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
- Part I - Probability
- Part II - A/B Test
- Part III - Regression

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.

In [225]:

```
import pandas as pd
import numpy as np
import random
import matplotlib.pyplot as plt
%matplotlib inline
#We are setting the seed to assure you get the same answers on quizzes as we set up
random.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:

In [226]:

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

Out[226]:

	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

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

In [227]:

```
df.shape
```

Out[227]:

```
(294478, 5)
```

c. The number of unique users in the dataset.

In [228]:

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

Out[228]:

```
290584
```

d. The proportion of users converted.

In [229]:

```
(df[df.converted == 1].user_id.nunique() / df.user_id.nunique()) * 100
```

Out[229]:

```
12.104245244060237
```

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

In [230]:

```
df[(df.group=='treatment')  & (df.landing_page == 'old_page')].shape +  df[(df.group=='control')  &
(df.landing_page == 'new_page')].shape
```

Out[230]:

```
(1965, 5, 1928, 5)
```

In [231]:

```
1965 + 1928
```

Out[231]:

```
3893
```

f. Do any of the rows have missing values?

In [232]:

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

Out[232]:

```
0
```

`2.` For the rows where **treatment** does not match with **new_page** or **control** does not match with **old_page**, we cannot be sure if this row truly received the new or old page. Use **Quiz 2** in the classroom to figure out 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**.

In [233]:

```
df2 = df.copy()
```

```
df2 = df.copy()
drop_index = list(df2[(df2.group=='treatment')  & (df2.landing_page == 'old_page')].index)
df2.drop(index=drop_index,inplace=True)
```

In [234]:

```
drop_index = list(df2[(df2.group=='control')  & (df2.landing_page == 'new_page')].index)
df2.drop(index=drop_index,inplace=True)
```

In [235]:

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

Out[235]:

0

`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**?

In [236]:

```
df2.user_id.count()
```

Out[236]:

290585

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

In [237]:

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

Out[237]:

	user_id	timestamp	group	landing_page	converted
2893	773192	2017-01-14 02:55:59.590927	treatment	new_page	0

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

In [238]:

```
df2[df2.user_id == 773192]
```

Out[238]:

	user_id	timestamp	group	landing_page	converted
1899	773192	2017-01-09 05:37:58.781806	treatment	new_page	0
2893	773192	2017-01-14 02:55:59.590927	treatment	new_page	0

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

In [239]:

```
df2.drop(index=1899,inplace=True)
```

In [240]:

```
df2[df2.user_id == 773192]
```

Out[240]:

	user_id	timestamp	group	landing_page	converted
2893	773192	2017-01-14 02:55:59.590927	treatment	new_page	0

`4.` Use **df2** in the cells below 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?

In [241]:

```
df2[(df2.converted == 1)].count()['user_id']/ df2.count()['user_id']
```

Out[241]:

```
0.11959708724499628
```

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

In [242]:

```
df2[(df2.group == 'control') & (df2.converted == 1)].count()['user_id']/ df2[(df2.group == 'control')].count()['user_id']
```

Out[242]:

```
0.1203863045004612
```

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

In [243]:

```
df2[(df2.group == 'treatment') & (df2.converted == 1)].count()['user_id']/ df2[(df2.group == 'treatment')].count()['user_id']
```

Out[243]:

```
0.11880806551510564
```

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

In [244]:

```
1/2
```

Out[244]:

```
0.5
```

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

Answer.
From the above results, we can see clearly, the probability that individual will convert is almost same (around 12%) for both groups (control and treatment). So I think, there is no solid evidence to suggest that the new page will lead 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.

H_{o} : p_{old} >= p_{new},
H_{1} : p_{new} > p_{old}

`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 **conversion rate** for p_{new} under the null?

In [245]:

```
p_new = df2.converted.mean()
print(p_new)
```

```
0.119597087245
```

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

In [246]:

```
p_old = df2.converted.mean()
print(p_old)
```

```
0.119597087245
```

c. What is n_{new}, the number of individuals in the treatment group?

In [247]:

```
n_new = df2[df2.group == 'treatment'].user_id.nunique()
print(n_new)
```

```
145310
```

d. What is n_{old}, the number of individuals in the control group?

In [248]:

```
n_old = df2[df2.group == 'control'].user_id.nunique()
print(n_old)
```

145274

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

In [249]:

```
new_page_converted  = np.random.choice([1,0],n_new,p =[p_new, 1-p_new])
```

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

In [250]:

```
old_page_converted  = np.random.choice([1,0],n_old,p =[p_old, 1-p_old])
```

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

In [251]:

```
new_page_converted.mean() - old_page_converted.mean()
```

Out[251]:

-0.00053237684227855353

h. Create 10,000 p_{new} - p_{old} values using the same simulation process you used in parts (a) through (g) above. Store all 10,000 values in a NumPy array called **p_diffs**.

In [252]:

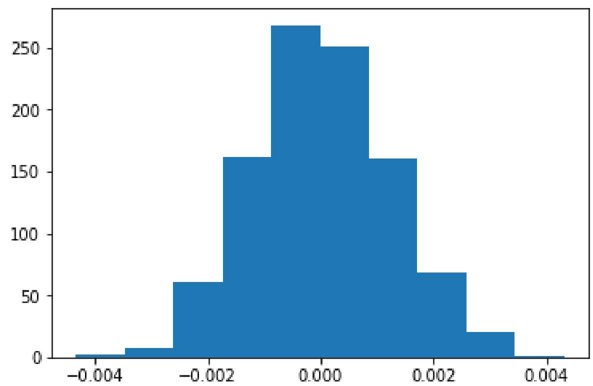
```
p_diffs = []

for i in range(1000):
    new_page_converted  = np.random.choice([1,0],n_new,p =[p_new, 1-p_new])
    old_page_converted  = np.random.choice([1,0],n_old,p =[p_old, 1-p_old])
    p_diffs.append(new_page_converted.mean() - old_page_converted.mean())
```

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.

In [253]:

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



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

In [254]:

In [254]:

```
actual_diff = df2[df2.group == 'treatment'].converted.mean() - df2[df2.group == 'control'].converte
d.mean()
print(actual_diff)
len (p_diffs[p_diffs > actual_diff]) / len(p_diffs)
```

```
-0.00157823898536
```

Out[254]:

```
0.921
```

k. Please explain using the vocabulary you've learned in this course 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?

Answer

- In part j, we have computed the pvalue.
- pvalue gives us the probability of observing the statistic when Null is True.
- Since, pvalue is large (much greater than the alpha level) , we would fail to reject the Null.
- So we will retain the Null, this means, conversion rate of old page is equal or better than the conversion rate of 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.

In [255]:

```
df2[df2.landing_page == 'old_page'].converted.sum()
```

Out[255]:

```
17489
```

In [256]:

```
import statsmodels.api as sm

convert_old = df2[df2.landing_page == 'old_page'].converted.sum()
convert_new = df2[df2.landing_page == 'new_page'].converted.sum()
n_old = df2[df2.landing_page == 'old_page'].shape[0]
n_new = df2[df2.landing_page == 'new_page'].shape[0]
print(convert_old)
print(convert_new)
print(n_old)
print(n_new)
```

```
17489
17264
145274
145310
```

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

In [257]:

```
z_score,p_value = sm.stats.proportions_ztest([convert_new, convert_old], [n_new,
n_old],alternative='larger')
```

In [258]:

```
print(z_score)
print(p_value)
```

```
-1.31092419842
```

```
0.905058312759
```

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.**?

Answer

- Again from Z test, pvalue is 0.905 which is greater than alpha level of 0.05. So, we would fail to reject the null.
- Hence we have reached to the same conclusion as above (in parts j and k) i.e. *"conversion rate of old page is equal or better than the conversion rate of new page.*

Part III - A regression approach

`1.` In this final part, you will see that the result you achieved in the A/B test in Part II above can also be achieved 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?

Answer
Since this type of problem can be classified as classification problem, **binary classification** to be more precise, becauase there are only two possible outcomes i.e. conversion or no conversion, *Logistic Regression* would be a good approach here.

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 in df2 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**.

In [259]:

```
df2.head(10)
```

Out[259]:

	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 [260]:

```
df2['ab_page'] = pd.get_dummies(df2.group)['treatment']
df2['intercept'] =1
```

In [261]:

```
df2.head(10)
```

Out[261]:

	user_id	timestamp	group	landing_page	converted	ab_page	intercept

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

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

In [262]:

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

```
Optimization terminated successfully.
         Current function value: 0.366118
         Iterations 6
```

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

In [263]:

```
results.summary2()
```

Out[263]:

Model:	Logit	No. Iterations:	6.0000
Dependent Variable:	converted	Pseudo R-squared:	0.000
Date:	2020-05-16 19:47	AIC:	212780.3502
No. Observations:	290584	BIC:	212801.5095
Df Model:	1	Log-Likelihood:	-1.0639e+05
Df Residuals:	290582	LL-Null:	-1.0639e+05
Converged:	1.0000	Scale:	1.0000

	Coef.	Std.Err.	z	P>|z|	[0.025	0.975]
intercept	-1.9888	0.0081	-246.6690	0.0000	-2.0046	-1.9730
ab_page	-0.0150	0.0114	-1.3109	0.1899	-0.0374	0.0074

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 **Part II**?

Answer
The pvalue associated with ab_page is 0.1899. It is different from the value found in part II because in regression model, we have following null and alternative hypotheses.

H_{o} $p_{old} = p_{new}$,
H_{1} : $p_{new} != p_{old}$

So, in this case, we have performed two tailed test while in part II we had performed one tailed 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?

Answer
We should consider other factors also while predicting converstion rate. One factor can be notice from given dataset is the *Timestamp*. If we include timestamp into the consideration, then we can determine, At what time of day most of the users visit our website? or may be, At what time of the day users convert more?

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

In [264]:

```
countries_df = pd.read_csv('countries.csv')
countries_df.head()
```

Out[264]:

	user_id	country
0	834778	UK
1	928468	US
2	822059	UK
3	711597	UK
4	710616	UK

In [265]:

```
countries_df.country.unique() #three values for country column.
```

Out[265]:

```
array(['UK', 'US', 'CA'], dtype=object)
```

In [266]:

```
df_merged = pd.merge(df2,countries_df,on='user_id') #merging data set on column user_id
```

In [267]:

```
df_merged.head(10)
```

Out[267]:

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

	user_id	timestamp	group	landing_page	converted	ab_page	intercept	country
9	839785	2017-01-15 18:11:06.610965	treatment	new_page	1	1	1	CA

In [268]:

```
countries_dummies = pd.get_dummies(df_merged.country)
```

In [269]:

```
df_merged = pd.concat([df_merged  ,countries_dummies],axis=1)
df_merged.drop(['UK'],1,inplace=True) #dropping extra dummy variabe column.
df_merged.head(10)
```

Out[269]:

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

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.

In [270]:

```
model = sm.Logit(df_merged['converted'], df_merged[['intercept','CA','US','ab_page']])
results = model.fit()
```

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

In [272]:

```
results.summary2()
```

Out[272]:

Model:	Logit	No. Iterations:	6.0000
Dependent Variable:	converted	Pseudo R-squared:	0.000
Date:	2020-05-16 20:08	AIC:	212781.1253
No. Observations:	290584	BIC:	212823.4439
Df Model:	3	Log-Likelihood:	-1.0639e+05
Df Residuals:	290580	LL-Null:	-1.0639e+05
Converged:	1.0000	Scale:	1.0000

	Coef.	Std.Err.	z	P>|z|	[0.025	0.975]
intercept	-1.9794	0.0127	-155.4145	0.0000	-2.0044	-1.9544
CA	-0.0506	0.0284	-1.7835	0.0745	-0.1063	0.0050

US	-0.0099	0.0133	-0.7433	0.4573	-0.0359	0.0162
ab_page	-0.0149	0.0114	-1.3069	0.1912	-0.0374	0.0075

Conclusion
Keeping significance level (alpha level) of 0.05 in mind, we can say all variables above have p-value greater than 0.05. So, we fail to reject the null, that means,*"conversion rate of old page is equal to the conversion rate of new page even after including countries into the consideration".*

In [273]:

```
from subprocess import call
call(['python', '-m', 'nbconvert', 'Analyze_ab_test_results_notebook.ipynb'])
```

Out[273]:

0