Evaluating the Benefits of Sample Splitting for Double Machine Learning

Michael Sarkis, Jack McCarthy

4/24/2022

Double Machine Learning

In this section we will replicate the methods put forth in the paper by Chernozhukov et al., dubbed "CCDDHNR", which establishes an unbiased Bayesian machine learning framework for treatment effect estimation. This work deals with treatment effect estimation for complex data which has a high number of confounding parameters relative to a low dimensional treatment effect.

Simulated Dataset

First we will generated a toy dataset on which we may test each method. We will assume the data is generated according to the following simpler model derived from the general form considered in CCDDHNR:

$$\begin{aligned} Y_i &= D_i\theta + g_0(X_i) + \epsilon_i, \quad \epsilon_i \sim N(0,1), \\ D_i &= m_0(X_i) + \tau_i, \qquad \tau_i \sim N(0,1), \end{aligned}$$

where Y is the outcome, D is the treatment, and X is the vector of covariates. $g_0(X)$ and $m_0(X)$ relate the covariates to the value of the response and the treatment respectively. We define these "nuisance" functions to be

$$\begin{aligned} g_0(x) &= x_1 + \sigma(x_3), \\ m_0(x) &= x_3 + \sigma(x_1), \end{aligned}$$

where $\sigma(x)$ is the sigmoid function

$$\sigma(x) = \frac{1}{1+e^{-x}}.$$

We also define $X_i \sim N(0, \Sigma)$ where $\Sigma_{ij} = 0.5^{|i-j|}$.

Assuming a continuous response and treatment, N total observations, a p-dimensional covariate vector, and a true value of $\theta = 0.5$, we may generate a toy dataset as follows:

```
# data dimensions
N = 250
p = 100
theta = 0.5
```

```r
# covariance matrix
i_mat <- matrix(rep(1:p, p), p)
j_mat <- matrix(rep(1:p, each=p), p)
Sigma <- 0.5^abs(i_mat - j_mat)

generate_data <- function(N, p, theta, S) {
  # generate covariates
  X <- mvrnorm(n=N, mu=rep(0, p), Sigma=S)

  # generate treatment
  D <- m_0(X) + rnorm(N)

  # generate response
  Y <- D*theta + g_0(X) + rnorm(N)

  data <- as.data.frame(cbind(Y, D, X))
}
```

where we will use the **generate_data** function to obtain distributions on the predicted value of θ through multiple fits to many random datasets.

Naive ML

A naive approach to estimating θ would be to estimate $D\theta + g_0(X)$ using some machine learning method. In line with the demonstration in Chernozhukov et al., we will split the samples into two index sets of equal size, R (auxiliary) and S (primary). We will then use the auxiliary set generate the estimate $D\hat{\theta} + \hat{g_0}(X)$, and use the primary set to estimate θ as

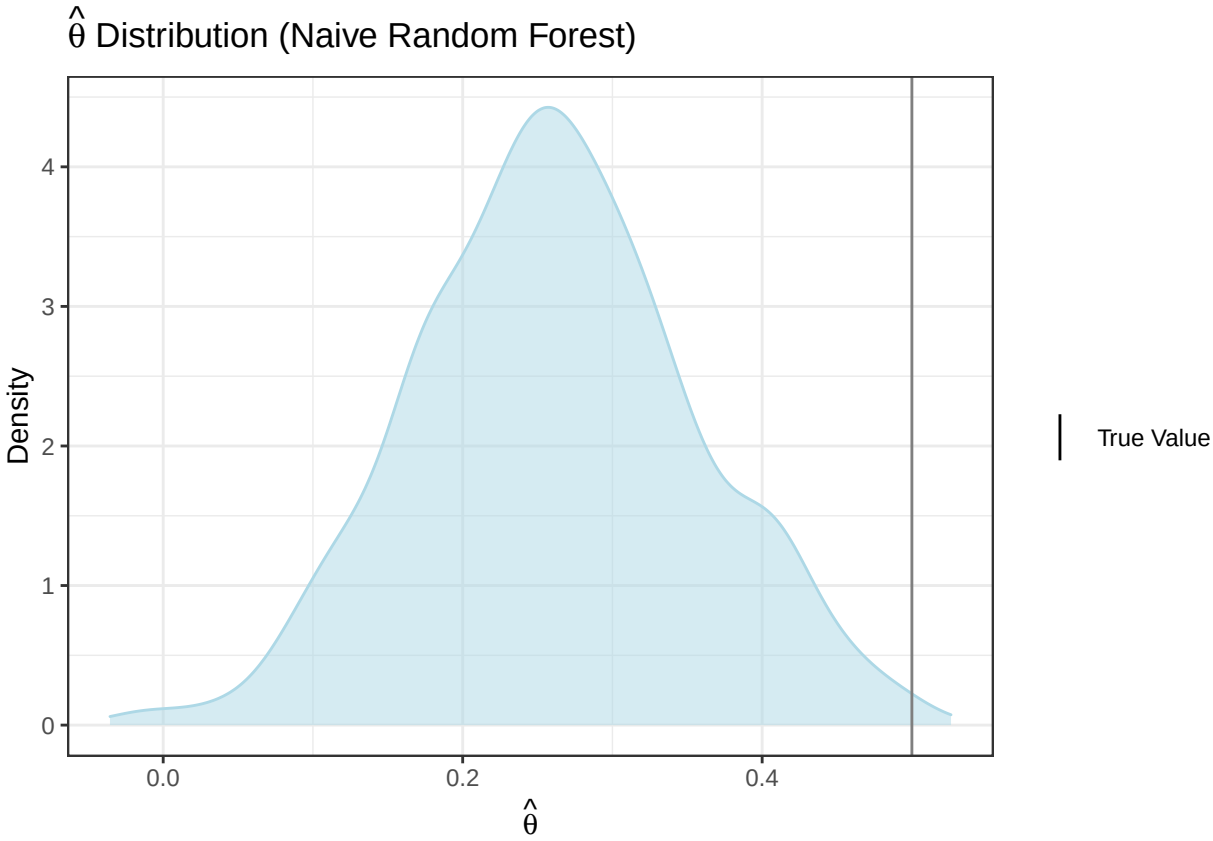
$$\hat{\theta} = \left(\frac{1}{n}\sum_{i \in S} D_i^2\right)^{-1} \frac{1}{n}\sum_{i \in S} D_i(Y_i - \hat{g_0}(X_i)).$$

We do so for our simulated dataset below using random forest regressors.

```r
n_sim <- 1000
theta.est <- numeric(n_sim)

for (i in 1:n_sim) {
  # obtain new data
  data <- generate_data(N, p, theta, Sigma)

  # sample splitting
  idx <- sample.int(N, N/2, replace=F)
  s1 <- data[idx,]
  s2 <- data[-idx,]

  # fit random forest
  rf <- randomForest(Y ~ ., ntree=100, data=s1)

  # obtain theta estimate
  g.hat <- predict(rf, s2)
  theta.est[i] <- with(s2, mean(D * (Y - g.hat)) / mean(D^2))
}
```

```
save(theta.est, file="theta_naive.RData")
```

Having obtained a vector of 500 estimates of $\hat{\theta}$, we may plot the distribution of said estimates to assess whether or not this approach is biased.



As we can see, the estimates of θ are quite distant from the true value of 0.5, indicating this naive ML method for estimating treatment effects is biased. Double machine learning was introduced to address this issue.

Double ML

Instead of estimating $D\theta + g_0(X)$, we may instead first estimate $V = D - m_0(X)$ by regressing D on X and obtaining the residuals. This will give the estimate $\hat{V} = D - \hat{m}_0(X)$, which we use in conjunction with $\hat{g}_0(X)$ to obtain the following estimate of θ:

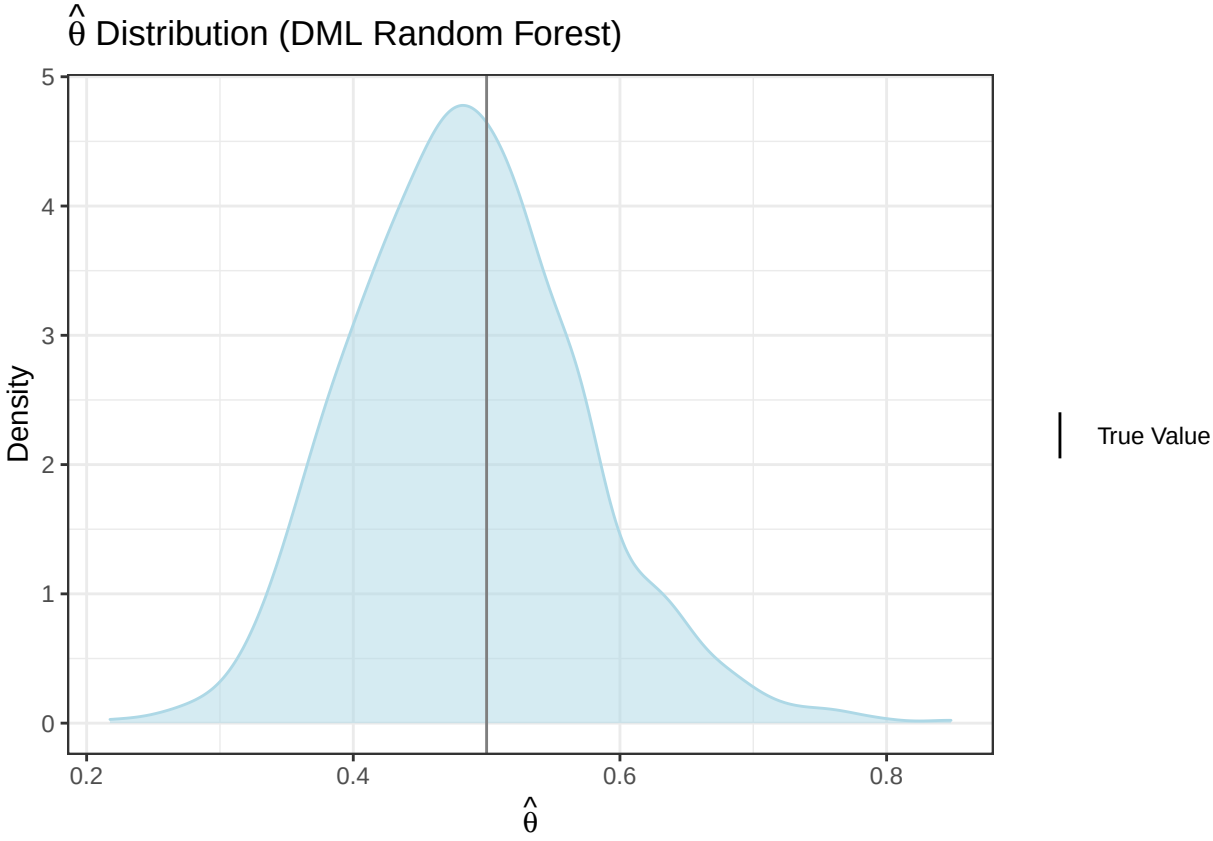
$$\hat{\theta} = \left(\frac{1}{n}\sum_{i\in S}\hat{V}_i D_i\right)^{-1}\frac{1}{n}\sum_{i\in S}\hat{V}_i(Y_i - \hat{g}_0(X_i)).$$

This method is implemented in R below, again with random forest regressors.

```
theta.dml <- numeric(n_sim)

for (i in 1:n_sim) {
  # obtain new data
```

```
  data <- generate_data(N, p, theta, Sigma)

  # sample splitting
  idx <- sample.int(N, N/2, replace=F)
  s1 <- data[idx,]
  s2 <- data[-idx,]

  # fit treatment
  rf.D <- randomForest(D ~ . - Y, ntree=100, data=s1)

  # fit response
  rf.Y <- randomForest(Y ~ . - D, ntree=100, data=s1)

  # obtain theta estimate
  V.hat <- s2$D - predict(rf.D, s2)
  g.hat <- predict(rf.Y, s2)
  theta.dml[i] <- with(s2, mean(V.hat * (Y - g.hat)) / mean(V.hat * D))
}

save(theta.dml, file="theta_dml.RData")
```

Having accumulated 1000 samples of $\hat{\theta}$, we may again investigate their distribution.



This time, the distribution appears centered directly around the true value of $\theta = 0.5$, which indicates that DML was able to avoid bias in treatment effect estimation.

We have therefore been able to replicate the results of Chernozhukov et al., demonstrating the utility of

double machine learning in produced unbiased treatment effect estimates. Notably, the examples above both made use of sample splitting in order to produce these estimates. The actual benefit of this intermediate step will be the subject of investigation in the following section.