

Pol Sci 630: Problem Set 4 - Regression Model Estimation

Prepared by: Anh Le (anh.le@duke.edu)

Due Date: Tuesday, September 22nd, 2015, 10 AM (Beginning of Class)

Note 1: It is absolutely essential that you show all your work, including intermediary steps, and comment on your R code to earn full credit. Showing all steps and commenting on code them will also be required in future problem sets.

Note 2: Please use a **single** PDF file created through knitr to submit your answers. knitr allows you to combine R code and L^AT_EX code in one document, meaning that you can include both the answers to R programming and math problems. Also submit the source code that generates the PDF file (i.e. either .Rnw or .Rmd files)

Note 3: Make sure that the PDF files you submit do not include any references to your identity. The grading will happen anonymously. You can submit your answer at the following website: <http://ps630-f15.herokuapp.com/>

1. Create a data frame

a) (2 points)

Create a data frame with 1000 rows and 3 variables:

1. `var_norm`: a normal variable with mean = 5, sd = 10
2. `var_binom`: a binomial variable with number of trial = 10, probability of success = 0.5
3. `var_poisson`: a Poisson variable with $\lambda = 4$

(Recall how to generate random sample from various distributions from previous labs.)

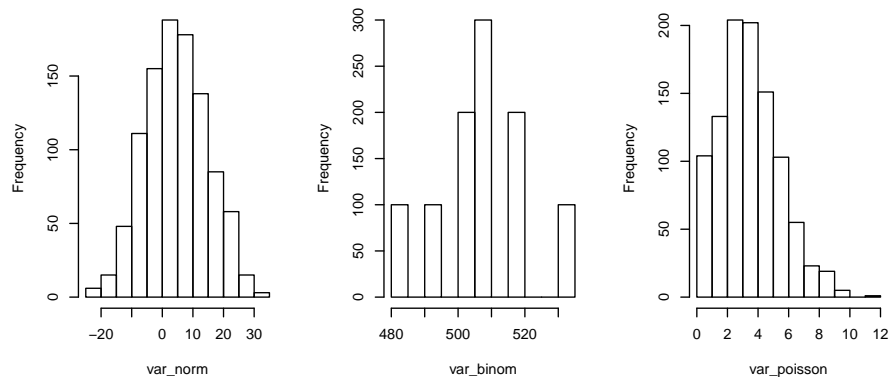
b) (2 points)

Plot the histograms of the three variables, arranging them nicely (with `fig.width()`, `fig.height()`, `par(mfrow)` as you see fit). Brownie point if you plot using a for loop instead of writing `hist` three times.

Solution

```
# Create the data frame
my_dataframe <- data.frame(var_norm = rnorm(1000, mean = 5, sd = 10),
                           var_binom = rbinom(1000, n = 10, prob = 0.5),
                           var_poisson = rpois(1000, lambda = 4))

# Plot the histogram (nicely)
par(mfrow = c(1, 3))
for (i in 1:3) {
  hist(my_dataframe[, i],
       xlab = colnames(my_dataframe)[i], main = NULL)
}
```



2. Subset data frame

a) 1 point

Download the following data from WDI and clean it as follows. Briefly comment on what each command does.

```
library(WDI)

## Loading required package: RJSONIO

d_wdi <- WDI(indicator = c("NY.GDP.PCAP.CD", "SP.DYN.IMRT.IN", "SH.MED.PHYS.ZS"),
            start = 2005, end = 2010, extra = TRUE)
d_wdi <- d_wdi[d_wdi$region != "Aggregates",
              c("country", "year", "NY.GDP.PCAP.CD", "SP.DYN.IMRT.IN", "SH.MED.PHYS.ZS")]
colnames(d_wdi)[3:5] <- c('gdppc', 'infant_mortality', 'number_of_physician')
d_wdi <- na.omit(d_wdi)
```

infant_mortality: number of mortality per 1000 live births
number_of_physician: number of physician per 1000 people

b) 3 points

Use subsetting techniques to do the following:

1. Show the GDP per capita of Brazil across years
2. Show the country-years where infant mortality > 100 per 1000 live birth
3. Show the country-years where GDP per capita is above average
4. Show the country-years where GDP per capita is above average, but number of physician is below average

Solution

```
library(WDI)

# Download data from WDI, specifying the indicators and start / end year
d_wdi <- WDI(indicator = c("NY.GDP.PCAP.CD", "SP.DYN.IMRT.IN", "SH.MED.PHYS.ZS"),
             start = 2008, end = 2010, extra = TRUE)

# Remove aggregates rows, selecting wanted columns by name
d_wdi <- d_wdi[d_wdi$region != "Aggregates",
              c("country", "year", "NY.GDP.PCAP.CD", "SP.DYN.IMRT.IN", "SH.MED.PHYS.ZS")]

# Rename some of the columns
colnames(d_wdi)[3:5] <- c('gdppc', 'infant_mortality', 'number_of_physician')

# Remove all missing data
d_wdi <- na.omit(d_wdi)

# 1. Show the GDP per capita of Brazil across years
d_wdi[d_wdi$country == "Brazil", c("country", "year", "gdppc")]

##      country year      gdppc
## 94  Brazil 2008  8836.914
## 95  Brazil 2010 11318.057

# 2. Show the country-years where infant mortality > 100 per 1000 live birth
d_wdi[d_wdi$infant_mortality > 100, c("country", "year", "infant_mortality")]

##              country year infant_mortality
## 34              Angola 2009             112.2
## 120 Central African Republic 2009             103.6
## 562              Sierra Leone 2010             107.0
## 563              Sierra Leone 2008             116.2
```

```
# 3. Show the country-years where GDP per capita is above average
d_wdi[d_wdi$gdppc > mean(d_wdi$gdppc), c("country", "year", "gdppc")]
```

```
##          country year    gdppc
## 16          Andorra 2010 42952.72
## 17          Andorra 2009 46401.09
## 20  United Arab Emirates 2010 33885.93
## 43          Austria 2010 46593.39
## 48          Australia 2010 51801.05
## 62          Barbados 2010 15854.36
## 67          Belgium 2010 44360.90
## 69          Belgium 2008 48561.36
## 76          Bahrain 2008 23037.64
## 77          Bahrain 2010 20545.75
## 88  Brunei Darussalam 2008 37094.86
## 89  Brunei Darussalam 2010 30882.40
## 99          Bahamas, The 2008 23674.14
## 113          Canada 2008 46400.44
## 114          Canada 2010 47463.63
## 124          Switzerland 2010 74277.12
## 154          Cyprus 2010 30438.90
## 155          Cyprus 2008 34950.35
## 157  Czech Republic 2008 22649.38
## 158  Czech Republic 2010 19763.96
## 160          Germany 2008 45632.84
## 162          Germany 2010 41725.85
## 167          Denmark 2009 57895.50
## 168          Denmark 2010 57647.67
## 182          Estonia 2008 18087.68
## 190          Spain 2010 30737.83
## 202          Finland 2009 47107.16
## 203          Finland 2010 46205.17
## 204          Finland 2008 53401.31
## 214          France 2008 45415.81
## 215          France 2010 40708.50
## 222  United Kingdom 2010 38362.22
## 245          Greece 2010 26863.01
## 246          Greece 2008 31700.49
## 267          Croatia 2008 15887.42
## 273          Hungary 2008 15598.32
## 278          Ireland 2008 60968.84
## 279          Ireland 2010 47903.68
## 280          Israel 2010 30551.12
## 295          Iceland 2008 55446.76
## 297          Iceland 2010 41695.89
## 298          Italy 2009 36995.11
```

```

## 299          Italy 2010 35877.87
## 300          Italy 2008 40659.67
## 310          Japan 2010 42909.23
## 312          Japan 2008 37865.62
## 334      Korea, Rep. 2008 20474.89
## 336      Korea, Rep. 2010 22151.21
## 337          Kuwait 2010 38580.41
## 338          Kuwait 2008 54540.42
## 339          Kuwait 2009 37158.42
## 367      Lithuania 2008 14961.72
## 371      Luxembourg 2010 102863.10
## 423          Malta 2010 19694.08
## 458      Netherlands 2010 50341.25
## 459      Netherlands 2008 56628.75
## 460          Norway 2009 80017.78
## 461          Norway 2010 87646.27
## 462          Norway 2008 96880.51
## 467      New Zealand 2010 33394.07
## 472          Oman 2010 20922.66
## 474          Oman 2008 23483.63
## 504      Portugal 2010 22539.99
## 512          Qatar 2010 71510.19
## 538      Saudi Arabia 2008 19714.40
## 539      Saudi Arabia 2010 19326.58
## 540      Saudi Arabia 2009 16013.28
## 550          Sweden 2008 55746.84
## 551          Sweden 2010 52076.43
## 552          Sweden 2009 46207.06
## 553      Singapore 2010 46569.69
## 556      Slovenia 2008 27501.82
## 558      Slovenia 2010 23417.64
## 560      Slovak Republic 2010 16509.90
## 626      Trinidad and Tobago 2010 15494.70
## 643      United States 2010 48374.06
## 644      United States 2009 47001.56

# 4. Show the country-years where GDP per capita is above average,
# but number of physician is below average
d_wdi[d_wdi$gdppc > mean(d_wdi$gdppc) &
      d_wdi$number_of_physician < mean(d_wdi$number_of_physician),
      c("country", "year", "gdppc")]

##          country year    gdppc
## 76          Bahrain 2008 23037.64
## 77          Bahrain 2010 20545.75
## 88      Brunei Darussalam 2008 37094.86

```

```
## 89      Brunei Darussalam 2010 30882.40
## 538      Saudi Arabia 2008 19714.40
## 539      Saudi Arabia 2010 19326.58
## 540      Saudi Arabia 2009 16013.28
## 626 Trinidad and Tobago 2010 15494.70
```

3. Build linear model

a) 2 points

Download variables of interest (may be maximum of 4 variables) and build a linear model using `lm()`. Show the `summary()` of results

b) 2 points

Show the result with `stargazer`, customizing:

- The labels of the independent variables (i.e. the covariate)
- The label of the dependent variable
- Make the model name (i.e. OLS) show up

Hint: The options to do those things are in `help(stargazer)`. I have worded the task in a way that should help you find the relevant options.

Solution

```
m1 <- lm(infant_mortality ~ gdppc + number_of_physician, data = d_wdi)
summary(m1)

##
## Call:
## lm(formula = infant_mortality ~ gdppc + number_of_physician,
##     data = d_wdi)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -30.985 -12.604  -2.951   7.699  68.971
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  4.751e+01  1.748e+00  27.185  < 2e-16 ***
## gdppc        -2.822e-04  7.432e-05  -3.797  0.000184 ***
## number_of_physician -9.480e+00  9.492e-01  -9.988  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
##
## Residual standard error: 18.34 on 248 degrees of freedom
## Multiple R-squared: 0.5121, Adjusted R-squared: 0.5082
## F-statistic: 130.1 on 2 and 248 DF, p-value: < 2.2e-16

library(stargazer)

##
## Please cite as:
##
## Hlavac, Marek (2014). stargazer: LaTeX code and ASCII text for
## well-formatted regression and summary statistics tables.
## R package version 5.1. http://CRAN.R-project.org/package=stargazer

stargazer(m1,
  covariate.labels = c("GDP per capita", "Number of Physician"),
  dep.var.labels = c("Infant Mortality (per 1000 births)"),
  model.names = TRUE)
```

Table 1:

	<i>Dependent variable:</i>
	Infant Mortality (per 1000 births)
	<i>OLS</i>
GDP per capita	−0.0003*** (0.0001)
Number of Physician	−9.480*** (0.949)
Constant	47.508*** (1.748)
Observations	251
R ²	0.512
Adjusted R ²	0.508
Residual Std. Error	18.344 (df = 248)
F Statistic	130.149*** (df = 2; 248)
<i>Note:</i>	*p<0.1; **p<0.05; ***p<0.01

4. Verify the formula for R square

1. Extract the residuals and predicted values (fitted values) from the model object
2. Calculate three “sum of squares” (TSS, RegSS, RSS)
3. Calculate the R square and compare it with the R square reported by R

Note: the data you feed to `lm()` may have missing data, so R has to modify the data a little before using it. To extract the data that are actually used by `lm()`, use `my_model$model`. Use this data to calculate \bar{y} in the sum of squares.

Solution

```
res <- m1$residuals # Residuals
pred <- m1$fitted.values # Predicted values
y <- m1$model$infant_mortality # Data of Y that is used by lm()

# Calculate 3 sum of squares
TSS <- sum( (y - mean(y)) ** 2 )
RegSS <- sum( (pred - mean(y)) ** 2 )
RSS <- sum( res ** 2 )

Rsquare <- 1 - RSS / TSS
Rsquare

## [1] 0.5120972
```

The calculated R square is 0.5120972, the same as reported by R.