Learning Hypothesis Testing with Python: A Practical Guide with Examples

Authored by

Mohammed looti

October 29, 2025

A Hypothesis Test is a formal procedure in inferential statistics used to assess the plausibility of a statistical hypothesis regarding a population parameter. This process allows us to make informed decisions about populations based on sample data, leading us to either reject or fail to reject the proposed hypothesis.

This comprehensive tutorial demonstrates how to efficiently perform three common types of t-tests using the powerful **scipy.stats** library in Python:

The **One Sample t-test**, for comparing a single mean to a known value.
The **Two Sample t-test** (Independent Samples t-test), for comparing the means of two distinct, independent groups.
The **Paired Samples t-test** (Dependent Samples t-test), for analyzing differences in paired observations (e.g., before and after measurements).

Let's delve into practical examples for each method.

Example 1: One Sample t-test in Python

The **One Sample t-test** is employed when the objective is to determine whether the unknown population mean (mu) is statistically equal to a specific hypothesized value. This test is foundational for comparing sample averages against pre-established standards or theoretical expectations.

Consider a scenario where conservationists are studying a specific species of turtle. We want to test the claim that the average weight of this population is 310 pounds. We must gather empirical data to validate or reject this claim.

To conduct the test, we collect a simple random sample of turtles, resulting in the following measured weights (in pounds):

Weights: 300, 315, 320, 311, 314, 309, 300, 308, 305, 303, 305, 301, 303

We utilize the powerful ttest_1samp() function available within the **scipy.stats** library to execute this one sample t-test, comparing our sample mean against the hypothesized population mean of 310:

import scipy.stats as stats

#define data
data =

#perform one sample t-test

```
stats.ttest_1samp(a=data, popmean=310)

Ttest_1sampResult(statistic=-1.5848116313861254, pvalue=0.1389944275158753)
```

The output provides two critical values: the calculated **t test statistic** of **-1.5848** and the corresponding two-sided p-value of **0.1389**. These values are essential for the final decision-making process in hypothesis testing.

The two competing hypotheses established for this test are:

H0: μ = 310 (The Null Hypothesis states that the true mean weight for this species of turtle is 310 pounds.)
HA: μ ? 310 (The Alternative Hypothesis states that the mean weight is *not* 310 pounds.)

To make a decision, we compare the p-value (0.1389) against our predetermined significance level ($alpha$), typically set at 0.05. Since the obtained p-value is greater than $alpha$ (0.1389 > 0.05), we **fail to reject the null hypothesis**. This implies that based on our collected sample, we lack sufficient statistical evidence to definitively conclude that the mean weight of the turtle species differs from 310 pounds.

Example 2: Two Sample t-test in Python

A **Two Sample t-test**, also known as an Independent Samples t-test, is designed to compare the means of two entirely separate and independent populations. This test is critical for scenarios where we need to determine if a statistically significant difference exists between the averages of two groups.

Expanding on the previous example, suppose we now want to compare the average weight between two distinct species of turtles (Species A and Species B). Our goal is to assess whether the mean weight of Species A is equal to the mean weight of Species B.

We gather separate random samples from both populations, yielding the following weight measurements:

Sample 1 (Species A): 300, 315, 320, 311, 314, 309, 300, 308, 305, 303, 305, 301, 303

Sample 2 (Species B): 335, 329, 322, 321, 324, 319, 304, 308, 305, 311, 307, 300, 305

To perform this comparison, we use the ttest_ind() function from the **scipy.stats** library. This function calculates the t-statistic and p-value necessary for evaluating the difference between the two sample means:

import scipy.stats as stats

#define array of turtle weights for each sample
sample1 =
sample2 =

#perform two sample t-test
stats.ttest_ind(a=sample1, b=sample2)

Ttest_indResult(statistic=-2.1009029257555696, pvalue=0.04633501389516516)

The resulting **t test statistic** is **-2.1009**, and the corresponding two-sided p-value is **0.0463**. We now use these results to test the following hypotheses:

The hypotheses for this comparison are formulated as:

H0: µ1 = µ2 (The mean weight of Species A is statistically equal to the mean weight of Species B.)
HA: µ1 ? µ2 (The mean weights between the two species are not equal.)

By comparing the calculated p-value (0.0463) to our standard significance level of $alpha = 0.05$, we find that the p-value is less than the threshold (0.0463 < 0.05). Consequently, we **reject the null hypothesis**. This outcome suggests strong statistical evidence that the mean weight between the two species is significantly different.

Example 3: Paired Samples t-test in Python

The **Paired Samples t-test** (or dependent t-test) is specifically designed for situations where observations in one sample are directly related or matched to observations in the second sample. This is commonly used in 'before-and-after' studies, crossover trials, or when comparing two different treatments applied to the same subjects.

Imagine we are evaluating the effectiveness of a new training program designed to increase the maximum vertical jump (measured in inches) of basketball players. Since we are measuring the same individuals twice, the 'before' and 'after' measurements are paired, making this test appropriate.

We recruit a simple random sample of 12 college basketball players. We measure their initial vertical jump, implement the training program for one month, and then measure their jump height again. The collected data is summarized below:

Before Training: 22, 24, 20, 19, 19, 20, 22, 25, 24, 23, 22, 21

After Training: 23, 25, 20, 24, 18, 22, 23, 28, 24, 25, 24, 20

To perform this paired comparison, we utilize the `ttest_rel()` function from the scipy.stats library. The "rel" suffix signifies a test on related (dependent) samples:

import scipy.stats as stats

#define before and after max jump heights
before =
after =

#perform paired samples t-test
stats.ttest_rel(a=before, b=after)

Ttest_relResult(statistic=-2.5289026942943655, pvalue=0.02802807458682508)

The statistical output shows a **t test statistic** of **-2.5289** and a two-sided p-value of **0.0280**. We proceed by defining our hypotheses based on whether the training program caused a change:

The hypotheses tested are:

H0: μ1 = μ2 (The mean jump height before and after using the program is statistically identical.)
HA: μ1 ? μ2 (The mean jump height has changed significantly following the program.)

Given that the calculated p-value (0.0280) is less than our significance level of $alpha = 0.05$, we **reject the null hypothesis**. This outcome suggests strong statistical evidence that the training program successfully altered the mean maximum vertical jump height of the basketball players.

Additional Resources for t-Tests

Understanding the implementation of these statistical tests in Python using the **scipy.stats** library is crucial for robust data analysis. For quick validation or learning purposes, you may find the following external online calculators useful for automatically performing various t-tests: