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 (https://review.udacity.com/#!/projects/37e27304-ad47-4eb0-a1ab-8c12f60e43d0/rubric).

Part I - Probability

To get started, let's import our libraries.

In [0]:

```
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 [0]:

```
df= pd.read_csv("ab_data.csv")
df.head(5)
```

Out[2]:

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

In [0]:

```
df.shape[0]
```

Out[3]:

```
294478
```

c. The number of unique users in the dataset.

In [0]:

```
len(df.user_id.unique())
```

Out[4]:

```
290584
```

d. The proportion of users converted.

In [0]:

```
x= (df[['converted']]==1).sum()
y= (df[['converted']]==0).sum()
percent= ((x)/(x+ y))*100
percent
```

Out[5]:

```
converted    11.965919
dtype: float64
```

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

In [0]:

```
df2 = df.query("(group == 'control' and landing_page == 'new_page') or (group == 'tı
df2.shape[0]
```

Out[6]:

```
3893
```

f. Do any of the rows have missing values?

In [0]:

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

Out[7]:

```
user_id         0
timestamp       0
group           0
landing_page    0
converted       0
dtype: int64
```

`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**.

In [0]:

```
df2 = df.query("(group == 'control' and landing_page == 'old_page') or (group == 'tı
```

In [0]:

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

Out[9]:

```
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 [0]:

```
df2.user_id.nunique
```

Out[10]:

```
<bound method IndexOpsMixin.nunique of 0         851104
1         804228
2         661590
3         853541
4         864975
           ...
294473    751197
294474    945152
294475    734608
294476    697314
294477    715931
Name: user_id, Length: 290585, dtype: int64>
```

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

In [0]:

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

Out[11]:

```
array([773192])
```

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

In [0]:

```
df2[df2.duplicated(['user_id'], keep=False)]
```

Out[12]:

	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 [0]:

```
df2 = df2.drop_duplicates(['user_id'], keep='first')
```

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?

In [0]:

```
x= (df2[['converted']]==1).sum()
y= (df2[['converted']]==0).sum()
percent= ((x)/(x+ y))
percent
```

Out[14]:

```
converted    0.119597
dtype: float64
```

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

In [0]:

```
control_group= df2[df2['group']=='control']
x= (control_group[['converted']]==1).sum()
y= (control_group[['converted']]==0).sum()
percent= ((x)/(x+ y))
percent
```

Out[15]:

```
converted    0.120386
dtype: float64
```

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

In [0]:

```
treatment_group= df2[df2['group']=='treatment']
x= (treatment_group[['converted']]==1).sum()
y= (treatment_group[['converted']]==0).sum()
percent= ((x)/(x+ y))
percent
```

Out[16]:

```
converted    0.118808
dtype: float64
```

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

In [0]:

```
x= (df2[['landing_page']]=="new_page").sum()
y= (df2[['landing_page']]=="old_page").sum()
percent= ((x)/(x+ y))
percent
```

Out[17]:

```
landing_page    0.500062
dtype: float64
```

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.

- **The control group converted at a slightly higher rate that the treatment group**
- **Probability of a person of received the new page is .50 it indicates that it is not possible for there to be a difference in conversion based on being given more opportunities to do so**

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?

In [0]:

```
p_new = df2['converted'].mean()
p_new
```

Out[18]:

```
0.11959708724499628
```

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

In [0]:

```
p_old = df2['converted'].mean()
p_old
```

Out[19]:

```
0.11959708724499628
```

c. What is n_{new}?

In [0]:

```
n_new = len(df2.query("landing_page == 'new_page'"))
n_new
```

Out[20]:

```
145310
```

d. What is n_{old}?

In [0]:

```
n_old = len(df2.query("landing_page == 'old_page'"))
n_old
```

Out[21]:

```
145274
```

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**.

In [0]:

```
new_page_converted = np.random.binomial(n_new,p_new)
new_page_converted
```

Out[22]:

```
17568
```

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**.

In [0]:

```
old_page_converted = np.random.binomial(n_old,p_old)
old_page_converted
```

Out[23]:

```
17551
```

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

In [0]:

```
(new_page_converted/n_new) - (old_page_converted/n_old)
```

Out[24]:

```
8.706027780146 23e-05
```

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**.

In [0]:

```
p_diffs = []
for _ in range(10000):
    new_page_converted = np.random.binomial(n_new,p_new)
    old_page_converted = np.random.binomial(n_old, p_old)
    p_diff = new_page_converted/n_new - old_page_converted/n_old
    p_diffs.append(p_diff)
```

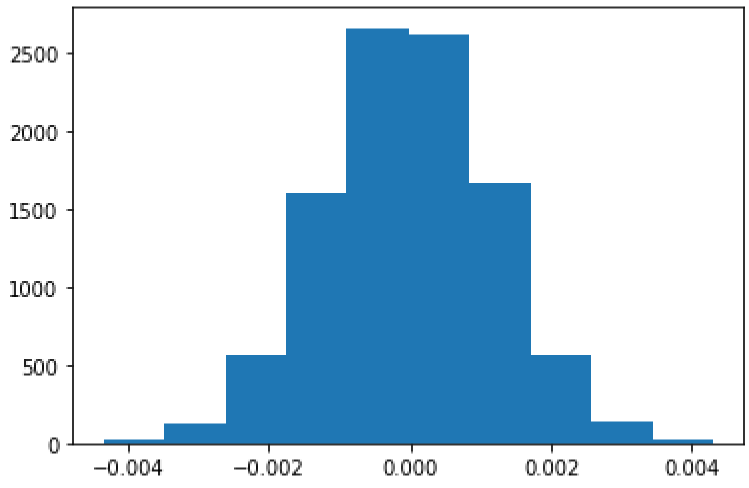
i. Plot a histogram of the **p_diffs**. Does this plot look like what you expected? Use the matching problem in the classroom to assure you fully understand what was computed here.

In [0]:

```
plt.hist(p_diffs)
```

Out[26]:

```
(array([  22.,  129.,  561., 1606., 2663., 2627., 1666.,  562.,  138.,
          26.]),
 array([-4.35204808e-03, -3.48689555e-03, -2.62174302e-03, -1.75659048
e-03,
       -8.91437951e-04, -2.62854182e-05,  8.38867114e-04,  1.70401965
e-03,
        2.56917218e-03,  3.43432471e-03,  4.29947724e-03]),
 <a list of 10 Patch objects>)
```



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

In [0]:

```
p_diff_orig = df[df['landing_page'] == 'new_page']['converted'].mean() -  df[df['la
p_diffs = np.array(p_diffs)
p_diff_proportion = (p_diff_orig < p_diffs).mean()
p_diff_proportion
```

Out[27]:

```
0.9151
```

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?

- **If null hypothesis is true pvalue gives the probability of statistics tested.**
- **In this case, the new page doesn't have better conversion rates than the old 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 [0]:

```
import statsmodels.api as sm

convert_old = sum(df2.query("landing_page == 'old_page'")['converted'])
convert_new = sum(df2.query("landing_page == 'new_page'")['converted'])
n_old = len(df2.query("landing_page == 'old_page'"))
n_new = len(df2.query("landing_page == 'new_page'"))
```

```
/usr/local/lib/python3.6/dist-packages/statsmodels/tools/_testing.py:1
9: FutureWarning: pandas.util.testing is deprecated. Use the functions
in the public API at pandas.testing instead.
  import pandas.util.testing as tm
```

m. Now use `stats.proportions_ztest` to compute your test statistic and p-value. Here (http://knowledgetack.com/python/statsmodels/proportions_ztest/) is a helpful link on using the built in.

In [0]:

```
z_score, p_value = sm.stats.proportions_ztest([convert_old, convert_new], [n_old, n_
print('z_score :: ',z_score)
print('p_value :: ',p_value)
```

```
z_score ::  1.3109241984234394
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.**?

In [0]:

```
from scipy.stats import norm
print(norm.cdf(z_score))
print(norm.ppf(1-(0.05)))
```

```
0.9050583127590245
1.6448536269514722
```

zscore is a measure of how many standard deviations below or above the population mean a raw score is. We find that the z-score of 1.3109241984234394 is less than the critical value of 1.6448536269514722 which means we can't reject the null hypothesis. We can conclude that old page conversions are slightly better than new page conversions. Eventhough the values are different from findings in parts j. and k but it suggests there is no significant difference between old page and new page conversions

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?

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**.

In [0]:

```
df2['intercept'] = 1
df2[['control', 'ab_page']]=pd.get_dummies(df2['group'])
df2.drop(labels=['control'], axis=1, inplace=True)
df2.head()
```

Out[31]:

	user_id	timestamp	group	landing_page	converted	intercept	ab_page
0	851104	2017-01-21 22:11:48.556739	control	old_page	0	1	0
1	804228	2017-01-12 08:01:45.159739	control	old_page	0	1	0
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	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.

In [0]:

```
import statsmodels.api as sm
import scipy.stats as stats
logit = sm.Logit(df2['converted'],df2[['intercept' ,'ab_page']])
results = logit.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 [0]:

```
stats.chisqprob = lambda chisq, df: stats.chi2.sf(chisq, df)
results.summary()
```

Out[33]:

Logit Regression Results

Dep. Variable:	converted	**No. Observations:**	290584
Model:	Logit	**Df Residuals:**	290582
Method:	MLE	**Df Model:**	1
Date:	Tue, 07 Apr 2020	**Pseudo R-squ.:**	8.077e-06
Time:	08:05:12	**Log-Likelihood:**	-1.0639e+05
converged:	True	**LL-Null:**	-1.0639e+05
Covariance Type:	nonrobust	**LLR p-value:**	0.1899

	coef	std err	z	P>|z|	[0.025	0.975]
intercept	-1.9888	0.008	-246.669	0.000	-2.005	-1.973
ab_page	-0.0150	0.011	-1.311	0.190	-0.037	0.007

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 is 0.190.The p-value here suggests that that new page is not statistically significant as 0.19 > 0.05 .In Part II it was a one sided test and in this section it was a two sided test. Here we test for not equal in our hypotheses whereas in Part II it was for different.

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?

There can be multiple factors that affect whether or not an individual converts. Factors like age can offer signifficant insights. To ensure that it's best fit it's always better to include more features however at the same time we should ensure to not take ample features since we do not want to overfit

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 (https://pandas.pydata.org/pandas-docs/stable/generated/pandas.DataFrame.join.html)](https://pandas.pydata.org/pandas-docs/stable/generated/pandas.DataFrame.join.html) 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 [0]:

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

In [0]:

```
df_new[['CA','UK','US']]=pd.get_dummies(df_new['country'])
mod = sm.Logit(df_new['converted'], df_new[['intercept', 'CA', 'UK']])
results = mod.fit()
results.summary()
```

```
Optimization terminated successfully.
         Current function value: 0.366116
         Iterations 6
```

Out[35]:

Logit Regression Results

Dep. Variable:	converted	**No. Observations:**	290584
Model:	Logit	**Df Residuals:**	290581
Method:	MLE	**Df Model:**	2
Date:	Tue, 07 Apr 2020	**Pseudo R-squ.:**	1.521e-05
Time:	08:05:14	**Log-Likelihood:**	-1.0639e+05
converged:	True	**LL-Null:**	-1.0639e+05
Covariance Type:	nonrobust	**LLR p-value:**	0.1984

	coef	**std err**	**z**	**P>|z|**	**[0.025**	**0.975]**
intercept	-1.9967	0.007	-292.314	0.000	-2.010	-1.983
CA	-0.0408	0.027	-1.518	0.129	-0.093	0.012
UK	0.0099	0.013	0.746	0.456	-0.016	0.036

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 [0]:

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

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

Out[36]:

Logit Regression Results

Dep. Variable:	converted	**No. Observations:**	290584
Model:	Logit	**Df Residuals:**	290580
Method:	MLE	**Df Model:**	3
Date:	Tue, 07 Apr 2020	**Pseudo R-squ.:**	2.323e-05
Time:	08:05:17	**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
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
ab_page	-0.0149	0.011	-1.307	0.191	-0.037	0.007

bold text

Conclusions

We can accept Null Hypothesis as there is no significant difference in conversion rates. We can reject alternate hypothesis.These results are based on given dataset. There may be limitations due to incorrect data or missing columns etc.

Applications :

These can be used in many practical appraoches such as trying to find out the sustainability of the system. It is used by Netflix to check for the amount of people it can support at a point of time. It is also used by E-commerce companies to monitor their traffic such as Amazon

In [ ]:

```
```