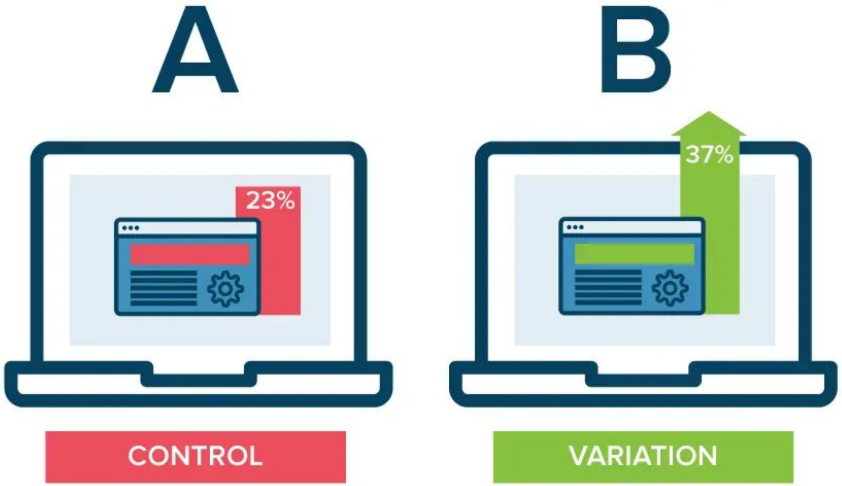


Landing page A/B testing in python

By
Ahmad Faishal Akbar

Background & Objective

An e-commerce company has **developed** a **new landing page** in order to **increase conversion rate by 3%** (from 12% to 15%), meaning the percentage number of users who decide to pay for the company's product. The goal of this project is to help the company understand if they should implement this new page or keep the old page by **running an A/B Testing based on conversion rate** of both pages.

About The Data

	user_id	timestamp	group	landing_page	converted
0	851104	2017-01-21 22:11:48.556739	control	old_page	False
1	804228	2017-01-12 08:01:45.159739	control	old_page	False
2	661590	2017-01-11 16:55:06.154213	treatment	new_page	False
3	853541	2017-01-08 18:28:03.143765	treatment	new_page	False
4	864975	2017-01-21 01:52:26.210827	control	old_page	True

This is a **user conversion data** who saw the new or old landing page occurring between **02/01/2017** and **24/01/2017**. We got this data from Kaggle.com, if you'd like to see the data, please check in the link below:

https://www.kaggle.com/datasets/zhangluyuan/ab-testing/data

About The Data

This dataset contains 5 variables as follows:

- **user_id**: User unique identifier (ID). Nominal, a 6-digit integral number uniquely assigned to each user.
- **timestamp**: Date and time when a user saw the landing page.
- **group**: Group name. Nominal, ‘control’ (user who saw old landing page) or ‘treatment’ (user who saw new landing page).
- **landing_page**: Kind of landing page. Nominal, ‘old_page’ or ‘new_page’.
- **converted**: Boolean, if it’s true, the user paid for the company’s product, if it’s false, the user didn’t do it.

Methodology

- Data Cleaning
- Experimental Design
- Exploratory Data Analysis (EDA)
- Sanity Check
- Hypothesis Testing
- Conclusion

Data cleaning

In this step, We performed various Data Cleaning processes such as **cleaning missing values and duplicate data** (if any), and **validating user assignment**. If you'd like to review these processes and the entire processes, please check my notebook here:

https://colab.research.google.com/drive/1W1ZWQ1ZTwemxe9E7GqLNXPURytqAjl9S?usp=sharing

Methodology

- Data Cleaning
- Experimental Design
- Exploratory Data Analysis (EDA)
- Sanity Check
- Hypothesis Testing
- Conclusion

Experimental Design: group assignment and hypothesis

- **Control** group: the users who saw the **old** landing page.
- **Treatment** group: the users who saw the **new** landing page.
- **H0**: There is **no** statistically significant **difference** between **conversion rate** of **Control** and **Treatment** Groups.
- **HA**: There is statistically significant **difference** between **conversion rate** of **Control** and **Treatment** Groups.

Experimental Design: Minimum sample size

```
#Calculating the baseline of conversion rate from control group
conv_rate_control = df_cleaned2[df_cleaned2['group']=='control']['converted'].mean()
conv_rate_control
```

```
0.12030492030492031
```

The goal of creating new design of landing page is to increase conversion rate 3% (from 12% to 15%)

```
#Specifying the minimum target of conversion rate
conv_rate_new = 0.15
```

```
#Calculating standardized effect size
from statsmodels.stats.proportion import proportion_effectsize
std_effect_size = proportion_effectsize(conv_rate_new, conv_rate_control)
std_effect_size
```

```
0.08697780511967357
```

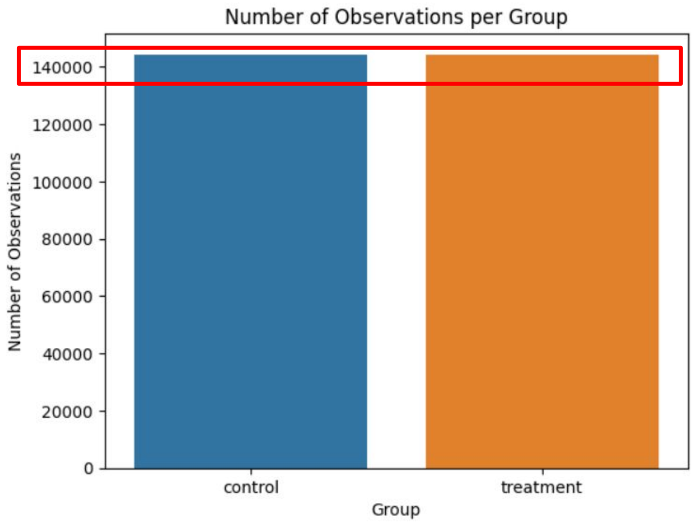
```
#Import power module
from statsmodels.stats import power

#Estimating sample size
sample_size = power.TTestIndPower().solve_power(effect_size=std_effect_size,
                                                power=0.80,
                                                alpha=0.05,
                                                nobs1=None)
sample_size
```

```
2075.968558513573
```

We need minimum sample size 2076 for each group.

Experimental Design: Minimum sample size



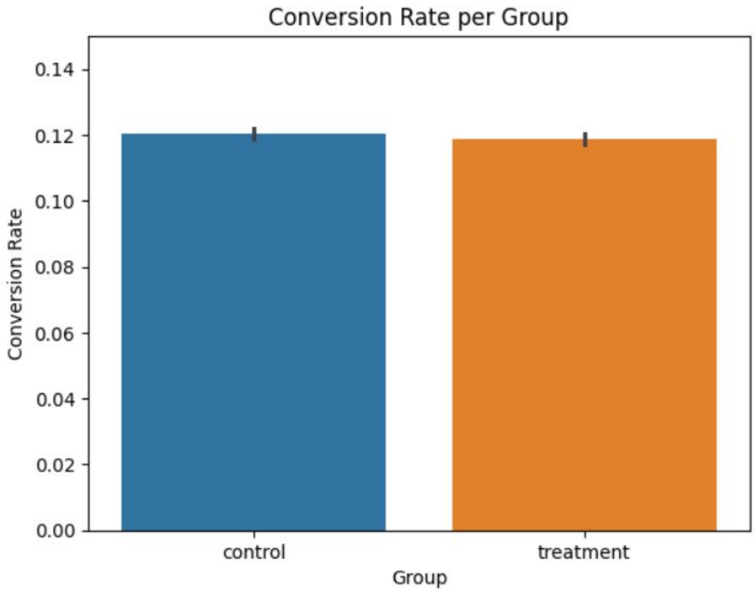
Based on the goal of creating new design landing page to increase conversion rate from 12% to 15%, We can calculate the minimum sample size using **power analysis** for each group (check on my notebook). We **need at least 2076 samples**. We **have around 140000 samples** for each group. So, the sample we have for each group has **met the minimum sample size.**

Methodology

- Data Cleaning
- Experimental Design
- Exploratory Data Analysis (EDA)
- Sanity Check
- Hypothesis Testing
- Conclusion

Exploratory Data Analysis

Group	Conversion rate
Control	0.120305
Treatment	0.118831



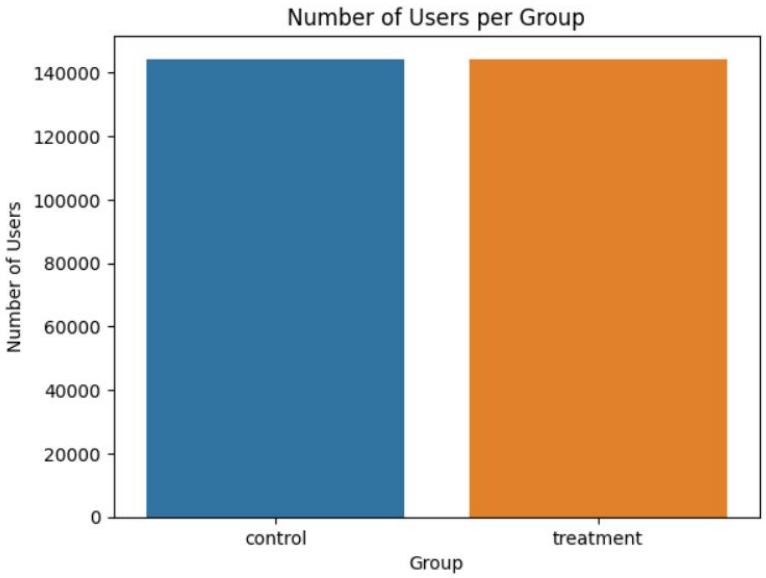
Control group has **slightly higher** conversion rate than treatment group. Is it **statistically significant?** We'll examine it in **hypothesis testing** step.

Methodology

- Data Cleaning
- Experimental Design
- Exploratory Data Analysis (EDA)
- Sanity Check
- Hypothesis Testing
- Conclusion

Sanity Check: Sample Ratio Mismatch (SRM)

Group	Number of Users
Control	144300
Treatment	144389



Treatment group has **slightly higher** number of users than control group. Is it **statistically significant?** We need to examine this using **Chi-square goodness-of-fit test**. The **Null hypothesis** states that the **allocated traffic matches** the **experiment design** which is **50/50** in this case. If the **p-value** is **lower than** a strict significance threshold of **5%**, then we **reject the Null hypothesis**.

Sanity Check: Sample Ratio Mismatch (SRM)

```
# Assign the unique counts to each variant
control_users = df_cleaned2[df_cleaned2['group'] == 'control']['user_id'].nunique()
treatment_users = df_cleaned2[df_cleaned2['group'] == 'treatment']['user_id'].nunique()
total_users = control_users + treatment_users
print("Control group unique users:",control_users)
print("Treatment group unique users:",treatment_users)


# Calculate the percentages and create two lists
control_perc = control_users / total_users
treatment_perc = treatment_users / total_users
print("Percentage of users in the Control group:",100*round(control_perc,4),"%")
print("Percentage of users in the Treatment group:",100*round(treatment_perc,4),"%")


observed = [ control_users, treatment_users ]
expected = [ total_users/2, total_users/2 ]


# Run chisquare test on observed and expected lists and print the results
from scipy.stats import chisquare
chi = chisquare(observed, f_exp=expected)


print(chi)
if chi[1] < 0.05:
    print("SRM may be present")
else:
    print("SRM likely not present")
```

```
Control group unique users: 144300
Treatment group unique users: 144389
Percentage of users in the Control group: 49.980000000000004 %
Percentage of users in the Treatment group: 50.019999999999996 %
Power_divergenceResult(statistic=0.027437831022311208, pvalue=0.8684373649715974)
SRM likely not present
```

Sanity Check: Sample Ratio Mismatch (SRM)

This is the result of **Chi-square goodness-of-fit test:**

Percentage of users in the **Control** group: **49.98 %**

Percentage of users in the **Treatment group**: **50.02 %**

Power_divergenceResult(statistic=0.027437831022311208, **pvalue=0.8684373649715974**)

SRM likely not present

Pvalue > 0.05, We **fail to reject null hypothesis**, so We can **continue running the experiment.**

Methodology

- Data Cleaning
- Experimental Design
- Exploratory Data Analysis (EDA)
- Sanity Check
- Hypothesis Testing
- Conclusion

Hypothesis Testing

```python
from statsmodels.stats.proportion import proportions_ztest, proportion_confint

# Calculate the number of users in groups Control and Treatment
n_control = df_cleaned2[df_cleaned2['group'] == 'control']['user_id'].nunique()
n_treatment = df_cleaned2[df_cleaned2['group'] == 'treatment']['user_id'].nunique()
print('Control Group users:',n_control)
print('Treatment Group users:',n_treatment, '\n')


# Compute unique converted users in each group and assign to lists
converted_control = df_cleaned2[df_cleaned2['group'] == 'control'].groupby('user_id')['converted'].max().sum()
converted_treatment = df_cleaned2[df_cleaned2['group'] == 'treatment'].groupby('user_id')['converted'].max().sum()


converted_abtest = [converted_control, converted_treatment]
n_abtest = [n_control, n_treatment]


# Calculate the z_stat, p-value, and 95% confidence intervals
z_stat, pvalue = proportions_ztest(converted_abtest, nobs=n_abtest)
(A_lo95, B_lo95), (A_up95, B_up95) = proportion_confint(converted_abtest, nobs=n_abtest, alpha=0.05)


print(f'p-value: {pvalue:.4f}')
print(f'Control Group 95% CI : [{A_lo95:.4f}, {A_up95:.4f}]')
print(f'Treatment Group 95% CI : [{B_lo95:.4f}, {B_up95:.4f}]', '\n')

if pvalue < 0.05:
  print('Reject null hypothesis')
else:
  print('Fail to reject null hypothesis')
```

```
Control Group users: 144300
Treatment Group users: 144389

p-value: 0.2226
Control Group 95% CI : [0.1186, 0.1220]
Treatment Group 95% CI : [0.1172, 0.1205]

Fail to reject null hypothesis
```

Hypothesis Testing

To examine statistically significant **difference** between **conversion rate** of **Control** and **Treatment** Groups, We used **proportions_ztest** from **statsmodels.stats.proportion**. This is the result of the test:

- **p-value: 0.2226**
- **Control** Group **95% CI : [0.1186, 0.1220]**
- **Treatment** Group **95% CI : [0.1172, 0.1205]**

P-value > 0.05, it means We **fail to reject null hypothesis**. So, there is **no** statistically significant **difference** between **conversion rate** of **Control** and **Treatment** Groups.

Methodology

- Data Cleaning
- Experimental Design
- Exploratory Data Analysis (EDA)
- Sanity Check
- Hypothesis Testing
- Conclusion

Conclusion

- There is **no** statistically significant **difference** between **conversion rate** of **Control** and **Treatment** Groups.
- The **new landing page did not hit the target** (15% conversion rate).
- We **should not implement** this **new landing page**, **keep** the **old landing page**, **or** perhaps **run the experiment longer** considering We ran this experiment in just 3 weeks. May be We need more time to see the significant impact of the new landing page.

References

datacamp.com:

- A/B Testing in Python
- Hypothesis Testing in Python

Thank you ...