Week 4: Double Machine Learning Using Random Forest

```r
library(randomForest)
library(hdm)
library(ggplot2)
library(dplyr)
library(tidyr)
library(lmtest)
library(sandwich)
library(caret)
library(ranger)
```

Double Machine Learning

We consider the data set GrowthData which is included in the package *hdm*. We evaluate the *Convergence Hypothesis* using Double Machine Learning.

```r
## function to get data
getdata <- function(...) {
  e <- new.env()
  name <- data(..., envir = e)[1]
  e[[name]]
}

## now load your data calling getdata()
growth <- getdata(GrowthData)
dim(growth)
```

```
[1] 90 63
```

```
## create the outcome variable y, treatment d, and control covariates W
y <- growth$Outcome
x <- growth[-which(colnames(growth) %in% c("Outcome", "intercept", "gdpsh465"))]
d <- growth$gdpsh465
```

The sample contains 90 countries and 63 controls. With $p \approx 60$ and $n = 90$, p/n is not small. We expect the least squares method to provide a poor estimate of β_1. We expect the method based on partialling-out with Lasso to provide a high quality estimate of β_1. To check this hypothesis, we analyze the relation between the output variable Y and the other country's characteristics by running a linear regression in the first step.

Unpenalized Linear Regression

```
## fit the regular OLS
fit <- lm(y ~ ., data = growth[-which(colnames(growth) %in% c("Outcome", "intercept"))])
est <- summary(fit)$coef["gdpsh465", 1]

hcv_coefs <- vcovHC(fit, type = "HC1") ## HC - "heteroskedasticity cosistent"
se <- sqrt(diag(hcv_coefs))[2] ## estimated std errors

## calculate the 95% confidence interval for 'gdpsh465'
lower_ci <- est - 1.96 * se
upper_ci <- est + 1.96 * se

cat("95% Confidence Interval from unpenalized OLS: [", lower_ci, ",", upper_ci, "] \n", "wit
```

```
95% Confidence Interval from unpenalized OLS: [ -0.07292335 , 0.05416737 ]
 with coefficient estimate:  -0.009377989
```

Unpenalized OLS provides a rather noisy estimate (high standard error) of the speed of convergence, and does not allow us to answer the question about the convergence hypothesis since the confidence interval includes zero.

Double Machine Learning Using Random Forest

```r
## double machine learning with cross fitting
dml2_for_plm <- function(x, d, y, dreg, yreg, nfold = 5) {

  nobs <- nrow(x) ## number of observations
  foldid <- rep.int(1:nfold, times = ceiling(nobs / nfold))[sample.int(nobs)] ## define fold
  I <- split(1:nobs, foldid) ## split observation indices into folds
  ytil <- dtil <- rep(NA, nobs)
  cat("fold: ")
  for (b in seq_along(I)) {
    dfit <- dreg(x[-I[[b]], ], d[-I[[b]]]) ## take a fold out
    yfit <- yreg(x[-I[[b]], ], y[-I[[b]]]) ## take a foldt out
    dhat <- predict(dfit, x[I[[b]], ], type = "raw") ## predict the left-out fold
    yhat <- predict(yfit, x[I[[b]], ], type = "raw") ## predict the left-out fold
    dtil[I[[b]]] <- (d[I[[b]]] - dhat) ## record residual for the left-out fold
    ytil[I[[b]]] <- (y[I[[b]]] - yhat) ## record residial for the left-out fold
    cat(b, " ")
  }
  rfit <- lm(ytil ~ dtil) ## estimate the main parameter by regressing one residual on the o
  coef.est <- coef(rfit)[2] ## extract coefficient
  se <- sqrt(vcovHC(rfit)[2, 2]) ## record robust standard error

  ## calculate the 95% confidence interval for 'gdpsh465'
  lower_ci <- coef.est - 1.96 * se
  upper_ci <- coef.est + 1.96 * se

  cat("\n 95% Confidence Interval from Double Machine Learning: [", lower_ci, ",", upper_ci,
  return(list(coef.est = coef.est, se = se, dtil = dtil, ytil = ytil)) ## save output and re
}
```

```r
## treatment regression
dreg <- function(x, d) {
  train(
    x = x, y = d,
    method = "ranger",
    trControl = trainControl(method = "cv", number = 5),
    tuneGrid = expand.grid(
      mtry = floor(sqrt(ncol(x))),      ## fixed at default
      splitrule = "variance",
      min.node.size = c(1, 5, 10)       ## tune min node size
    ),
    num.trees = 500                     ## fix number of trees
  )
```

```r
}

## outcome regression
yreg <- function(x, y) {
  train(
    x = x, y = y,
    method = "ranger",
    trControl = trainControl(method = "cv", number = 5),
    tuneGrid = expand.grid(
      mtry = floor(sqrt(ncol(x))),
      splitrule = "variance",
      min.node.size = c(1, 5, 10)
    ),
    num.trees = 500
  )
}

## execute double ML using random forest
set.seed(1)
dml2_rf <- dml2_for_plm(x, d, y, dreg, yreg, nfold = 5)
```

```
fold: 1  2  3  4  5
 95% Confidence Interval from Double Machine Learning: [ -0.06324313 , -0.003580002 ]
 with coefficient estimate:  -0.03341156
```