04-01: Basic Statistic Testing

In this lecture we're going to review some of the basics of statistical testing in python. We're going to talk about hypothesis testing, statistical significance, and using scipy to run student's t-tests.

We use statistics in a lot of different ways in data science, and on this lecture, I want to refresh your knowledge of hypothesis testing, which is a core data analysis activity behind experimentation. The goal of hypothesis testing is to determine if, for instance, the two different conditions we have in an experiment have resulted in different impacts

Hypothesis Testing

When we do hypothesis testing, we actually have two statements of interest:

- the first is our actual explanation, which we call the alternative hypothesis (or H1)
- and the second is that the explanation we have is not sufficient, and we call this the null hypothesis. (or H0)

Our actual testing method is to determine whether the null hypothesis is true or not. If we find that there is a difference between groups, then we can reject the null hypothesis and we accept our alternative.

Now, scipy is an interesting collection of libraries for data science and you'll use most or perpahs all of these libraries. It includes numpy and pandas, but also plotting libraries such as matplotlib, and a number of scientific library functions as well

In [1]:

```
# Let's import our usual numpy and pandas libraries
import numpy as np
import pandas as pd

# Now let's bring in some new libraries from scipy
from scipy import stats
```

In [2]:

```
# Let's see an example of this; we're going to use some grade data
df=pd.read_csv ('datasets/grades.csv')
df.head()
```

Out[2]:

	student_id	assignment1_grade	assignment1_submission	assignment2_grade	assignme
0	B73F2C11-70F0-E37D-8B10-1D20AFED50B1	92.733946	2015-11-02 06:55:34.282000000	83.030552	02::
1	98A0FAE0-A19A-13D2-4BB5-CFBFD94031D1	86.790821	2015-11-29 14:57:44.429000000	86.290821	17:4
2	D0F62040-CEB0-904C-F563-2F8620916C4E	85.512541	2016-01-09 05:36:02.389000000	85.512541	06::
3	FFDF2B2C-F514-EF7F-6538-A6A53518E9DC	86.030665	2016-04-30 06:50:39.801000000	68.824532	17::
4	5ECBEEB6-F1CE-80AE-3164-E45E99473FB4	64.813800	2015-12-13 17:06:10.750000000	51.491040	12::

In [3]:

```
# If we take a look at the data frame inside, we see we have six different assig
nments. Lets look at some
# summary statistics for this DataFrame
print("There are {} rows and {} columns".format(df.shape[0], df.shape[1]))
```

```
There are 2315 rows and 13 columns
```

Classifying Learners into Categorical Data

In [4]:

```
# For the purpose of this lecture, let's segment this population into two piece
s. Let's say those who finish
# the first assignment by the end of December 2015, we'll call them early finish
ers, and those who finish it
# sometime after that, we'll call them late finishers.

early_finishers=df[pd.to_datetime(df['assignment1_submission']) < '2016']
early_finishers.head()
```

Out[4]:

	student_id	assignment1_grade	assignment1_submission	assignment2_grade	assignm
0	B73F2C11-70F0-E37D-8B10-1D20AFED50B1	92.733946	2015-11-02 06:55:34.282000000	83.030552	02:
1	98A0FAE0-A19A-13D2-4BB5-CFBFD94031D1	86.790821	2015-11-29 14:57:44.429000000	86.290821	17:
4	5ECBEEB6-F1CE-80AE-3164-E45E99473FB4	64.813800	2015-12-13 17:06:10.750000000	51.491040	12:
5	D09000A0-827B-C0FF-3433-BF8FF286E15B	71.647278	2015-12-28 04:35:32.836000000	64.052550	21:
8	C9D51293-BD58-F113-4167-A7C0BAFCB6E5	66.595568	2015-12-25 02:29:28.415000000	52.916454	01:

In [5]:

```
# So, you have lots of skills now with pandas, how would you go about getting th
e late_finishers dataframe?
# Why don't you pause the video and give it a try. Take the complement of the se
t above.
late_finishers=df[pd.to_datetime(df['assignment1_submission']) >= '2016']
late_finishers.head()
```

Out[5]:

	student_id	assignment1_grade	assignment1_submission	assignment2_grade	assignme
2	D0F62040-CEB0-904C-F563-2F8620916C4E	85.512541	2016-01-09 05:36:02.389000000	85.512541	06::
3	FFDF2B2C-F514-EF7F-6538-A6A53518E9DC	86.030665	2016-04-30 06:50:39.801000000	68.824532	17:2
6	3217BE3F-E4B0-C3B6-9F64-462456819CE4	87.498744	2016-03-05 11:05:25.408000000	69.998995	07:2
7	F1CB5AA1-B3DE-5460-FAFF-BE951FD38B5F	80.576090	2016-01-24 18:24:25.619000000	72.518481	13::
9	E2C617C2-4654-622C-AB50-1550C4BE42A0	59.270882	2016-03-06 12:06:26.185000000	59.270882	02:(

In [6]:

```
# Here's my solution. First, the dataframe df and the early_finishers share inde
x values, so I really just
# want everything in the df which is not in early_finishers
# early finishers is a subset of df.
boolean = df.index.isin(early_finishers.index)
late_finishers = df[~boolean]
late_finishers.head()
```

Out[6]:

	student_id	assignment1_grade	assignment1_submission	assignment2_grade	assignme
2	D0F62040-CEB0-904C-F563-2F8620916C4E	85.512541	2016-01-09 05:36:02.389000000	85.512541	06::
3	FFDF2B2C-F514-EF7F-6538-A6A53518E9DC	86.030665	2016-04-30 06:50:39.801000000	68.824532	17::
6	3217BE3F-E4B0-C3B6-9F64-462456819CE4	87.498744	2016-03-05 11:05:25.408000000	69.998995	07::
7	F1CB5AA1-B3DE-5460-FAFF-BE951FD38B5F	80.576090	2016-01-24 18:24:25.619000000	72.518481	13::
9	E2C617C2-4654-622C-AB50-1550C4BE42A0	59.270882	2016-03-06 12:06:26.185000000	59.270882	02:(

There are lots of other ways to do this.

1. For instance, you could just copy and paste the first projection and change the sign from less than to greater than or equal to.
    - This is ok, but if you decide you want to change the date down the road you have to remember to change it in two places.
2. You could also do a join of the dataframe df with early_finishers - if you do a left join you only keep the items in the left dataframe, so this would have been a good answer.
3. You also could have written a function that determines if someone is early or late, and then called .apply() on the dataframe and added a new column to the dataframe. This is a pretty reasonable answer as well.

In [7]:

```
# As you've seen, the pandas data frame object has a variety of statistical func
tions associated with it. If
# we call the mean function directly on the data frame, we see that each of the
 means for the assignments are
# calculated. Let's compare the means for our two populations

print(early_finishers['assignment1_grade'].mean())
print(late_finishers['assignment1_grade'].mean())
```

```
74.94728457024303
74.0450648477065
```

Using Hypothesis Testing (T-Tests) to evaluate Data

Ok, these look pretty similar. But, are they the same? What do we mean by similar? This is where the students' t-test comes in. It allows us to form the alternative hypothesis ("These are different") as well as the null hypothesis ("These are the same") and then test that null hypothesis. So mathematically,

$$H_0 : \mu = 75$$
$$H_1 : \mu \neq 75$$

When doing hypothesis testing, we have to choose a significance level as a threshold for how much of a chance we're willing to accept. This significance level is typically called alpha α. For this example, let's use a threshold of 0.05 for our alpha or 5%. **Now this is a commonly used number but it's really quite arbitrary.**

The SciPy library contains a number of different statistical tests and forms a basis for hypothesis testing in Python and we're going to use the `ttest_ind()` function which does an **independent t-test (meaning the populations are not related to one another).** The result of `ttest_ind()` are the `t-statistic` and a `p-value` .

It's this latter value, **the probability**, which is most important to us, as it **indicates the chance** (between 0 and 1) **of our null hypothesis being True**.

In [8]:

```
# Let's bring in our ttest_ind function
from scipy.stats import ttest_ind

# Let's run this function with our two populations, looking at the assignment 1
 grades
ttest_ind(early_finishers['assignment1_grade'], late_finishers['assignment1_grad
e'])
```

Out[8]:

```
Ttest_indResult(statistic=1.322354085372139, pvalue=0.18618101101714
55)
```

So here we see that the probability is 0.18, and this is above our alpha value of 0.05. **This means that we cannot reject the null hypothesis.** The null hypothesis (H_0) was that the two populations are the same, and we don't have enough certainty in our evidence (because it is greater than alpha) to come to a conclusion to the contrary. This doesn't mean that we have **proven** the populations are the same.

In [9]:

```
# Why don't we check the other assignment grades?
print(ttest_ind(early_finishers['assignment2_grade'], late_finishers['assignment
2_grade']))
print(ttest_ind(early_finishers['assignment3_grade'], late_finishers['assignment
3_grade']))
print(ttest_ind(early_finishers['assignment4_grade'], late_finishers['assignment
4_grade']))
print(ttest_ind(early_finishers['assignment5_grade'], late_finishers['assignment
5_grade']))
print(ttest_ind(early_finishers['assignment6_grade'], late_finishers['assignment
6_grade']))
```

```
Ttest_indResult(statistic=1.2514717608216366, pvalue=0.2108889627004
424)
Ttest_indResult(statistic=1.6133726558705392, pvalue=0.1067999810222
7865)
Ttest_indResult(statistic=0.049671157386456125, pvalue=0.96038872978
9337)
Ttest_indResult(statistic=-0.05279315545404755, pvalue=0.95790127397
46492)
Ttest_indResult(statistic=-0.11609743352612056, pvalue=0.90758540119
89656)
```

Further Experiments

Ok, so it looks like in this data we do not have enough evidence to suggest the populations differ with respect to grade. Let's take a look at those p-values for a moment though, because they are saying things that can inform experimental design down the road. For instance, one of the assignments, assignment 3, has a p-value around 0.1. This means that if we accepted a level of chance similarity of 11% this would have been considered statistically significant. As a research, this would suggest to me that there is something here worth considering following up on. For instance, if we had a small number of participants (we don't) or if there was something unique about this assignment as it relates to our experiment (whatever it was) then there may be followup experiments we could run.

Limitations of the P-Values

P-values have come under fire recently for being insuficient for telling us enough about the interactions which are happening, and two other techniques, confidence intervalues and bayesian analyses, are being used more regularly. One issue with p-values is that as you run more tests you are likely to get a value which is statistically significant just by chance.

In [10]:

```
# Lets see a simulation of this. First, lets create a data frame of 100 columns,
each with 100 numbers
df1=pd.DataFrame([np.random.random(100) for x in range(100)])
df1.shape
```

Out[10]:

```
(100, 100)
```

In [11]:

```
# Pause this and reflect -- do you understand the list comprehension and how I c
reated this DataFrame? You
# don't have to use a list comprehension to do this, but you should be able to r
ead this and figure out how it
# works as this is a commonly used approach on web forums.
```

In [12]:

```
# Ok, let's create a second dataframe
df2=pd.DataFrame([np.random.random(100) for x in range(100)])
df2.shape
```

Out[12]:

```
(100, 100)
```

In [13]:

```
# Are these two DataFrames the same? Maybe a better question is, for a given row
inside of df1, is it the same
# as the row inside df2?

# Let's take a look. Let's say our critical value is 0.1, or and alpha of 10%. A
nd we're going to compare each
# column in df1 to the same numbered column in df2. And we'll report when the p-
value isn't less than 10%,
# which means that we have sufficient evidence to say that the columns are diffe
rent.

# Let's write this in a function called test_columns
def test_columns(alpha=0.1):
    # I want to keep track of how many differ
    num_diff=0
    # And now we can just iterate over the columns
    for col in df1.columns:
        # we can run out ttest_ind between the two dataframes , also note: tuple
unpacking.
        teststat,pval=ttest_ind(df1[col],df2[col])
        # and we check the pvalue versus the alpha
        if pval<=alpha:
            # And now we'll just print out if they are different and increment t
he num_diff
            print("Col {} is statistically significantly different at alpha={},
 pval={}".format(col,alpha,pval))
            num_diff=num_diff+1
    # and let's print out some summary stats
    print("Total number different was {}, which is {}%".format(num_diff,float(nu
m_diff)/len(df1.columns)*100))

# And now lets actually run this
test_columns(0.05)
```

```
Col 8 is statistically significantly different at alpha=0.05, pval=
0.040394482824999266
Col 22 is statistically significantly different at alpha=0.05, pval=
0.028775596099077153
Col 36 is statistically significantly different at alpha=0.05, pval=
0.027595331942215046
Col 45 is statistically significantly different at alpha=0.05, pval=
0.018616520225533666
Col 75 is statistically significantly different at alpha=0.05, pval=
0.0054097376131833925
Col 80 is statistically significantly different at alpha=0.05, pval=
0.02457251202079053
Total number different was 6, which is 6.0%
```

In [14]:

```
# Interesting, so we see that there are a bunch of columns that are different! I
n fact, that number looks a
# lot like the alpha value we chose. So what's going on - shouldn't all of the c
olumns be the same? Remember
# that all the ttest does is check if two sets are similar given some level of c
onfidence, in our case, 10%.
# The more random comparisons you do, the more will just happen to be the same b
y chance. In this example, we
# checked 100 columns, so we would expect there to be roughly 10 of them if our
 alpha was 0.1.

# We can test some other alpha values as well
test_columns(0.05)
```

```
Col 8 is statistically significantly different at alpha=0.05, pval=
0.040394482824999266
Col 22 is statistically significantly different at alpha=0.05, pval=
0.028775596099077153
Col 36 is statistically significantly different at alpha=0.05, pval=
0.027595331942215046
Col 45 is statistically significantly different at alpha=0.05, pval=
0.018616520225533666
Col 75 is statistically significantly different at alpha=0.05, pval=
0.0054097376131833925
Col 80 is statistically significantly different at alpha=0.05, pval=
0.02457251202079053
Total number different was 6, which is 6.0%
```

In [17]:

```
# So, keep this in mind when you are doing statistical tests like the t-test whi
ch has a p-value. Understand
# that this p-value isn't magic, that it's a threshold for you when reporting re
sults and trying to answer
# your hypothesis. What's a reasonable threshold? Depends on your question, and
 you need to engage domain
# experts to better understand what they would consider significant.

# Just for fun, lets recreate that second dataframe using a non-normal distribut
ion, I'll arbitrarily chose
# chi squared
df2=pd.DataFrame([np.random.chisquare(df=1,size=100) for x in range(100)])
test_columns()
```

```
Col 0 is statistically significantly different at alpha=0.1, pval=3.
947099095449562e-05
Col 1 is statistically significantly different at alpha=0.1, pval=0.
0034921477656637825
Col 2 is statistically significantly different at alpha=0.1, pval=3.
385651416158829e-05
Col 3 is statistically significantly different at alpha=0.1, pval=0.
0010057724737930188
Col 4 is statistically significantly different at alpha=0.1, pval=0.
00011698887207642596
Col 5 is statistically significantly different at alpha=0.1, pval=0.
002012604140690967
Col 6 is statistically significantly different at alpha=0.1, pval=0.
0002459239346874621
Col 7 is statistically significantly different at alpha=0.1, pval=0.
00013917078112813346
Col 8 is statistically significantly different at alpha=0.1, pval=0.
00026162430524005417
Col 9 is statistically significantly different at alpha=0.1, pval=0.
0005430388492998936
Col 10 is statistically significantly different at alpha=0.1, pval=
0.00011134971234062867
Col 11 is statistically significantly different at alpha=0.1, pval=
0.001947792126602745
Col 12 is statistically significantly different at alpha=0.1, pval=
0.0018221722156074652
Col 13 is statistically significantly different at alpha=0.1, pval=
1.3687748299940754e-05
Col 14 is statistically significantly different at alpha=0.1, pval=
0.00184188122454627 7
Col 15 is statistically significantly different at alpha=0.1, pval=
0.010629591618320552
Col 16 is statistically significantly different at alpha=0.1, pval=
0.00020470409608789374
Col 17 is statistically significantly different at alpha=0.1, pval=
0.007112807536908285
Col 18 is statistically significantly different at alpha=0.1, pval=
0.0016698318956031688
Col 19 is statistically significantly different at alpha=0.1, pval=
0.0003753652462699119
Col 20 is statistically significantly different at alpha=0.1, pval=
0.007139953378405039
Col 21 is statistically significantly different at alpha=0.1, pval=
0.000312748296229491
Col 22 is statistically significantly different at alpha=0.1, pval=
0.0005544100555922932
Col 23 is statistically significantly different at alpha=0.1, pval=
0.00019418457682373135
Col 24 is statistically significantly different at alpha=0.1, pval=
0.002299090381050996
Col 25 is statistically significantly different at alpha=0.1, pval=
0.000996993381307106
Col 26 is statistically significantly different at alpha=0.1, pval=
0.0012105395439893116
Col 27 is statistically significantly different at alpha=0.1, pval=
2.631879527331853e-06
Col 28 is statistically significantly different at alpha=0.1, pval=
0.002261001649196019
Col 29 is statistically significantly different at alpha=0.1, pval=
0.004322166281964971
Col 30 is statistically significantly different at alpha=0.1, pval=
```

```
0.0037090051397635545
Col 31 is statistically significantly different at alpha=0.1, pval=
0.00434516298538967
Col 32 is statistically significantly different at alpha=0.1, pval=
0.0007855752938791443
Col 33 is statistically significantly different at alpha=0.1, pval=
0.00434086290104116
Col 34 is statistically significantly different at alpha=0.1, pval=
7.215093549818593e-05
Col 35 is statistically significantly different at alpha=0.1, pval=
1.5827641703177774e-05
Col 36 is statistically significantly different at alpha=0.1, pval=
0.0036405484445250825
Col 37 is statistically significantly different at alpha=0.1, pval=
0.0003469098888396098
Col 38 is statistically significantly different at alpha=0.1, pval=
2.3457452194037483e-05
Col 39 is statistically significantly different at alpha=0.1, pval=
4.4288074751257864e-05
Col 40 is statistically significantly different at alpha=0.1, pval=
0.02381878845187583
Col 41 is statistically significantly different at alpha=0.1, pval=
0.0006893870942072854
Col 42 is statistically significantly different at alpha=0.1, pval=
0.0010565633956565357
Col 43 is statistically significantly different at alpha=0.1, pval=
0.002367448954685127
Col 44 is statistically significantly different at alpha=0.1, pval=
0.0009159242631613687
Col 45 is statistically significantly different at alpha=0.1, pval=
1.1488088053789511e-05
Col 46 is statistically significantly different at alpha=0.1, pval=
0.001799277645804236
Col 47 is statistically significantly different at alpha=0.1, pval=
0.0003684539882054862
Col 48 is statistically significantly different at alpha=0.1, pval=
0.0010992312304164483
Col 49 is statistically significantly different at alpha=0.1, pval=
4.104454473676489e-05
Col 50 is statistically significantly different at alpha=0.1, pval=
0.0001460362405016095
Col 51 is statistically significantly different at alpha=0.1, pval=
0.0011512077694935995
Col 52 is statistically significantly different at alpha=0.1, pval=
0.00034989120622442384
Col 53 is statistically significantly different at alpha=0.1, pval=
5.5310629347799495e-06
Col 54 is statistically significantly different at alpha=0.1, pval=
0.07775487563551316
Col 55 is statistically significantly different at alpha=0.1, pval=
0.0005278774407072997
Col 56 is statistically significantly different at alpha=0.1, pval=
6.247506950352275e-07
Col 57 is statistically significantly different at alpha=0.1, pval=
0.0033139699226201963
Col 58 is statistically significantly different at alpha=0.1, pval=
8.276817045586344e-05
Col 59 is statistically significantly different at alpha=0.1, pval=
0.009582021162370159
Col 60 is statistically significantly different at alpha=0.1, pval=
0.00048310173366168506
```

```
Col 61 is statistically significantly different at alpha=0.1, pval=
1.6117869802064275e-05
Col 62 is statistically significantly different at alpha=0.1, pval=
0.0006000853744699587
Col 63 is statistically significantly different at alpha=0.1, pval=
6.192655785066324e-05
Col 64 is statistically significantly different at alpha=0.1, pval=
0.0004449515307216615
Col 65 is statistically significantly different at alpha=0.1, pval=
0.0017751781342693318
Col 66 is statistically significantly different at alpha=0.1, pval=
0.0032751498489150647
Col 67 is statistically significantly different at alpha=0.1, pval=
0.006645288330260071
Col 68 is statistically significantly different at alpha=0.1, pval=
0.0010534324833306205
Col 69 is statistically significantly different at alpha=0.1, pval=
0.056084914625760936
Col 70 is statistically significantly different at alpha=0.1, pval=
0.0019553778853853387
Col 71 is statistically significantly different at alpha=0.1, pval=
6.05871326381493e-05
Col 72 is statistically significantly different at alpha=0.1, pval=
1.3253288561583667e-06
Col 73 is statistically significantly different at alpha=0.1, pval=
0.00013268929953854372
Col 74 is statistically significantly different at alpha=0.1, pval=
0.0346603875252702
Col 75 is statistically significantly different at alpha=0.1, pval=
0.004366164921909572
Col 76 is statistically significantly different at alpha=0.1, pval=
0.00747758607318877
Col 77 is statistically significantly different at alpha=0.1, pval=
0.0034664750710030051
Col 78 is statistically significantly different at alpha=0.1, pval=
0.001140074294709592
Col 79 is statistically significantly different at alpha=0.1, pval=
0.0018483541121686636
Col 80 is statistically significantly different at alpha=0.1, pval=
0.00032844676380595363
Col 81 is statistically significantly different at alpha=0.1, pval=
0.0027579884413592923
Col 82 is statistically significantly different at alpha=0.1, pval=
0.002834701358304507
Col 83 is statistically significantly different at alpha=0.1, pval=
3.14598354511748e-05
Col 84 is statistically significantly different at alpha=0.1, pval=
0.005022519649328865
Col 85 is statistically significantly different at alpha=0.1, pval=
0.016270909161640822
Col 86 is statistically significantly different at alpha=0.1, pval=
2.6587962913824178e-05
Col 87 is statistically significantly different at alpha=0.1, pval=
0.0008782876458182688
Col 88 is statistically significantly different at alpha=0.1, pval=
0.017333863309704312
Col 89 is statistically significantly different at alpha=0.1, pval=
0.001938026045263344
Col 90 is statistically significantly different at alpha=0.1, pval=
0.007304994267962801
Col 91 is statistically significantly different at alpha=0.1, pval=
```

```
2.453298143245935e-05
Col 92 is statistically significantly different at alpha=0.1, pval=
0.009082737410365206
Col 93 is statistically significantly different at alpha=0.1, pval=
2.1831881880376394e-05
Col 94 is statistically significantly different at alpha=0.1, pval=
1.2603707591495165e-06
Col 95 is statistically significantly different at alpha=0.1, pval=
0.006270908050595888
Col 96 is statistically significantly different at alpha=0.1, pval=
0.003187363468672461
Col 97 is statistically significantly different at alpha=0.1, pval=
0.0015839853572104728
Col 98 is statistically significantly different at alpha=0.1, pval=
0.00010766753340497722
Col 99 is statistically significantly different at alpha=0.1, pval=
0.00016114530527999558
Total number different was 100, which is 100.0%
```

Now we see that all or most columns test to be statistically significant at the 10% level.

Summary

In this lecture, we've discussed just some of the basics of hypothesis testing in Python. I introduced you to the SciPy library, which you can use for the students t test. We've discussed some of the practical issues which arise from looking for statistical significance. There's much more to learn about hypothesis testing, for instance, there are different tests used, depending on the shape of your data and different ways to report results instead of just p-values such as confidence intervals or bayesian analyses. But this should give you a basic idea of where to start when comparing two populations for differences, which is a common task for data scientists.