A/B Testing

Oamen Modupe
oamenmodupe@gmail.com

TABLE OF CONTENTS

1. Discover

1.1 Understand the problem and Define the hypothesis

The company wants to know if introducing new tailored ads will increase the daily average click-through rate. We hypothesize that if users are exposed to relevant ads, they are more likely to click on them and purchase the product, leading to increased CTRs and affiliate revenue.

Null Hypothesis: There is no increase in CTR after tailored ads
Alternative hypothesis: There is an increase in CTR after tailored ads

This project aims to use A/B testing to determine if the CTR metric will change and if the change is reliable. We will also create an a/b testing package with reusable functions.

A/B testing is a decision-making support and research methodology that allows you to measure the impact of a change in a product/service based on a sample of data. It answers the questions:

a. Which metrics or user behavior will change, and how much will they change?
b. Is the change reliable?

1.2 Decide if it's worth testing

There are a few reasons why an a/b test might not be the best approach, like:

a. Insufficient data: You might want to consider other alternatives like qualitative research methods, user surveys, etc
b. Too risky: It might be too risky to test on users as new features might be shaky to implement, we might have latency issues, a drop in user engagement, crashes, etc.
c. Too vague to test: unclear hypothesis, undefined metrics
d. Too many tests already running: If we have too many tests already running, we might not have enough data for the control group, or it might dilute user experience, etc.

1.3 Decide on how to design the feature and build it

This is usually done by software engineers and product managers.

1.4 Determine how to measure the success of the feature: KPI and metric(s) to track

- KPI: Increase user daily click-through rate by at least 5% over the next year.
- Success Metric: Click-Through Rate(CTR)
- Guardrail Metric: Revenue per user, Daily Active Users

The Success Metric is the main metric we are trying to track. It is based on the hypothesis and is the dependent variable.
The Guardrail Metric is used to track the baseline of other metrics. Apart from the success metric, some other metrics might be impacted by the change we are introducing.

1.5 Data Collection

The data was generated with chatGPT to simulate an actual ab test with this prompt.

```
"I want to work on an a/b testing project to practice, I want you to create a raw dataset with raw tracking events for a marketing campaign, I want to test if the introduction of new ads will lead to an increase in ctr, before placing them in a control or test group, I will do the splitting myself"
```

1.6 Determine how to run the test

a. For how long should we run the test?
b. How do we monitor the performance of the test: Here, we will create a Databricks dashboard to monitor the success and guardrail metrics
c. Agree on the Minimum Detectable Effect(MDE): MDE is the smallest change or difference between two groups in an A/B test that you care about detecting. It helps you determine how big of an effect you must observe to reach statistical significance and not just change due to random chance. For example, since we want to increase our CTR by at least 5%, our MDE will be 5%.
d. Agree on the alpha and beta:
   Alpha(α) is the probability of making a Type I error. It represents the significance level of the test. In Simple Terms, It's the risk you're willing to take of incorrectly rejecting the null hypothesis when it's true. For example, if you set alpha to 0.05, you're accepting a 5% chance of making this mistake.
   Beta (β): Beta is the probability of making a Type II error. It represents the likelihood of failing to reject the null hypothesis when the alternative hypothesis is true.
   In Simple Terms, It's the risk of not detecting a difference or effect when there is one. For example, if the beta is 0.20, there's a 20% chance you won't detect a true effect.
   Power: Power is the probability of correctly rejecting the null hypothesis when the alternative hypothesis is true. It's calculated as $1 - \beta$
   In Simple Terms. It's how likely you are to detect a real effect if it exists. For example, if the power is 0.80, you have an 80% chance of finding an effect when there is one.

2. Play

2.1 Git Repo

- Create a Repo in GitHub and open the project folder on the local machine
- Run git init in the terminal
- Create a master branch in GitHub and set it to default(only if the default branch in local and GitHub are different)
- Delete the main branch in GitHub
- Run these commands in Terminal: git remote add origin <git url>, git pull origin master
- Create dev branch: git branch dev => git checkout dev

2.2 Create Package

A package will be created and installed into Jupyter and VSCode for use. It contains reusable functions for tasks like visualization, preprocessing, splitting, testing, etc.
Process:

- Create the folder (custom_package)
- Create a sub-folder(package)
- Create a setup.py file within custom_package and input values
- Create modules(py files) and functions for the package
- Run these commands: cd package, pip install build, python -m build, cd dist, pip install <package-file.whl> , cd ../..

2.3 Collect Data, aggregate, and explore(1_Exploration_And_Aggregation.ipynb)

The data was simulated with ChatGPT. Here's a sample structure for the dataset:

- user_id: Unique identifier for each user.
- timestamp: Time of the event.
- ad_id: Identifier for the ad.
- event_type: Type of event (either "impression" or "click").

Here, we explore the data, check for nulls, duplicates, etc and take care of any problems.
Next, we perform low-level-aggregations and transformations:

a. Average daily CTR and revenue per user
b. Remove any users with a CTR of 1
c. Average daily users

2.4 Calculate the sample size

2.4.1 Determine whether we need a 1-tailed or 2-tailed test based on the hypothesis

We hypothesized that showing users more tailored ads would lead to an increase in CTR. We want to know if the mean of the CTR in the test group is larger than that of the control group, hence, a 1-tailed test. If we wanted to know if the mean was different(bigger/smaller), it would be a 2-tailed test.
One-tailed tests, also known as one-sided tests, and two-tailed tests, sometimes known as two-sided tests, are the two types of hypothesis tests. One side of a statistic, such as "the mean is greater than 10" or "the mean is less than 10," is the focus of one-tailed tests. The z-score is on one side of the statistic, and one-tailed tests only address one tail of the distribution.

The z-score is on both sides of the statistic, and two-tailed tests deal with both tails of the distribution. A two-tailed test is required for a hypothesis such as "the mean is not equal to 10," since it asserts that the mean may be either less than or larger than 10.

2.4.2 Calculate the critical values

We will do this with the Scipy library.

2.4.3 Determine the metric type

Click-Through Rate (CTR) is a continuous metric representing a proportion or rate that can take any value within a given range. In your case, CTR values range from 0 to 1, and since it can take on any value within this range, it is considered a continuous variable.

Why CTR is Continuous:

- Range: CTR values, including any fractional values, can vary continuously between 0 and 1.
- Nature of Data: It represents the proportion of clicks out of the total impressions, a continuous measurement rather than a discrete count.

In contrast, a binomial metric typically refers to count data, where each observation falls into one of two categories (success or failure). While CTR is derived from binomial data (click/no-click), the resulting metric itself is continuous.

2.4.4 Calculate the sample size

We start by splitting the low-level aggregated data into pretest(before the test_date = '2024-06-24') and after the test. Next, we separate the daily active users to avoid wrong data from aggregating.
Next, we create functions to calculate the sample size:

- _binomial_sample_size_calc: To calculate the sample size for the binomial metric
- _continuous_sample_size_calc: To calculate the sample size for the continuous metric
- Sample_size_calc: to collect info from the user

Using the continuous metric, we get a sample size of 898, hence our control group sample size is 898, and the test sample size is 898 + MDE.
We check for pretest bias as it can affect the results of the test, users in two experiment groups can have meaningfully different average behaviors before your experiment applies any intervention to them

2.5 Run the test

At this point, we start the test, i.e. expose the test group to the new feature and the control group to nothing.

2.6 Set up Monitoring

We need to set up monitoring to measure the success and guardrail metrics as well as other metrics like latency, product performance, etc. This is simply to make sure there is no negative impact on the company's baseline, drop in engagement, errors, etc.
This will be done by uploading the dataset to Databricks and creating a dashboard to monitor metrics.

2.8 Collect Data and Test

After the ab test period, we then collect the data for the test and control groups and run the statistical tests. We check for normality and homogeneity in the samples.
If the samples are normal, test for homogeneity with the Levene test.

- If the samples are normal and have homogeneity, test with a T-test
- If the samples are normal and have no homogeneity, use Welch's test
- If the samples are not normal, use the Mann-Whitney test directly.

A t-test is a statistical test that compares the means of two samples. It is used in hypothesis testing, with a null hypothesis that the difference in group means is zero and an alternate hypothesis that the difference in group means is different from zero.

Welch's t-test, or unequal variances t-test, is a two-sample location test that is used to test the (null) hypothesis that two populations have equal means. It is named for its creator, Bernard Lewis Welch, and is an adaptation of Student's t-test, and is more reliable when the two samples have unequal variances and possibly unequal sample sizes.

The Mann-Whitney U test is used to compare differences between two independent groups when the dependent variable is either ordinal or continuous, but not normally distributed

The testing is done with the test.py module in the custom ab_testing_kit package.
This script performs statistical tests for comparing two groups in an A/B test scenario. It evaluates the normality of the data, checks for equal variances, and performs appropriate statistical tests to determine if there is a significant difference between the test and control groups.

Functions
1. _normality_test
Description: Conducts the Shapiro-Wilk test for normality on two datasets (test and control). The Shapiro-Wilk test assesses whether the data is normally distributed.
Parameters:

- test (pd.DataFrame): DataFrame containing the test/experimental group data.
- control (pd.DataFrame): DataFrame containing the control group data.
- column (str): The column name in the DataFrame that holds the metric to be tested.
- alpha (float): Significance level for the test (e.g., 0.05).

Returns:

- bool: True if both datasets are normally distributed (p-value >= alpha); False otherwise.

2. _equal_variance_test
Description: Performs Levene's test to assess the equality of variances between the test and control groups. Levene's test checks if the variance is equal among the two groups.
Parameters:

- test (pd.DataFrame): DataFrame containing the test/experimental group data.

- control (pd.DataFrame): DataFrame containing the control group data.
- column (str): The column name in the DataFrame that holds the metric to be tested.
- alpha (float): Significance level for the test (e.g., 0.05).
- center (str): Method for measuring the center of the distribution ('mean', 'median', 'trimmed').

Returns:

- bool: True if variances are equal (p-value >= alpha); False otherwise.

3. ab_test

Description: Performs an A/B test to compare the means of two groups using different statistical tests based on the properties of the data:

- Student's T-test: Used if both samples are normal and have equal variance.
- Welch's T-test: Used if both samples are normal but have unequal variance.
- Mann-Whitney U test: Used if neither sample is normal.

Parameters:

- test (pd.DataFrame): DataFrame containing the test/experimental group data.
- control (pd.DataFrame): DataFrame containing the control group data.
- column (str): The column name in the DataFrame that holds the metric to be tested.
- alpha (float): Significance level for the tests (e.g., 0.05).
- center (str): Method for measuring the center of the distribution ('mean', 'median', 'trimmed').

Returns:

dict: Contains the following keys:

- statistic: Test statistic from the selected statistical test.
- p-value: p-value from the selected statistical test

3 Deploy

3.1 Github Workflow

Workflow Name: Pylint

Trigger: The workflow is triggered on every push to the repository.

Steps

1. Checkout Code

```
- uses: actions/checkout@v3
```

Description: Checks out the code from the repository so that subsequent steps can access it.

Purpose: To ensure that the latest code is available for analysis.

2. Set Up Python

```
name: Set up Python ${{ matrix.python-version }}
 uses: actions/setup-python@v3
 with:
   python-version: ${{ matrix.python-version }}
```

Description: Set up the specified Python version in the workflow environment.
Purpose: Ensures that the correct Python version (in this case, Python 3.10) is used for the linting and conversion processes.

3. Install Dependencies

```
- name: Install dependencies
  run: |
    python -m pip install --upgrade pip
    pip install pylint
    pip install jupytext
```

Description: Installs the necessary Python packages:
pip: Python package installer.
pylint: A code analysis tool for checking coding standards.
jupytext: Tool to convert Jupyter notebooks to scripts and vice versa.
Purpose: To prepare the environment with the tools required for linting and file conversion.

4. Convert .ipynb Files to .py

```
- name: Convert .pynb files to .py
  run: |
    jupytext --to py $(git ls-files '*.ipynb')
```

Description: Converts Jupyter notebook files (.ipynb) in the repository to Python script files (.py).
Purpose: Ensures that code in Jupyter Notebooks is also analyzed by Pylint. This is useful for maintaining code quality across different file formats.

5. Run Pylint

```
- name: Run pylint
  run: |
    pylint $(git ls-files '*.py') --fail-under=8.0 || echo "Pylint failed
with score below threshold."
```

Description: Runs pylint on all Python files (.py) in the repository.
Parameters:
--fail-under=8.0: Fails the job if the pylint score is below 8.0.
Purpose: To check code quality and adherence to coding standards. If pylint reports a score below the threshold, it prints a message indicating failure but does not stop the workflow.

3.2 PyPi Upload

This guide provides step-by-step instructions for preparing and uploading a Python package to the Python Package Index (PyPI). PyPI is a repository for Python software, where you can distribute your packages to other users.

Prerequisites

- Python Installed: Ensure you have Python installed on your system.
- PyPI Account: Create an account on PyPI.

- API Token: Generate an API token from your PyPI account for authentication.

1. Build the Distribution:

```
python -m build
```

This command creates distribution files in the dist/ directory.

2. Install twine: twine is a tool for securely uploading your package to PyPI.

```
pip install twine
```

3. Upload Your Package to PyPI

```
twine upload dist/*
```

When prompted, enter your PyPI username and API token.

4. Verify the Upload: Check that your package is available on PyPI, Go to PyPI and search for your package.

Resources

1. https://stackoverflow.com/questions/33862420/ipython-notebook-how-to-reload-all-modules-in-a-specific-python-file
2. https://www.sciencedirect.com/topics/mathematics/tailed-test#:~:text=One%20tailed%2Dtests%20are%20concerned,one%20side%20of%20the%20statistic.
3. https://statistics.laerd.com/spss-tutorials/mann-whitney-u-test-using-spss-statistics.php
4. https://en.wikipedia.org/wiki/Welch%27s_t-test
5. https://pypi.org/project/ab-testing-kit/#description
6. https://github.com/actions/starter-workflows/issues/2303
7. https://www.youtube.com/watch?v=Kz6IlDCyOUY
8. https://packaging.python.org/en/latest/guides/making-a-pypi-friendly-readme/
9.