chap10-psm

yousri elfattah

2024-10-06

# Chapter 10

# Causal Inference with Observational Data

## Propensity Score Matching (PSM)

### Loading the Cattaneo2 dataset

# Load necessary library  
library(haven) # for read\_dta function  
  
# Load the dataset  
data <- haven::read\_dta("cattaneo2.dta")  
  
# Convert to standard data frame for full column display  
data\_df <- as.data.frame(data)  
  
# Inspect the first few rows and column names  
head(data\_df)

## bweight mmarried mhisp fhisp foreign alcohol deadkids mage medu fage fedu  
## 1 3459 1 0 0 0 0 0 24 14 28 16  
## 2 3260 0 0 0 1 0 0 20 10 0 0  
## 3 3572 1 0 0 1 0 0 22 9 30 9  
## 4 2948 1 0 0 0 0 0 26 12 30 12  
## 5 2410 1 0 0 0 0 0 20 12 21 14  
## 6 3147 0 0 0 0 0 0 27 12 40 12  
## nprenatal monthslb order msmoke mbsmoke mrace frace prenatal birthmonth  
## 1 10 30 2 0 0 1 1 1 12  
## 2 6 42 3 0 0 0 0 1 7  
## 3 10 17 3 0 0 1 1 1 3  
## 4 10 34 2 0 0 1 1 1 1  
## 5 12 0 1 0 0 1 1 1 3  
## 6 9 0 1 0 0 1 1 1 4  
## lbweight fbaby prenatal1  
## 1 0 0 1  
## 2 0 0 1  
## 3 0 0 1  
## 4 0 0 1  
## 5 1 1 1  
## 6 0 1 1

# Check the structure of the dataset to see data types  
# Use capture.output to capture the output of str(data)  
str\_output <- capture.output(str(data))  
writeLines(str\_output, "str\_data\_output.txt")  
  
# Print to console to verify  
cat("Output of str(data) saved to str\_data\_output.txt\n")

## Output of str(data) saved to str\_data\_output.txt

### Average Treatment Effect (ATE)

The code calculates the mean birth weights for smoking and non-smoking mothers and computes the ATE by subtracting the mean birth weight of non-smokers from that of smokers. Standard Error and Confidence Interval via Bootstrapping: We can estimate the SE and CI using a non-parametric bootstrap method. The bootstrap involves repeatedly resampling the matched dataset with replacement, recalculating the ATE for each resample, and then using the distribution of these ATE estimates to compute the SE and CI.

# Load necessary libraries  
library(MatchIt)  
library(dplyr)

##   
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':  
##   
## filter, lag

## The following objects are masked from 'package:base':  
##   
## intersect, setdiff, setequal, union

library(boot)  
  
# Assuming 'data' is your dataset  
data$mbsmoke <- as.factor(data$mbsmoke)  
  
# Perform propensity score matching  
psm <- matchit(mbsmoke ~ mmarried + prenatal1 + mage + fbaby,  
 data = data,  
 method = "nearest",  
 distance = "logit")  
  
# Extract the matched data, specifying a different name for weights  
matched\_data <- match.data(psm, weights = "matching\_weights")  
  
# Calculate ATE  
mean\_bw\_smoke <- matched\_data %>%  
 filter(mbsmoke == 1) %>%  
 summarize(mean\_bweight = mean(bweight, na.rm = TRUE)) %>%  
 pull(mean\_bweight)  
  
mean\_bw\_nosmoke <- matched\_data %>%  
 filter(mbsmoke == 0) %>%  
 summarize(mean\_bweight = mean(bweight, na.rm = TRUE)) %>%  
 pull(mean\_bweight)  
  
ATE <- mean\_bw\_smoke - mean\_bw\_nosmoke  
  
print(paste("Mean birth weight for smoking mothers:", mean\_bw\_smoke))

## [1] "Mean birth weight for smoking mothers: 3137.65972222222"

print(paste("Mean birth weight for non-smoking mothers:", mean\_bw\_nosmoke))

## [1] "Mean birth weight for non-smoking mothers: 3349.72337962963"

print(paste("Average Treatment Effect = ", ATE))

## [1] "Average Treatment Effect = -212.063657407407"

# Define a function for bootstrapping the ATE  
bootstrap\_ate <- function(data, indices) {  
 sampled\_data <- data[indices, ]  
   
 mean\_bw\_smoke\_boot <- sampled\_data %>%  
 filter(mbsmoke == 1) %>%  
 summarize(mean\_bweight = mean(bweight, na.rm = TRUE)) %>%  
 pull(mean\_bweight)  
   
 mean\_bw\_nosmoke\_boot <- sampled\_data %>%  
 filter(mbsmoke == 0) %>%  
 summarize(mean\_bweight = mean(bweight, na.rm = TRUE)) %>%  
 pull(mean\_bweight)  
   
 return(mean\_bw\_smoke\_boot - mean\_bw\_nosmoke\_boot)  
}  
  
# Perform bootstrapping with 1000 replications  
set.seed(123) # For reproducibility  
boot\_results <- boot(data = matched\_data, statistic = bootstrap\_ate, R = 1000)  
  
# Calculate 95% CI from bootstrapped results  
boot\_ci <- boot.ci(boot\_results, type = "perc")  
  
# Print results  
print(paste("Bootstrap Standard Error = ", sd(boot\_results$t)))

## [1] "Bootstrap Standard Error = 29.5673094759983"

print(paste("95% Confidence Interval = (", boot\_ci$percent[4], ", ", boot\_ci$percent[5], ")", sep=""))

## [1] "95% Confidence Interval = (-267.989098351548, -153.375395391343)"

# Alternatively, you can use boot\_ci$perc to see percentile delimiters:   
print(boot\_ci)

## BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS  
## Based on 1000 bootstrap replicates  
##   
## CALL :   
## boot.ci(boot.out = boot\_results, type = "perc")  
##   
## Intervals :   
## Level Percentile   
## 95% (-268.0, -153.4 )   
## Calculations and Intervals on Original Scale

## Balance Plot for Propensity Scores

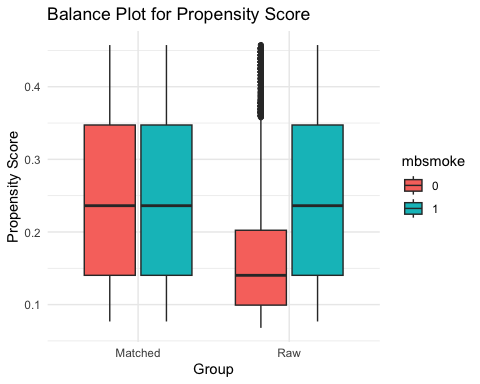
### Explanation

* **Subset Data to Common Columns**: Use intersect to retain only the common columns in data and matched\_data.
* **Add Propensity Score Column**: Add the computed propensity score to both raw and matched datasets.
* **Add Group Column**: Differentiate between raw and matched data.
* **Combine Data**: Use rbind after ensuring both datasets have the same structure.

# Load necessary libraries  
# library(MatchIt)  
library(ggplot2)  
library(tidyverse)

## ── Attaching core tidyverse packages ──────────────────────── tidyverse 2.0.0 ──  
## ✔ forcats 1.0.0 ✔ stringr 1.5.1  
## ✔ lubridate 1.9.3 ✔ tibble 3.2.1  
## ✔ purrr 1.0.2 ✔ tidyr 1.3.1  
## ✔ readr 2.1.5   
## ── Conflicts ────────────────────────────────────────── tidyverse\_conflicts() ──  
## ✖ dplyr::filter() masks stats::filter()  
## ✖ dplyr::lag() masks stats::lag()  
## ℹ Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors

# # Assuming 'data' is your dataset  
# data$mbsmoke <- as.factor(data$mbsmoke)  
#   
# # Perform propensity score matching  
# psm <- matchit(mbsmoke ~ mmarried + prenatal1 + mage + fbaby,  
# data = data,  
# method = "nearest",  
# distance = "logit")  
#   
# # Extract the matched data, ensuring there's no name conflict with weights  
# matched\_data <- match.data(psm, weights = "matching\_weights")  
  
# Extract propensity scores  
data$propensity\_score <- psm$distance  
matched\_data$propensity\_score <- matched\_data$distance  
  
# Extract common columns  
common\_columns <- intersect(names(data), names(matched\_data))  
  
# Subset data to have the same columns  
data\_common <- data[common\_columns]  
matched\_data\_common <- matched\_data[common\_columns]  
  
# Add Group column for comparison  
data\_common$Group <- "Raw"  
matched\_data\_common$Group <- "Matched"  
  
# Combine raw and matched propensity scores  
combined\_data <- rbind(data\_common, matched\_data\_common)  
  
# Function to create box plot for propensity scores  
create\_propensity\_score\_plot <- function(combined\_data) {  
 ggplot(combined\_data, aes(x = Group, y = propensity\_score, fill = mbsmoke)) +  
 geom\_boxplot() +  
 theme\_minimal() +  
 labs(title = "Balance Plot for Propensity Score",   
 x = "Group",   
 y = "Propensity Score")  
}  
  
# Create and print the propensity score balance plot  
print(create\_propensity\_score\_plot(combined\_data))



### Session Info

sessionInfo()

## R version 4.4.1 (2024-06-14)  
## Platform: aarch64-apple-darwin20  
## Running under: macOS 15.0.1  
##   
## Matrix products: default  
## BLAS: /Library/Frameworks/R.framework/Versions/4.4-arm64/Resources/lib/libRblas.0.dylib   
## LAPACK: /Library/Frameworks/R.framework/Versions/4.4-arm64/Resources/lib/libRlapack.dylib; LAPACK version 3.12.0  
##   
## locale:  
## [1] en\_US.UTF-8/en\_US.UTF-8/en\_US.UTF-8/C/en\_US.UTF-8/en\_US.UTF-8  
##   
## time zone: America/Los\_Angeles  
## tzcode source: internal  
##   
## attached base packages:  
## [1] stats graphics grDevices utils datasets methods base   
##   
## other attached packages:  
## [1] lubridate\_1.9.3 forcats\_1.0.0 stringr\_1.5.1 purrr\_1.0.2   
## [5] readr\_2.1.5 tidyr\_1.3.1 tibble\_3.2.1 tidyverse\_2.0.0  
## [9] ggplot2\_3.5.1 boot\_1.3-31 dplyr\_1.1.4 MatchIt\_4.5.5   
## [13] haven\_2.5.4   
##   
## loaded via a namespace (and not attached):  
## [1] gtable\_0.3.5 highr\_0.11 compiler\_4.4.1 tidyselect\_1.2.1   
## [5] Rcpp\_1.0.13 scales\_1.3.0 yaml\_2.3.10 fastmap\_1.2.0   
## [9] R6\_2.5.1 labeling\_0.4.3 generics\_0.1.3 knitr\_1.48   
## [13] backports\_1.5.0 chk\_0.9.2 munsell\_0.5.1 pillar\_1.9.0   
## [17] tzdb\_0.4.0 rlang\_1.1.4 utf8\_1.2.4 stringi\_1.8.4   
## [21] xfun\_0.47 timechange\_0.3.0 cli\_3.6.3 withr\_3.0.1   
## [25] magrittr\_2.0.3 digest\_0.6.37 grid\_4.4.1 rstudioapi\_0.16.0  
## [29] hms\_1.1.3 lifecycle\_1.0.4 vctrs\_0.6.5 evaluate\_0.24.0   
## [33] glue\_1.7.0 farver\_2.1.2 fansi\_1.0.6 colorspace\_2.1-1   
## [37] rmarkdown\_2.28 tools\_4.4.1 pkgconfig\_2.0.3 htmltools\_0.5.8.1