# Assignment 3
# Econometrics I

Universidad Carlos III de Madrid

*Gabriel Merlo*

## 1. Predicting wages in USA

```
# install (if missing) packages
list_packages <- c("dplyr", "formatR", "glmnet", "hdm", "Hmisc", "oaxaca")
new_packages <- list_packages[!(list_packages %in% installed.packages()[,"Package"])]
if(length(new_packages)) install.packages(new_packages)

# Load packages
sapply(list_packages, require, character.only = TRUE)
```

### (a) Load, prepare and summarize the data

```
## Load prepare and summarize the data

# Load Census data from the US for the year 2012
data("cps2012")

# Check data and summarize
str(cps2012)
summary(cps2012) #no missing obs.
head(cps2012)
describe(cps2012)
```

### (b) Apply Ridge-Regression with cross-validation (CV)

The ridge regression coefficient estimates $\hat{\beta}^R$ are the values that minimize

$$\sum_{i=1}^{n}\left(y_i-\beta_0-\sum_{j=1}^{p}\beta_j x_{ij}\right)^2+\lambda\sum_{j=1}^{p}\beta_j^2$$

Where $\lambda \geq 0$ is a tuning parameter that shrinks the estimates of $\beta_j$ towards zero as $\lambda \to \infty$ (with the extreme case were all coefficient estimates are zero and the model contains no predictors). When $\lambda = 0$ the ridge coefficient estimates are the same as least squares (OLS) estimates.

Implementing ridge regression requires a method for selecting a value for $\lambda$. This selection can be done using cross-validation (CV). A grid of values for $\lambda$ is chosen and the CV error for each value is computed. The optimal value of $\lambda$ is that for which the CV error is smallest.

It's important to note that by default, the `glmnet()` function standardizes the variables so that they are on the same scale. This is done because ridge regression coefficients are depending on the scaling of the predictors.

**(i) Apply ridge regression to the previous dataset for the the default grid of values of lambda. Plot the 10-fold CV MSE as a function of lambda.**

```
# Set seed
set.seed(143)

# Define training set (75% of total data)
train <- cps2012 %>% sample_frac(0.7)

# Define test set
test <- cps2012 %>% setdiff(train)

# Define dependent variable y: lnw
y_train <- train %>% select(lnw) %>% unlist() %>% as.numeric()
y_test <- test %>% select(lnw) %>% unlist() %>% as.numeric()

# Define matrix of predictors
x_train <- train %>% select(female, widowed, divorced, separated, nevermarried, hsd08,
    hsd911, hsg, cg, ad, mw, so, we, exp1, exp2, exp3) %>% data.matrix()

x_test <- test %>% select(female, widowed, divorced, separated, nevermarried, hsd08,
    hsd911, hsg, cg, ad, mw, so, we, exp1, exp2, exp3) %>% data.matrix()

## Apply ridge regression using glmnet

# Estimate ridge model using train data and 10-fold CV (we need to set alpha = 0
# to get ridge coefficients)
cv_ridge <- cv.glmnet(x_train, y_train, alpha = 0)
summary(cv_ridge)
```

```
##            Length Class  Mode
## lambda     100    -none- numeric
## cvm        100    -none- numeric
## cvsd       100    -none- numeric
## cvup       100    -none- numeric
## cvlo       100    -none- numeric
## nzero      100    -none- numeric
## call         4    -none- call
## name         1    -none- character
## glmnet.fit  12    elnet  list
## lambda.min   1    -none- numeric
## lambda.1se   1    -none- numeric
```

```
# Optimal lambda
opt_lamb_ridge <- cv_ridge$lambda.min

# Draw plot of training MSE as a function of lambda
plot(cv_ridge)
```

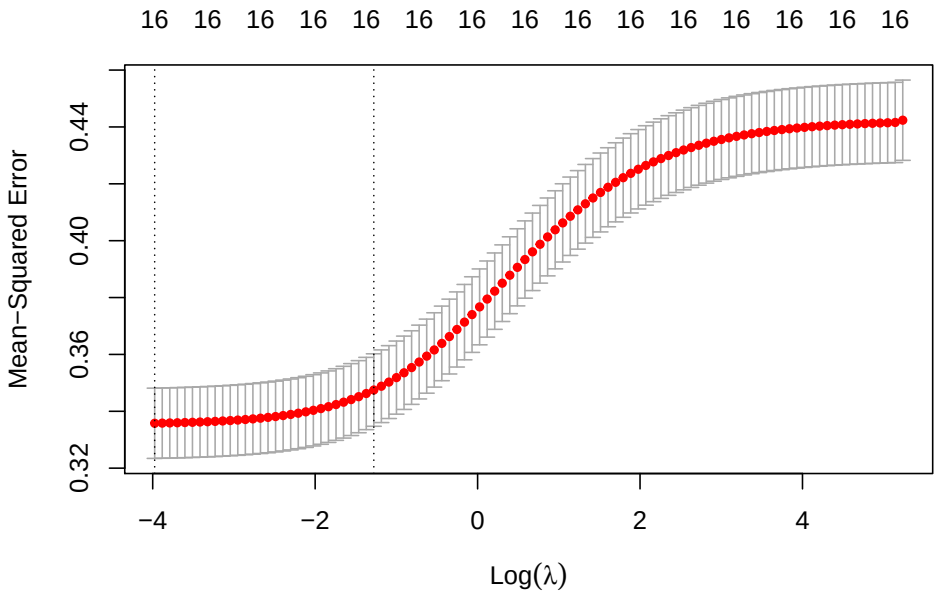


Figure 1: MSE for the ridge regression predictions, as a function of $\lambda$.

**(ii) Then, select the optimal lambda ($\lambda$) by cross-validation. How many variables are used in the Ridge fit?**

```
# Model estimation using optimal value of lambda
ridge_reg <- glmnet(x_train, y_train, alpha = 0, lambda = opt_lamb_ridge)
```

The number of variables used is 16, the same as in OLS (since ridge does not make variable selection).

**(iii) Why is the test MSE for Ridge often smaller than for OLS when lambda is not zero?**

The Mean Square Error (MSE) is a useful measure to evaluate model prediction performance. MSE will be small if the predicted values are very close to the true values observed in the data. In general, this prediction accuracy measure is calculated over unseen new data (also called test data).

In the following chunk the MSE is calculated using test data using both ridge and OLS methods.

```
## Test MSE

# Ridge
ridge_pred = predict(ridge_reg, s = opt_lamb_ridge, newx = x_test)
ridge_mse <- mean((y_test - ridge_pred)^2)
ridge_mse
```

```
## [1] 0.3301245
```

```
# OLS
ols_reg <- glmnet(x_train, y_train, alpha = 0, lambda = 0)
ols_pred = predict(ols_reg, newx = x_test)
ols_mse <- mean((y_test - ols_pred)^2)
ols_mse
```

```
## [1] 0.3295575
```

In general, ridge MSE will be smaller than OLS MSE given that ridge allows for a decrease in variance (at the cost of an increase in the bias). OLS estimates correspond to the ridge regression with $\lambda = 0$, and variance will decrease as $\lambda$ increases.

In this case $MSE_{OLS} < MSE_{ridge}$, which suggests OLS has a better adjustment than ridge, at least according to this performance measure. This is an example of why ridge is not always the best method.

**What is the optimal value of lambda? Is unrestricted OLS optimal here, in a test MSE sense?**

```
# Optimal value of lambda
opt_lamb_ridge
```

```
## [1] 0.01876455
```

The optimal lambda is $\lambda^* = 0.0188$. According to the MSE values found previously, OLS provides a better fit in terms of test data.

## (c) Apply LASSO with cross-validation (CV):

### (i) Apply Lasso regression to the previous dataset for the the default grid of values of lambda. Plot the 10-fold CV MSE as a function of lambda.

Lasso is an alternative to ridge regression that allows for variable selection. The lasso coefficients $\hat{\beta}_\lambda^L$ minimize the quantity

$$\sum_{i=1}^{n}\left(y_i - \beta_0 - \sum_{j=1}^{p}\beta_j x_{ij}\right)^2 + \lambda\sum_{j=1}^{p}|\beta_j|$$

As with the ridge regression, lasso shrinks the coefficients towards zero with the difference that the penalty term can force some of the coefficient estimates to be exactly equal to zero when $\lambda$ is sufficiently large.

```
# Estimate lasso model using train data and 10-fold CV (we need to set alpha = 1
# to get lasso coefficients)
cv_lasso <- cv.glmnet(x_train, y_train, alpha = 1)
summary(cv_lasso)
```

```
##            Length Class  Mode
## lambda     91     -none- numeric
## cvm        91     -none- numeric
## cvsd       91     -none- numeric
## cvup       91     -none- numeric
## cvlo       91     -none- numeric
## nzero      91     -none- numeric
## call        4     -none- call
## name        1     -none- character
## glmnet.fit 12     elnet  list
## lambda.min  1     -none- numeric
## lambda.1se  1     -none- numeric
```

```
# Optimal lambda
opt_lamb_lasso <- cv_lasso$lambda.min

# Draw plot of training MSE as a function of lambda
plot(cv_lasso)
```

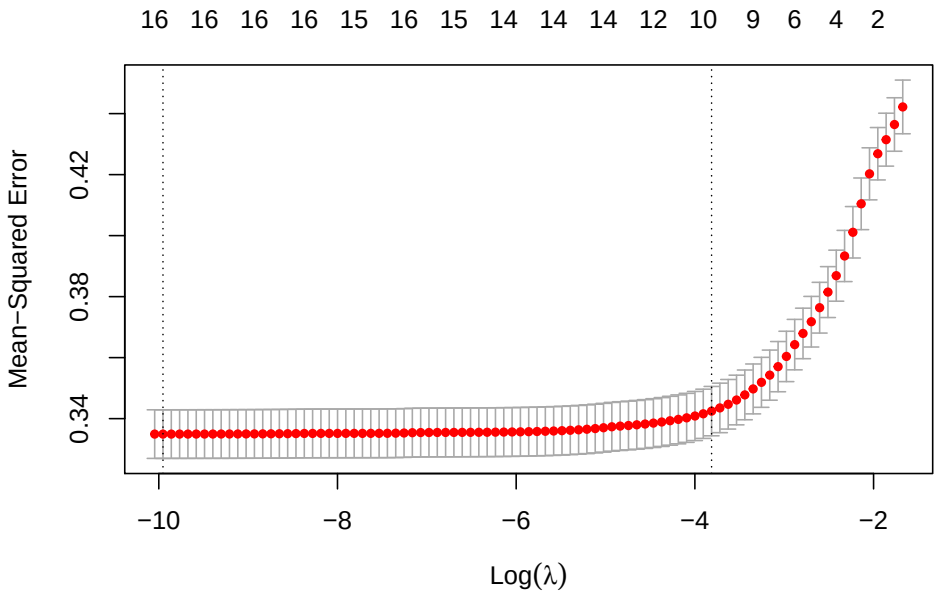


Figure 2: MSE for the lasso regression predictions, as a function of $\lambda$.

**(ii) Then, select the optimal lambda ($\lambda$) by cross-validation. What is the optimal lambda?**

```
# Optimal value of lambda
opt_lamb_lasso
```

```
## [1] 4.757496e-05
```

The optimal lambda is $\lambda^* = 0.000048$.

**(iii) How many variables are used in the optimal Lasso fit? What are their coefficients? Is there a big difference here between Ridge and Lasso (in terms of test MSE)?**

Estimated coefficients:

```
# Model estimation using optimal value of lambda
lasso_reg <- glmnet(x_train, y_train, alpha = 1, lambda = opt_lamb_lasso)
coef(lasso_reg)
```

```
## 17 x 1 sparse Matrix of class "dgCMatrix"
##                       s0
## (Intercept)   2.46138828
## female       -0.28546556
## widowed      -0.15207362
## divorced     -0.08674552
## separated    -0.08905415
## nevermarried -0.14283348
## hsd08        -0.57742882
## hsd911       -0.37721101
## hsg          -0.16841600
## cg            0.35727754
## ad            0.61207752
```

```
## mw           -0.10374091
## so           -0.05537310
## we           -0.01063416
## exp1          0.04488948
## exp2         -0.14929353
## exp3          0.01768608
```

```
## Test MSE

# Lasso
lasso_pred = predict(lasso_reg, s = opt_lamb_lasso, newx = x_test)
lasso_mse <- mean((y_test - lasso_pred)^2)
lasso_mse
```

```
## [1] 0.329565
```

Lasso regression with the optimal value of $\lambda$ found estimates the model using all variables.

In terms of prediction power, $MSE_{lasso} = 0.329565 < 0.3301245 = MSE_{ridge}$ which suggests that lasso has a better fit than ridge model.

**(iv) Which method of prediction would you choose and why? Is gender an important factor in the prediction model? Interpret the coeffcient of female.**

As stated before, lasso has the advantage of reducing the amount of parameters used for estimation. This is particularly useful when working with a big number of predictors and when the associated coefficient for many of them is close to zero (which means they are not particularly relevant). Considering MSE of lasso is smaller than ridge, lasso seems to be a better approach to estimate the model.

The female coefficient indicates that on average, women earn 28% less than men (keeping all the other variables constant).

## (d) Repeat (b) and (c) for a more flexible specification: You would like to analyse the effect of gender and interaction effects of other variables with gender on wage jointly. The dependent variable is still the logarithm of the wage.

```
# Relevant variables
x_var <- c("widowed", "divorced", "separated", "nevermarried", "hsd08", "hsd911",
    "hsg", "cg", "ad", "mw", "so", "we", "exp1", "exp2", "exp3")

# Generate x matrix with original variables and their interaction with female
int <- cps2012[, x_var] * cps2012$female

# Rename interaction variables
colnames(int) <- paste("int_female", colnames(int), sep = "_")

# Bind interaction variables to the original variables
cps2012_int <- cbind(cps2012[, c("lnw", "female", x_var)], int)

# Set seed
set.seed(143)

# Define training set (75% of total data)
train1 <- cps2012_int %>% sample_frac(0.7)

# Define test set
```

```r
test1 <- cps2012_int %>% setdiff(train1)

# Define dependent variable y: lnw
y_train1 <- train1 %>% select(lnw) %>% unlist() %>% as.numeric()
y_test1 <- test1 %>% select(lnw) %>% unlist() %>% as.numeric()

# Define matrix of predictors
x_train1 <- train1[, -1] %>% data.matrix()

x_test1 <- test1[, -1] %>% data.matrix()

## Apply ridge regression using glmnet and interaction variables

# Estimate ridge model using train data and 10-fold CV
cv_ridge_int <- cv.glmnet(x_train1, y_train1, alpha = 0)
summary(cv_ridge_int)
```

```
##            Length Class  Mode
## lambda     100    -none- numeric
## cvm        100    -none- numeric
## cvsd       100    -none- numeric
## cvup       100    -none- numeric
## cvlo       100    -none- numeric
## nzero      100    -none- numeric
## call         4    -none- call
## name         1    -none- character
## glmnet.fit  12    elnet  list
## lambda.min   1    -none- numeric
## lambda.1se   1    -none- numeric
```

```r
# Optimal lambda
opt_lamb_ridge_int <- cv_ridge_int$lambda.min

# Draw plot of training MSE as a function of lambda
plot(cv_ridge_int)
```

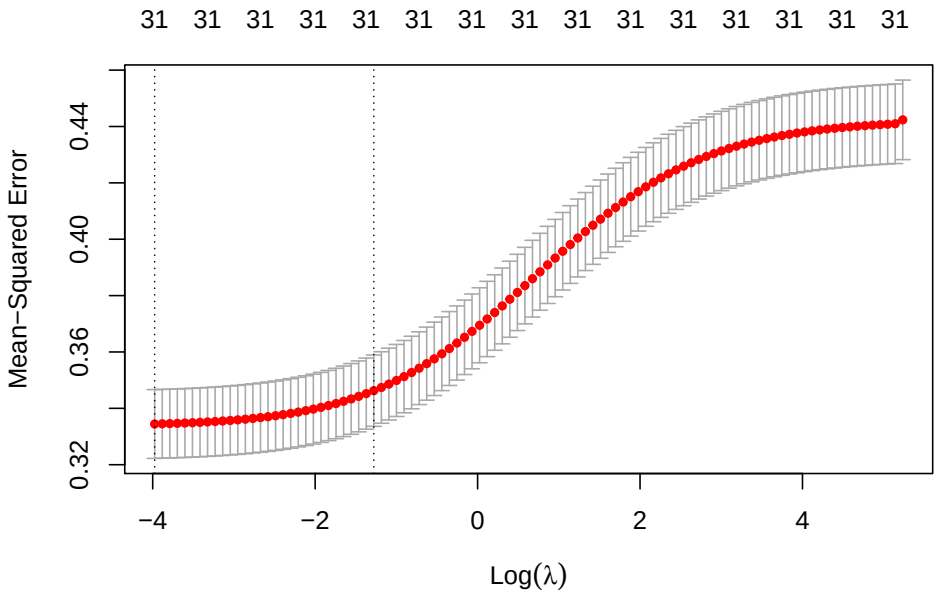


Figure 3: MSE for the ridge regression predictions (with interactions), as a function of $\lambda$.

```
# Optimal value of lambda
opt_lamb_ridge_int
```

```
## [1] 0.01876455
```

```
# Model estimation using optimal value of lambda
ridge_reg_int <- glmnet(x_train1, y_train1, alpha = 0, lambda = opt_lamb_ridge_int)
coef(ridge_reg_int)
```

```
## 32 x 1 sparse Matrix of class "dgCMatrix"
##                                      s0
## (Intercept)               2.597524087
## female                   -0.227474999
## widowed                  -0.188431461
## divorced                 -0.140164137
## separated                -0.073932970
## nevermarried             -0.214818786
## hsd08                    -0.515166784
## hsd911                   -0.344199014
## hsg                      -0.163167561
## cg                        0.327086019
## ad                        0.574082066
## mw                       -0.086053200
## so                       -0.041891821
## we                        0.011021288
## exp1                      0.017778883
## exp2                     -0.001123886
## exp3                     -0.005368745
## int_female_widowed        0.081015986
## int_female_divorced       0.112244660
```

```
## int_female_separated    -0.019343478
## int_female_nevermarried  0.148166136
## int_female_hsd08        -0.166621128
## int_female_hsd911       -0.113301571
## int_female_hsg          -0.016295822
## int_female_cg            0.014336011
## int_female_ad            0.008635434
## int_female_mw           -0.033043415
## int_female_so           -0.026024412
## int_female_we           -0.044703406
## int_female_exp1         -0.003059803
## int_female_exp2         -0.004736260
## int_female_exp3          0.001274730
```

The optimal lambda is $\lambda^* = 0.0188$, same value as the model without interactions. We can interpret this as an indication that the interaction variables do not help to improve the predictions.

```
## Apply lasso regression using glmnet and interaction variables

# Estimate lasso model using train data and 10-fold CV
cv_lasso_int <- cv.glmnet(x_train1, y_train1, alpha = 1)
summary(cv_ridge_int)
```

```
##            Length Class  Mode
## lambda     100    -none- numeric
## cvm        100    -none- numeric
## cvsd       100    -none- numeric
## cvup       100    -none- numeric
## cvlo       100    -none- numeric
## nzero      100    -none- numeric
## call         4    -none- call
## name         1    -none- character
## glmnet.fit  12    elnet  list
## lambda.min   1    -none- numeric
## lambda.1se   1    -none- numeric
```

```
# Optimal lambda
opt_lamb_lasso_int <- cv_lasso_int$lambda.min

# Draw plot of training MSE as a function of lambda
plot(cv_lasso_int)
```

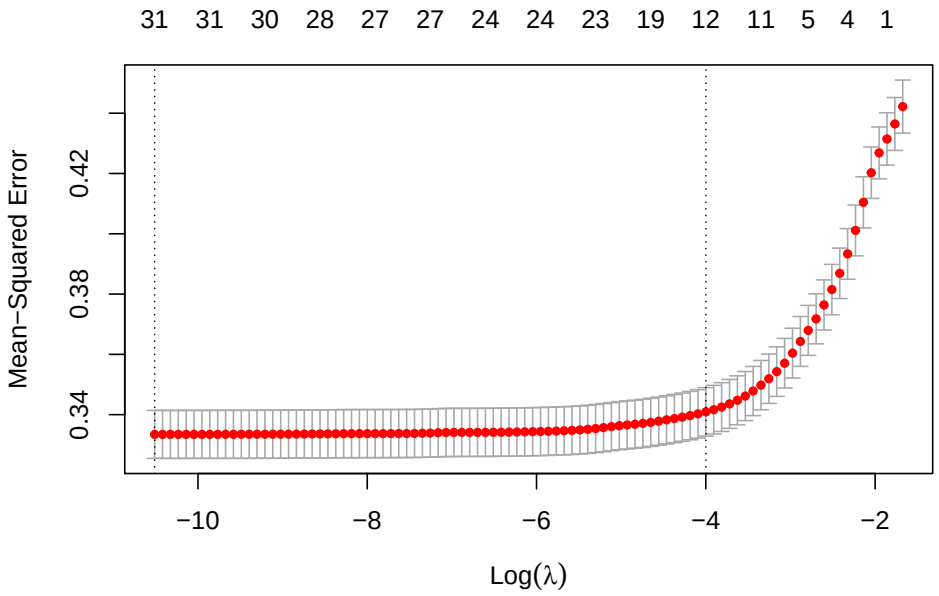


Figure 4: MSE for the lasso regression predictions (with interactions), as a function of $\lambda$.

```
# Optimal value of lambda
opt_lamb_lasso_int
```

```
## [1] 2.722414e-05
```

```
# Model estimation using optimal value of lambda
lasso_reg_int <- glmnet(x_train1, y_train1, alpha = 1, lambda = opt_lamb_lasso_int)
coef(lasso_reg_int)
```

```
## 32 x 1 sparse Matrix of class "dgCMatrix"
##                                      s0
## (Intercept)              2.4084893579
## female                  -0.2187056382
## widowed                 -0.2117191176
## divorced                -0.1546883090
## separated               -0.0829476825
## nevermarried            -0.2137916179
## hsd08                   -0.5259472497
## hsd911                  -0.3475076060
## hsg                     -0.1602834987
## cg                       0.3527853034
## ad                       0.6099085689
## mw                      -0.1000024365
## so                      -0.0557692320
## we                      -0.0013960553
## exp1                     0.0505323338
## exp2                    -0.1534501781
## exp3                     0.0155533405
## int_female_widowed       0.1098356064
## int_female_divorced      0.1364632184
```

```
## int_female_separated     0.0016441071
## int_female_nevermarried  0.1709034814
## int_female_hsd08        -0.2318141271
## int_female_hsd911       -0.1194630824
## int_female_hsg          -0.0127392058
## int_female_cg            0.0113209886
## int_female_ad           -0.0003418542
## int_female_mw           -0.0058943688
## int_female_so            0.0018014328
## int_female_we           -0.0194502652
## int_female_exp1         -0.0037001570
## int_female_exp2         -0.0422584088
## int_female_exp3          0.0135745694
```

The estimated coefficients suggest that many variables and interactions are not very good predictors of salaries. Among the interactions, it seems that being female and divorced, widowed or never married increase the mean salary with respect to married. Having a low level of education and being a woman also affects salary negatively.

Let's study prediction performance using MSE for test data.

```
## Test MSE

# Ridge
ridge_int_pred = predict(ridge_reg_int, s = opt_lamb_ridge_int, newx = x_test1)
ridge_int_mse <- mean((y_test1 - ridge_int_pred)^2)
ridge_int_mse
```

```
## [1] 0.353661
```

```
# Lasso
lasso_int_pred = predict(lasso_reg_int, s = opt_lamb_lasso_int, newx = x_test1)
lasso_int_mse <- mean((y_test1 - lasso_int_pred)^2)
lasso_int_mse
```

```
## [1] 0.3529753
```

In this case again we can see that lasso performs better than ridge using new unseen data.

## (e) Based on the variable selection provided by Lasso, study the Gender Pay Gap (GPG) with this data set, including an Oaxaca-Blinder decomposition for the GPG.

The variable selection used is the one related to the first LASSO regression made in part 1-c)(i)

```
# Remembering the coefficients needed
coef(cv_lasso)
```

```
## 17 x 1 sparse Matrix of class "dgCMatrix"
##                          1
## (Intercept)   2.684226838
## female       -0.246628273
## widowed       .
## divorced     -0.012803194
## separated     .
## nevermarried -0.114111137
## hsd08        -0.236665567
## hsd911       -0.232624863
```

```
## hsg          -0.136486893
## cg            0.296601503
## ad            0.537433576
## mw           -0.034454770
## so            .
## we            .
## exp1          0.006276259
## exp2          .
## exp3          .
```

```
# Regression used for Oaxaca Blinder decomposition
reg_formula <- lnw ~ divorced + nevermarried + hsd08 + hsd911 + hsg + cg + ad + mw + exp1 | female

# Oaxaca Blinder decomposition
oaxaca_dec <- oaxaca(formula = reg_formula, data = cps2012)

# Result of estimation
summary.oaxaca(oaxaca_dec$beta)
```

```
## $beta.A
##  (Intercept)     divorced nevermarried        hsd08       hsd911          hsg
##   2.62914109  -0.14339563  -0.22109175  -0.62245678  -0.36482106  -0.16820464
##           cg           ad           mw         exp1
##   0.34702991   0.60649845  -0.08237793   0.01186731
##
## $beta.B
##  (Intercept)     divorced nevermarried        hsd08       hsd911          hsg
##  2.366748561 -0.003282422 -0.040192411 -0.720570323 -0.511075232 -0.190742391
##           cg           ad           mw         exp1
##  0.353458948  0.584447708 -0.077648490  0.008607443
##
## $beta.diff
##  (Intercept)     divorced nevermarried        hsd08       hsd911          hsg
##  0.262392524 -0.140113209 -0.180899340  0.098113541  0.146254172  0.022537750
##           cg           ad           mw         exp1
## -0.006429038  0.022050743 -0.004729440  0.003259865
##
## $beta.R
##                 (Intercept)     divorced nevermarried      hsd08     hsd911
## [1,]  0.0000000    2.366749 -0.003282422  -0.04019241 -0.7205703 -0.5110752
## [2,]  1.0000000    2.629141 -0.143395631  -0.22109175 -0.6224568 -0.3648211
## [3,]  0.5000000    2.497945 -0.073339026  -0.13064208 -0.6715136 -0.4379481
## [4,]  0.5712428    2.516638 -0.083321079  -0.14352985 -0.6645237 -0.4275286
## [5,] -1.0000000    2.521679 -0.118445513  -0.15297437 -0.5967423 -0.3583868
## [6,] -2.0000000    2.642568 -0.072362355  -0.14303034 -0.6482486 -0.4069420
##             hsg        cg        ad          mw        exp1
## [1,] -0.1907424 0.3534589 0.5844477 -0.07764849 0.008607443
## [2,] -0.1682046 0.3470299 0.6064985 -0.08237793 0.011867308
## [3,] -0.1794735 0.3502444 0.5954731 -0.08001321 0.010237376
## [4,] -0.1778679 0.3497864 0.5970440 -0.08035015 0.010469618
## [5,] -0.1592192 0.3500703 0.5921838 -0.08682056 0.010464771
## [6,] -0.1806783 0.3504434 0.5997703 -0.08112082 0.010423112
```