Lab 6

Stat C131A

Feb 28, 2022

Welcome to Lab 6! In this lab, we will:

- Learn how to calculate Boferroni correction.
- Learn about how to calculate confidence intervals (CIs) for t-tests.
- Learn how to create confidence interval using bootstrap methods.
- Learn basic syntax of linear regression.

Bonferroni correction

Exercise 1.

Read in data.

```
craigslist <- read.csv("craigslist.csv",
                       header = TRUE, stringsAsFactors = FALSE)
one.bedrooms <- craigslist[craigslist$brs == 1, ]
```

Subset the dataset to only consider one-bedroom apartments. In lab 4, we performed t-test and permutation test on the mean price of Berkeley and Palo alto. In fact, there are 11 cities in the data set. We could conduct test on all the pairs of them.

(a) Perform t-test on the mean price difference for each pairwise city combinations. Save the p-values in vector `p.values`. For which city pairs we can not reject the null hypothesis that mean rent price is equal? (above level 0.05) Again, you can use `apply` or for loops, but `apply` is recommended.

```
cities <- unique(craigslist$location) # get the vector of unique city names
cities.combn <- combn(cities, 2) # this will give you all the possible combinations of two cities

# 'p.values'
p.values <- apply(cities.combn, 2, function(x){
  # The price vector for city 1 (x[1])
  # city1data <-
  # The price vector for city 2 (x[2])
  # city2data <-
  # return the p value of t test
  # p <-
  # return(p)
})
```

```r
# You can not reject the null hypothesis that mean rent price for the following city pairs.
# (Subset the `cities.combn`.)
# save the your answer as
# 'not.reject'
not.reject = c()
```

(b) Perform Bonferroni corrections on the p-values. For which city pairs we can not reject the null hypothesis that mean rent price is equal? (under level 0.05)

```
# insert code here save you answer as
# 'p.values.adj'
p.values.adj = c()
```

```
# You can not reject the null hypothesis that mean rent price for the following city pairs.
# (Subset the `cities.combn`.)
# (After Bonferroni corrections)
# insert code here save you answer as
# 'not.reject.adj'
not.reject.adj = c()
```

```
p.values
```

```
## NULL
```

```
not.reject
```

```
## NULL
```

```
p.values.adj
```

```
## NULL
```

```
not.reject.adj
```

```
## NULL
```

In the following three exercises, we will look at a comic book data created by FiveThirtyEight for their story *Comic Books Are Still Made By Men, For Men And About Men*, where they claimed that Comic books vastly under-represent women. Specifically, they said that characters who are women have lower appearance counts after analyzing the Marvel and DC dataset. Well, is that true? Let's apply our testing methods to their analysis!

```
# Read in data.
marvel <- read.csv("marvel-wikia-data.csv")
```

The data `marvel-wikia-data.csv` comes from Marvel Wikia. It has the following variables:

- `page_id`: The unique identifier for that characters page within the wikia
- `name`: The name of the character
- `urlslug`: The unique url within the wikia that takes you to the character
- `ID`: The identity status of the character (Secret Identity, Public identity, [on marvel only: No Dual Identity])
- `ALIGN`: If the character is Good, Bad or Neutral
- `EYE`: Eye color of the character
- `HAIR`: Hair color of the character
- `SEX`: Sex of the character (e.g. Male, Female, etc.)
- `GSM`: If the character is a gender or sexual minority (e.g. Homosexual characters, bisexual characters)
- `ALIVE`: If the character is alive or deceased
- `APPEARANCES`: The number of appearances of the character in comic books (as of Sep. 2, 2014. Number will become increasingly out of date as time goes on.)
- `FIRST APPEARANCE`: The month and year of the character's first appearance in a comic book, if available
- `YEAR`: The year of the character's first appearance in a comic book, if available

We will focus on two columns `SEX` and `APPEARANCES`. To make thing simpler, we created two vectors `female.logappearances` and `male.logappearances` for you, which contains the log appearances counts for female and male characters separately. You will use these two vectors to do hypothesis testing in the rest of the lab.

```
female.logappearances <- log(marvel$APPEARANCES[marvel$SEX == "Female Characters"])
male.logappearances <- log(marvel$APPEARANCES[marvel$SEX == "Male Characters"])
```

T-test confidence intervals

Exercise 2.

(a) Get the confidence interval for the means of the female and male log appearance using the `t.test()` function.

```
# Insert your code here for calculating the CI, and save the CIs as
# `log.female.ci` and `log.male.ci`
# log.female.ci <-
# log.male.ci <-
```

You may noticed that when you print the confidence interval, there is an additional line named `attr(,"conf.level")` with value 0.95. In `R`, all objects can have arbitrary additional attributes used to store metadata about the object. Attributes can be accessed using `attributes()` or `attr()`. For example, to get all the attributes of `log.male.ci`, run `attributes(log.male.ci)`. And to get the `conf.level` attribute of `log.male.ci`, run `attr(log.male.ci, "conf.level")`. Attributes do not influence the fact that `log.male.ci` is a two-dimensional vector.

(b) Get the confidence interval for the difference in the log appearance count for female and male using the `t.test()` function.

```
# Insert your code here for calculating the CI, and save the CI as
# `log.diff.ci`
# log.diff.ci <-
```

(c) Based on your calculation, which of the following statements do you support?

i. There is no significant difference in appearance counts for male and female in Marvel comics.
ii. The female character appearances count is significantly larger than the male in Marvel comics.
iii. The male character appearances count is significantly larger than the female in Marvel comics.

```
# Uncomment the line of your answer for this question:
# Ex1c.answer <- "i."
# Ex1c.answer <- "ii."
# Ex1c.answer <- "iii."
```

Bootstrap confidence intervals

Exercise 3.

In this exercise, you will use the bootstrap to get the confidence interval for the difference in means between female and male log appearance count.

(a) Calculate the observed difference.

```
# Insert your code here for calculating the observed difference
# `obs.d`
# obs.d <-
```

(b) Complete the following function to calculate the bootstrapped statistic. The function only needs to calculate one statistic; you will replicate it later.

```
# Complete the function for calculating the bootstrapped difference in means
bootOnce <- function() {
  # boot.male <-
  # boot.female <-
  # diff <-
  # return(diff)
}
```

You can test whether your code works or not by running the following chunk:

```
test <- bootOnce()
test  # This should give you a numeric value, which is a bootstrapped statistic
```

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

```
# Insert your code here and save the bootstrapped differences as
# `boot`
# boot <-
```

(d) Plot the histogram of your bootstrapped statistics.

```
# Insert your code for histogram here
```

(e) Calculate the bootstrapped CI at the 0.95 confidence level.___Hint:___ use the `quantile()` function to find the bounds; use lower bound = $(1 - 0.95)/2$th quantile, and upper bound = $1 - (1 - 0.95)/2$th quantile.

```
# Insert your code and save your bootstrapped CI as
# `boot.ci`
# boot.ci <-
```

(f) Based on your calculation, which of the following statements is supported?

i. I will fail to reject the hypothesis that there is no significant difference in appearance counts for male and female.
ii. I will reject the hypothesis that there is no significant difference in appearance counts for male and female.

```
# Uncomment the line of your answer for this question:
# Ex2f.answer <- "i."
# Ex2f.answer <- "ii."
```

Linear regression

Let us first look at a simulated example. You can ignore the particularities of the data generating process.

```
# Make some data
# X increases (noisily)
# Y is constructed so it is inversely related to xvar
set.seed(955)
xvar <- 1:20 + rnorm(20,sd=3)
yvar <- -2 * xvar + 3 + rnorm(20,sd=4)

# Make a data frame with the variables
dat <- data.frame(x=xvar, y=yvar)
# Show first few rows
head(dat)
```

```
##           x          y
## 1 -4.252354  14.802251
## 2  1.702318  -3.063383
## 3  4.323054  -9.771777
## 4  1.780628  -3.130846
## 5 11.537348 -25.120776
## 6  6.672130 -13.533446
```

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

```
# These two commands is 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
## -5.0503 -2.6031 -0.2142  2.6217  7.6811
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   2.7791     1.4737   1.886   0.0756 .
## dat$x        -1.9889     0.1295 -15.355 8.71e-12 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.642 on 18 degrees of freedom
## Multiple R-squared:  0.9291, Adjusted R-squared:  0.9251
## F-statistic: 235.8 on 1 and 18 DF,  p-value: 8.709e-12
```

The coefficients estimation can be accessed:

```
fit$coefficients
```

```
## (Intercept)       dat$x
##    2.779061   -1.988899
```

To get the estimated intercept term:

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

```
## (Intercept)
##    2.779061
```

To get the estimated slope term:

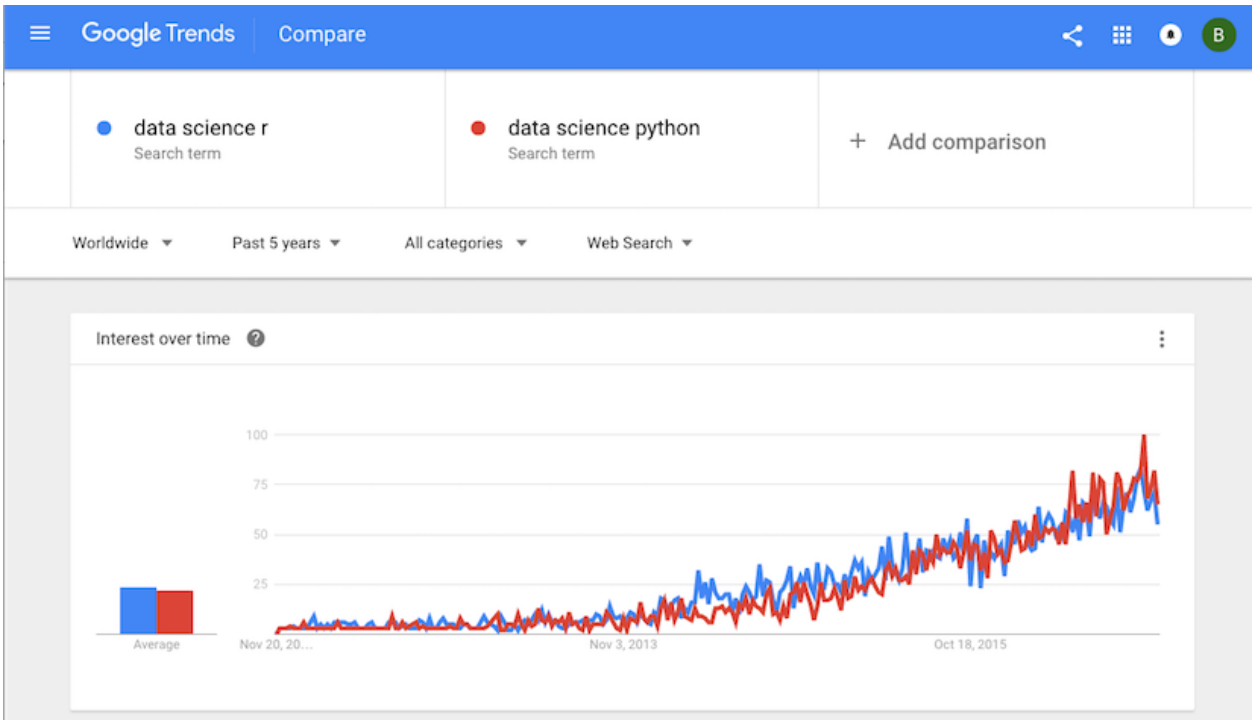
```
fit$coefficients[2]
```

```
##     dat$x
## -1.988899
```

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**



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 4 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 5.

```
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 `r.lm`.

(a) Which of the folowing 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
```