index

January 20, 2022

1 ANOVA - Lab

1.1 Introduction

In this lab, you'll get some brief practice generating an ANOVA table (AOV) and interpreting its output. You'll also perform some investigations to compare the method to the t-tests you previously employed to conduct hypothesis testing.

1.2 Objectives

In this lab you will:

- Use ANOVA for testing multiple pairwise comparisons
- Interpret results of an ANOVA and compare them to a t-test

1.3 Load the data

Start by loading in the data stored in the file `'ToothGrowth.csv'`:

[1]:
```
# Your code here
import pandas as pd
import numpy as np

df = pd.read_csv('ToothGrowth.csv')
```

1.4 Generate the ANOVA table

Now generate an ANOVA table in order to analyze the influence of the medication and dosage:

[2]:
```
# Your code here
import statsmodels.api as sm
from statsmodels.formula.api import ols

formula = 'len ~ C(supp) + C(dose)'
lm = ols(formula, df).fit()
table = sm.stats.anova_lm(lm, typ=2)
print(table)
```

```
              sum_sq    df          F        PR(>F)
C(supp)   205.350000   1.0  14.016638  4.292793e-04
```

```
C(dose)    2426.434333   2.0  82.810935  1.871163e-17
Residual    820.425000  56.0        NaN           NaN
```

1.5 Interpret the output

Make a brief comment regarding the statistics and the effect of supplement and dosage on tooth length:

[3]:
```
# Your comment here

# As we can see, All of values in the far right column are significantly
# smaller than 0.05, so we can reject the null hypothesis.

# Both dose and supplement type are impactful. At first glance, dosage seems
# to be the more impactful of the two.
```

1.6 Compare to t-tests

Now that you've had a chance to generate an ANOVA table, its interesting to compare the results to those from the t-tests you were working with earlier. With that, start by breaking the data into two samples: those given the OJ supplement, and those given the VC supplement. Afterward, you'll conduct a t-test to compare the tooth length of these two different samples:

[4]:
```
df.head()
```

[4]:

	len	supp	dose
0	4.2	VC	0.5
1	11.5	VC	0.5
2	7.3	VC	0.5
3	5.8	VC	0.5
4	6.4	VC	0.5

[5]:
```
# Your code here
import scipy.stats as stats
VC = df[df["supp"] == "VC"]["len"]
OJ = df[df["supp"] == "OJ"]["len"]

print(len(VC))
print(VC.std())

print("\n")

print(len(OJ))
print(OJ.std())
```

```
30
8.266028664664638
```

```
30
6.605561049722362
```

Now run a t-test between these two groups and print the associated two-sided p-value:

[6]:
```
# Calculate the 2-sided p-value for a t-test comparing the two supplement groups
results = stats.ttest_ind(VC, OJ, equal_var=False)
results.pvalue
```

[6]:
```
0.06063450788093387
```

1.7 A 2-Category ANOVA F-test is equivalent to a 2-tailed t-test!

Now, recalculate an ANOVA F-test with only the supplement variable. An ANOVA F-test between two categories is the same as performing a 2-tailed t-test! So, the p-value in the table should be identical to your calculation above.

> Note: there may be a small fractional difference (>0.001) between the two values due to a rounding error between implementations.

[7]:
```
# Your code here; conduct an ANOVA F-test of the oj and vc supplement groups.

formula = 'len ~ C(supp)'
lm = ols(formula, df).fit()
table = sm.stats.anova_lm(lm, typ=2)
print(table)

# Compare the p-value to that of the t-test above.
# They should match (there may be a tiny fractional difference due to
# rounding errors in varying implementations)
```

```
               sum_sq    df         F    PR(>F)
C(supp)    205.350000   1.0  3.668253  0.060393
Residual  3246.859333  58.0       NaN       NaN
```

1.8 Run multiple t-tests

While the 2-category ANOVA test is identical to a 2-tailed t-test, performing multiple t-tests leads to the multiple comparisons problem. To investigate this, look at the various sample groups you could create from the 2 features:

[9]:
```
for group in df.groupby(['supp', 'dose'])['len']:
    group_name = group[0]
    data = group[1]
    print(group_name)
```

```
('OJ', 0.5)
('OJ', 1.0)
('OJ', 2.0)
```

```
('VC', 0.5)
('VC', 1.0)
('VC', 2.0)
```

[39]:
```python
l = list(df.groupby(['supp', 'dose'])['len'])
pd.DataFrame(l)
```

[39]:
```
             0                                                   1
0  (OJ, 0.5)  30    15.2
31    21.5
32    17.6
33     9.7
34…
1  (OJ, 1.0)  40    19.7
41    23.3
42    23.6
43    26.4
44…
2  (OJ, 2.0)  50    25.5
51    26.4
52    22.4
53    24.5
54…
3  (VC, 0.5)  0     4.2
1    11.5
2     7.3
3     5.8
4     …
4  (VC, 1.0)  10    16.5
11    16.5
12    15.2
13    17.3
14…
5  (VC, 2.0)  20    23.6
21    18.5
22    33.9
23    25.5
24…
```

While bad practice, examine the effects of calculating multiple t-tests with the various combinations of these. To do this, generate all combinations of the above groups. For each pairwise combination, calculate the p-value of a 2-sided t-test. Print the group combinations and their associated p-value for the two-sided t-test.

[12]:
```python
# Your code here; reuse your t-test code above to calculate the p-value for
# a 2-sided t-test
# for all combinations of the supplement-dose groups listed above.
# (Since there isn't a control group, compare each group to every other group.)
```

```python
### From GitHub Solution
from itertools import combinations

groups = [group[0] for group in df.groupby(['supp', 'dose'])['len']]
combos = combinations(groups, 2)
for combo in combos:
    supp1 = combo[0][0]
    dose1 = combo[0][1]
    supp2 = combo[1][0]
    dose2 = combo[1][1]
    sample1 = df[(df.supp == supp1) & (df.dose == dose1)]['len']
    sample2 = df[(df.supp == supp2) & (df.dose == dose2)]['len']
    p = stats.ttest_ind(sample1, sample2, equal_var=False)[1]
    print(combo, p)

    # Note that while ANOVA also concluded that all factors were significant,
    # these p-values are substantially lower.
```

```
(('OJ', 0.5), ('OJ', 1.0)) 8.784919055161479e-05
(('OJ', 0.5), ('OJ', 2.0)) 1.3237838776972294e-06
(('OJ', 0.5), ('VC', 0.5)) 0.006358606764096813
(('OJ', 0.5), ('VC', 1.0)) 0.04601033257637553
(('OJ', 0.5), ('VC', 2.0)) 7.196253524006043e-06
(('OJ', 1.0), ('OJ', 2.0)) 0.039195142046244004
(('OJ', 1.0), ('VC', 0.5)) 3.6552067303259103e-08
(('OJ', 1.0), ('VC', 1.0)) 0.001038375872299884
(('OJ', 1.0), ('VC', 2.0)) 0.09652612338267014
(('OJ', 2.0), ('VC', 0.5)) 1.3621396478988818e-11
(('OJ', 2.0), ('VC', 1.0)) 2.3610742020468435e-07
(('OJ', 2.0), ('VC', 2.0)) 0.9638515887233756
(('VC', 0.5), ('VC', 1.0)) 6.811017702865016e-07
(('VC', 0.5), ('VC', 2.0)) 4.6815774144921145e-08
(('VC', 1.0), ('VC', 2.0)) 9.155603056638692e-05
```

1.9 Summary

In this lesson, you implemented the ANOVA technique to generalize testing methods to multiple groups and factors.