data = psw_df, weights = att_w)
lmtest::coeftest(psw_att, vcov. = vcovCL(psw_att, cluster = psw_df$pcg_id))
```

```
##
## t test of coefficients:
##
##             Estimate Std. Error t value  Pr(>|t|)
## (Intercept) 85.29467    5.05331 16.8790 < 2.2e-16 ***
## kuse        -4.62058    1.41182 -3.2728  0.001102 **
## male        -1.58995    1.14705 -1.3861  0.166020
## black       -2.74605    1.41605 -1.9392  0.052756 .
## age97        0.61577    0.20001  3.0787  0.002136 **
## pcged97      0.92698    0.36718  2.5246  0.011738 *
## mratio96     1.26018    0.33556  3.7555  0.000183 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

7 Appendix A: Function to Replicate Stata's Robust Standard Errors

Function by Jorge Cimentada to replicate robust standard errors in Stata:[9]

```
robustse <- function(x, coef = c("logit", "odd.ratio", "probs")) {
  suppressMessages(suppressWarnings(library(lmtest)))
  suppressMessages(suppressWarnings(library(sandwich)))
  sandwich1 <- function(object, ...) {
    sandwich(object) *
      nobs(object) / (nobs(object) - 1)
  }
  mod1 <- coeftest(x, vcov = sandwich1)
  if (coef == "logit") {
    return(mod1)
  } else if (coef == "odd.ratio") {
    mod1[, 1] <- exp(mod1[, 1])
    mod1[, 2] <- mod1[, 1] * mod1[, 2]
    return(mod1)
  } else {
    mod1[, 1] <- (mod1[, 1] / 4)
    mod1[, 2] <- mod1[, 2] / 4
    return(mod1)
  }
}
```

[9] https://cimentadaj.github.io/blog/2016-09-19-obtaining-robust-standard-errors-and-odds-ratios/obtaining-robust-standard-errors-and-odds-ratios-for-logistic-regression-in-r/

8 Appendix B: Rosenbaum's Sensitivity Analysis

The R package `rbounds` can be used to carry out Rosenbaum's sensitivity analysis. Please refer to Sections 11.5.1 and 11.5.2 of the PSA-R code. Below is an example from the `rbounds` package documentation:[10]

```
library(rbounds)

# Data: Matched Data of Lead Blood Levels in Children
trt <- c(38, 23, 41, 18, 37, 36, 23, 62, 31, 34, 24, 14, 21, 17, 16, 20, 15,
         10, 45, 39, 22, 35, 49, 48, 44, 35, 43, 39, 34, 13, 73, 25, 27)

ctrl <- c(16, 18, 18, 24, 19, 11, 10, 15, 16, 18, 18, 13, 19, 10, 16, 16, 24, 13,
          9, 14, 21, 19, 7, 18, 19, 12, 11, 22, 25, 16, 13, 11, 13)

hlsens(trt, ctrl)
```

```
##
##  Rosenbaum Sensitivity Test for Hodges-Lehmann Point Estimate
##
## Unconfounded estimate ....  15.5
##
##  Gamma Lower bound Upper bound
##      1        15.5        15.5
##      2        10.5        19.6
##      3         8.0        23.1
##      4         6.5        25.1
##      5         5.0        26.6
##      6         4.0        28.1
##
##  Note: Gamma is Odds of Differential Assignment To
##  Treatment Due to Unobserved Factors
##
```

[10] https://cran.r-project.org/web/packages/rbounds/rbounds.pdf