Horizontal Stratification of Higher Education and Gender Earnings Gap in the Early Labor Market

Inchan Hwang, inchan@yonsei.ac.kr

4 October 2024

Introduction to Doubly Robust Estimation for Categorical Treatments

Accurate estimation of causal effects across multiple treatment categories is vital in fields such as education, healthcare, and economics. Challenges such as selection bias and model misspecification often hinder the reliability of these estimates. To address these issues, Debiased Machine Learning (DML) provides a powerful framework. By integrating flexible machine learning techniques with the doubly robust framework, DML ensures consistent and unbiased estimates, even when one of the underlying models—propensity score or outcome regression—is misspecified.

This document focuses on the application of doubly robust estimation in settings involving three categorical treatments. By leveraging generalized propensity scores, we extend this methodology to three treatments.

1. The Doubly Robust Framework

Key Principles

Doubly robust estimation combines two complementary approaches: - Outcome regression, which models the outcome as a function of covariates, and - Propensity score weighting, which accounts for the probability of treatment assignment.

The doubly robust property ensures that if either the outcome regression model or the propensity score model is correctly specified, the resulting estimates remain consistent. This robustness is particularly valuable in complex causal inference settings.

Extending to Categorical Treatments

Let T represent the treatment variable, taking on values $T \in \{0, 1, 2\}$, where each value corresponds to one of the treatment categories. The expected outcomes for each category, denoted as $\mathbb{E}[Y(0)]$, $\mathbb{E}[Y(1)]$, and $\mathbb{E}[Y(2)]$, can be estimated using the following doubly robust formulas:

$$Y_i(0) = \frac{1}{N} \sum_{i=1}^{N} \left[\hat{m}_0(X_i) + \frac{\mathbb{I}(T_i = 0)(Y_i - \hat{m}_0(X_i))}{\hat{e}_0(X_i)} \right]$$

$$Y_i(1) = \frac{1}{N} \sum_{i=1}^{N} \left[\hat{m}_1(X_i) + \frac{\mathbb{I}(T_i = 1)(Y_i - \hat{m}_1(X_i))}{\hat{e}_1(X_i)} \right]$$

$$Y_i(2) = \frac{1}{N} \sum_{i=1}^{N} \left[\hat{m}_2(X_i) + \frac{\mathbb{I}(T_i = 2)(Y_i - \hat{m}_2(X_i))}{\hat{e}_2(X_i)} \right]$$

Where: - $\hat{m}_j(X_i)$: Estimated conditional expectation of Y given covariates for treatment j, - $\mathbb{I}(T_i = j)$: Indicator function, equal to 1 if $T_i = j$ and 0 otherwise, - $\hat{e}_j(X_i)$: Estimated propensity score for treatment j.

Bias Reduction Mechanism

The property of double robustness operates through bias reduction. For example, consider estimating $\mathbb{E}[Y(0)]$:

$$\mathbb{E}\left[\frac{\mathbb{I}(T=0)Y}{e_0(X)} - \frac{\mathbb{I}(T=0) - e_0(X)}{e_0(X)} m_0(X)\right].$$

By adding and subtracting $Y(0)$, the expanded equation becomes:

$$\mathbb{E}[Y(0)] + \mathbb{E}\left[\frac{Z - e_0(X)}{e_0(X)} (Y_0 - m_0(X))\right].$$

In the above equation, to obtain $\mathbb{E}[Y(0)]$ and $\mathbb{E}\left[\frac{Z-e_0(X)}{e_0(X)}(Y_0 - m_0(X))\right]$, the term must converge to zero. This can be broadly divided into two types of errors: $Z - e_0(X)$, which is related to the propensity score model, and $Y_0 - m_0(X)$, which is related to the outcome regression model. If the propensity score model is correctly specified, $Y_0 - m_0(X)$ converges to zero, making the entire term zero. Conversely, if the outcome regression model is correctly specified, $Z - e_0(X)$ becomes zero, causing the entire error term to become zero.

This feature, where the bias of the double robustness model is the product of the biases from the propensity score model and the outcome regression model, ensures that if at least one model is correctly specified, a consistent estimate can still be obtained.

Enhancements Through Flexible Machine Learning

The doubly robust framework achieves even greater accuracy when integrated with flexible machine learning models. Machine learning excels at modeling non-linear relationships and handling high-dimensional data, making it ideal for estimating $m_0(X)$ and $e_0(X)$ with high precision. Studies by Chernozhukov et al. (2018) and Semenova and Chernozhukov (2021) demonstrate that combining doubly robust properties with machine learning leads to:

- Root-n consistency: Estimates that converge to the true parameter at the optimal rate as sample size increases.
- Bias minimization: Improved reliability of causal inference, particularly in scenarios with complex covariate interactions.

By leveraging the strengths of machine learning, this approach provides a robust method for causal inference, especially in contexts involving multidimensional data and multiple treatment categories.

2. DML with Cross-Fitting: Step-by-Step Process and Code

This document explains the process of implementing Debiased Machine Learning (DML) using cross-fitting to address overfitting risks and obtain reliable causal estimates. The methodology is aligned with the following steps:

Step 0: Data Preparation

The following code snippet demonstrates the process of loading the dataset, defining key variables, and preparing the covariates for the analysis. This document focuses on explaining formulas and R code, so I use only a small amount of data and a subset of covariates for simplicity. For more detailed analyses or inquiries about the R code, please contact the author via email.

```r
file_path <- "./GOMS.txt"
data <- read.table(file_path, header =TRUE, sep=",")

# The treatment variable, "education," is a categorical variable consisting of
# three levels: 0, 1, and 2.

treatment <- "edu"
outcome <- "ln_hourly_income"
group <- "sex"

data$year <- factor(data$year)
data$faedu <- factor(data$faedu)
data$maedu <- factor(data$maedu)

# Covariates
covariates <- c("sex","year","faedu","maedu")
```

Step 1: Partition the Dataset

Randomly partition the total dataset I into J subsets: $I_1, I_2, \ldots, I_J$. Many scholars recommend setting $J = 5$ (Chernozhukov et al., 2018).

```r
set.seed(123)

# Randomly partition into 5
n <- nrow(data)
n.folds <- 5

random_indices <- sample(seq(n))
indices <- split(random_indices, cut(random_indices, n.folds, labels = FALSE))

W <- data[,treatment]
Y <- data[,outcome]

fmla.xw <- formula(paste("~ 0 +", paste(covariates,
                                        collapse=" + "), "+", treatment))
XW <- model.matrix(fmla.xw, data)

data.2 <- data
data.2[,treatment] <- 2
XW2 <- model.matrix(fmla.xw, data.2)  # setting W=2
data.1 <- data
data.1[,treatment] <- 1
XW1 <- model.matrix(fmla.xw, data.1)  # setting W=1
data.0 <- data
data.0[,treatment] <- 0
XW0 <- model.matrix(fmla.xw, data.0)  # setting W=0
```

```
XX <- model.matrix(formula(paste0("~", paste0(covariates,
                                              collapse="+"))), data=data)

mu.hat.2 <- rep(NA, n)
mu.hat.1 <- rep(NA, n)
mu.hat.0 <- rep(NA, n)
e.hat2 <- rep(NA, n)
e.hat1 <- rep(NA, n)
e.hat0 <- rep(NA, n)
weight <- data$f11_weight
```

Step 2: Train the Models

In this step, Using all observations except for I_j in the overall dataset:

a) Train the propensity score model $\hat{e}_j(X)$ and the outcome model $\hat{m}_j(X_i)$ based on covariates X. Here, training fits the model to predict probabilities as accurately as possible. A flexible machine learning method, random forest, is used for training. Through this training process, the propensity score model aims to identify the fitting model to predict the treatment value using covariates X. For the outcome model, the goal is to find the fitted model that can predict the actual outcome variable using covariates X and the treatment value.

2.1 Train propensity model using Random Forest

```
for (idx in indices) {
  # Propensity model: Perform manual cross-validation with randomForest
  propensity.model <- randomForest(
    x = XX[-idx,],
    y = as.factor(W[-idx]),  # Multinomial response requires factors
    ntree = 500,
    importance = TRUE,
    nodesize = 5,
    weights = weight[-idx]  # Weights for training
  )
}
```

2.2 Train outcome model using Random Forest

```
for (idx in indices) {
  # Outcome model: Perform manual cross-validation with randomForest
  outcome.model <- randomForest(x = XW[-idx,],
                                y = Y[-idx],
                                ntree = 500,
                                importance = TRUE,
                                nodesize = 5,
                                weights = weight[-idx])

}
```

Step 3: Predict Values for the Testing Subset

b) Using the fitted model identified in step (a), calculate the propensity score values and the outcome model values for observation I_j. The propensity score values calculated here are the predicted values $\hat{e}_0(X), \hat{e}_1(X), \hat{e}_2(X)$ when the treatment variable is 0, 1, and 2, respectively, and the outcome model values are $\hat{m}_0(X_i), \hat{m}_1(X_i), \hat{m}_2(X_i)$.

c) Using the propensity score values and outcome model values calculated in each I, we can calculate individual $Y_i(0), Y_i(1), Y_i(2)$ for each observation. Once $Y_i(0), Y_i(1), Y_i(2)$ are calculated for all observations in each I, the individual $Y_i(0), Y_i(1), Y_i(2)$ values for the entire dataset are obtained.

3.1 Predicted Propensity Scores

```
for (idx in indices) {
# Predict probabilities for propensity scores
  e.hat.predictions <- predict(propensity.model,
                               newdata = XX[idx,], type = "prob")

# Extract probabilities for W = 0, 1, 2
e.hat0[idx] <- e.hat.predictions[, "0"]  # Probability for W = 0
e.hat1[idx] <- e.hat.predictions[, "1"]  # Probability for W = 1
e.hat2[idx] <- e.hat.predictions[, "2"]  # Probability for W = 2
}
```

3.2 Predict Outcomes

```
for (idx in indices) {
  outcome.model <- randomForest(x = XW[-idx,],
                                y = Y[-idx],
                                ntree = 500,
                                importance = TRUE,
                                nodesize = 5,
                                weights = weight[-idx])
# Predict outcomes using the outcome model
  mu.hat.2[idx] <- predict(outcome.model, newdata = XW2[idx,])
  mu.hat.1[idx] <- predict(outcome.model, newdata = XW1[idx,])
  mu.hat.0[idx] <- predict(outcome.model, newdata = XW0[idx,])
}
```

Step 4: Calculate Doubly Robust Scores

Using the calculated $Y_i(0)$, $Y_i(1)$, $Y_i(2)$ values for the entire dataset, we analyze the study's focus: the effect of college selectivity on the gender wage gap on the gender wage gap among four-year college graduates. To calculate the gender wage gap associated with college selectivity, we conduct linear regression analyses using each of $Y_i(0)$, $Y_i(1)$, $Y_i(2)$ as the dependent variable and gender as the independent variable. This approach allows us to estimate the predicted values for each gender by college selectivity, with the regression coefficient for the gender variable representing the gender wage gap.

```
# Trimming data with propensity scores below 0.01 or above 0.99.
data$valid_data <- with(data,
                        (e.hat0 > 0.01 & e.hat0 < 0.99) &
                          (e.hat1 > 0.01 & e.hat1 < 0.99) &
```

```r
                          (e.hat2 > 0.01 & e.hat2 < 0.99))

filtered_data <- subset(data, valid_data == 1)

expected.outcome.2 <- mu.hat.2 + (W == 2) / e.hat2 * (Y - mu.hat.2)
expected.outcome.1 <- mu.hat.1 + (W == 1) / e.hat1 * (Y - mu.hat.1)
expected.outcome.0 <- mu.hat.0 + (W == 0) / e.hat0 * (Y - mu.hat.0)

expected.outcome.2male <- expected.outcome.2[data$sex == 1 &
                                               data$valid_data == 1]
expected.outcome.1male <- expected.outcome.1[data$sex == 1 &
                                               data$valid_data == 1]
expected.outcome.0male <- expected.outcome.0[data$sex == 1 &
                                               data$valid_data == 1]
expected.outcome.2female <- expected.outcome.2[data$sex == 2 &
                                                 data$valid_data == 1]
expected.outcome.1female <- expected.outcome.1[data$sex == 2 &
                                                 data$valid_data == 1]
expected.outcome.0female <- expected.outcome.0[data$sex == 2 &
                                                 data$valid_data == 1]

expected.outcome.2 <- expected.outcome.2[data$valid_data == 1]
expected.outcome.1 <- expected.outcome.1[data$valid_data == 1]
expected.outcome.0 <- expected.outcome.0[data$valid_data == 1]

gapclosing_2 <- formula(paste0('expected.outcome.2 ~ factor(', group, ')'))
gapclosing_ols2 <- lm(gapclosing_2,
                      data=transform(filtered_data,
                      expected.outcome.2=expected.outcome.2))
summary(gapclosing_ols2)

gapclosing_1 <- formula(paste0('expected.outcome.1 ~ factor(', group, ')'))
gapclosing_ols1 <- lm(gapclosing_1,
                      data=transform(filtered_data,
                      expected.outcome.1=expected.outcome.1))
summary(gapclosing_ols1)

gapclosing_0 <- formula(paste0('expected.outcome.0 ~ factor(', group, ')'))
gapclosing_ols0 <- lm(gapclosing_0,
                      data=transform(filtered_data,
                       expected.outcome.0=expected.outcome.0))
summary(gapclosing_ols0)
```

Step 5: Average Treatment effects

If you want to calculate the Average Treatment Effect (ATE), you can use the following R code. For nominal treatments, the most common causal estimands are pairwise comparisons (Lechner, 2001).

```r
# comparing (education 0 vs education 1)
```

```
aipw.scores01 <- (mu.hat.1 - mu.hat.0 +
                    (W == 1) / e.hat1 * (Y - mu.hat.1) -
                    (W == 0) / e.hat0 * (Y - mu.hat.0))

aipw.scores01_male <- aipw.scores01[data$sex == 1 & data$valid_data == 1]
aipw.scores01_female <- aipw.scores01[data$sex == 2 & data$valid_data == 1]

ate.aipw.est01_male <- mean(aipw.scores01_male, na.rm = TRUE)
ate.aipw.est01_female <- mean(aipw.scores01_female, na.rm = TRUE)

ate.aipw.est01_male
ate.aipw.est01_female
```

Reference

- Chernozhukov, V., Chetverikov, D., Demirer, M., Duflo, E., Hansen, C., Newey, W., & Robins, J. (2018). Double/debiased machine learning for treatment and structural parameters. The Econometrics Journal, 21(1), C1-C68.

- Lechner, M. (2001). Identification and estimation of causal effects of multiple treatments under the conditional independence assumption. Econometric Evaluation of Labour Market Policies, 43-58. Physica-Verlag HD.

- Semenova, V., & Chernozhukov, V. (2021). Debiased machine learning of conditional average treatment effects and other causal functions. The Econometrics Journal, 24(2), 264-289.