# pc4_students.R

*user*

*2020-03-19*

```r
################################################################################
## Course: Machine Learning for Economists and Business Analysts
## Topic: High-Dimensional Confounding
################################################################################

#rm(list = ls())
set.seed(100239)

#getwd()
#setwd("")

#  Load Packages
library("fBasics")
```

```
## Warning: package 'fBasics' was built under R version 3.6.1
## Loading required package: timeDate
## Loading required package: timeSeries
## Warning: package 'timeSeries' was built under R version 3.6.1
```

```r
library("glmnet")
```

```
## Loading required package: Matrix
## Loading required package: foreach
## Loaded glmnet 2.0-18
```

```r
library("AER")
```

```
## Warning: package 'AER' was built under R version 3.6.2
## Loading required package: car
## Warning: package 'car' was built under R version 3.6.2
## Loading required package: carData
## Warning: package 'carData' was built under R version 3.6.1
##
## Attaching package: 'car'
## The following object is masked from 'package:fBasics':
##
##     densityPlot
## Loading required package: lmtest
## Loading required package: zoo
##
## Attaching package: 'zoo'
```

```
## The following object is masked from 'package:timeSeries':
##
##     time<-

## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric

## Loading required package: sandwich

## Loading required package: survival
```

```
library("hdm")
```

```
## Warning: package 'hdm' was built under R version 3.6.3
```

```
library("lmtest")
library("sandwich")
library("tidyverse")
```

```
## -- Attaching packages ----------------------------------------------------------------- tidyverse 1.2

## v ggplot2 3.2.0     v purrr   0.3.2
## v tibble  2.1.3     v dplyr   0.8.2
## v tidyr   0.8.3     v stringr 1.4.0
## v readr   1.3.1     v forcats 0.4.0

## -- Conflicts -------------------------------------------------------------- tidyverse_conflicts
## x purrr::accumulate() masks foreach::accumulate()
## x tidyr::expand()     masks Matrix::expand()
## x dplyr::filter()     masks timeSeries::filter(), stats::filter()
## x dplyr::lag()        masks timeSeries::lag(), stats::lag()
## x dplyr::recode()     masks car::recode()
## x purrr::some()       masks car::some()
## x purrr::when()       masks foreach::when()
```

```
# Load data
df <- read.csv("job_corps.csv",header=TRUE, sep=",")

########################################
### Exercise 1: Conventional Methods ###
########################################

################################################################################
# Task 1
################################################################################

# Table with Descriptive Statistics
desc <- fBasics::basicStats(df) %>% t() %>% as.data.frame() %>%
  select(Mean, Stdev, Minimum, Maximum, nobs)
print(round(desc, digits=2))
```

```
##                  Mean  Stdev Minimum Maximum  nobs
## EARNY4         204.44 195.69       0 2409.91 10516
## assignment       0.60   0.49       0    1.00 10516
## participation    0.44   0.50       0    1.00 10516
## female           0.43   0.49       0    1.00 10516
## age_1            0.41   0.49       0    1.00 10516
## age_2            0.31   0.46       0    1.00 10516
```

```
## age_3           0.27   0.45       0    1.00 10516
## ed0_6           0.26   0.44       0    1.00 10516
## ed6_12          0.36   0.48       0    1.00 10516
## hs_ged          0.24   0.43       0    1.00 10516
## white           0.26   0.44       0    1.00 10516
## black           0.49   0.50       0    1.00 10516
## hisp            0.17   0.38       0    1.00 10516
## oth_eth         0.07   0.26       0    1.00 10516
## haschld         0.20   0.40       0    1.00 10516
## livespou        0.06   0.24       0    1.00 10516
## everwork        0.80   0.40       0    1.00 10516
## yr_work         0.64   0.48       0    1.00 10516
## currjob         0.21   0.40       0    1.00 10516
## job0_3          0.22   0.41       0    1.00 10516
## job3_9          0.30   0.46       0    1.00 10516
## job9_12         0.21   0.41       0    1.00 10516
## earn1           0.11   0.31       0    1.00 10516
## earn2           0.27   0.44       0    1.00 10516
## earn3           0.14   0.34       0    1.00 10516
## earn4           0.07   0.25       0    1.00 10516
## badhlth         0.13   0.34       0    1.00 10516
## welf_kid        0.20   0.40       0    1.00 10516
## got_fs          0.45   0.50       0    1.00 10516
## publich         0.22   0.41       0    1.00 10516
## got_afdc        0.31   0.46       0    1.00 10516
## harduse         0.06   0.24       0    1.00 10516
## potuse          0.24   0.43       0    1.00 10516
## evarrst         0.24   0.43       0    1.00 10516
## pmsa            0.32   0.47       0    1.00 10516
## msa             0.47   0.50       0    1.00 10516
```

```r
###############################################################################
# Task 2
###############################################################################

# Univariate OLS
ols <- lm(EARNY4 ~ participation, data = df)
summary(ols)
```

```
##
## Call:
## lm(formula = EARNY4 ~ participation, data = df)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -211.44 -168.41  -25.03  101.08 2210.97
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)    198.936      2.550  78.023  < 2e-16 ***
## participation   12.503      3.842   3.254  0.00114 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 195.6 on 10514 degrees of freedom
```

```
## Multiple R-squared:  0.001006,   Adjusted R-squared:  0.0009111
## F-statistic: 10.59 on 1 and 10514 DF,  p-value: 0.001141
```

```r
# Robust standard errors
coeftest(ols, vcov = vcovHC(ols, type = "HC1"))
```

```
##
## t test of coefficients:
##
##                 Estimate Std. Error t value  Pr(>|t|)
## (Intercept)   198.9359     2.5047 79.4250 < 2.2e-16 ***
## participation  12.5026     3.8604  3.2387  0.001204 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
################################################################################
# Task 3
################################################################################

# IV results
iv <- ivreg(formula = EARNY4 ~ participation | assignment, data = df)
summary(iv)
```

```
##
## Call:
## ivreg(formula = EARNY4 ~ participation | assignment, data = df)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -213.84 -168.37  -23.14  100.57 2212.87
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)    197.042      3.058  64.434  < 2e-16 ***
## participation   16.803      5.428   3.096  0.00197 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 195.6 on 10514 degrees of freedom
## Multiple R-Squared: 0.0008871,   Adjusted R-squared: 0.0007921
## Wald test: 9.584 on 1 and 10514 DF,  p-value: 0.001968
```

```r
# Robust standard errors
coeftest(iv, vcov = vcovHC(ols, type = "HC1"))
```

```
##
## t test of coefficients:
##
##                 Estimate Std. Error t value  Pr(>|t|)
## (Intercept)   197.0422     2.5047 78.6690 < 2.2e-16 ***
## participation  16.8027     3.8604  4.3526 1.358e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
###############################################
### Exercise 2: Double Selection Procedure ###
###############################################
```

```r
################################################################################
# Task 1
################################################################################

# Predict earnings
st1 <- rlasso(as.matrix(df[,c(4:ncol(df))]), as.matrix(df$EARNY4))
summary(st1)
```

```
##
## Call:
## rlasso.default(x = as.matrix(df[, c(4:ncol(df))]), y = as.matrix(df$EARNY4))
##
## Post-Lasso Estimation:  TRUE
##
## Total number of variables: 33
## Number of selected variables: 16
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -384.45 -131.90  -19.95   90.80 2194.30
##
##             Estimate
## (Intercept)  189.994
## female       -56.801
## age_1         -3.582
## age_2          0.000
## age_3          8.514
## ed0_6          0.000
## ed6_12         0.000
## hs_ged        29.675
## white         22.872
## black        -24.628
## hisp           0.000
## oth_eth        0.000
## haschld        0.000
## livespou       0.000
## everwork      12.135
## yr_work       41.323
## currjob       18.280
## job0_3         0.000
## job3_9         0.000
## job9_12        5.634
## earn1        -25.530
## earn2          0.000
## earn3         24.463
## earn4         64.541
## badhlth        0.000
## welf_kid     -12.330
## got_fs        -6.239
## publich        0.000
## got_afdc       0.000
## harduse        0.000
## potuse       -16.297
```

```
## evarrst        0.000
## pmsa           0.000
## msa            0.000
##
## Residual standard error: 185.5
## Multiple R-squared:  0.1018
## Adjusted R-squared:  0.1005
## Joint significance test:
##  the sup score statistic for joint significance test is  1818 with a p-value of     0
```

```
# Store selected variables
n1<- names(st1$coefficients[(st1$coefficients != 0) == TRUE])[-1]

################################################################################
# Task 2
################################################################################

# Predict participation
st2 <- rlasso(as.matrix(df[,c(4:ncol(df))]), as.matrix(df$participation))
summary(st2)
```

```
##
## Call:
## rlasso.default(x = as.matrix(df[, c(4:ncol(df))]), y = as.matrix(df$participation))
##
## Post-Lasso Estimation:  TRUE
##
## Total number of variables: 33
## Number of selected variables: 1
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -0.4462 -0.4462 -0.4462  0.5538  0.6449
##
##             Estimate
## (Intercept)    0.446
## female         0.000
## age_1          0.000
## age_2          0.000
## age_3          0.000
## ed0_6          0.000
## ed6_12         0.000
## hs_ged         0.000
## white          0.000
## black          0.000
## hisp           0.000
## oth_eth        0.000
## haschld        0.000
## livespou      -0.091
## everwork       0.000
## yr_work        0.000
## currjob        0.000
## job0_3         0.000
## job3_9         0.000
## job9_12        0.000
```

```
## earn1          0.000
## earn2          0.000
## earn3          0.000
## earn4          0.000
## badhlth        0.000
## welf_kid       0.000
## got_fs         0.000
## publich        0.000
## got_afdc       0.000
## harduse        0.000
## potuse         0.000
## evarrst        0.000
## pmsa           0.000
## msa            0.000
##
## Residual standard error: 0.496
## Multiple R-squared:  0.002016
## Adjusted R-squared:  0.001921
## Joint significance test:
##  the sup score statistic for joint significance test is 0.9382 with a p-value of 0.054
```

```r
# Store selected variables
n2<- names(st2$coefficients[(st2$coefficients != 0) == TRUE])[-1]

###############################################################################
# Task 3
###############################################################################

# Take uniion of selected covariates
selected_covariates <- c("participation", unique(c(n1, n2)))

# Setup the formula of the linear regression model
sumx <- paste(selected_covariates, collapse = " + ")
linear <- paste("EARNY4",paste(sumx, sep=" + "), sep=" ~ ")
linear <- as.formula(linear)

# Post-Lasso OLS regression
ols <- lm(linear, data = df)
summary(ols)
```

```
##
## Call:
## lm(formula = linear, data = df)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -378.47 -132.12  -20.12   91.01 2201.29
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)    183.337      6.762  27.113  < 2e-16 ***
## participation   15.678      3.650   4.295 1.76e-05 ***
## female         -56.965      3.794 -15.015  < 2e-16 ***
## age_1           -4.133      4.565  -0.905 0.365348
## age_3            8.685      4.931   1.761 0.078229 .
```

```
## hs_ged          29.911      4.861   6.153 7.87e-10 ***
## white           23.023      5.167   4.456 8.44e-06 ***
## black          -24.811      4.509  -5.502 3.84e-08 ***
## everwork        12.168      6.236   1.951 0.051065 .
## yr_work         41.219      5.878   7.013 2.48e-12 ***
## currjob         18.076      5.020   3.601 0.000319 ***
## job9_12          5.685      5.675   1.002 0.316465
## earn1          -25.373      6.506  -3.900 9.68e-05 ***
## earn3           24.666      6.107   4.039 5.40e-05 ***
## earn4           65.047      8.651   7.519 5.99e-14 ***
## welf_kid       -12.366      4.794  -2.580 0.009903 **
## got_fs          -5.893      3.945  -1.494 0.135292
## potuse         -16.390      4.280  -3.830 0.000129 ***
## livespou        -2.070      7.570  -0.273 0.784519
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 185.5 on 10497 degrees of freedom
## Multiple R-squared:  0.1034, Adjusted R-squared:  0.1019
## F-statistic: 67.26 on 18 and 10497 DF,  p-value: < 2.2e-16
```

```
# Robust standard errors
coeftest(ols, vcov = vcovHC(ols, type = "HC1"))
```

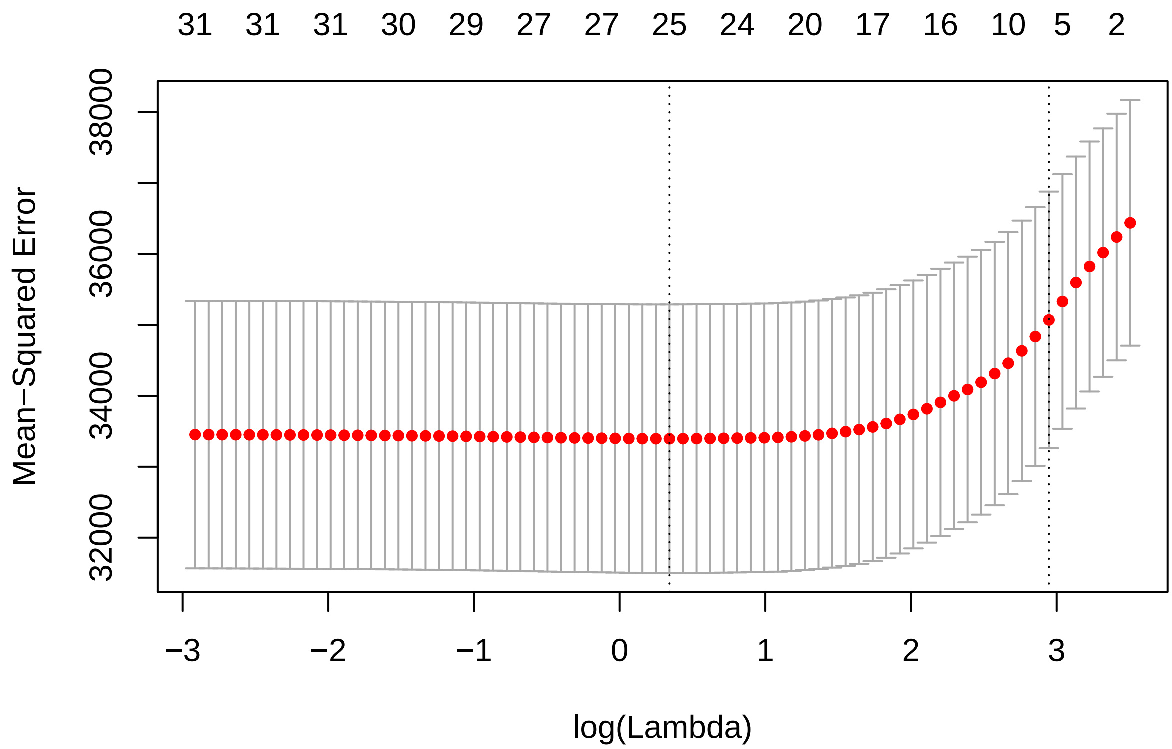
```
##
## t test of coefficients:
##
##                Estimate Std. Error  t value  Pr(>|t|)
## (Intercept)    183.3367     6.8100  26.9219 < 2.2e-16 ***
## participation   15.6776     3.6616   4.2816 1.872e-05 ***
## female         -56.9647     3.7024 -15.3857 < 2.2e-16 ***
## age_1           -4.1325     4.6049  -0.8974 0.3695081
## age_3            8.6852     4.9891   1.7408 0.0817412 .
## hs_ged          29.9115     4.9989   5.9836 2.254e-09 ***
## white           23.0225     5.4843   4.1979 2.717e-05 ***
## black          -24.8106     4.5131  -5.4975 3.943e-08 ***
## everwork        12.1681     5.6126   2.1680 0.0301820 *
## yr_work         41.2189     5.5136   7.4758 8.286e-14 ***
## currjob         18.0755     5.3711   3.3653 0.0007673 ***
## job9_12          5.6854     6.4497   0.8815 0.3780699
## earn1          -25.3734     6.2281  -4.0740 4.655e-05 ***
## earn3           24.6663     6.8610   3.5952 0.0003257 ***
## earn4           65.0470    10.9807   5.9238 3.246e-09 ***
## welf_kid       -12.3665     4.6543  -2.6570 0.0078957 **
## got_fs          -5.8926     3.9438  -1.4941 0.1351761
## potuse         -16.3902     4.1337  -3.9650 7.389e-05 ***
## livespou        -2.0699     7.6720  -0.2698 0.7873174
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
###############################################################################
# Task 4
###############################################################################

# Double Selection Procedure
```

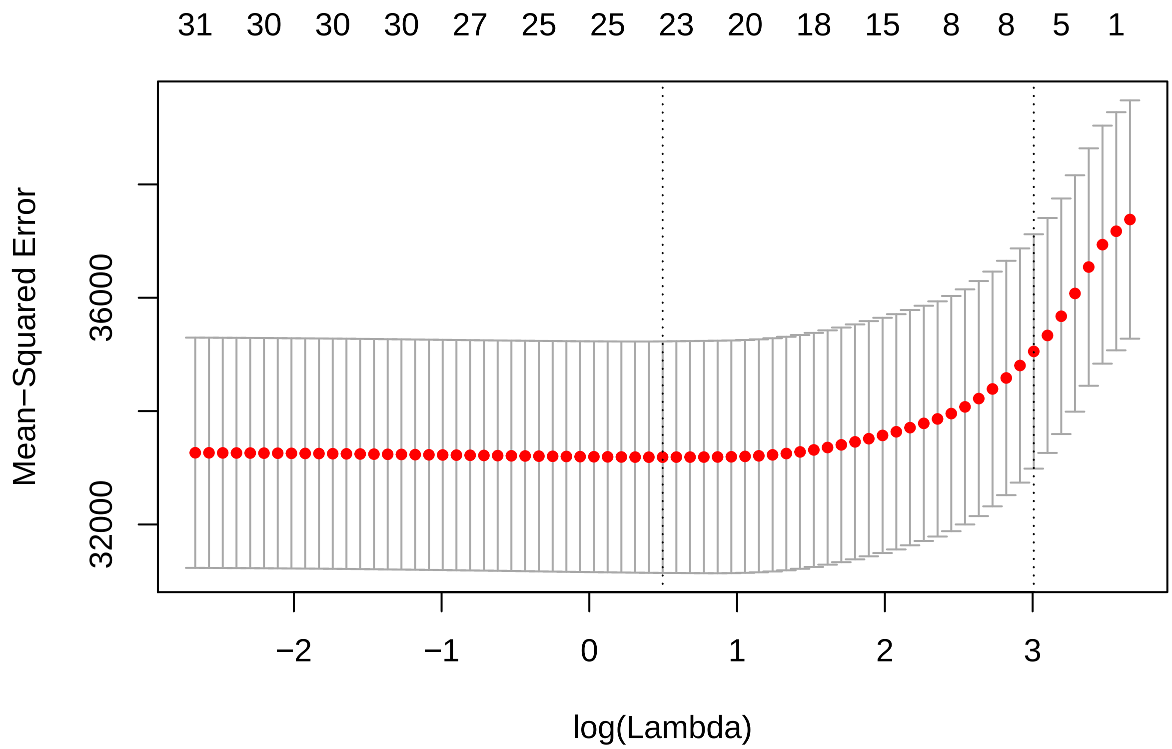
```r
dsp <- rlassoEffect(as.matrix(df[,c(4:ncol(df))]), as.matrix(df$EARNY4)
          , as.matrix(df$participation), model = TRUE, method = "double selection")
summary(dsp)
```

```
## [1] "Estimates and significance testing of the effect of target variables"
##    Estimate. Std. Error t value Pr(>|t|)
## d1    15.678      3.661   4.282 1.85e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

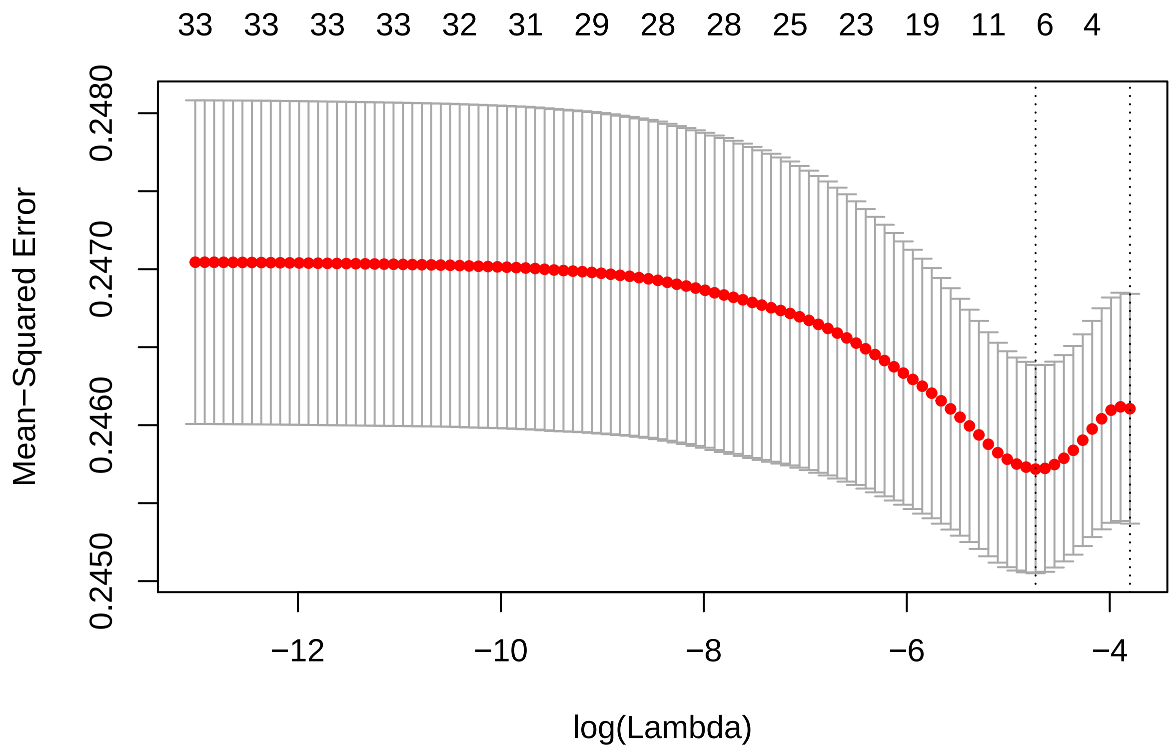
```r
###########################################
### Exercise 3: Double Machine Learning ###
###########################################

###############################################################################
# Task 1
###############################################################################
set.seed(1001)

# Partition Samples for Cross-Fitting
df_part <- modelr::resample_partition(df, c(obs_A = 0.5, obs_B = 0.5))
df_obs_A <- as.data.frame(df_part$obs_A) # Sample A
df_obs_B <- as.data.frame(df_part$obs_B) # Sample B

##  Generate Variables
# Outcome
earnings_obs_A <- as.matrix(df_obs_A[,1])
earnings_obs_B <- as.matrix(df_obs_B[,1])

# Treatment
treat = 3 #Select treatment 2= offer to participate, 3 = actual participation
treat_obs_A <- as.matrix(df_obs_A[,treat])
treat_obs_B <- as.matrix(df_obs_B[,treat])

# Covariates
covariates_obs_A <- as.matrix(df_obs_A[,c(4:ncol(df_obs_A))])
covariates_obs_B <- as.matrix(df_obs_B[,c(4:ncol(df_obs_B))])

###############################################################################
# Task 2
###############################################################################

##  Conditional Potential Earnings under Non-Participation
p = 1 # 1 for LASSO, 0 for Ridge
set.seed(100237)

## Using Sample A to Predict Sample B
# Potential Earnings under Non-Treatment
lasso_y0_A <- cv.glmnet(covariates_obs_A[treat_obs_A==0,], earnings_obs_A[treat_obs_A==0,],
                              alpha=p, type.measure = 'mse')
plot(lasso_y0_A)
```



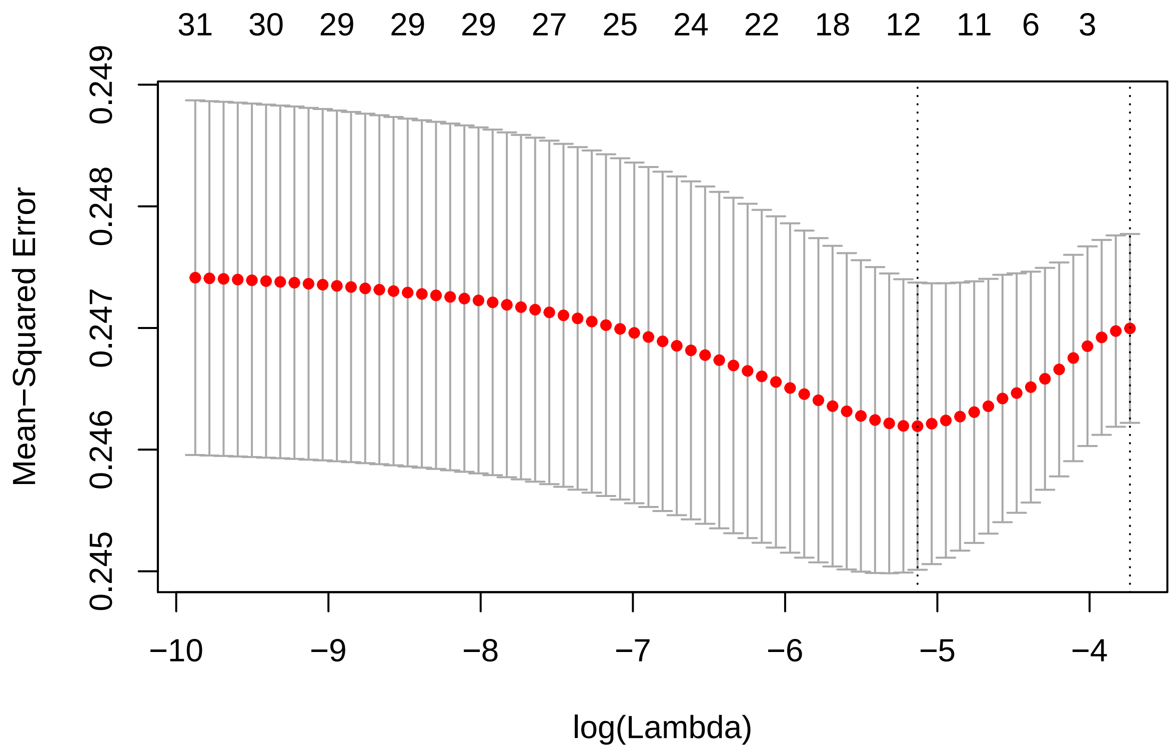

```
fit_y0_A <- glmnet(covariates_obs_A[treat_obs_A==0,], earnings_obs_A[treat_obs_A==0,]
                   ,lambda = lasso_y0_A$lambda.min)
y0hat_B <- predict(fit_y0_A, covariates_obs_B)

## Using Sample B to Predict Sample A
# Potential Earnings under Non-Treatment
lasso_y0_B <- cv.glmnet(covariates_obs_B[treat_obs_B==0,], earnings_obs_B[treat_obs_B==0,],
                        alpha=p, type.measure = 'mse')
plot(lasso_y0_B)
```




```r
fit_y0_B <- glmnet(covariates_obs_B[treat_obs_B==0,], earnings_obs_B[treat_obs_B==0,]
                        ,lambda = lasso_y0_B$lambda.min)
y0hat_A <- predict(fit_y0_B, covariates_obs_A, type = 'response')

################################################################################
# Task 3
################################################################################

##  Propensity Score
p = 1 # 1 for LASSO, 0 for Ridge
set.seed(100236)

# Using Sample A to Predict Sample B
lasso_p_A <- cv.glmnet(covariates_obs_A, treat_obs_A, alpha=p, type.measure = 'mse')
plot(lasso_p_A)
```




```
fit_p_A <- glmnet(covariates_obs_A, treat_obs_A,lambda = lasso_p_A$lambda.min)
pscore_B <- predict(fit_p_A, covariates_obs_B)

# Using Sample B to Predict Sample A
lasso_p_B <- cv.glmnet(covariates_obs_B, treat_obs_B, alpha=p, type.measure = 'mse')
plot(lasso_p_B)
```




```r
fit_p_B <- glmnet(covariates_obs_B, treat_obs_B,lambda = lasso_p_B$lambda.min)
pscore_A <- predict(fit_p_B, covariates_obs_A)

################################################################################
# Task 4
################################################################################

## Efficient Score
p_A = mean(pscore_A)
p_B = mean(pscore_B)

# Generate Modified Outcome in each sample
Y_star_A = invisible(treat_obs_A*(earnings_obs_A - y0hat_A)/p_A
            - (1-treat_obs_A)*pscore_A*(earnings_obs_A - y0hat_A)/(p_A*(1-pscore_A)))

Y_star_B = invisible(treat_obs_B*(earnings_obs_B - y0hat_B)/p_B
            - (1-treat_obs_B)*pscore_B*(earnings_obs_B - y0hat_B)/(p_B*(1-pscore_B)))

Y_star <- 0.5*(mean(Y_star_A) + mean(Y_star_B))

################################################################################
# Task 5
################################################################################

# Average Treatment Effect for Treated (ATET)
ATET <- round(Y_star, digits=3)
```

```r
# Estimate variance for each sample
var_A = mean((Y_star_A- treat_obs_A*Y_star/p_A)^2)
var_B = mean((Y_star_B - treat_obs_B*Y_star/p_B)^2)

# Split sample estimator for standard error
SD_ATET <- round(sqrt(0.5*(var_A + (mean(Y_star_A) - Y_star)^2
                    + var_B + (mean(Y_star_B) - Y_star)^2)
                    /(length(Y_star_A) + length(Y_star_B)))),digits=3)

print(paste0("Average Treatment Effect for Treated (ATET): ", ATET))
```

```
## [1] "Average Treatment Effect for Treated (ATET): 15.375"
```

```r
print(paste0("Standard Error for ATET: ", SD_ATET))
```

```
## [1] "Standard Error for ATET: 3.663"
```