Learning Z-Tests: A Practical Guide to One and Two Sample Z-Tests in Python

Authored by

Mohammed looti

November 2, 2025

RECOMMENDED CITATION

Mohammed looti (2025). *Learning Z-Tests: A Practical Guide to One and Two Sample Z-Tests in Python*. PSYCHOLOGICAL STATISTICS. Retrieved from https://statistics.arabpsychology.com/?p=8547

In the expansive discipline of **statistical inference**, the **Z-test** stands as a foundational method for drawing conclusions about population parameters based on sample data. This powerful test is primarily utilized in two scenarios: determining if a single sample mean significantly deviates from a known population mean, or assessing whether the means of two distinct samples exhibit a statistically significant difference. A critical distinction of the **Z-test**, differentiating it from the T-test, is the requirement that the **population standard deviation (σ)** must be either known, or the sample size must be sufficiently large (generally N > 30) to justify the application of the **Central Limit Theorem**.

Modern data analysis relies heavily on robust software tools, and Python is the preferred environment for statistical computing. Libraries such as **NumPy** and **Pandas** handle data manipulation efficiently, but for rigorous **hypothesis testing**, we turn to the specialized package: **statsmodels**. This library is specifically designed to provide classes and functions for statistical modeling, estimation, and testing, delivering results suitable for professional-grade research and data science applications.

The core functionality we will utilize is the `ztest()` function, which is nested within the `statsmodels.stats.weightstats` package. This function is engineered for versatility, allowing practitioners to effortlessly switch between one-sample and two-sample scenarios by simply adjusting the input parameters. By mastering the application of `ztest()`, analysts can streamline the process of obtaining reliable statistical conclusions from diverse datasets.

Prerequisites and Assumptions for the Z-Test

Before any statistical procedure can be validly applied, it is essential to confirm that the underlying data structure satisfies the test's critical assumptions. Failure to meet these prerequisites can severely compromise the accuracy of the calculated **Z-statistic** and the subsequent **P-value**, potentially leading to erroneous inferences about the studied populations. For the **Z-test**, these requirements are particularly stringent regarding data distribution and variance knowledge.

The primary distributional requirement is that the data must follow a **normal distribution**. Although strict normality is required for smaller sample sizes, the test demonstrates robustness when the sample size is large ($N \geq 30$), a situation where the **Central Limit Theorem** comes into effect, ensuring the distribution of sample means approaches normality regardless of the population distribution shape. Additionally, it is mandatory that all observations within the sample are **independent**; that is, the value recorded for one data point must not influence or be influenced by the value of any other data point.

A non-negotiable requirement for the classical **Z-test** is the knowledge of the **population standard deviation (σ)**. If this parameter is unknown, the statistically correct procedure shifts to a **t-test**, which uses the sample standard deviation to estimate the population variance, introducing

additional uncertainty accounted for by the T-distribution. However, in practice, if the population standard deviation is unknown but the sample size is extensive (N > 30), the sample standard deviation can be used as a reliable approximation of the population parameter, thereby allowing the continued, practical application of the Z-test.

Understanding the ztest() Function Syntax

To effectively execute Z-tests within the Python ecosystem, practitioners must first gain mastery over the syntax and parameters of the `ztest()` function. This function is designed to be highly adaptable, capable of executing both single-sample tests (comparison against a hypothesized mean) and two-sample tests (comparison between two means) based entirely on the configuration of its input arguments. The function returns a crucial two-element tuple consisting of the calculated **Z-statistic** and the corresponding **two-tailed P-value**.

The `ztest()` function is imported from the `statsmodels.stats.weightstats` module. Its primary parameters govern the input data and define the core assumption of the **null hypothesis** being tested. Proper specification of these arguments is paramount for ensuring the correct statistical procedure is executed:

statsmodels.stats.weightstats.ztest(*x1*, *x2=None*, *value=0)*

The roles of the three primary arguments dictate the test structure:

x1: This is the mandatory input, representing the data points for the first sample or the only sample under investigation. It must be provided as an array-like structure (e.g., a Python list or NumPy array).
x2: This parameter accepts data for the second sample. If `x2` is explicitly set to `None` (which is the default behavior), the function automatically initiates a **one-sample Z-test**, comparing `x1` against the hypothesized mean defined by `value`. If `x2` contains data, a **two-sample Z-test** is performed.
value: This critical argument specifies the value asserted by the **null hypothesis** (H_0). In a one-sample test, it is the hypothesized population mean (μ). In a two-sample test, it represents the hypothesized difference between the two population means (μ1 - μ2). For standard comparisons testing for equality (i.e., no difference), this parameter is typically set to 0.

Having established the theoretical framework and the necessary syntax, we now move to practical application, beginning with the procedure for comparing a single sample mean against a defined population standard.

Performing a One-Sample Z-Test in Python (Example 1)

The primary use case for the one-sample **Z-test** involves assessing whether a specific sample mean is statistically representative of, or significantly different from, a benchmark population mean that is assumed to be known. This methodology is fundamental in quality control, psychological research, and medical trials where new interventions are measured against established norms.

Consider a classical example from cognitive research: the average **IQ** score in the general population is standardized at μ = 100, with a known population standard deviation of σ = 15, reflecting its definition as a **normally distributed** variable. A pharmaceutical company develops a new cognitive enhancer and recruits 20 subjects to test its efficacy. The researcher's goal is to determine if the mean IQ score of this sample, post-treatment, deviates significantly from the population mean of 100.

The necessary statistical hypotheses are structured formally: the **null hypothesis** (H_0) posits that the drug has no effect, meaning the sample mean is equal to the population mean (μ = 100). The alternative hypothesis (H_a) asserts that the drug does cause a change (μ ? 100). We execute the `ztest()` function, providing only the sample data (`x1`) and setting the `value` parameter to 100, which is the hypothesized population mean under H_0.

from statsmodels.stats.weightstats import ztest as ztest

Sample data: IQ levels for 20 patients after drug treatment
data =

# Perform one sample z-test against a hypothesized mean of 100
ztest(data, value=100)

(1.5976240527147705, 0.1101266701438426)

The execution of this Python code yields the numerical output required for the inferential step: the calculated Z-statistic and its paired **P-value**. The next crucial phase involves correctly interpreting these quantitative measures within the framework of statistical significance to draw a definitive conclusion regarding the drug's effect.

Interpreting One-Sample Z-Test Results

The numerical output `(1.5976240527147705, 0.1101266701438426)` is the culmination of the statistical calculation. The first element, **1.5976**, is the **test statistic (Z-score)**, which quantifies the distance, in standard error units, between the observed sample mean and the hypothesized population mean of 100. The second element, **0.1101**, is the corresponding two-tailed **P-value**, representing the probability of observing a sample mean as extreme as, or more extreme than, the one observed, assuming the **null hypothesis** (H_0) is true.

The decision rule in **hypothesis testing** requires comparing the calculated **P-value** against a predetermined significance level, traditionally denoted as $alpha$. In most scientific and social research, $alpha$ is set at 0.05. If the P-value is less than or equal to 0.05, we reject H_0, concluding that the observed difference is statistically significant. Conversely, if the P-value is greater than 0.05, we retain H_0, concluding that there is insufficient evidence to claim a significant difference.

Applying this rule to our example, the calculated **P-value** (0.1101) is substantially larger than the standard significance threshold of 0.05. Consequently, we must fail to reject the **null hypothesis**. This statistical outcome suggests that, based on the data collected from the 20 subjects, there is no statistically significant evidence to conclude that the new cognitive enhancement drug alters the mean IQ level when compared to the established population mean of 100.

Performing a Two-Sample Z-Test in Python (Example 2)

When the research objective shifts to comparing the means of two independent groups, the **two-sample Z-test** becomes the appropriate statistical instrument. This procedure is designed to determine if any observed difference between the two sample means is likely due to genuine differences in the underlying populations or merely attributable to random sampling variability. As with the one-sample test, this procedure requires the assumption that the population standard deviations for both groups are known or that both samples are large.

Imagine a geographical comparative study aiming to evaluate the mean IQ scores between two different populations residing in City A and City B. We assume that IQ scores in both regions are **normally distributed** and that we have reliable estimates or knowledge of the population standard deviations for both cities. A researcher collects random samples of 20 individuals from each city, resulting in two separate datasets for comparison.

The **null hypothesis** (H_0) for this comparative test assumes that there is no difference between the population means (μ_A - μ_B = 0). The alternative hypothesis (H_a) suggests that a difference does exist (μ_A - μ_B ? 0). To execute this test using `ztest()`, we supply both the `cityA` data (`x1`) and the `cityB` data (`x2`). By keeping the `value` parameter at its default of 0, we are testing the assertion of zero mean difference under H_0.

from statsmodels.stats.weightstats import ztest as ztest

Sample data for City A
cityA =

Sample data for City B
cityB =

```
# Perform two sample z-test, testing for a mean difference of 0
ztest(cityA, cityB, value=0)
```

```
(-1.9953236073282115, 0.046007596761332065)
```

The generated output, `(-1.9953236073282115, 0.046007596761332065)`, provides the Z-statistic and the **P-value** necessary for the final determination. The negative Z-statistic (**-1.9953**) suggests that the mean of the first sample (City A) is lower than the mean of the second sample (City B). The statistical significance of this observed difference is determined by examining the accompanying **P-value** of **0.0460**.

Since this P-value (0.0460) is marginally less than the conventional $alpha = 0.05$ threshold, we have achieved sufficient statistical evidence to reject the **null hypothesis**. This leads us to conclude that there is a statistically significant difference in the mean IQ levels between the populations represented by City A and City B.

Conclusion and Best Practices in Hypothesis Testing

The **`ztest()`** function, a component of the powerful **statsmodels** library, provides a streamlined and reliable mechanism for conducting both one-sample and two-sample Z-tests within Python. This functionality is essential for data scientists and analysts tasked with making inferences about population parameters. The reliability of the conclusions drawn hinges entirely upon rigorous adherence to the core assumptions of the test, particularly the assurance of a known population standard deviation or reliance on the large-sample applicability afforded by the **Central Limit Theorem**.

Mastering the workflow of **hypothesis testing** is a core competency in data analysis. This process involves four key steps: accurately defining the **null hypothesis** and alternative hypothesis, selecting the appropriate statistical test (Z-test vs. T-test), executing the test using tools like **`ztest()`**, and finally, correctly interpreting the resulting Z-statistic and **P-value** relative to the chosen significance level.

By consistently applying these principles, analysts can move beyond descriptive statistics to make robust, evidence-based conclusions that directly inform business strategies, medical research outcomes, and policy decisions across various industries.

For those looking to expand their knowledge of statistical inference using Python, the following additional resources cover other essential statistical tests and procedures commonly implemented in data analysis workflows: