Analyze A/B Test Results

This project will assure you have mastered the subjects covered in the statistics lessons. The hope is to have this project be as comprehensive of these topics as possible. Good luck!

Table of Contents

Introduction

A/B tests are very commonly performed by data analysts and data scientists. It is important that you get some practice working with the difficulties of these

For this project, you will be working to understand the results of an A/B test run by an e-commerce website. Your goal is to work through this notebook to help the company understand if they should implement the new page, keep the old page, or perhaps run the experiment longer to make their decision.

As you work through this notebook, follow along in the classroom and answer the corresponding quiz questions associated with each question. The labels for each classroom concept are provided for each question. This will assure you are on the right track as you work through the project, and you can feel more confident in your final submission meeting the criteria. As a final check, assure you meet all the criteria on the RUBRIC.

Part I - Probability

To get started, let's import our libraries.

```
mport pandas as pd
mport numpy as np
mport random
mport matplotlib.pyplot as plt
matplotlib inline
We are setting the seed to assure you get the same answers on quizzes as we set up
andom.seed(42)
```

`1.` Now, read in the `ab_data.csv` data. Store it in `df`. **Use your dataframe to answer the questions in Quiz 1 of the classroom.**

a. Read in the dataset and take a look at the top few rows here:

```
df=pd.read_csv('/content/ab_data.csv')
df.head()
```

AFFICHER LA SORTIE MASQUÉE

b. Use the below cell to find the number of rows in the dataset.

```
df.shape
```

AFFICHER LA SORTIE MASQUÉE

c. The number of unique users in the dataset.

```
df.user_id.nunique()
```

AFFICHER LA SORTIE MASQUÉE

d. The proportion of users converted.

```
df[df['converted']==1].shape[0]/df.user_id.nunique()
```

AFFICHER LA SORTIE MASQUÉE

e. The number of times the `new_page` and `treatment` don't line up.

```
tot_dontlineup=df[(df['landing_page']=='old_page') & (df['group']=='treatment')].sh
print(tot_dontlineup)
```

AFFICHER LA SORTIE MASQUÉE

f. Do any of the rows have missing values?

```
df.isnull().sum()
```

AFFICHER LA SORTIE MASQUÉE

`2.` For the rows where **treatment** is not aligned with **new_page** or **control** is not aligned with **old_page**, we cannot be sure if this row truly received the new or old page. Use **Quiz 2** in the classroom to provide how we should handle these rows.

a. Now use the answer to the quiz to create a new dataset that meets the specifications from the quiz. Store your new dataframe in **df2**.

```
df
```

	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
...	...	...	...	...	...
294473	751197	2017-01-03 22:28:38.630509	control	old_page	0
294474	945152	2017-01-12 00:51:57.078372	control	old_page	0
294475	734608	2017-01-22 11:45:03.439544	control	old_page	0
294476	697314	2017-01-15 01:20:28.957438	control	old_page	0
294477	715931	2017-01-16 12:40:24.467417	treatment	new_page	0

294478 rows × 5 columns

```
df
```

	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
...	...	...	...	...	...
294473	751197	2017-01-03 22:28:38.630509	control	old_page	0
294474	945152	2017-01-12 00:51:57.078372	control	old_page	0
294475	734608	2017-01-22 11:45:03.439544	control	old_page	0
294476	697314	2017-01-15 01:20:28.957438	control	old_page	0
294477	715931	2017-01-16 12:40:24.467417	treatment	new_page	0

294478 rows × 5 columns

```
df2_1=df[(df['landing_page']=='new_page') & (df['group']=='treatment')]
df2_2=df[(df['landing_page']=='old_page') & (df['group']=='control')]
```

```
df2_2=df[(df['landing_page']== old_page ) & (df['group']== control )]
```

```
df2 = pd.concat([df2_1, df2_2], ignore_index=True)
```

```
df2
```

	user_id	timestamp	group	landing_page	converted
0	661590	2017-01-11 16:55:06.154213	treatment	new_page	0
1	853541	2017-01-08 18:28:03.143765	treatment	new_page	0
2	679687	2017-01-19 03:26:46.940749	treatment	new_page	1
3	817355	2017-01-04 17:58:08.979471	treatment	new_page	1
4	839785	2017-01-15 18:11:06.610965	treatment	new_page	1
...	...	...	...	...	...
290580	718310	2017-01-21 22:44:20.378320	control	old_page	0
290581	751197	2017-01-03 22:28:38.630509	control	old_page	0
290582	945152	2017-01-12 00:51:57.078372	control	old_page	0
290583	734608	2017-01-22 11:45:03.439544	control	old_page	0
290584	697314	2017-01-15 01:20:28.957438	control	old_page	0

290585 rows × 5 columns

```
# Double Check all of the correct rows were removed - this should be 0
df2[((df2['group'] == 'treatment') == (df2['landing_page'] == 'new_page')) == False
```

AFFICHER LA SORTIE MASQUÉE

3. Use **df2** and the cells below to answer questions for **Quiz3** in the classroom.

a. How many unique **user_id**s are in **df2**?

```
df2.user_id.nunique()
```

AFFICHER LA SORTIE MASQUÉE

b. There is one **user_id** repeated in **df2**. What is it?

```
df2[df2.user_id.duplicated()].shape[0]
```

AFFICHER LA SORTIE MASQUÉE

c. What is the row information for the repeat **user_id**?

```
df2[df2.user_id.duplicated()]
```

AFFICHER LA SORTIE MASQUÉE

d. Remove **one** of the rows with a duplicate **user_id**, but keep your dataframe as **df2**.

```
df2.drop_duplicates(subset='user_id',inplace=True)
```

```
df2[df2.user_id.duplicated()].shape[0]
```

```
0
```

4. Use **df2** in the below cells to answer the quiz questions related to **Quiz 4** in the classroom.

a. What is the probability of an individual converting regardless of the page they receive?

```
df2.converted.sum()/df2.user_id.nunique()
```

AFFICHER LA SORTIE MASQUÉE

b. Given that an individual was in the `control` group, what is the probability they converted?

```
df2[df2['group']=='control'].converted.sum()/df2[df2['group']=='control'].user_id.r
```

AFFICHER LA SORTIE MASQUÉE

c. Given that an individual was in the `treatment` group, what is the probability they converted?

```
df2[df2['group']=='treatment'].converted.sum()/df2[df2['group']=='treatment'].user_
```

AFFICHER LA SORTIE MASQUÉE

d. What is the probability that an individual received the new page?

```
df2[df2['landing_page']=='new_page'].shape[0]/df2.user_id.nunique()
```

AFFICHER LA SORTIE MASQUÉE

e. Consider your results from a. through d. above, and explain below whether you think there is sufficient evidence to say that the new treatment page leads to more conversions.

Based on previous results, we can't say that new treatement page leads to more conversions.

Part II - A/B Test

Notice that because of the time stamp associated with each event, you could technically run a hypothesis test continuously as each observation was observed.

However, then the hard question is do you stop as soon as one page is considered significantly better than another or does it need to happen consistently for a certain amount of time? How long do you run to render a decision that neither page is better than another?

These questions are the difficult parts associated with A/B tests in general.

1. For now, consider you need to make the decision just based on all the data provided. If you want to assume that the old page is better unless the new page proves to be definitely better at a Type I error rate of 5%, what should your null and alternative hypotheses be? You can state your hypothesis in terms of words or in terms of p_{old} and p_{new}, which are the converted rates for the old and new pages.

H0: pnew-pold=<0

H1: pnew-pold>0

2. Assume under the null hypothesis, p_{new} and p_{old} both have "true" success rates equal to the **converted** success rate regardless of page - that is p_{new} and p_{old} are equal. Furthermore, assume they are equal to the **converted** rate in **ab_data.csv** regardless of the page.

Use a sample size for each page equal to the ones in **ab_data.csv**.

Perform the sampling distribution for the difference in **converted** between the two pages over 10,000 iterations of calculating an estimate from the null.

Use the cells below to provide the necessary parts of this simulation. If this doesn't make complete sense right now, don't worry - you are going to work through the problems below to complete this problem. You can use **Quiz 5** in the classroom to make sure you are on the right track.

a. What is the **convert rate** for p_{new} under the null?

```
pnew=df2[df2['converted']==1].shape[0]/df2.shape[0]
print(pnew)
```

AFFICHER LA SORTIE MASQUÉE

b. What is the **convert rate** for p_{old} under the null?

```
pold=df2[df2['converted']==1].shape[0]/df2.shape[0]
print(pold)
```

AFFICHER LA SORTIE MASQUÉE

c. What is n_{new}?

```
df2[df2['landing_page']=='new_page'].shape[0]
```

AFFICHER LA SORTIE MASQUÉE

d. What is n_{old}?

```
df2[df2['landing_page']=='old_page'].shape[0]
```

AFFICHER LA SORTIE MASQUÉE

e. Simulate n_{new} transactions with a convert rate of p_{new} under the null. Store these n_{new} 1's and 0's in **new_page_converted**.

```
new_page_converted=df2[df2['landing_page']=='new_page']
```

f. Simulate n_{old} transactions with a convert rate of p_{old} under the null. Store these n_{old} 1's and 0's in **old_page_converted**.

```
old_page_converted=df2[df2['landing_page']=='old_page']
```

g. Find p_{new} - p_{old} for your simulated values from part (e) and (f).

```
pnew-pold
```

```
0.0
```

h. Simulate 10,000 p_{new} - p_{old} values using this same process similarly to the one you calculated in parts **a. through g.** above. Store all 10,000 values in a numpy array called **p_diffs**.

```
p_diffs = []

for i in range(10000):
    new_page = df2.sample(new_page_converted.shape[0], replace=True)
    conv_new = new_page.query('converted == 1')
    pnew = conv_new.shape[0]/new_page.shape[0]
    old_page = df2.sample(old_page_converted.shape[0], replace=True)
    conv_old = old_page.query('converted == 1')
    pold = conv_old.shape[0]/old_page.shape[0]
    p_diffs.append(pnew - pold)
p_diffs
```

```
0.00180793439000748l8,
-0.00018082323563760327,
0.0004383819538636702,
-0.0004219075875697559,
0.0007893779038924686,
-0.0028788860592589838,
-0.00022903874172650052,
0.0003420737284628439,
-0.0016542154827044603,
0.0013813098384978933,
-0.0013856617581076702,
0.001050739411975657,
-0.0005872217921532169,
-9.15912614610076e-05,
0.0006449735484359148,
0.001085655101567412,
-0.0017021103788756875,
-0.0009381734025125515,
0.0007549158432268166,
0.000837511552130063,
0.0014364071834141318,
-0.0007452994405535984,
0.0019318961682384395,
-0.00034609821666729823,
-0.00014662891836066116,
0.0013333347898472625,
0.0013951775438386244,
-4.361109669466512e-05,
-0.0013994629539441916,
0.00045915366786647427,
0.0012089654277315648,
0.0012160945454847971,
-0.0002291478855282747,
0.0006381599244846825,
-0.0008969147716147013,
-0.0015365591275977447,
-0.00017433533771978038,
0.0011815523323396998,
0.0009543970791367762,
-0.0004495236232190343
```

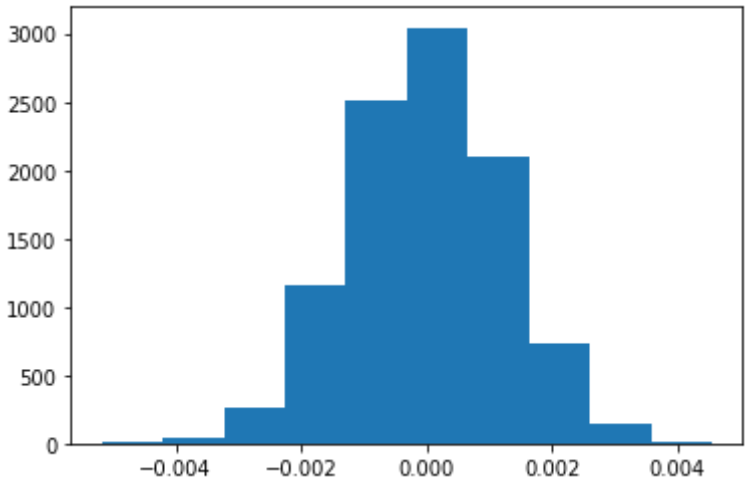
```
-0.0004495236222180343,

0.0016979631038942233,
-2.940743308911531e-05,
0.0004521865604440039 6,
-0.0011516007908707604,
0.001009782631904571,
-0.00026359630321871885,
-0.0015299740234814702,
-0.002025625018636512,
-0.0006971692030598164,
0.0002938803922086647,
0.0006173882104818923,
0.0007206430788423562,
-0.00018802227363882984,
0.0035354891071563005,
0.0007825404047346191,
0.0028198341463862853,
0.0015188749894246706,
0.000844453078973989,
-0.001027993729995555,
0.0006931071966734953
```

i. Plot a histogram of the **p_diffs**. Does this plot look like what you expected? Use the matching problem in the classroom to assure you fully understand what was computed here.

```
plt.hist(np.array(p_diffs))
```

```
(array([   5.,   42.,  268., 1162., 2503., 3043., 2097.,  727.,  139.,
         14.]),
 array([-0.00517779, -0.00420456, -0.00323134, -0.00225811, -0.00128489,
        -0.00031166,  0.00066156,  0.00163479,  0.00260801,  0.00358124,
         0.00455446]),
 <a list of 10 Patch objects>)
```



j. What proportion of the **p_diffs** are greater than the actual difference observed in **ab_data.csv**?

```
#let's calculate obs_diff
pold_obs = old_page_converted.query('converted == 1').user_id.nunique() / old_page_
```

```
pnew_obs = new_page_converted.query('converted == 1').user_id.nunique() / new_page_

# difference of pold_obs and pnew_obs
obs_diff=pnew_obs-pold_obs

# proportion of the p_diffs are greater than the actual difference observed
(np.array(p_diffs) > obs_diff).mean()
```

AFFICHER LA SORTIE MASQUÉE

k. In words, explain what you just computed in part **j.** What is this value called in scientific studies? What does this value mean in terms of whether or not there is a difference between the new and old pages?

In j. we computed p-value, and we found that it's greater than 0.05, then we could not reject H0. So there no impact in conversion when we switch from old page to new page.

l. We could also use a built-in to achieve similar results. Though using the built-in might be easier to code, the above portions are a walkthrough of the ideas that are critical to correctly thinking about statistical significance. Fill in the below to calculate the number of conversions for each page, as well as the number of individuals who received each page. Let `n_old` and `n_new` refer the the number of rows associated with the old page and new pages, respectively.

```
import statsmodels.api as sm

convert_old =old_page_converted.query('converted == 1').user_id.nunique()
convert_new =new_page_converted.query('converted == 1').user_id.nunique()
n_old = old_page_converted.user_id.nunique()
n_new = new_page_converted.user_id.nunique()
```

AFFICHER LA SORTIE MASQUÉE

m. Now use `stats.proportions_ztest` to compute your test statistic and p-value. Here is a helpful link on using the built in.

```
from statsmodels.stats.proportion import proportions_ztest

converted = np.array([convert_old , convert_new])
sizes_samples = np.array([n_old, n_new])
# note, no need for a Ho value here - it's derived from the other parameters
stat, p_value = proportions_ztest(count=converted, nobs=sizes_samples,alternative='

print(p_value)
```

```
0.9050583127590245
```

n. What do the z-score and p-value you computed in the previous question mean for the conversion rates of the old and new pages? Do they agree with the findings in parts **j.** and **k.**?

p-value is greater than 0.05, so again we could not reject H0. wich confirm the observation in pats j. and k.

Part III - A regression approach

1. In this final part, you will see that the result you acheived in the previous A/B test can also be acheived by performing regression.

a. Since each row is either a conversion or no conversion, what type of regression should you be performing in this case?

In this case we can use logistic regression.

b. The goal is to use **statsmodels** to fit the regression model you specified in part **a.** to see if there is a significant difference in conversion based on which page a customer receives. However, you first need to create a column for the intercept, and create a dummy variable column for which page each user received. Add an **intercept** column, as well as an **ab_page** column, which is 1 when an individual receives the **treatment** and 0 if **control**.

```
df2['intercept'] = 1
df2[['old_page','new_page']] = pd.get_dummies(df2['landing_page'])
df2 = df2.drop('old_page', axis=1)
df2['ab_page']=df2['group'].replace({'control': 0, 'treatment': 1})
```

```
df2 = df2.drop('group', axis=1)
df2 = df2.drop('landing_page', axis=1)
df2
```

	user_id	timestamp	converted	intercept	new_page	ab_page
0	661590	2017-01-11 16:55:06.154213	0	1	0	1
1	853541	2017-01-08 18:28:03.143765	0	1	0	1
2	679687	2017-01-19 03:26:46.940749	1	1	0	1
3	817355	2017-01-04 17:58:08.979471	1	1	0	1
4	839785	2017-01-15 18:11:06.610965	1	1	0	1
...	...	...	...	...	...	...
290580	718310	2017-01-21 22:44:20.378320	0	1	1	0

c. Use **statsmodels** to import your regression model. Instantiate the model, and fit the model using the two columns you created in part **b.** to predict whether or not an individual converts.

```
log_mod = sm.Logit(df2['converted'], df2[['intercept', 'ab_page']])
results = log_mod.fit()
```

AFFICHER LA SORTIE MASQUÉE

d. Provide the summary of your model below, and use it as necessary to answer the following questions.

```
results.summary()
```

AFFICHER LA SORTIE MASQUÉE

e. What is the p-value associated with **ab_page**? Why does it differ from the value you found in **Part II**?

Hint: What are the null and alternative hypotheses associated with your regression model, and how do they compare to the null and alternative hypotheses in the **Part II**?

p-value for ab_page is 0.19, but in part II, it was 0.9.This difference is due to That regression model is set up as a two-tailed or two-sided test, whereas in Part II it was a one-sided test.

f. Now, you are considering other things that might influence whether or not an individual converts. Discuss why it is a good idea to consider other factors to add into your regression model. Are there any disadvantages to adding additional terms into your regression model?

It's a good idea to add another factor, but we need to pay attention that this factor must be independent to other factors.

g. Now along with testing if the conversion rate changes for different pages, also add an effect based on which country a user lives. You will need to read in the **countries.csv** dataset and merge together your datasets on the approporiate rows. Here are the docs for joining tables.

Does it appear that country had an impact on conversion? Don't forget to create dummy variables for these country columns - **Hint: You will need two columns for the three dummy variables.** Provide the statistical output as well as a written response to answer this question.

```
countries_df = pd.read_csv('./countries.csv')
df_new = countries_df.set_index('user_id').join(df2.set_index('user_id'), how='inne
```

```
### Create the necessary dummy variables
df_new[['CA','UK','US']]=pd.get_dummies(df_new['country'])
```

```
log_mod1 = sm.Logit(df_new['converted'], df_new[['intercept','ab_page','CA','UK']])
results1 = log_mod1.fit()
results1.summary()
```

```
Optimization terminated successfully.
         Current function value: 0.366113
         Iterations 6
```

Logit Regression Results

Dep. Variable:	converted	**No. Observations:**	290584
Model:	Logit	**Df Residuals:**	290580
Method:	MLE	**Df Model:**	3
Date:	Tue, 19 Jan 2021	**Pseudo R-squ.:**	2.323e-05
Time:	17:38:14	**Log-Likelihood:**	-1.0639e+05
converged:	True	**LL-Null:**	-1.0639e+05
Covariance Type:	nonrobust	**LLR p-value:**	0.1760

	coef	std err	z	P>|z|	[0.025	0.975]
intercept	-1.9893	0.009	-223.763	0.000	-2.007	-1.972
ab_page	-0.0149	0.011	-1.307	0.191	-0.037	0.007
CA	-0.0408	0.027	-1.516	0.130	-0.093	0.012
UK	0.0099	0.013	0.743	0.457	-0.016	0.036

p-value is greater than 0.05 so we can say that H0 could not be rejected. then country has not impact on page

h. Though you have now looked at the individual factors of country and page on conversion, we would now like to look at an interaction between page and country to see if there significant effects on conversion. Create the necessary additional columns, and fit the new model.

Provide the summary results, and your conclusions based on the results

```
## create new variables
df_new['UK_ab_page'] = df_new['UK'] * df_new['ab_page']
df_new['CA_ab_page'] = df_new['CA'] * df_new['ab_page']


### Fit Your Linear Model And Obtain the Results
log_mod2 = sm.Logit(df_new['converted'], df_new[['intercept','ab_page','CA','UK','U
results2 = log_mod2.fit()
results2.summary()
```

AFFICHER LA SORTIE MASQUÉE

even when we combine country and ab_page, p-value is still greater than 0.05 so we can say that H0 could not be rejected. then country has not impact on page

Conclusions

In this project we studied the impact of new page on convesion. we proved that the new page has no impact on conversion using two methods:

- A/B testing
- Logistic Regression

Then we tried to study the impact of new page and country on conversion. We used multiple logistic regression to prove that country has no impact on conversion.