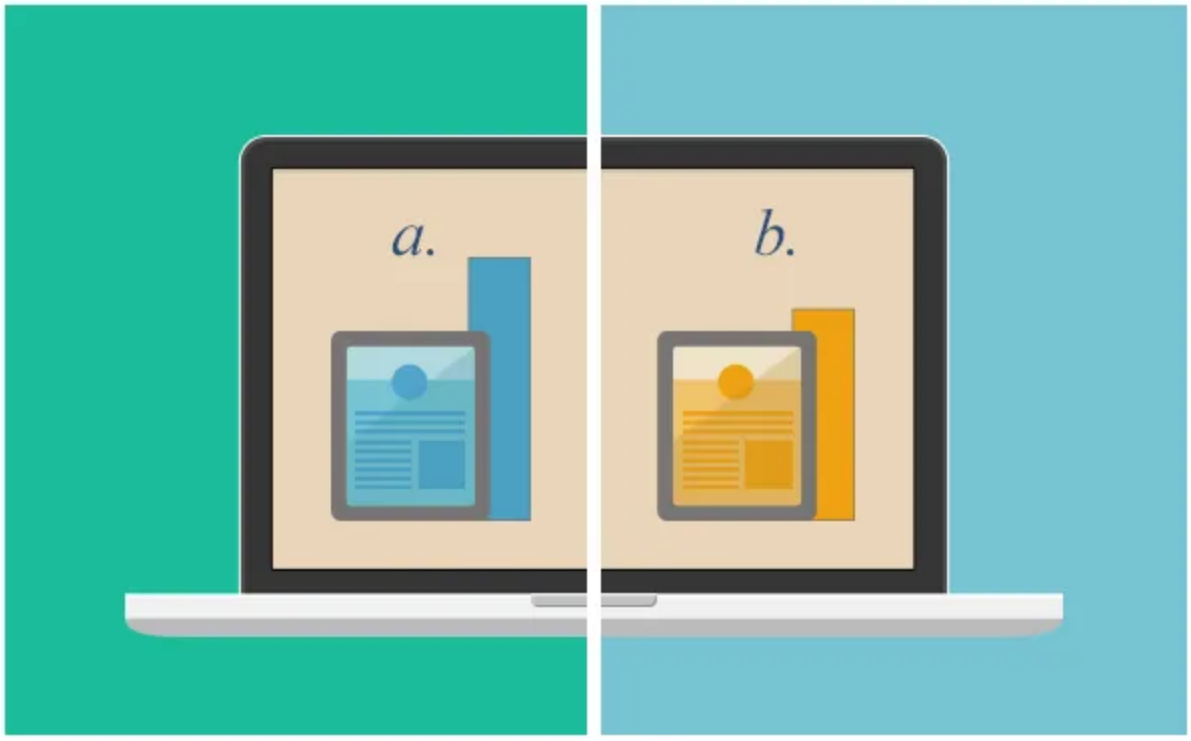
Learn How to Perform A/B Tests in Python

Practical Guide with Code and Examples to Optimize the Efficiency of Elements in Any Area.

🐼 panData · Follow
6 min read · Apr 6, 2023



A/B tests are an experimentation technique widely used in the world of digital marketing and data analysis to determine which version of an element (such as a landing page or an ad) is most effective in achieving a specific goal, such as increasing conversions or website dwell time.

What is an A/B Test?

Before we dive into the details of how to perform A/B tests in Python, it's important to understand what an A/B test is and how it works.

In an A/B test, you create two versions of an element and randomly show them to users. You then monitor user interactions with each version and compare the results to determine which version is more effective.

. . .

Setting Up the Environment

Before you start working with A/B tests in Python, you'll need to set up your development environment. For this article, we'll be using the numpy library to generate random data, the scipy library to perform statistical tests, and the matplotlib library to visualize the results.

Google Colaboratory

Edit description

colab.research.google.com

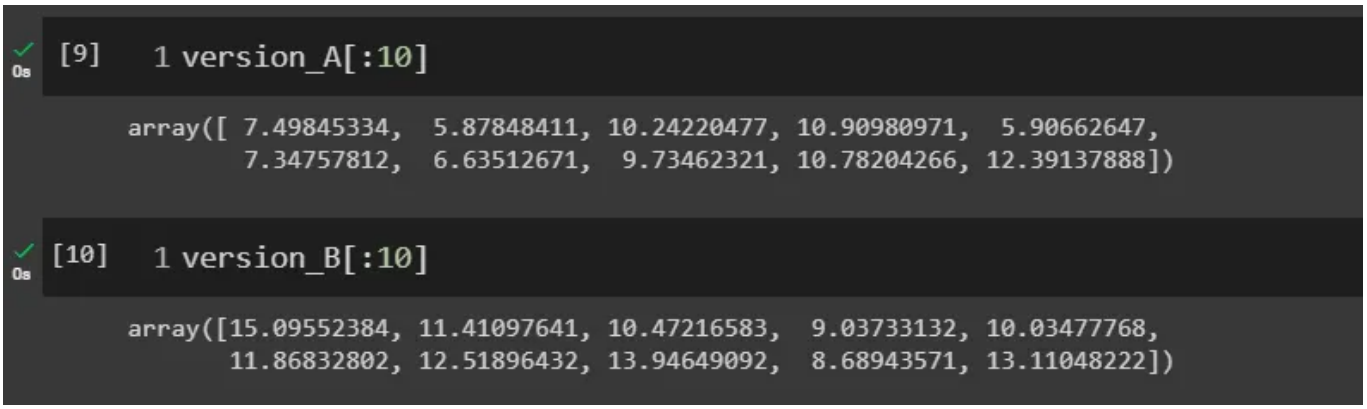
To install these libraries, open the terminal and execute the following commands:

```
pip install numpy
pip install scipy
pip install matplotlib
```

Generating Random Data To perform an A/B test, you'll need data to analyze. Let's start by generating some random data for our tests using the numpy library. The code below creates two sets of sample data, one for each version of the element we're testing:

```
import numpy as np

# Generating the data
version_A = np.random.normal(loc=10, scale=2, size=1000)
version_B = np.random.normal(loc=12, scale=2, size=1000)
```



🐼 panData

In this example, we're generating random data using a normal distribution. loc defines the mean of the distribution and scale defines the standard deviation. The parameter size defines the sample size.

. . .

Statistical Hypothesis

T-Test

The t-test is a type of statistical test that is used to test hypotheses and determine whether an observed **difference between the means of two samples is statistically significant** or occurred by chance.

In simple terms, the t-test is a tool that helps to find out whether two things are really different or whether they are only different by chance. It helps to make decisions based on statistical evidence.

P-Value

The p-value is a numerical result that is produced by a statistical test, such as the t-test. It indicates the probability of obtaining a result equal to or more extreme than the observed result, if the null hypothesis is true.

The null hypothesis is a statement that is tested in a statistical test. For example, in the t-test, the null hypothesis is that the two samples have the same mean. If the p-value ≤ 0.05, then the null hypothesis is rejected. This means that there is sufficient statistical evidence to conclude that the two samples have different means.

In summary, the p-value is a number that **helps to interpret the result of a statistical test.** It indicates whether the result is statistically significant or

whether it may have occurred by chance.

The lower the p-value, the stronger the evidence against the null hypothesis and the stronger the evidence in favor of the alternative hypothesis.

. . .

Performing A/B Test

With our generated data, we can perform the A/B test using the scipy library. The code below performs a t-test to compare the means of the two data sets:

```
from scipy.stats import ttest_ind

# Performing the t-test
t, p = ttest_ind(version_A, version_B)

# Printing the result
print(f"t = {t:.3f}")
print(f"p = {p:.3f}")
```

```
t = -22.337
p = 0.000
```

🐼 panData

The result of the t-test includes two values: **t** is the test statistic and **p** is the p-value. The p-value is a measure that highlights the significance of the difference between the two means.

A low p-value indicates that it is unlikely that the observed differences between the two versions are due to chance.

The p-value of 0.000 indicates that the **observed differences between the two versions are statistically significant and that the difference between the means is very unlikely to be due to chance.** This result suggests that there is a statistically significant difference between the two versions being tested.

The negative value of t indicates that **the mean of version A is smaller than the mean of version B.** However, the absolute value of t is very large, indicating that **there is a large difference between the means of the two samples.**

In summary, **we can conclude that version B is statistically superior to version A based on the results of the A/B test.** However, it is important to remember that the result of the A/B test is just a tool to aid decision-making and that other considerations, such as context and target audience, should also be taken into account.

. . .

Visualizing the Results

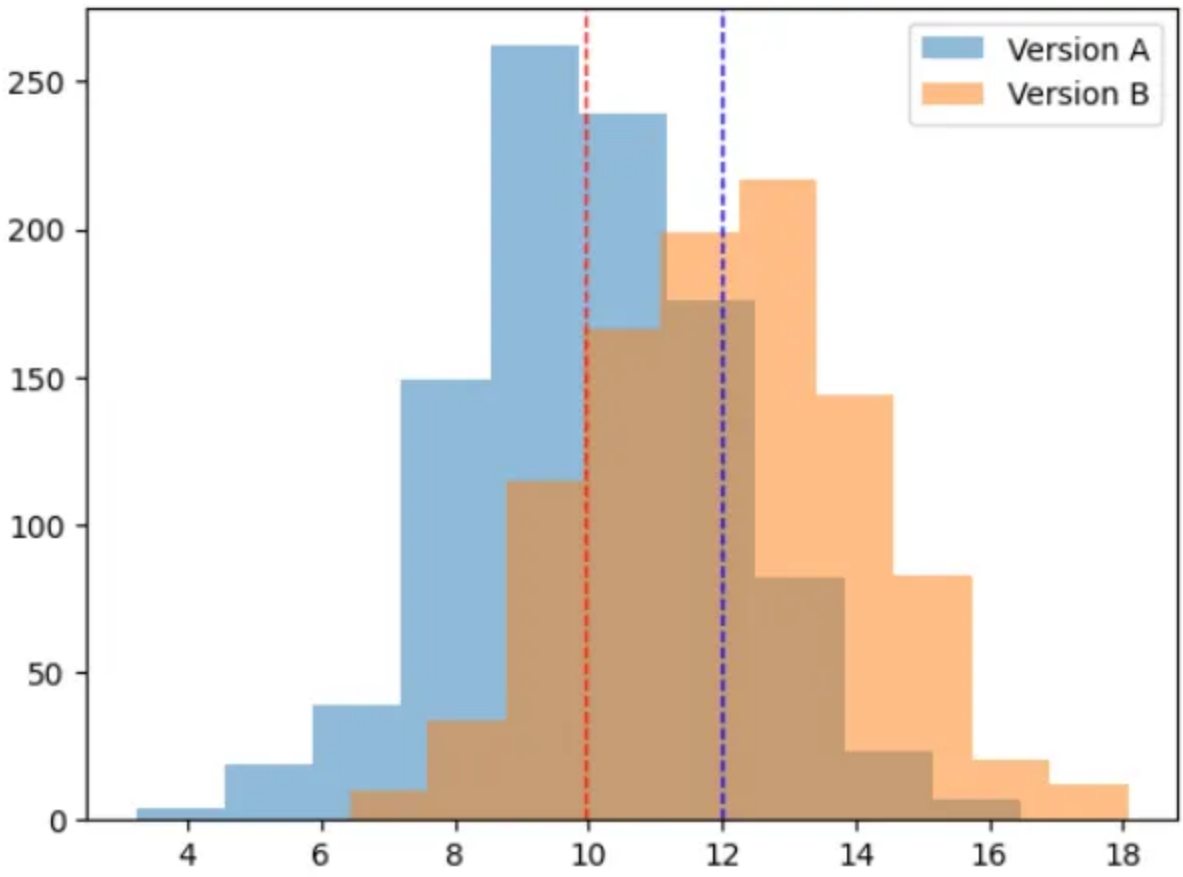
Finally, we can visualize the results of our A/B test using the matplotlib library.

```
import matplotlib.pyplot as plt

# Calculating the means
mean_A = np.mean(version_A)
mean_B = np.mean(version_B

# Plotting the data
plt.hist(version_A, alpha=0.5, label='Version A')
plt.hist(version_B, alpha=0.5, label='Version B')
```

```
plt.axvline(mean_A, color='r', linestyle='dashed', linewidth=1)
plt.axvline(mean_B, color='b', linestyle='dashed', linewidth=1)
plt.legend(loc='upper right')
plt.show()
```



🐼 panData

In this code, we calculate the means of each version using the np.mean function from Numpy and add a vertical line using the axvline method from Matplotlib for each mean. The lines are drawn in red for version A and blue for version B, with a dashed style and line width of 1.

The result is a chart that clearly shows where the means of each version lie, allowing for a more precise visualization of the differences between them.

. . .

Conclusion

A/B testing is a powerful tool for optimizing the effectiveness of elements such as landing pages, ads, and other digital marketing elements. In this article, we explored how to perform A/B testing in Python, using libraries like numpy, scipy, and matplotlib to generate random data, perform statistical tests, and visualize the results.

If you want to deepen your knowledge of A/B testing, I recommend exploring other statistical techniques, such as proportion hypothesis tests or non-parametric tests, and experimenting with different user sampling and segmentation strategies.

FAQs

What is A/B testing?

A/B testing is an experimentation technique in which you create two versions of an element and randomly show them to users to determine which version is more effective.

How do I know if the mean is statistically significant?

You can use the p-value of a statistical test to determine if your mean difference is statistically significant. A low p-value indicates that the observed differences between the two versions are unlikely to be due to chance.

How do I choose the size of my sample for an A/B test?

The sample size depends on the size of your population and the size of the effect you expect to see.

What are some best practices for conducting A/B tests?

Some best practices for conducting A/B tests include setting a clear goal for your test, testing only one variable at a time, testing the test on a control group, and collecting enough data to obtain statistically significant results.

Thank you, 🐼💔.

Statistics | Data | Python | Data Science | Marketing

Written by 🐼 panData

Follow

1.1K Followers

🐼 Derived from the Latin word "pan," which means "all" or "every," a personal repository of Data I have studied (and somewhat self-taught).

More from 🐼 panData

🐼 panData in Level Up Coding

Revolutionizing Data Analysis

Integrating Excel and ChatGPT in Advanced Exploratory Explorations

19 min read · Jan 14

38

🐼 panData

🐼 Exploratory Data Analysis (EDA): Techniques and Methods f...

Learn how to properly prepare data, select and optimize models, and evaluate their...

10 min read · Apr 8, 2023

35

🐼 panData in Python in Plain English

HOW to Create a Machine Learning Model in Python for Sales...

Step-by-Step Guide to Building a Sales Prediction Model Using Python and Scikit-...

9 min read · Mar 28, 2023

95 1

🐼 panData in Level Up Coding

Mastering Data Preprocessing in Python Pandas (with code)

Learn about the essential techniques for data preprocessing in Python using Pandas, and...

9 min read · Mar 6, 2023

242 3

See all from 🐼 panData

Recommended from Medium

Python Fundamentals in Stackademic

A/B Testing: Step-by-Step Guide with Code Examples

The Steps of A/B Testing

4 min read · Aug 27, 2023

Wenkai Bao in Expedia Group Technology

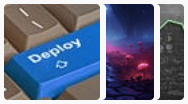
How to Size For Online Experiments With Ratio Metrics

Sizing Derivation and Evaluation

8 min read · Mar 28, 2023

170 4

Lists



Predictive Modeling w/ Python

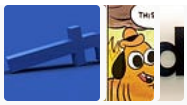
20 stories · 824 saves

Coding & Development

11 stories · 398 saves

Practical Guides to Machine Learning

10 stories · 953 saves



Modern Marketing

58 stories · 388 saves

Chandradip Banerjee

P value and Feature Selection

P value and Feature Selection

8 min read · Sep 21, 2023

4

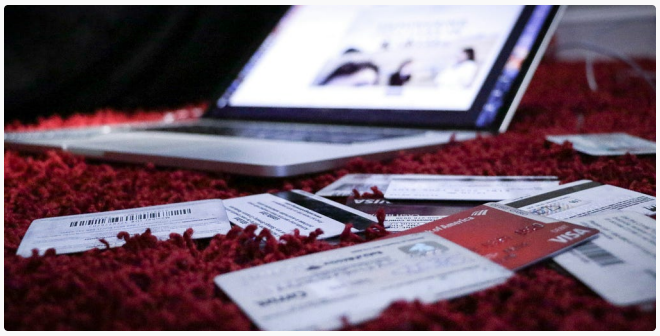
Bragadeesh Sundararajan

Performing A/B Testing in Python: A Real-World Use Case

A/B testing is a powerful method for comparing two versions of something to...

✨ · 5 min read · Sep 17, 2023

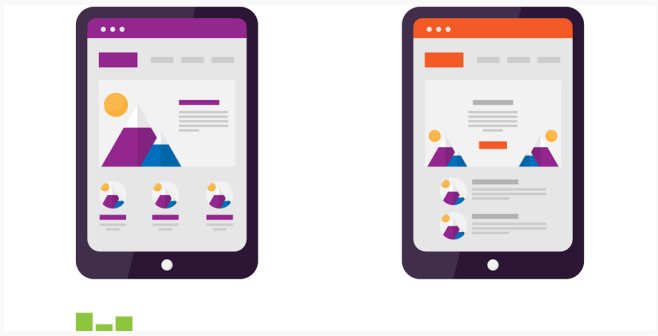
2



Muhammad Arief Rachman

Credit Scoring Prediction

Scorecard modeling



Youssef Hosni in Towards AI

Mastering A/B Testing: A Real World Business Example [Part 1]

Step-by-Step Walk-Through Real World A/B Testing Use Case

15 min read · Nov 15, 2023

11

✨ · 9 min read · Aug 15, 2023

177 1

See more recommendations