A/B testing

Created By	

What is A/B testing?

A/B testing is a popular statistical tool used by a data scientists. It is basically a randomized controlled experiment. It is used to find out which version is performing well in two versions. For e.g. an online newspaper launches two headlines for the same news. They might be interested to find out the headline which is attracting more users to read the entire article.

Another example could be: let's say we have two versions of the same webpage, where one version is updated and the other remains unchanged. Then by statistically testing the response of users of both the versions, the decision of keeping a certain version could be made. There could be many other examples.

Steps to perform A/B testing.

1. Generate a null hypothesis and an alternate hypothesis.

Null hypothesis: A null hypothesis states that a particular result is an output by a chance. In the context of A/B testing, a null hypothesis says that there will be no significant change in the output if we change the input. For. e.g. in the case of headlines, the null hypothesis says that: there will be no change in the number of users reading the full article if we change the headlines of the news. In general, we want to reject our null hypothesis.

Alternate hypothesis: An alternated hypothesis will suggest that there will be a significant change in the number of users reading the full article if we made some changes to the headlines. In general, we want the alternate hypothesis to get accepted.

Our goal is to collect enough arguments to reject the null hypothesis.

2. Create group A and Group B/ or control group and test group.

   The control group or group A is the group of users who get original headlines, and the test group or group B are the group of people getting updated headline.

   Let's say we have 1000 users, we randomly assign 500 users in group A and 500 users to group B.

   we need to take care of two things: we must have enough samples/users so that w can make an informed decision, and second, there must be no sample bias i.e. users must be assigned randomly to group A and group B.

3. Get the baseline conversion rate and desired lift rate:

   Baseline conversion rate: it is the percentage of users reading the article with original headlines. For example, if 10 out of 100 users are reading the complete article, then the conversion rate is 0.10.

   Desired lift rate: as its name suggest, it is the desired change we want to see after implementing changes in the headlines. For example, if we want a 2% lift rate, that means at least 12 users must read the complete article after changing the headline.

4. Understand hypothesis testing errors

   There are 2 types of errors in hypothesis testing: Type I error and Type II error

   Type I error: When we reject the null hypothesis when it is true. we accept version B even when it is not performing better than version A.

Type II error: It happens when we fail to reject the null hypothesis when it is false. That means, we accept version A, even when it is not performing better than version B.

5. Perform two-sample T-test:

   Two-sample T-test is the most popular hypothesis test. Before understanding the test we must know the following terms:

   - Significance level (alpha): Normally it is 1% or 5%. It is the probability of rejecting a null hypothesis when it is true.
   - p-value: It is the probability that the result is just because of a random chance. Normally smaller the p-value, the stronger the chances to reject the null hypothesis. Normally, we take the p-value as 0.05.
   - Confidence interval: The confidence interval is an observed range in which a given percentage of test outcomes fall. The confidence interval is 1 - p-value, i.e. if p-value is 5%, then confidence interval is 95%.

6. Calculate T-statistics:

   Perform t-statistic by using the formula:

$$T - statistic = \frac{Observed\ value - hypothesized\ value}{Standard\ Error}$$

$$Stamdard\ Error = \sqrt{\frac{2 * Variance(sample)}{N}}$$

Python implementation of A/B testing

Let's implement an A/B test on a data set where we want to check the impact of changes made on the website page on the conversion rate of the users (we want users to get a premium account). The old version of the webpage is named “old_page” in the data, and the new version is named “new_page”. There are two groups control and treatment, the control group has given the old page, and the treatment group has given the new_page. The conversion is either 0 (not converted into a premium account) or 1 (users converted onto a premium account). We want at least an uplift rate of 2% showing that the new_page is performing better than the old_page. So the hypothesis in this case is:

Null hypothesis: There is no change in conversion rate when new_page is shown to the users.

Alternate hypothesis: There is a significant (at least 2%) improvement in the conversion rate when new_page is shown to the users.

Ideally, we want to reject the null hypothesis.

The python implementation is quite simple. The steps are. listed below:

1. Import libraries

```
import numpy as np
import pandas as pd
import seaborn as sns
import scipy.stats as ss
```

2. Load the data. The csv file “ab_data.csv” is available at Kaggle for free download.

```
ab_data = pd.read_csv('ab_data.csv')
print(ab_data[:10]) # print first 10 rows of the data
```

The output of this step looks like this:

	user_id	timestamp	group	landing_page	converted
0	851104	2017-01-21 22:11:48.556739	control	old_page	0
1	804228	2017-01-12 08:01:45.159739	control	old_page	0
2	661590	2017-01-11 16:55:06.154213	treatment	new_page	0
3	853541	2017-01-08 18:28:03.143765	treatment	new_page	0
4	864975	2017-01-21 01:52:26.210827	control	old_page	1
5	936923	2017-01-10 15:20:49.083499	control	old_page	0
6	679687	2017-01-19 03:26:46.940749	treatment	new_page	1
7	719014	2017-01-17 01:48:29.539573	control	old_page	0
8	817355	2017-01-04 17:58:08.979471	treatment	new_page	1
9	839785	2017-01-15 18:11:06.610965	treatment	new_page	1

In this data, we have 5 columns: user_id, timestamp, group (control or treatment), landing_page (old page or new page), and converted (0: not converted, 1: converted)

3. You may check the information of the data by using the command:

```
ab_data.info()
```

The output is:

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 294478 entries, 0 to 294477
Data columns (total 5 columns):
 #   Column        Non-Null Count   Dtype
---  ------        --------------   -----
 0   user_id       294478 non-null  int64
 1   timestamp     294478 non-null  object
 2   group         294478 non-null  object
 3   landing_page  294478 non-null  object
 4   converted     294478 non-null  int64
dtypes: int64(2), object(3)
memory usage: 11.2+ MB
```

4. We need to make sure that the control group is seeing the old page and the treatment group sees the new page of the website.

```
print(pd.crosstab(ab_data['group'],ab_data['landing_page']))
```

The output is:

```
landing_page  new_page  old_page
group
control           1928    145274
treatment       145311      1965
```

5. There might be multiple entries in the data, we should remove these multiple entries as it could skew the data, and as a result statistic analysis will be biased.

```
session_counts = ab_data['user_id'].value_counts(ascending=False)
multi_users = session_counts[session_counts>1].count()

print(f"There are {multi_users} users appearing multiple times in the data")
```

The output is:

```
There are 3894 users appearing multiple times in the data
```

6. As we have 3894 multiple entries, we need to remove these rows from the ab_data. It is done as:

```
users_to_drop = session_counts[session_counts > 1].index
ab_data = ab_data[~ab_data['user_id'].isin(users_to_drop)]
print(f'The updated dataset now has {ab_data.shape[0]} entries')
```

The output of this is:

```
The updated dataset now has 286690 entries
```

7. Now, we will do random sampling on the ab_data by using the pandas “data frame. sample()” method. We need 4720 samples in each group, and we have set random_state =22. The python code is shown below:

```
required_n = 4720
control_sample = ab_data[ab_data['group'] == 'control'].sample(n=required_n, random_state=22)
treatment_sample = ab_data[ab_data['group'] == 'treatment'].sample(n=required_n, random_state=22)

ab_test = pd.concat([control_sample, treatment_sample], axis=0)
ab_test.reset_index(drop=True, inplace=True)
ab_test
```

The output of this cell is:

```
      user_id                   timestamp      group landing_page  converted
0      763854  2017-01-21 03:43:17.188315    control     old_page          0
1      690555  2017-01-18 06:38:13.079449    control     old_page          0
2      861520  2017-01-06 21:13:40.044766    control     old_page          0
3      630778  2017-01-05 16:42:36.995204    control     old_page          0
4      656634  2017-01-04 15:31:21.676130    control     old_page          0
...       ...                         ...        ...          ...        ...
9435   908512  2017-01-14 22:02:29.922674  treatment     new_page          0
9436   873211  2017-01-05 00:57:16.167151  treatment     new_page          0
9437   631276  2017-01-20 18:56:58.167809  treatment     new_page          0
9438   662301  2017-01-03 08:10:57.768806  treatment     new_page          0
9439   944623  2017-01-19 10:56:01.648653  treatment     new_page          1

[9440 rows x 5 columns]
```

8. Now we should check the current conversion rate for the control group and treatment group.

```
conversion_rates = ab_test.groupby('group')['converted']

std_p = lambda x: np.std(x, ddof=0)              # Std. deviation of the proportion
se_p = lambda x: ss.sem(x, ddof=0)            # Std. error of the proportion (std / sqrt(n))

conversion_rates = conversion_rates.agg([np.mean, std_p, se_p])
conversion_rates.columns = ['conversion_rate', 'std_deviation', 'std_error']


conversion_rates.style.format('{:.3f}')
```

The output of this code is:

```
           conversion_rate  std_deviation  std_error
group
control           0.123305       0.328787   0.004786
treatment         0.125636       0.331438   0.004824
```

The output shows that the conversion rate for the control group is 12.3% and for the treatment group is 12.5%. which is not increased by 2% as we expected. But, to be more sure, we will conduct a t-test to find the p-value.

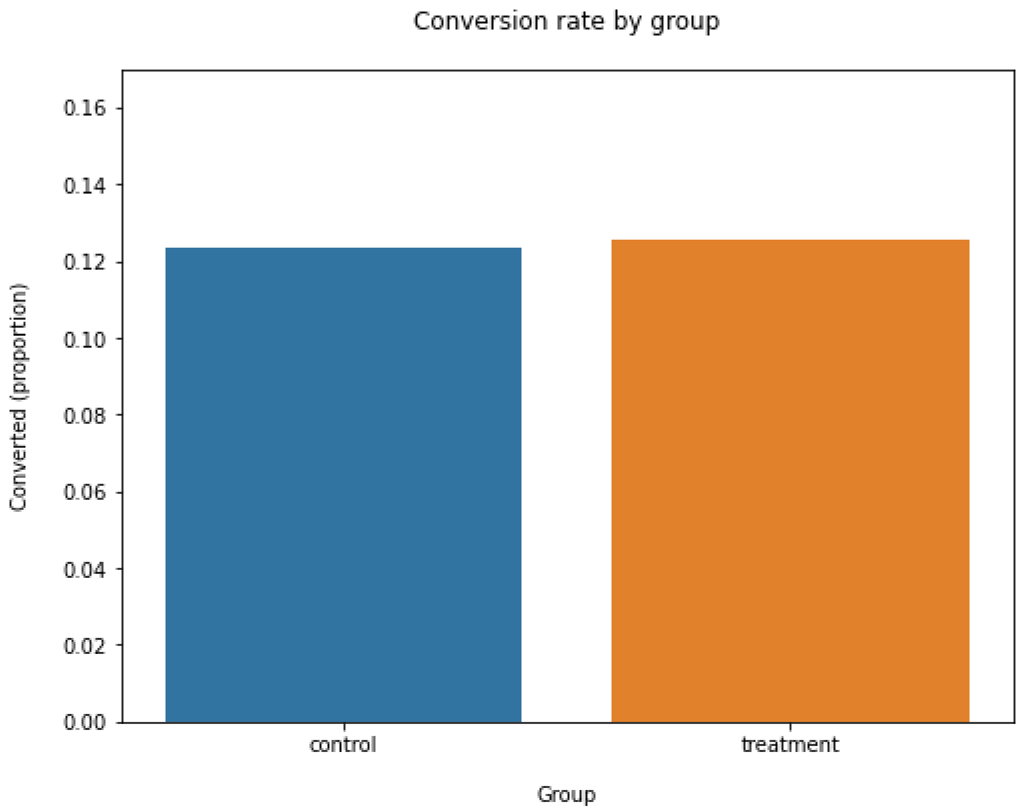
9. We can visualize the results: conversion in the control group and conversion in the treatment group to see the effect of new_page on conversion rate. This is an optional step.

```
import matplotlib.pyplot as plt
plt.figure(figsize=(8,6))

sns.barplot(x=ab_test['group'], y=ab_test['converted'], ci=False)

plt.ylim(0, 0.17)
plt.title('Conversion rate by group', pad=20)
plt.xlabel('Group', labelpad=15)
plt.ylabel('Converted (proportion)', labelpad=15);
```

The output plot is:



10. The final step in this A/B test is to perform the z test to get p-value and conversion intervals. The proportions_ztest and proportions_confint are imported.

```
from statsmodels.stats.proportion import proportions_ztest, proportion_confint
control_results = ab_test[ab_test['group'] == 'control']['converted']
treatment_results = ab_test[ab_test['group'] == 'treatment']['converted']
n_con = control_results.count()
n_treat = treatment_results.count()
successes = [control_results.sum(), treatment_results.sum()]
nobs = [n_con, n_treat]

z_stat, pval = proportions_ztest(successes, nobs=nobs)
(lower_con, lower_treat), (upper_con, upper_treat) = proportion_confint(successes, nobs=nobs, alpha=0.05)

print(f'z statistic: {z_stat:.2f}')
print(f'p-value: {pval:.3f}')
print(f'ci 95% for control group: [{lower_con:.3f}, {upper_con:.3f}]')
print(f'ci 95% for treatment group: [{lower_treat:.3f}, {upper_treat:.3f}]')
```

The output of this cell is:

```
z statistic: -0.34
p-value: 0.732
ci 95% for control group: [0.114, 0.133]
ci 95% for treatment group: [0.116, 0.135]
```

The output shows that the p-value is 0.732 which is way higher than 0.05, hence we can’t reject our null hypothesis. This means that the new webpage is not converting more users into premium account users.

In this A/B testing model, the original model A performs better than the new model B.