Introduction to R Programming

Ozan Bakış[1]
[1]Bahcesehir University, Department of Economics and BETAM

Outline

1. Regression with cross-sectional data

Multiple regression: notation I

- linear regression model (estimated by OLS):

$$y_i = \beta_0 + \beta_1 x_{i1} + \ldots + \beta_k x_{ik} + u_i, \quad i = 1, \ldots, n.$$

- **Application:** estimation of wage equation using DCPS1988 data from AER (Applied Econometrics with R) package. CPS= Current Population Survey. $\Rightarrow$ cross-section data on male workers (excluding self-employment and ed unpaid family work) aged 18 to 70 with positive annual income.

```r
f_url = "https://github.com/obakis/econ_data/raw/master/hls2011.rds"
download.file(url = f_url, destfile = "hls2011.rds", mode="wb")
hls = readRDS("hls2011.rds")
```

- Before regression let us look our variables of interest.

Multiple regression: notation II

```
head(hls,3)
```

```
##   id exper educ emp_sect emp_type hwage nuts1   wts urban female
## 1  1    33    2      pub   f-time  8.75    N9  45.8     1      0
## 2  2     2    2     priv   f-time  2.92   N12 178.8     0      1
## 3  3    22    5     priv   f-time  2.53    N2  66.9     1      1
```

```
vars = c("hwage","educ", "female","exper","emp_sect")
str(hls[,vars])
```

```
## 'data.frame': 762 obs. of  5 variables:
##  $ hwage   : num  8.75 2.92 2.53 58.33 3.89 ...
##  $ educ    : int  2 2 5 15 8 8 5 15 5 15 ...
##  $ female  : int  0 1 1 0 0 0 0 1 0 1 ...
##  $ exper   : int  33 2 22 21 16 49 22 6 17 2 ...
##  $ emp_sect: Factor w/ 3 levels "other","priv",..: 3 2 2 2 2 2 2 2 2 3 ...
```

Multiple regression: notation III

- Note that `emp_sect` is a `factor` variable with 12 levels. In R, categorical (nominal) and ordered categorical (ordinal) variables are called `factor`s. Each possible value of a categorical variable is called a level. In a regression a set of dummy variables will be automatically created by R. More precisely, if we have n groups/levels, $n-1$ dummy variables will be created.

```
r2e_1 = lm(log(hwage) ~ exper + I(exper^2) + educ + emp_sect, data=hls)
```

- Operators `+,-,:,*,/,^` have special meanings in a `formula` object. To ensure arithmetic meaning, we need either to protect by insulation in a function, e.g., `log(x1 * x2)` or to use `I()` function.

```
summary(r2e_1)
```

Multiple regression: notation IV

```
##
## Call:
## lm(formula = log(hwage) ~ exper + I(exper^2) + educ + emp_sect,
##     data = hls)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2.1542 -0.2863 -0.0271  0.2702  2.1929
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  2.51e-01   1.63e-01    1.54     0.12
## exper        3.17e-02   4.56e-03    6.96  7.6e-12 ***
## I(exper^2)  -4.93e-04   9.52e-05   -5.18  2.9e-07 ***
## educ         7.18e-02   4.57e-03   15.70  < 2e-16 ***
## emp_sectpriv 1.15e-01   1.52e-01    0.76     0.45
## emp_sectpub  7.59e-01   1.57e-01    4.84  1.5e-06 ***
```

Multiple regression: notation V

```
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.447 on 756 degrees of freedom
## Multiple R-squared:  0.569,Adjusted R-squared:  0.567
## F-statistic:  200 on 5 and 756 DF,  p-value: <2e-16
```

- Generic functions related to `lm` object (See `help(lm)` and `names(cps.lm)` for details):

`print()`	simple printed display
`summary()`	standard regression output
`coef()`	(or `coefficients()`) extract regression coefficients
`residuals()`	(or `resid()`) extract residuals
`fitted()`	(or `fitted.values()`) extract fitted values
`predict()`	predictions for new data
`plot()`	diagnostic plots
`confint()`	confidence intervals for the regression coefficients
`AIC()`	information criteria including AIC, BIC/SBC

Multiple regression: notation VI

- The `lm()` command, relies on `model.matrix()` for the creation of dummy variables.

```
dummy <- factor(LETTERS[1:4])
model.matrix( ~ dummy)
```

```
##   (Intercept) dummyB dummyC dummyD
## 1           1      0      0      0
## 2           1      1      0      0
## 3           1      0      1      0
## 4           1      0      0      1
## attr(,"assign")
## [1] 0 1 1 1
## attr(,"contrasts")
## attr(,"contrasts")$dummy
## [1] "contr.treatment"
```

- To change the base level of a factor variable (ex. ”region” variable) we can use `relevel` function

Multiple regression: notation VII

```r
table(hls$emp_sect)
```

```
##
## other  priv   pub
##     9   557   196
```

```r
levels(hls$emp_sect)
```

```
## [1] "other" "priv"  "pub"
```

```r
contrasts(hls$emp_sect) #other is base level
```

```
##       priv pub
## other    0   0
## priv     1   0
## pub      0   1
```

```r
hls$emp_sect <- relevel(hls$emp_sect, ref = "pub")
r2e_2 <- update(r2e_1, formula = . ~ .) ## we change nothing here!
summary(r2e_2)$coef
```

Multiple regression: notation VIII

```
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)    1.009864   9.47e-02   10.66 8.14e-25
## exper          0.031745   4.56e-03    6.96 7.62e-12
## I(exper^2)    -0.000493   9.52e-05   -5.18 2.86e-07
## educ           0.071799   4.57e-03   15.70 2.70e-48
## emp_sectother -0.758748   1.57e-01   -4.84 1.54e-06
## emp_sectpriv  -0.643402   4.36e-02  -14.74 1.98e-43
```

`update()` is used for updating an **`lm`** object. Since we do not change the LHS or the RHS of the **`formula`**, above our goal is just re-doing the same regression with new base level for **`region`** variable.

- What if we want to add or remove some variables

```
r2e_3 <- update(r2e_2, formula = . ~ . - emp_sect)
summary(r2e_3)$coef
```

Multiple regression: notation IX

```
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.014760   0.075683   0.195 8.45e-01
## exper        0.046859   0.005032   9.313 1.32e-19
## I(exper^2)  -0.000696   0.000106  -6.550 1.06e-10
## educ         0.107217   0.004419  24.263 1.11e-96
```

```
r2e_4 <- update(r2e_3, formula = . ~ . + female)
summary(r2e_4)$coef
```

```
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.014430   0.076451  0.1887 8.50e-01
## exper        0.046877   0.005068  9.2504 2.24e-19
## I(exper^2)  -0.000696   0.000107 -6.5258 1.24e-10
## educ         0.107196   0.004472 23.9696 6.51e-95
## female       0.001440   0.045617  0.0316 9.75e-01
```

```
confint(r2e_4) # by default: level = 0.95
```

Multiple regression: notation X

```
##                   2.5 %    97.5 %
## (Intercept) -0.135650  0.164510
## exper        0.036929  0.056825
## I(exper^2)  -0.000905 -0.000486
## educ         0.098417  0.115975
## female      -0.088111  0.090990
```

```r
confint(r2e_4, level=0.9)
```

```
##                     5 %     95 %
## (Intercept) -0.111474  0.14033
## exper        0.038531  0.05522
## I(exper^2)  -0.000871 -0.00052
## educ         0.099831  0.11456
## female      -0.073685  0.07656
```

Interactions I

Formula	Description
`y~a+x`	Model without interaction: identical slopes with respect to `x` but different intercepts with respect to `a`.
`y~a*x` `y~a+x+a:x`	Model with interaction: the term `a:x` gives the difference in slopes compared with the reference category.
`y~a/x` `y~a+x%in%a`	Model with interaction: produces the same fitted values as the model above but using a nested coefficient coding. An explicit slope estimate is computed for each category in `a`.

Interactions II

```r
#install.packages("lmtest")
library(lmtest) # for inference
# need to convert female into factor variable
hls$female=factor(hls$female)

## main effects + interaction
r2e_5 = lm(log(hwage) ~ exper + I(exper^2) + educ*female, data=hls)
coeftest(r2e_5)

##
## t test of coefficients:
##
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)   0.044031   0.079164    0.56     0.58
## exper         0.046917   0.005064    9.26  < 2e-16 ***
## I(exper^2)   -0.000695   0.000107   -6.53  1.2e-10 ***
## educ          0.103704   0.005095   20.35  < 2e-16 ***
```

Interactions III

```
## female1       -0.142456   0.110656   -1.29     0.20
## educ:female1   0.013875   0.009722    1.43     0.15
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
## nested models
r2e_6 = lm(log(hwage) ~ female/(0+exper + I(exper^2) + educ), data=hls)
coeftest(r2e_6)
```

```
##
## t test of coefficients:
##
##                       Estimate Std. Error t value Pr(>|t|)
## female0               0.075992   0.090083    0.84     0.40
## female1              -0.233356   0.153313   -1.52     0.13
## female0:exper         0.046082   0.006650    6.93  9.0e-12 ***
## female1:exper         0.052638   0.008358    6.30  5.1e-10 ***
```

Interactions IV

```
## female0:I(exper^2) -0.000708   0.000144   -4.90  1.2e-06 ***
## female1:I(exper^2) -0.000707   0.000159   -4.46  9.6e-06 ***
## female0:educ        0.102708   0.005146   19.96  < 2e-16 ***
## female1:educ        0.122231   0.009077   13.47  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

F test: linear restrictions I

- Consider the following model

```r
coeftest(r2e_4)
```

```
##
## t test of coefficients:
##
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.014430   0.076451    0.19     0.85
## exper        0.046877   0.005068    9.25  < 2e-16 ***
## I(exper^2)  -0.000696   0.000107   -6.53  1.2e-10 ***
## educ         0.107196   0.004472   23.97  < 2e-16 ***
## female       0.001440   0.045617    0.03     0.97
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

- We want to test $H_0 : \beta_3 = 0.07, \beta_4 = 0$. These are called exclusion restrictions.

F test: linear restrictions II

```r
#install.packages("car")
library(car)
linearHypothesis(r2e_4, c("educ=0.07","female=0")) # reject null
```

```
## Linear hypothesis test
##
## Hypothesis:
## educ = 0.07
## female = 0
##
## Model 1: restricted model
## Model 2: log(hwage) ~ exper + I(exper^2) + educ + female
##
##   Res.Df RSS Df Sum of Sq    F Pr(>F)
## 1    759 213
## 2    757 194  2      18.2 35.4  2e-15 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

F test: linear restrictions III

```r
linearHypothesis(r2e_4, "educ - 2*exper  = 0") # cannot reject null
```

```
## Linear hypothesis test
##
## Hypothesis:
## - 2 exper  + educ = 0
##
## Model 1: restricted model
## Model 2: log(hwage) ~ exper + I(exper^2) + educ + female
##
##   Res.Df RSS Df Sum of Sq    F Pr(>F)
## 1    758 195
## 2    757 194  1     0.441 1.72   0.19
```

Heteroskedasticity robust std. erros I

```r
library(lmtest) # for coeftest
coeftest(r2e_4) # assuming homoskedasticity
```

```
##
## t test of coefficients:
##
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.014430   0.076451    0.19     0.85
## exper        0.046877   0.005068    9.25  < 2e-16 ***
## I(exper^2)  -0.000696   0.000107   -6.53  1.2e-10 ***
## educ         0.107196   0.004472   23.97  < 2e-16 ***
## female       0.001440   0.045617    0.03     0.97
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Heteroskedasticity robust std. erros II

```r
library(sandwich) # for vcovHC
coeftest(r2e_4, vcov = vcovHC) # heteroskedasticity robust, R default: "HC3"
```

```
##
## t test of coefficients:
##
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.014430   0.079666    0.18     0.86
## exper        0.046877   0.005106    9.18  < 2e-16 ***
## I(exper^2)  -0.000696   0.000108   -6.42  2.4e-10 ***
## educ         0.107196   0.005053   21.21  < 2e-16 ***
## female       0.001440   0.044464    0.03     0.97
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Heteroskedasticity robust std. erros III

```r
coeftest(r2e_4, vcov = vcovHC(r2e_4, "HC3")) # robust, R default: "HC3"
```

```
##
## t test of coefficients:
##
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.014430   0.079666    0.18     0.86
## exper        0.046877   0.005106    9.18  < 2e-16 ***
## I(exper^2)  -0.000696   0.000108   -6.42  2.4e-10 ***
## educ         0.107196   0.005053   21.21  < 2e-16 ***
## female       0.001440   0.044464    0.03     0.97
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Heteroskedasticity robust std. erros IV

```r
coeftest(r2e_4, vcov = vcovHC(r2e_4, "HC1")) # robust, Stata default: "HC1"
```

```
##
## t test of coefficients:
##
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.014430   0.078957    0.18     0.86
## exper        0.046877   0.005002    9.37  < 2e-16 ***
## I(exper^2)  -0.000696   0.000105   -6.62  6.7e-11 ***
## educ         0.107196   0.005022   21.34  < 2e-16 ***
## female       0.001440   0.044203    0.03     0.97
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```