Lab06

2023-03-06

In this lab we will go over

1. Linear Regression
2. Review Parametric Models
3. Confidence Intervals

Linear regression

Let us first look at a simulated example,

Exercise 1 What are the values of different parameters β_0, β_1, e?

```
# Your answer here ...
```

To do regression of y on x as a predictor, we can call the `lm` function:

```
# These two commands are equivalent
fit <- lm(y ~ x, data=dat)
fit <- lm(dat$y ~ dat$x)
```

To get detailed information about the fit, such as coefficient estimates, t-statistics and p-values:

```
summary(fit)
```

```
##
## Call:
## lm(formula = dat$y ~ dat$x)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -8.2041 -2.1343  0.6057  2.1197  9.7063
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   2.3445     2.0164   1.163     0.26
## dat$x        -2.0669     0.1683 -12.279 3.48e-10 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.341 on 18 degrees of freedom
## Multiple R-squared:  0.8933, Adjusted R-squared:  0.8874
## F-statistic: 150.8 on 1 and 18 DF,  p-value: 3.483e-10
```

The coefficients estimation can be accessed:

```
fit$coefficients
```

```
## (Intercept)       dat$x
##    2.344487   -2.066860
```

To get the estimated intercept term:

```
fit$coefficients[1]
```

```
## (Intercept)
##    2.344487
```

To get the estimated slope term:

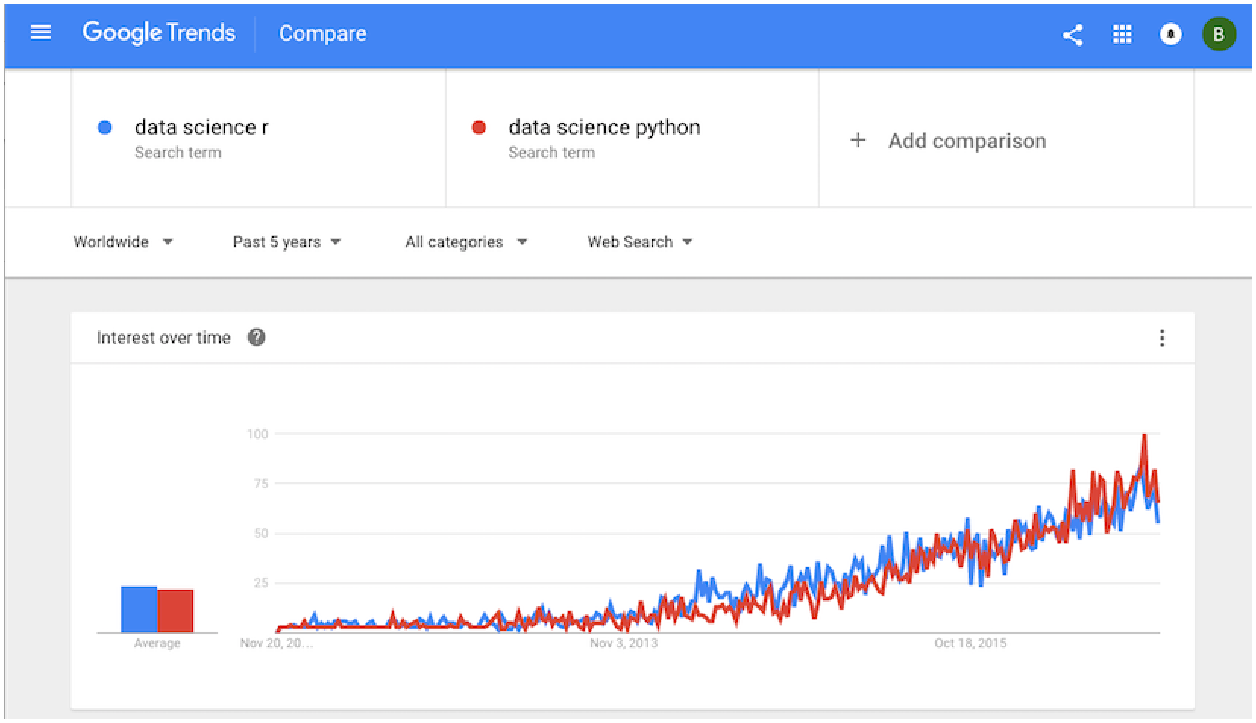
```
fit$coefficients[2]
```

```
##    dat$x
## -2.06686
```

Google Trends is a Google web tool providing equally-spaced time series data on the search volume. You may compare different topics to discover how peoples' interests change over time. You already code in at least one of the two most widely-used programming languages for data science (R and Python). Which language is more popular over time?

The following picture shows how to retrieve the Google Trends data. You may download and analyze your topics of interests as well. The dataset we obtained contains the following variables:

- **week**: beginning date of the week (recent 5 years)
- **python**: trend of the search term **Data science Python**
- **r**: trend of the search term **Data science r**



Google Trends | Compare

● data science r
Search term

● data science python
Search term

+ Add comparison

Worldwide ▾ Past 5 years ▾ All categories ▾ Web Search ▾

Interest over time

Read the data.

```
data_science <- read.csv("data_science.csv")
# convert string to date object
data_science$week <- as.Date(data_science$week, "%Y-%m-%d")
# create a numeric column representing the time
data_science$time <- as.numeric(data_science$week)
data_science$time <- data_science$time - data_science$time[1] + 1
```

Exercise 2 The plot in the Google Trend page looks somewhat linear. So we will try linear model first. Note that in the `lm` function for the model $y = \beta_0 + \beta_1 x + e$, you don't need to add the intercept term explicitly. Fitting the model with an intercept term is the default when you pass the formula as `y ~ x`. If you would like to fit a model without an intercept($y = \beta_1 x + e,$), you need the formula `y ~ x - 1`.

Run a regression of the `r` index on the predictor `time`. Get the estimate of the slope.

```
# Insert your code here and save the estimated slope as
# `r.slope`
# r.slope <-
# r.slope
```

Exercise 3. Let's expand the analysis to include R and python

```
python.lm <- lm(python ~ time, data = data_science)
r.lm <- lm(r ~ time, data = data_science)
```

```
summary(r.lm)
```

```
##
## Call:
## lm(formula = r ~ time, data = data_science)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -19.348  -7.076  -0.648   5.873  40.142
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -10.034179   1.116414  -8.988   <2e-16 ***
## time          0.038423   0.001065  36.089   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 9.019 on 258 degrees of freedom
## Multiple R-squared:  0.8347, Adjusted R-squared:  0.834
## F-statistic:  1302 on 1 and 258 DF,  p-value: < 2.2e-16
```

Read the output of `summary(r.lm)`.

(a) Which of the following formula will you use to predict the search index of `data science r` at time `t`.

A. -10.034179 + 0.038423 t

B. 1.116414 + 0.001065 t

C. -10.034179 + 1.116414 t

D. 0.038423 + 0.001065 t

(b) Calculate the t-statistics for the intercept and the slope using the coefficient estimates and standard deviations. (Please copy and paste the numbers you need from the output of `summary` function.) Are your results consistent with the ones given in the summary?

```
# Insert your code here and save the t-statistic for the intercept


# Insert your code here and save the t-statistic for the slope
```

(c) Calculate the p-value for the intercept and the slope using the test statistics you get in (b). What are the null hypothesis and your conclusion? Please use the normal distribution to approximate the distribution of test statistics under the null. Are your results consistent with the ones given in the summary?

```
# Insert your code here and save the p-value for the intercept


# Insert your code here and save the t-statistic for the slope
```

(d) Construct the confidence interval for the intercept and the slope using the coefficient estimates and standard deviations. (Please copy and paste the numbers you need from the output of `summary` function.) Will you accept or reject the null given the confidence interval you calculated?

```
# Insert your code here and save the confidence interval for the intercept

# Insert your code here and save the confidence interval for the slope
```

Confidence Intervals

Exercise 4.

```
summary(r.lm)
```

```
##
## Call:
## lm(formula = r ~ time, data = data_science)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -19.348  -7.076  -0.648   5.873  40.142
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -10.034179   1.116414  -8.988   <2e-16 ***
## time          0.038423   0.001065  36.089   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 9.019 on 258 degrees of freedom
## Multiple R-squared:  0.8347, Adjusted R-squared:  0.834
## F-statistic:  1302 on 1 and 258 DF,  p-value: < 2.2e-16
```

(a) Get the coefficients of the linear models fit on `data science r` search index using function `coef`. (This is equivalent to the dollar sign plus "coefficients")

```
# Insert your code here and save the coefficients vector
```

(b) Get the confidence intervals of the linear models fit on `data science r` search index using function `confint`.

```
# Insert your code here and save the confidence intervals
```

(c) Get bootstrap confidence intervals of β_0 and β_1. Complete the following codes and print out your results. Compare the bootstrap CI with the parametric CI. What do you observe? How to make a conclusion on the null hypothesis based on your confidence interval?

```
# We already got the estimate in r.lm
# Insert your code here and save the confidence intervals
bootOnce <- function() {
  # boot.index <- sample()

  # Use the index to get Y and X
  # y.sample <-
  # x.sample <-

  # Fit the linear regression with the new data
  # fit.model <-

  # Get the estimate and return the results
  # coef <-
  # return()
}
```

Replicate the function `bootOnce` 1000 times to get 1000 bootstrapped statistics.

```
# Complete the codes
# boot.stats <-
```

```
# Subtract the original estimates from boot.stats
# quantile.seq <- boot.stats - r.lm$coefficients

# Get the quantiles


# Confidence intervals
```

Compare the bootstrap CI with the parametric CI. What do you observe? How to make a conclusion on the null hypothesis based on your confidence interval?