INFO370 Problem Set 4: Inference, Monte Carlo Simulations

Your name:

Deadline: Wed, Feb 5th 10:30am

Instructions

The aim of this problem set is to learn about statistical hypotheses, hypothesis testing, and t-test. It follows broadly the approach of the lab 5 in the sense that you are first asked to generate random data under H_0, and later to use the corresponding formula. Hower, here we look at a slightly different task where the question is to compare two continuous outcomes, not a single proportion.

1 Are sons taller than fathers? Monte-Carlo approach (60pt)

In this dataset you use fathers-sons data. This is one of the original datasets used by Pearson for his work with linear regression. The data is copied from *UsingR* R package.

You will proceed as follows: first, you compute the difference between fathers' and sons' average height. Thereafter you create two samples of random normal numbers as in the data above, using the mean and standard deviation over combined fathers and sons. Call one of these samples "fake fathers" and the other "fake sons". What is the difference of their means? And now you repeat this exercise many-many times and see if you can get as big a difference between the fake fathers and fake sons as you got between real fathers and sons.

1. (3pt) load the *fatherson.csv* data. Perform basic description of it: what is the number of observations? Are there any missings or otherwise invalid entries?

Note: *fheight* and *sheight* are fathers' and sons' height, respectively (in cm).

2. (3pt) Describe fathers and sons: compute the mean, median, standard deviation, and range of their heights. According to these figures, who are taller: fathers or sons? Compute the mean difference between fathers and sons. (The answer is 2.53cm.)

3. (6pt) Lets add a graphical comparison. Plot histograms of both fathers' and sons' heights. Comment the histograms/density plots. Which distribution do they resemble?

This was the basic description of the data. Now onward to the comparison. We proceed as follows: Imagine that there is no real difference between fathers and sons. We call this null-hypothesis H_0. Hence whatever difference we see in the actual data is just random sampling noise. We would like to have a huge number of fathers and sons to test it, but unfortunately we do not have that. So we do this instead: we create a large number of fake fathers and fake sons, both drawn from the same distribution. Thereafter we compare the mean heights: how much taller are the fake sons compared to the fake fathers? We repeat this process many times and at the end we report how often did we find a difference that is similar to what we observe in the real data.

4. (6pt) Let's state our H_0: fathers and sons are of similar height (in average). Hence we have to create fake fathers and fake sons from a similar distribution. An obvious choice for this is the distribution of combined fathers' and sons' height.

   Compute the overall mean μ and standard deviation σ of combined fathers' and sons' heights.

   Hint: you can use `pd.concat` to combine two series together into a longer series.

5. (6pt) Now create two sets of random normals, "fake fathers" and "fake sons", both with the same overall mean and overall standard deviation that you just computed above.

   What is the average fathers' and sons' difference? Compare the result with that you found in the previous problem.

   Hint: say, the average is 170 and standard deviation is 10. You can create the corresponding normals like:

```
fakefathers = np.random.normal(170, 10, size=10)  # 10 fathers
fakesons = np.random.normal(170, 10, size=10)  # 10 sons
fakefathers

## array([178.52262296, 163.85141232, 184.1126095 , 167.58137249,
##        184.60252523, 158.5284813 , 161.65481958, 167.66126827,
##        169.31483097, 150.17138357])

fakesons

## array([185.47541168, 163.99801508, 166.51167163, 172.1163825 ,
##        169.13870591, 164.29609591, 159.33902028, 164.28541772,
##        153.01651141, 164.84325331])
```

compute the mean difference:

```
np.mean(fakefathers) - np.mean(fakesons)

## 2.298084075221823
```

Now compare this number with what you see in data.

Instead of just 10 fathers and sons as in this example, use the actual number, 1000 or so, you find in data.

Comment: assume the data is *not paired* (see OIS section 7.3). This may not be quite correct here, but the data description does not mention how the fathers and sons are selected. Non-paired data also tends to have more practical applications.

6. (6pt) Now repeat the previous question a large number `R` (1000 or more) times. Each time store the mean difference for fake fathers and fake sons, so you end up with `R` different values for the mean difference.

7. (6pt) What is the mean of the mean differences? If you did your simulations correctly, it should be close to 0. Explain why do you get this result.

8. (9pt) What is the largest mean difference (in absolute value) in your sample?

   Hint: `np.abs` computes absolute value.

9. (9pt) find 95% confidence interval (CI) of your sample of mean differences. Does the difference in actual data, 2.53cm in favor of sons, fall into the CI?

   Hint: use `np.percentile(2.5)` and a similar expression for the 97.5th percentile.

10. (6pt) Finally, based on the simulations, what is your conclusion: is the observed difference 2.53 just a random fluke, or are sons really taller than fathers?

2 Now repeat the above with t-test (40pt)

Above we spent a lot of effort with sampling, random numbers and such. In practice, it is usually not possible to sample millions of fathers and sons. And even more, even if feasible, it is much easier just to do a t-test. Below we ask you to *compute the* t*-value yourself*, do not use any pre-existing functions!

1. (8pt) Compute standard error SE of the difference.

   Hint: read OIS 7.3, p 267. You probably have to walk back and read about various other concepts the book is using in 7.3.

2. (8pt) Compute 95% CI.

   Use the 5% two-tail confidence level to look up t_{cr} values in t-distribution table. OIS has such a table in Appendix C.2, and google can find a million of those.

   95% CI is given by $\mu \pm t_{cr} \cdot SE$ where μ is the mean, SE is its standard error, and t_{cr} is the critical value from the table.

   Hint 1: what is the *degrees of freedom* in current case? Consult OIS 7.3.

   Hint 2: we need 2-tailed test as sons can be both taller and shorter than fathers.

3. (6pt) What will you conclude based on CI: can you reject H_0, fathers and sons are of equal height, at 5% level?

4. (6pt) Now perform the opposite operation: compute the t-value. When the you have mean μ and standard error SE, you can compute the t-value by

$$t = \frac{\mu}{SE}$$

Assume we do not have paired data here.

Hint: the answer should by quite large, > 8.

5. (6pt) What is the likelihood that such a `t` value happens just by random chance? Consult the `t`-table.

6. (6pt) Finally, state your conclusion: are fathers taller than sons? Do all of your three methods: simulations, 95% CI, and `t`-value agree?

3 Extra credit challenge (10pt = 1 EC)

How long time do you need to simulate to get the mean difference between fake fathers and fake sons to equal 2.53?

If you did the previous tasks well, you noticed that simulated differences are way smaller than the actual differences, and even millions of experiments do not bring you close. But how long time do you have to run the simulations to actually get close?

1. (3pt) First, time your simulations. Run a large number of simulations, say 1M, and measure how long it takes on your computer. It should take at least 5 seconds for your measurements to be precise enough. Now you can easily calculate how long it would take to run 10^{12} or so experiments.

   Hint: check out `%timeit` magic macro.

2. (3pt) Second, what is the probability to receive such enormous `t`-values? As these are off the `t` tables, you have to compute the corresponding probability yourself.

   Assume we are dealing with normal distribution. (Not quite but we are close.) You have to compute the probability you get a value larger than the `t` value you computed. This can be done along the lines:

```
from scipy import stats
norm = stats.norm()
norm.cdf(-1.96)  # close to 0.025

## 0.024997895148220435
```

where you replace 1.96 with your actual `t`-value.

Explain: why does the example use `norm.cdf(-1.96)` instead of `norm.cdf(1.96)`?

3. (2pt) How many iterations you need? Let's do a shortcut–if probability `p` is small, you need roughly 3/`p` iterations. So if `p` = 0.001, you need 3000 iterations.

4. (2pt) Based on the timings you did above, how many years do you have to run the simulations?

   If one had started the computer the year your grandfather was born, would it be there now?

   If the first Seattle inhabitants had started it when they moved here following the melting ice, 10,000 or so years ago?

   If the last dinosaurs had started it 66,000,000 years ago? (But it must have been in Idaho or somewhere else, the land where Seattle is now did not exist back then.)