



★ Member-only story

A/B Testing with Python: A Complete Guide



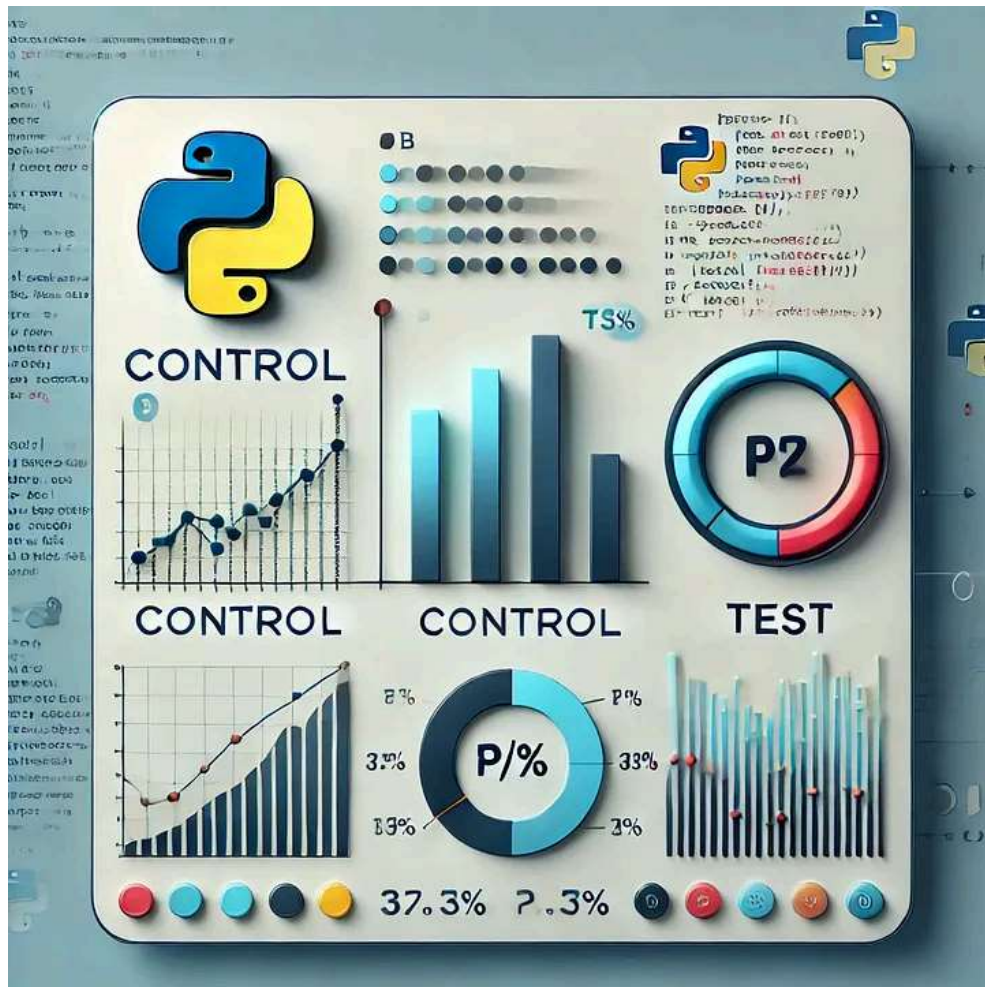
Ritesh Gupta · Follow

6 min read · 1 hour ago



Conducting an A/B Test Using Python: A Step-by-Step Guide

A/B testing is a powerful technique used by businesses and data scientists to compare two versions of a product, campaign, or website to determine which performs better. In this article, we will demonstrate how to conduct an A/B test in Python using a statistical approach. We will use a dataset that compares two groups, Control and Test (or Variant), and measure revenue differences between them.



What is A/B Testing?

A/B testing is a statistical method used to compare two or more groups to determine whether one group performs better than another. For example, in digital marketing, an A/B test might compare two different landing pages to see which one drives more conversions.

The two main hypotheses involved in A/B testing are:

- **Null Hypothesis (H0):** There is no significant difference between the groups.
- **Alternative Hypothesis (H1):** There is a significant difference between the groups.

If the test result rejects the null hypothesis, it indicates that the changes implemented in the Test group have a significant impact.

Step-by-Step A/B Test with Python

1. Import Libraries

To get started, we need to import necessary Python libraries. This includes libraries for data handling, visualization, and statistical testing.

```
import itertools
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import statsmodels.stats.api as sms
from scipy.stats import mannwhitneyu
import warnings
warnings.filterwarnings("ignore")
```

2. Data Preparation

The dataset typically contains revenue information for both Control and Test groups. For this example, let's assume you have two pandas DataFrames:

`control` and `test`. Each DataFrame contains a column called `REVENUE` representing the income generated by each group.

3. Statistical Test: Mann-Whitney U Test

To determine whether there is a statistically significant difference between the revenue of the Control and Test groups, we can use the Mann-Whitney U test, which is a non-parametric test suited for comparing two independent samples.

```
test_stat, pvalue = mannwhitneyu(control["REVENUE"], test['REVENUE'])
print('Test stat = %.4f, p-value = %.4f' % (test_stat, pvalue))
```

The output of this test will provide a test statistic and a p-value.

4. Interpretation of Results

Based on the p-value obtained, we can interpret the results as follows:

- If the p-value is less than the chosen significance level (typically 0.05), we reject the null hypothesis, indicating that there is a significant difference between the two groups.
- If the p-value is greater than 0.05, we fail to reject the null hypothesis, meaning there is no significant difference.

In the output of our example:

```
Test stat = 12521564.0000, p-value = 0.4783
```

The p-value is 0.4783, which is greater than 0.05, so we **fail to reject the null hypothesis**. This means there is no statistically significant difference in terms of revenue between the Control and Test groups.

Code Example Recap

Below is the full code used for this A/B test:

```
import itertools
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import statsmodels.stats.api as sms
from scipy.stats import mannwhitneyu
import warnings
warnings.filterwarnings("ignore")

# Mann-Whitney U test
test_stat, pvalue = mannwhitneyu(control["REVENUE"], test["REVENUE"])
print('Test stat = %.4f, p-value = %.4f' % (test_stat, pvalue))

# Interpretation of the result
if pvalue < 0.05:
    print("Reject the null hypothesis: The two groups are significantly different")
else:
    print("Fail to reject the null hypothesis: No significant difference between groups")
```

Advanced Topics in A/B Testing

Now that we've covered the basics, let's dive into some advanced topics that will help you run more effective and accurate A/B tests.

1. Choosing the Right Statistical Test

In A/B testing, it's crucial to choose the right statistical test based on the type of data you're working with. The Mann-Whitney U test, which we used above, is a non-parametric test suited for non-normal distributions. Here are other commonly used tests in A/B testing:

- **T-test:** Used when comparing the means of two groups and assuming the data follows a normal distribution. The **independent t-test** compares the means of two independent samples.

- **Chi-square test:** Used for categorical data (e.g., conversion rates), where you want to compare proportions between two groups.
- **ANOVA (Analysis of Variance):** Used when comparing means across three or more groups.

Example of a t-test implementation:

```
from scipy.stats import ttest_ind

t_stat, pvalue = ttest_ind(control['REVENUE'], test['REVENUE'])
print(f'Test stat = {t_stat:.4f}, p-value = {pvalue:.4f}')
```

2. Effect Size and Statistical Power

While the p-value helps us determine if the difference between groups is statistically significant, it doesn't tell us how large that difference is. This is where **effect size** comes into play. The effect size quantifies the magnitude of the difference, giving context to the results.

Cohen's d is a common way to calculate effect size:

```
def cohen_d(control, test):
    diff = control.mean() - test.mean()
    pooled_sd = np.sqrt(((len(control) - 1) * control.var() + (len(test) - 1) *
    return diff / pooled_sd

effect_size = cohen_d(control['REVENUE'], test['REVENUE'])
print(f'Cohen\'s d: {effect_size:.4f}')
```

You should also consider the **statistical power** of the test, which is the probability of correctly rejecting the null hypothesis. Running a power analysis ensures your test is sufficiently powered to detect meaningful effects. Python's `statsmodels` library provides a function for this:

```
from statsmodels.stats.power import tt_ind_solve_power

# Example: Power analysis for the t-test
power = tt_ind_solve_power(effect_size=effect_size, nobs1=len(control), alpha=0.
print(f'Statistical Power: {power:.4f}')
```

3. Handling Multiple Variants (A/B/n Testing)

In many cases, you might want to test more than just two versions (A/B/n tests). For example, you may have three or more variants of a website or product. This requires comparing more than two groups, often using ANOVA for continuous data and **chi-square tests** for categorical data.

Here's an example of running ANOVA for three or more groups:

```
from scipy.stats import f_oneway

group1 = np.random.randn(100) + 0.5 # Example data for group 1
group2 = np.random.randn(100)        # Example data for group 2
group3 = np.random.randn(100) - 0.5 # Example data for group 3

f_stat, pvalue = f_oneway(group1, group2, group3)
print(f'F-statistic: {f_stat:.4f}, p-value: {pvalue:.4f}')
```

If the p-value from ANOVA is significant, you can follow up with post-hoc tests like Tukey's HSD to identify which specific groups differ from each other.

4. Visualizing Results

Data visualization can help communicate the results of an A/B test effectively. Here are some common plots used for A/B testing:

- **Box plots** for comparing distributions of Control vs Test groups:

```
sns.boxplot(data=[control['REVENUE'], test['REVENUE']], palette="Set2")
plt.xticks([0, 1], ['Control', 'Test'])
plt.ylabel('Revenue')
plt.title('Revenue Distribution: Control vs Test')
plt.show()
```

- **Conversion rate plots** (for categorical data):

```
conversion_rates = pd.DataFrame({
    'Group': ['Control', 'Test'],
    'Conversion Rate': [control_conversion_rate, test_conversion_rate]
})

sns.barplot(x='Group', y='Conversion Rate', data=conversion_rates)
```

```
plt.title('Conversion Rates: Control vs Test')
plt.show()
```

Visualizations can also be customized to display cumulative conversions or uplift charts that show the difference in conversion rates over time.

5. Confidence Intervals

Confidence intervals (CI) provide a range of values within which the true difference between the groups is likely to fall. For example, a 95% confidence interval suggests that there's a 95% chance that the true difference lies within the interval.

To compute a 95% confidence interval in Python, we can use the `sms.DescrStatsW` function from the `statsmodels` library:

```
import statsmodels.api as sm

control_stats = sms.DescrStatsW(control['REVENUE'])
test_stats = sms.DescrStatsW(test['REVENUE'])

diff_stats = sms.CompareMeans(control_stats, test_stats)
confidence_interval = diff_stats.tconfint_diff(usevar='unequal')
print(f'95% Confidence Interval: {confidence_interval}')
```

6. Avoiding Common Pitfalls

- **Peeking at Results:** Checking results too frequently can inflate the likelihood of false positives. Make sure to wait until the test reaches the predetermined sample size before drawing conclusions.
- **Simpson's Paradox:** Be careful when breaking data into smaller subgroups, as the trends observed in individual subgroups may reverse when the data is combined.

7. Running A/B Tests for Online Businesses

For e-commerce stores like your **Asli Kart**, A/B testing can be used for a variety of purposes:

- **Product page optimizations:** Testing different layouts or pricing models.
- **Checkout flow improvements:** Reducing cart abandonment by tweaking user flow.
- **Marketing campaigns:** Email subject line or ad creative comparisons.

With accurate and well-planned A/B testing, **Asli Kart** can better understand customer preferences, leading to higher conversions and customer satisfaction.

Conclusion

A/B testing is an invaluable tool for making data-driven decisions. Python provides a range of libraries that make it easy to implement and analyze A/B tests. Whether you're comparing website versions, marketing campaigns, or product features, the right combination of statistical tests, effect size measurement, and visualizations will ensure that your results are both significant and actionable.

As you become more familiar with A/B testing, you can explore even more advanced topics such as sequential testing (to minimize sample size), Bayesian A/B testing, or multivariate testing for deeper insights.

Happy testing!

Python

Data Science

Machine Learning

Deep Learning

NLP



Written by Ritesh Gupta

3.4K Followers

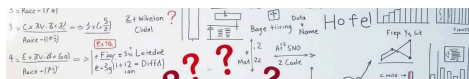
Data Scientist, I write Article on Machine Learning| Deep Learning| NLP | Open CV | AI

Lover ❤️

Follow



More from Ritesh Gupta



 Ritesh Gupta

Can You Handle These 25 Toughest Data Science Interview Questions?

The role of a Data Scientist demands a unique blend of skills, including statistics, machine...

★ Sep 25



 Ritesh Gupta

10 Automated EDA Tools That Will Save You Hours Of Work

Exploratory Data Analysis (EDA) is the process of analyzing and summarizing the...

★ Jan 29, 2023  6



 Ritesh Gupta

14 Life Changing Lessons From Chanakya Niti everyone should...

Who is Chanakya?

★ Jan 30, 2023  1



 Ritesh Gupta

Meta's Llama 3.2: A Game-Changer in Generative AI

Generative AI continues to advance, and one of the most significant updates is the launch...

★ Sep 26



See all from Ritesh Gupta

Recommended from Medium



Nivedita Bhadra

A Comprehensive Guide to A/B Testing in Python

Implementing A/B Testing in an Email Campaign Using Python

★ Aug 5 1



Actual Values			
	Setosa	Versicolor	Virginica
Setosa	16 (cell 1)	0 (cell 2)	0 (cell 3)
Versicolor	0 (cell 4)	17 (cell 5)	1 (cell 6)
Virginica	0 (cell 7)	0 (cell 8)	11 (cell 9)

Abhishek Jain

Confusion matrix for multiclass classification

How to Calculate FN, FP, TN, and TP Values?

Sep 19

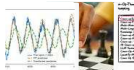


Lists



Predictive Modeling w/ Python

20 stories · 1576 saves



Natural Language Processing

1741 stories · 1322 saves



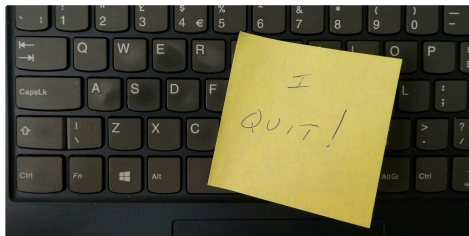
Practical Guides to Machine Learning

10 stories · 1915 saves



Coding & Development

11 stories · 834 saves



Imad Adrees in The Pythoneers

3 Reasons Most Data Scientists Resign after 1.7 Years

The reality of the 'sexiest' job in the world.

★ Sep 12 11



Abdur Rahman in Python in Plain English

7 Uncommon but Extremely Useful Python Libraries for 2024

I wish I knew it earlier...

★ 5d ago




 Gustavo Santos  in Towards Data Science

Make Your Way from Pandas to PySpark

Learn a few basic commands to start transitioning from Pandas to PySpark

★ Sep 26  4



 Joseph Robinson, Ph.D. in Towards AI

5 Essential Machine Learning Techniques to Master Your Data...

A Comprehensive Data Science Guide to Preprocessing for Success: From Missing...

★ Sep 25  4



See more recommendations