KEY_Practice16_Intro_Stats_III

August 28, 2019

1 Practice with Statistics (Part 3)!

Remember: * Significance tests tell you the probability that a change happens purely by chance * A **t-test** is a significance test which compares *two groups* * A **correlation** is a measure of the statistical relationship between two variables

First, import numpy and pandas and the scipy statistical module:

[0]:
```
# load numpy and pandas and scipy.stats

import numpy as np
import pandas as pd
import scipy.stats as stats
```

[10]:
```
# mount Google Drive
from google.colab import drive
drive.mount('/content/gdrive')
path = '/content/gdrive/My Drive/SummerExperience-master/'
```

```
Drive already mounted at /content/gdrive; to attempt to forcibly remount, call
drive.mount("/content/gdrive", force_remount=True).
```

Load in the sample data from the Lesson:

[0]:
```
# read the csv file: '../SampleData/detroit_weather_2.csv'
data_table = pd.read_csv(path + 'SampleData/detroit_weather_2.csv')
```

[12]:
```
# Print the head of the table to remind you of the format:

data_table.head()
```

[12]:

	YEAR	MONTH	DAY	Temperature
0	1937	1	1	0.50
1	1937	1	2	0.17
2	1937	1	3	-1.06
3	1937	1	4	-3.89
4	1937	1	5	-0.17

Now, we will repeat the analysis we did in the lesson but comparing two different decades. Which decades would you like to test?

[13]:
```
# Pick two decades that we didn't look at during the lesson
# extract the temperatures into numpy arrays

temps_1960 = np.array(data_table.query('YEAR >= 1960 and YEAR <␣
↪1970')["Temperature"] )
temps_today = np.array(data_table.query('YEAR >= 2010 and YEAR <␣
↪2020')["Temperature"] )

# note that the current decade isn't over...
# even though there isn't yet data for 2020, we include it in the query so our␣
↪code will still work in the year 2021!

print(temps_1960, temps_today)
```

```
[-2.33  0.11  1.28 … -2.11 -0.67 -2.06] [ -3.78 -10.17 -12.11 …   5.67
6.44  11.22]
```

[14]:
```
# Calculate the means of your data to see if they differ

print(np.mean(temps_1960))
print(np.mean(temps_today))
```

```
8.915261428962499
9.666400235086689
```

[15]:
```
# Calculate the difference between the two means

print(np.mean(temps_today) - np.mean(temps_1960)) # .75 degree increase
```

```
0.7511388061241906
```

[16]:
```
# Perform a t-test of these data to calculate the p value

stats.ttest_ind(temps_1960, temps_today).pvalue
```

[16]:
```
0.002958034438690926
```

Now let's practice calculating some correlations. For this we will read in the `tips` dataset.

[17]:
```
#read in tips data
tips = pd.read_csv(path + 'SampleData/tips.csv')

tips.head()
```

[17]:

	total_bill	tip	sex	smoker	day	time	size
0	16.99	1.01	Female	No	Sun	Dinner	2
1	10.34	1.66	Male	No	Sun	Dinner	3
2	21.01	3.50	Male	No	Sun	Dinner	3
3	23.68	3.31	Male	No	Sun	Dinner	2
4	24.59	3.61	Female	No	Sun	Dinner	4

We want to calculate the correlations of `total_bill`, `tip` and `size`. Since we are getting the correlations for three variables, what size do we expect the resulting **correlation matrix** to be?

ANSWER: 3 x 3 matrix

[18]:
```
# create correlation matrix for total_bill, tip and size
# HINT: what parameter do we need to use when our observations are along the␣
 ↪rows?
corrs = np.corrcoef(tips[['total_bill', 'tip','size']], rowvar=False)
print(corrs)
```

```
[[1.         0.67573411 0.59831513]
 [0.67573411 1.         0.48929878]
 [0.59831513 0.48929878 1.        ]]
```

Which two variables have the strongest correlation?

ANSWER: `total_bill` and `tip`

Which two variables have the weakest correlation?

ANSWER: `tip` and `size`

CHALLENGE: What if these relationships are different between lunch and dinner? Create two subsets of tips for the lunch and dinner times and repeat the correlation analysis.

[19]:
```
# create two subsets of tips, one for lunch and one for dinner
lunch = tips.query('time == "Lunch"')
dinner = tips.query('time == "Dinner"')

# compute the correlation matrix for each of the data subsets you created
lunch_corrs = np.corrcoef(lunch[['total_bill', 'tip','size']], rowvar=False)
din_corrs = np.corrcoef(dinner[['total_bill', 'tip','size']], rowvar=False)

#print the correlations
print(lunch_corrs)
print(din_corrs)
```

```
[[1.         0.80542384 0.708662  ]
 [0.80542384 1.         0.64785392]
 [0.708662   0.64785392 1.        ]]
[[1.         0.63287125 0.55701503]
 [0.63287125 1.         0.42850163]
 [0.55701503 0.42850163 1.        ]]
```

What do you notice when you compare the results between the different times of day?

Answer: correlations are much stronger during the lunch shift than the dinner shift

Do you have a *hypothesis* for why this might be? *HINT*: Does it have to do with a difference in the amount people generally spend for those meals? Or maybe the number of *samples* we have for each condition (i.e. meal time)?

How could you test those hypotheses?

[20]:
```
# find the average meal price for each meal time
print(np.mean(lunch['total_bill']))
print(np.mean(dinner['total_bill']))

# find the number of samples we have for each meal time
# HINT: use len()
print(len(lunch))
print(len(dinner))
```

```
17.16867647058823
20.7971590909091
68
176
```

What do you notice here? When we look at the *sample size*, we find that we have a lot fewer *samples* from the lunch shift compared to the dinner shift. When our sample size is smaller, this means that each individual sample contributes more to our final statistic, here correlation. Also, we see that the average meal price is lower for lunch as well. Both of these differences are things we must consider when comparing the correlations of these two sets.

Nice job! You just practiced:

- Using statistical tests to determine if two groups are significantly different
- Using correlations to determine the relationships between variables