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 [61]:

```
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 s
et 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 [62]:

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

Out[62]:

	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 [63]:

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

Out[63]:

```
294478
```

c. The number of unique users in the dataset.

In [64]:

```
df['user_id'].nunique()
```

Out[64]:

```
290584
```

d. The proportion of users converted.

In [65]:

```
df.converted.mean()
```

Out[65]:

```
0.11965919355605512
```

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

In [66]:

```
len(df[(df['landing_page']=='new_page')!=(df['group']=='treatment')])
```

Out[66]:

```
3893
```

f. Do any of the rows have missing values?

In [67]:

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

Out[67]:

```
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 [68]:

```
df2 = df
```

In [69]:

```
df2.info()
```

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

In [70]:

```
df2.drop(df[(df['landing_page']=='new_page')!=(df['group']=='treatment')].inde
x , inplace = True)
```

In [71]:

```
df2.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 290585 entries, 0 to 294477
Data columns (total 5 columns):
user_id         290585 non-null int64
timestamp       290585 non-null object
group           290585 non-null object
landing_page    290585 non-null object
converted       290585 non-null int64
dtypes: int64(2), object(3)
memory usage: 13.3+ MB
```

In [72]:

```
df2.shape[0]
```

Out[72]:

```
290585
```

In [73]:

```
# 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[73]:

```
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 [74]:

```
df2['user_id'].nunique()
```

Out[74]:

```
290584
```

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

In [75]:

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

Out[75]:

```
1
```

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

In [76]:

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

Out[76]:

	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 [77]:

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

In [78]:

```
df2.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 290584 entries, 0 to 294477
Data columns (total 5 columns):
user_id         290584 non-null int64
timestamp       290584 non-null object
group           290584 non-null object
landing_page    290584 non-null object
converted       290584 non-null int64
dtypes: int64(2), object(3)
memory usage: 13.3+ MB
```

In [79]:

```
df2.shape[0]
```

Out[79]:

```
290584
```

`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 [80]:

```
df2.converted.mean()*100
```

Out[80]:

```
11.959708724499627
```

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

In [81]:

```
df2.groupby('group').mean()*100
```

Out[81]:

group	user_id	converted
control	7.881641e+07	12.038630
treatment	7.878457e+07	11.880807

The probability of an individuals that are in the control group and they converted is 0.120386

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

In [82]:

```
df2.groupby('group').mean()*100
```

Out[82]:

group	user_id	converted
control	7.881641e+07	12.038630
treatment	7.878457e+07	11.880807

The probability of an individuals that are in the treatment group and they converted is 0.118808

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

In [83]:

```
len(df2.query("group =='treatment'"))/df2.shape[0]
```

Out[83]:

```
0.5000619442226688
```

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.

Your answer goes here.:

I've found the probability for each group, where the **treatment** group have converted by 11.8%, the **contol** group have converted by 12.03% and the probability of an individual converting regardless of the page are 11.95%. Therefor, since all the obtaiend results are colse from each other and there's no huge diffrences between them but **old page** dose better and give a high percentage by a small change.

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.

Put your answer here.:

H0 null: Pnew <= Pold

H1 alternative: Pnew > Pold

`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 [84]:

```
Pnew = df2['converted'].mean()
Pnew*100
```

Out[84]:

```
11.959708724499627
```

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

In [85]:

```
Pold = df2['converted'].mean()
Pold*100
```

Out[85]:

```
11.959708724499627
```

c. What is n_{new}?

In [86]:

```
Nnew=len(df2.query(" group == 'treatment'"))
Nnew
```

Out[86]:

```
145310
```

d. What is n_{old}?

In [87]:

```
Nold=len(df2.query(" group == 'control'"))
Nold
```

Out[87]:

```
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 [88]:

```
new_page_converted=np.random.choice([0,1],size=Nnew,p=[Pnew,(1-Pnew)])
```

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 [89]:

```
old_page_converted=np.random.choice([0,1],size=Nold,p=[Pold,(1-Pold)])
```

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

In [90]:

```
new_page_converted = new_page_converted[:145274]
```

In [91]:

```
diffrance=(new_page_converted/Nnew)-(old_page_converted/Nold)
diffrance
```

Out[91]:

```
array([-1.7053719e-09, -1.7053719e-09, -1.7053719e-09, ...,
       -1.7053719e-09, -1.7053719e-09, -1.7053719e-09])
```

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**.

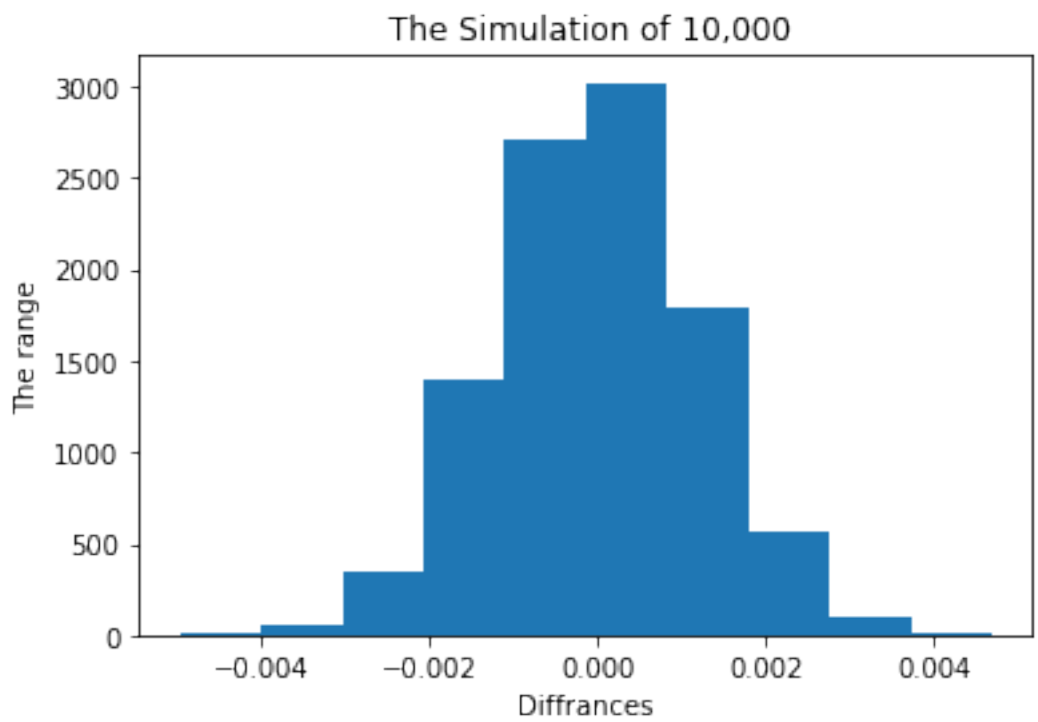
In [92]:

```
p_diffs=[]
for _ in range(10000):
 new_page_converted=np.random.choice([0,1],size=Nnew,p=[Pnew,(1-Pnew)]).mean()
 old_page_converted=np.random.choice([0,1],size=Nold,p=[Pold,(1-Pold)]).mean()
 DIFFRANCES=new_page_converted-old_page_converted
 p_diffs.append(DIFFRANCES)
```

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 [93]:

```
plt.hist(p_diffs)
plt.xlabel("Diffrances")
plt.ylabel("The range")
plt.title(" The Simulation of 10,000");
```



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

In [94]:

```
actual_difference = df.query("group =='treatment'")['converted'].mean()-df.que
ry("group =='control'")['converted'].mean()
actual_difference
```

Out[94]:

```
-0.0015782389853555567
```

In [95]:

```
#converting to array
p_diffs = np.array(p_diffs)
p_diffs
```

Out[95]:

```
array([ 1.17914349e-03,  6.56040180e-04,  9.65678588e-04, ...,
        3.05110740e-04, -4.56689121e-06, -1.64948497e-03])
```

In [96]:

```
(actual_difference < p_diffs).mean()
```

Out[96]:

```
0.9065
```

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?

Put your answer here. :

The value that called in the scientific studies is P-value and we will not rejecting the null hypothesis because the p-value is 0.9 ans it's larger than the α = 0.05. However, there is no difference between the new and old pages.

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 [97]:

```
import statsmodels.api as sm

convert_old = sum(df.query("group =='control'")['converted'])
convert_new = sum(df.query("group =='treatment'")['converted'])
n_old = len(df.query("group =='control'")['converted'])
n_new = len(df.query("group =='treatment'")['converted'])
```

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

In [98]:

```
z_score, p_value = sm.stats.proportions_ztest([convert_old, convert_new], [n_o
ld, n_new], alternative='smaller')
print(z_score, p_value)
```

```
1.3109241984234394 0.9050583127590245
```

compute the significance of z-score.

In [99]:

```
from scipy.stats import norm

norm.cdf(z_score)
```

Out[99]:

```
0.9050583127590245
```

compute the critical value in confidence interval by 95%.

In [100]:

```
norm.ppf(1-(0.05))
```

Out[100]:

```
1.6448536269514722
```

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.**?

Put your answer here.

So, from what we obtained that the z_score is 1.31 and the critical value in the confidence interval is 1.64 that's leading us to not reject the null hypothesis. Moreover, the p-value is identical to the proportion in J and it's also fails to reject the null hypothesis.

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?

Put your answer here. :

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 [101]:

```
#Create intercept column
df2['intercept']=1

#Create dummies
ab_page = ['treatment', 'control']
df2['ab_page'] = pd.get_dummies(df2.group)['treatment']
```

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 [102]:

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

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

In [103]:

```
results = logit.fit()
results.summary()
```

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

Out[103]:

Logit Regression Results

Dep. Variable:	converted	**No. Observations:**	290584
Model:	Logit	**Df Residuals:**	290582
Method:	MLE	**Df Model:**	1
Date:	Fri, 03 May 2019	**Pseudo R-squ.:**	8.077e-06
Time:	17:14:45	**Log-Likelihood:**	-1.0639e+05
converged:	True	**LL-Null:**	-1.0639e+05
		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**?

Put your answer here. :

The p-value that is associated with ab_page is (0.19) and it's lower than the p-value we obtained from Part II. However, because we implemented a one-sided test and in thid part it is two-sided test.

The hypotheses that associated with regression model is :

H0: Pnew - Pold =0

H1: Pnew - Pold != 0

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?

Put your answer here. :

It will make our work more reliable like if we consider the **timestamp** as influence and check which time the page become crowded we can see which time they converted the most and the least but as disadvantages it maybe give us inaccurate outputs.

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 [104]:

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

Out[104]:

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

In [105]:

```
df_new = countries_df.set_index('user_id').join(df2.set_index('user_id'), how=
'inner')
df_new.head()
```

Out[105]:

	country	timestamp	group	landing_page	converted	intercept	ab_page
user_id							
834778	UK	2017-01-14 23:08:43.304998	control	old_page	0	1	0
928468	US	2017-01-23 14:44:16.387854	treatment	new_page	0	1	1
822059	UK	2017-01-16 14:04:14.719771	treatment	new_page	1	1	1
711597	UK	2017-01-22 03:14:24.763511	control	old_page	0	1	0
710616	UK	2017-01-16 13:14:44.000513	treatment	new_page	0	1	1

To check the values for each country and how many country we have in the data set.

In [106]:

```
df_new['country'].value_counts()
```

Out[106]:

```
US    203619
UK     72466
CA     14499
Name: country, dtype: int64
```

In [107]:

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

Out[107]:

user_id	country	timestamp	group	landing_page	converted	intercept	ab_page
834778	UK	2017-01-14 23:08:43.304998	control	old_page	0	1	0
928468	US	2017-01-23 14:44:16.387854	treatment	new_page	0	1	1
822059	UK	2017-01-16 14:04:14.719771	treatment	new_page	1	1	1
711597	UK	2017-01-22 03:14:24.763511	control	old_page	0	1	0
710616	UK	2017-01-16 13:14:44.000513	treatment	new_page	0	1	1

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 [108]:

```
### Fit Your Linear Model And Obtain the Results
df['intercept'] = 1
log_mod = sm.Logit(df_new['converted'], df_new[['CA', 'US']])
results = log_mod.fit()
results.summary()
```

```
Optimization terminated successfully.
         Current function value: 0.447174
         Iterations 6
```

Out[108]:

Logit Regression Results

Dep. Variable:	converted	**No. Observations:**	290584
Model:	Logit	**Df Residuals:**	290582
Method:	MLE	**Df Model:**	1
Date:	Fri, 03 May 2019	**Pseudo R-squ.:**	-0.2214
Time:	17:14:47	**Log-Likelihood:**	-1.2994e+05
converged:	True	**LL-Null:**	-1.0639e+05
		LLR p-value:	1.000

	coef	std err	z	P>|z|	[0.025	0.975]
CA	-2.0375	0.026	-78.364	0.000	-2.088	-1.987
US	-1.9967	0.007	-292.314	0.000	-2.010	-1.983

In [109]:

```
log_mod = sm.Logit(df_new['converted'], df_new[['intercept', 'CA', 'US','ab_pa
ge']])
results = log_mod.fit()
results.summary()
```

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

Out[109]:

Logit Regression Results

Dep. Variable:	converted	**No. Observations:**	290584
Model:	Logit	**Df Residuals:**	290580
Method:	MLE	**Df Model:**	3
Date:	Fri, 03 May 2019	**Pseudo R-squ.:**	2.323e-05
Time:	17:14:48	**Log-Likelihood:**	-1.0639e+05
converged:	True	**LL-Null:**	-1.0639e+05
		LLR p-value:	0.1760

	coef	std err	z	P>|z|	[0.025	0.975]
intercept	-1.9794	0.013	-155.415	0.000	-2.004	-1.954
CA	-0.0506	0.028	-1.784	0.074	-0.106	0.005
US	-0.0099	0.013	-0.743	0.457	-0.036	0.016
ab_page	-0.0149	0.011	-1.307	0.191	-0.037	0.007

From what we have obtained, there is no significant influance on the convertion based on the country.

Conclusions

From the calcultions that we have done, we faild to find any statistcal evidance that telling us to reject the null hypothesis.