How to Perform a Correlation Test in Python (With Example)

BY ZACH BOBBITT APRIL 6, 2022

One way to quantify the relationship between two variables is to use the Pearson correlation coefficient, which measures the linear association between two variables.

It always takes on a value between -1 and 1 where:

- **-1** indicates a perfectly negative linear correlation
- **0** indicates no linear correlation
- **1** indicates a perfectly positive linear correlation

To determine if a correlation coefficient is statistically significant, you can calculate the corresponding t-score and p-value.

The formula to calculate the t-score of a correlation coefficient (r) is:

$$\mathbf{t} = r * \sqrt{n-2} / \sqrt{1-r^2}$$

The p-value is then calculated as the corresponding two-sided p-value for the t-distribution with n-2 degrees of freedom.

Example: Correlation Test in Python

To determine if the correlation coefficient between two variables is statistically significant, you can perform a correlation test in Python using the **pearsonr** function from the **SciPy** library.

This function returns the correlation coefficient between two variables along with the two-tailed p-value.

For example, suppose we have the following two arrays in Python:

```
#create two arrays
x = [3, 4, 4, 5, 7, 8, 10, 12, 13, 15]
y = [2, 4, 4, 5, 4, 7, 8, 19, 14, 10]
```

We can import the **pearsonr** function and calculate the Pearson correlation coefficient between the two arrays:

```
from scipy.stats.stats import pearsonr

#calculation correlation coefficient and p-value be
pearsonr(x, y)

(0.8076177030748631, 0.004717255828132089)
```

Here's how to interpret the output:

- Pearson correlation coefficient (r): **0.8076**
- Two-tailed p-value: **0.0047**

Since the correlation coefficient is close to 1, this tells us that there is a strong positive association between the two variables.

And since the corresponding p-value is less than .05, we conclude that there is a statistically significant association between the two variables.

Note that we can also extract the individual correlation coefficient and p-value from the **pearsonr** function as well:

```
#extract correlation coefficient (rounded to 4 deci
r = round(pearsonr(x, y)[0], 4)

print(r)

0.8076

#extract p-value (rounded to 4 decimal places)
p = round(pearsonr(x, y)[1], 4)

print(p)

0.0047
```

These values are a bit easier to read compared to the output from the original **pearsonr** function.

Additional Resources

The following tutorials provide additional information about correlation coefficients:

An Introduction to the Pearson Correlation Coefficient