Analyze_ab_test_results_notebook

April 6, 2018

0.1 Analyze A/B Test Results

You may either submit your notebook through the workspace here, or you may work from your local machine and submit through the next page. Either way assure that your code passes the project RUBRIC. **Please save regularly

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!

0.2 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.

```
In [1]: 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 [2]: df=pd.read_csv("ab_data.csv")
        df.head()
```

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 [3]: df.shape[0]
```

```
Out[3]: 294478
```

c. The number of unique users in the dataset.

```
In [4]: df.nunique()[0]
```

```
Out[4]: 290584
```

d. The proportion of users converted.

```
In [5]: df.query('converted==1').nunique()[0]/df.nunique()[0]
```

```
Out[5]: 0.12104245244060237
```

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

```
In [6]: df.query('group=="treatment" and landing_page!="new_page"').nunique()[0]+df.query('group
```

```
Out[6]: 3893
```

f. Do any of the rows have missing values?

```
In [7]: df.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
```

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 [8]: df1=df.query('group=="treatment" and landing_page!="new_page"')
        df1=df1.append(df.query('group!="treatment" and landing_page=="new_page"'))
        df1=df1.append(df.query('group=="control" and landing_page!="old_page"'))
        df1=df1.append(df.query('group!="control" and landing_page=="old_page"'))

        df2=df.drop(df1.index.values)
```

```
In [9]: # Double Check all of the correct rows were removed - this should be 0
        df2[((df2['group'] == 'treatment') == (df2['landing_page'] == 'new_page')) == False].sha
```

```
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 [10]: df2.shape[0]
```

```
Out[10]: 290585
```

```
In [11]: df2.nunique()[0]
```

```
Out[11]: 290584
```

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

```
In [12]: df2[df2.duplicated(['user_id'])]
```

```
Out[12]:       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 [13]: df2[df2.duplicated(['user_id'])]
```

```
Out[13]:       user_id                   timestamp      group landing_page  converted
         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 [14]: df2=df2.drop(2893)
         df2.shape
```

```
Out[14]: (290584, 5)
```

```
In [15]: df2.head()
```

```
Out[15]:    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
```

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 [16]: df2[df2.converted==1].shape[0]/df2.shape[0]
```

```
Out[16]: 0.11959708724499628
```

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

```
In [17]: dfz=df2.query("group=='control'")
         dfz[dfz.converted==1].shape[0]/dfz.shape[0]
```

```
Out[17]: 0.1203863045004612
```

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

```
In [18]: dfz=df2.query("group=='treatment'")
         dfz[dfz.converted==1].shape[0]/dfz.shape[0]
```

```
Out[18]: 0.11880806551510564
```

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

```
In [19]: df2[df2.landing_page=='new_page'].shape[0]/df2.shape[0]
```

```
Out[19]: 0.5000619442226688
```

e. Use the results in the previous two portions of this question to suggest if you think there is evidence that one page leads to more conversions? Write your response below.

Your answer goes here. 50% of individual received the new page.so there is no evidence that one page leads to more conversions.Both are practically same.

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. Null- p_{new} <= p_{old}
Alternative-New page is better than old page i.e 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 **convert rate** for p_{new} under the null?

```
In [20]: p_new=df2[df2.converted==1].shape[0]/df2.shape[0]
         p_new

Out[20]: 0.11959708724499628
```

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

```
In [21]: p_old=df2[df2.converted==1].shape[0]/df2.shape[0]
         p_old

Out[21]: 0.11959708724499628
```

c. What is n_{new}?

```
In [22]: n_new=df2[df2.landing_page=='new_page'].count()[0]
         n_new

Out[22]: 145310
```

d. What is n_{old}?

```
In [23]: n_old=df2[df2.landing_page=='old_page'].count()[0]
         n_old

Out[23]: 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 [24]: new_page_converted = np.random.choice([0,1],n_new, p=(p_new,1-p_new))
         new_page_converted
```

```
Out[24]: array([0, 0, 0, ..., 1, 1, 1])
```

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 [25]: #Simulate  n_old  transactions with a convert rate of  p_old  under the null
         old_page_converted = np.random.choice([0,1],n_old, p=(p_old,1-p_old))
         #Display old_page_converted
         old_page_converted
```

```
Out[25]: array([1, 0, 1, ..., 1, 1, 1])
```

```
In [26]: new_page_converted.mean()
```

```
Out[26]: 0.88028353175968621
```

```
In [27]: old_page_converted.mean()
```

```
Out[27]: 0.88041218662665033
```

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

```
In [28]: new_page_converted.mean()-old_page_converted.mean()
```

```
Out[28]: -0.00012865486696411743
```

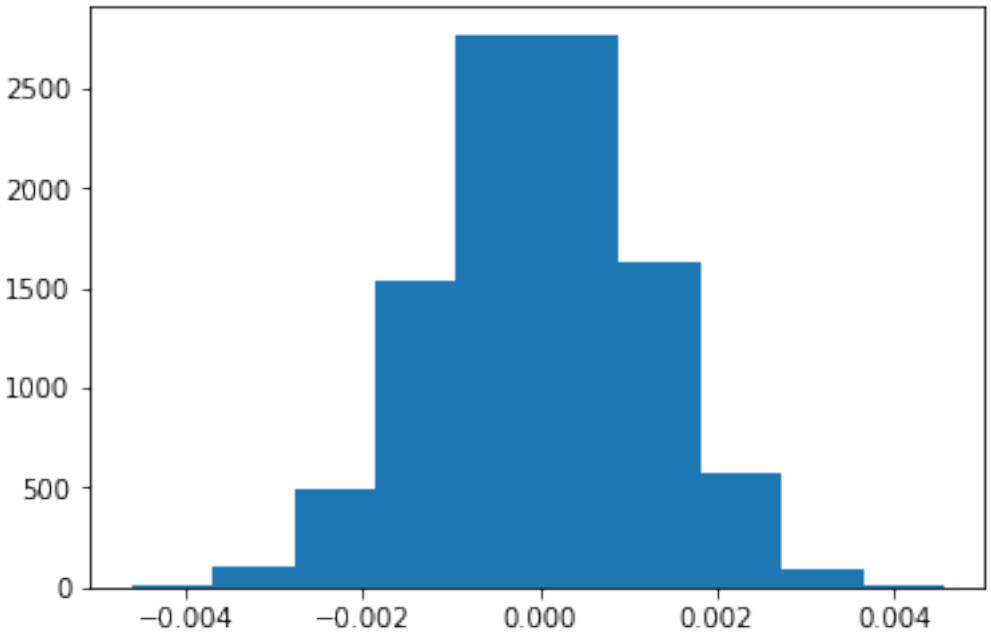
h. Simulate 10,000 p_{new} - p_{old} values using this same process similarly to the one you calculated in parts **a. through g.** above. Store all 10,000 values in **p_diffs**.

```
In [29]: p_diffs=[]
         for _ in range (10000):

             p_new_=np.random.choice([0,1],n_new, p=(p_new,1-p_new)).mean()
             p_old_=np.random.choice([0,1],n_old, p=(p_old,1-p_old)).mean()
             p_diffs.append(p_new_-p_old_)
```

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.

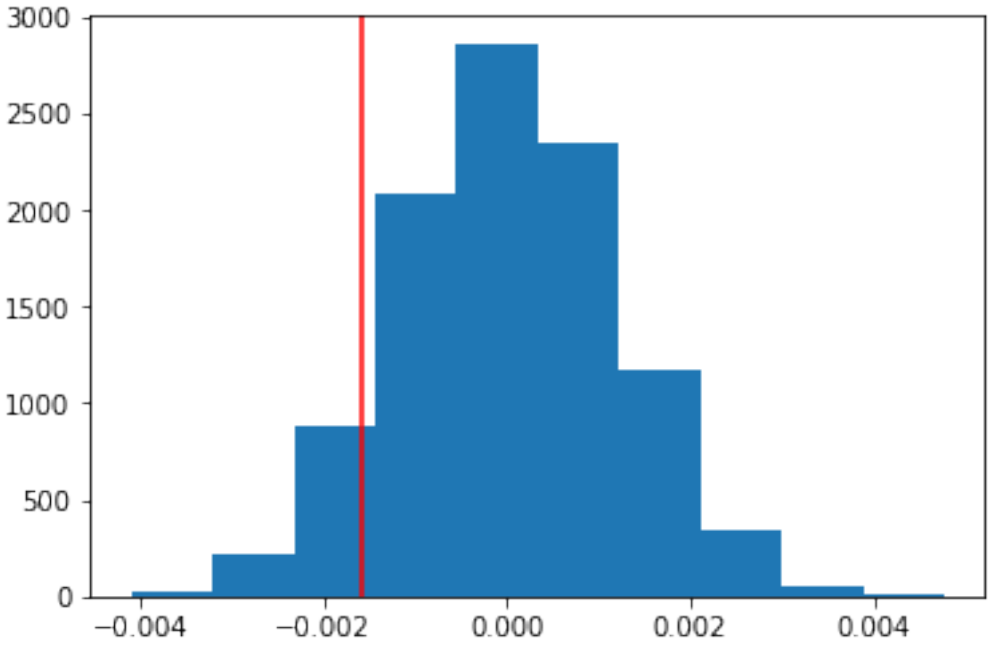
Yes.It is normally distributed as expected.

```
In [30]: plt.hist(p_diffs);
```



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

```
In [31]: #converting to array
         p_diffs=np.array(p_diffs)

In [32]: convert_new = df2.query('converted == 1 and landing_page == "new_page"')['user_id'].nun
         convert_old = df2.query('converted == 1 and landing_page == "old_page"')['user_id'].nun

In [33]: # Compute actual converted rate
         actual_cvt_new = float(convert_new)/ float(n_new)
         actual_cvt_old = float(convert_old)/ float(n_old)

In [34]: #observed difference in converted rate
         obs_diff=actual_cvt_new-actual_cvt_old
         obs_diff

Out[34]: -0.0015782389853555567

In [35]: #Distribution under the null hypothesis
         null_vals = np.random.normal(0, p_diffs.std(), p_diffs.size)

In [36]: #Plot Null distribution
         plt.hist(null_vals);
         #Plot vertical line for observed statistic
         plt.axvline(x=obs_diff,color ='red');
```




```
In [37]: (null_vals > obs_diff).mean()
```

```
Out[37]: 0.9083
```

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. Type I error rate of 5%, and Pold > Alpha, we fail to reject the null. Therefore, the data show,there is difference between new and old page.Old page has higher probablity of convert rate than new page

P-Value: The probability of observing our statistic or a more extreme statistic from the null hypothesis.

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 [38]: import statsmodels.api as sm
         convert_old,convert_new,n_old,n_new
```

```
/opt/conda/lib/python3.6/site-packages/statsmodels/compat/pandas.py:56: FutureWarning: The panda
  from pandas.core import datetools
```

```
Out[38]: (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 [39]: z_score, p_value = sm.stats.proportions_ztest(np.array([convert_new,convert_old]),np.ar

In [40]: z_score, p_value

Out[40]: (-1.3109241984234394, 0.90505831275902449)
```

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. Since the z-score of -1.3109241984234394 does not exceedsthe critical value of 1.959963984540054, we accept the null hypothesis. Yes they agree.The p-value also matches.

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 colun 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 [42]: #creating intercept and dummies
         df2['intercept']=1
         df2[['new_page','old_page']]=pd.get_dummies(df2['landing_page'])
         df2['ab_page']= pd.get_dummies(df2['group']) ['treatment']
         df2.head()
```

```
Out[42]:    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

            intercept  new_page  old_page  ab_page
         0          1         0         1        0
         1          1         0         1        0
         2          1         1         0        1
         3          1         1         0        1
         4          1         0         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 [43]: lm= sm.Logit(df2['converted'], df2[['intercept','ab_page']])
         res=lm.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 [44]: res.summary()

Out[44]: <class 'statsmodels.iolib.summary.Summary'>
         """
                                  Logit Regression Results
         ==============================================================================
         Dep. Variable:              converted   No. Observations:               290584
         Model:                          Logit   Df Residuals:                   290582
         Method:                           MLE   Df Model:                            1
         Date:                Fri, 06 Apr 2018   Pseudo R-squ.:               8.077e-06
         Time:                        19:17:53   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 the **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 associated with ab_page is 0.190. The null in c-e part is that there is no difference between the treatment and control group while alternative hypotheses is that there is difference between between the treatment and control group.

Part II assumes the old page is better unless the new page proves to be definitely better at a Type I error rate of 5%, compared to question c-e,they have different explainory varibale or factor for the result.

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.

Yes,it is a good idea to consider other factors to add into regression model.Other factor can be the time(timestamp variable). We can check if the converted rate depends on certain time of the day or certain day when user browse the website. For timestamp variable, we can further convert time as categorical variable which includes "Morning, afternoon, and evening", or "weekday and weekend". Disadavantage for adding additional terms into regression model is that it will make the model more complex and also, if new terms are dependable variable with the exisiting explanatory term, we need to add higher order term to help predict the result better.

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 varaibles.** Provide the statistical output as well as a written response to answer this question.

```
In [46]: import pandas as pd
         df_c=pd.read_csv('countries.csv')
         df_c.head()
```

Out[46]:

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

```
In [47]: #join datasets
         df3 = df2.merge(df_c, on ='user_id', how='left')
         df3.head()
```

Out[47]:

	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	

	intercept	new_page	old_page	ab_page	country
0	1	0	1	0	US
1	1	0	1	0	US
2	1	1	0	1	US
3	1	1	0	1	US
4	1	0	1	0	US

```
In [48]: df_c['country'].unique()

Out[48]: array(['UK', 'US', 'CA'], dtype=object)

In [49]: df3[['CA','UK','US']] = pd.get_dummies(df3['country'])
         df3 = df3.drop(df3['CA'])

In [50]: df3['intercept'] = 1
         #Create Logit regression model for country,CA and old page as baseline
         logit3 = sm.Logit(df3['converted'], df3[['intercept','new_page','UK','US']])
         result = logit3.fit()
         result.summary()

Optimization terminated successfully.
         Current function value: 0.366115
         Iterations 6


Out[50]: <class 'statsmodels.iolib.summary.Summary'>
         """
                                  Logit Regression Results
         ==============================================================================
         Dep. Variable:              converted   No. Observations:               290582
         Model:                          Logit   Df Residuals:                   290578
         Method:                           MLE   Df Model:                            3
         Date:                Fri, 06 Apr 2018   Pseudo R-squ.:               2.325e-05
         Time:                        19:23:31   Log-Likelihood:            -1.0639e+05
         converged:                       True   LL-Null:                   -1.0639e+05
                                                 LLR p-value:                    0.1757
         ==============================================================================
                          coef    std err          z      P>|z|      [0.025      0.975]
         ------------------------------------------------------------------------------
         intercept     -2.0300      0.027    -76.248      0.000      -2.082      -1.978
         new_page      -0.0150      0.011     -1.308      0.191      -0.037       0.007
         UK             0.0506      0.028      1.784      0.075      -0.005       0.106
         US             0.0408      0.027      1.516      0.129      -0.012       0.093
         ==============================================================================
         """

In [51]: 1/np.exp(-0.0150),np.exp(0.0506),np.exp(0.0408)

Out[51]: (1.0151130646157189, 1.0519020483004984, 1.0416437559600236)
```

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 [53]:
```
#Create a new interacton between new page and country US and UK
df3['UK_new_page'] = df3['new_page']* df3['UK']
df3['US_new_page'] = df3['new_page']* df3['US']
df3.head()
```

Out[53]:

	user_id	timestamp	group	landing_page	converted	intercept	new_page	old_page	ab_page	country	CA	UK	US	UK_new_page	US_new_page
2	661590	2017-01-11 16:55:06.154213	treatment	new_page	0	1	1	0	1	US	0	0	1	0	1
3	853541	2017-01-08 18:28:03.143765	treatment	new_page	0	1	1	0	1	US	0	0	1	0	1
4	864975	2017-01-21 01:52:26.210827	control	old_page	1	1	0	1	0	US	0	0	1	0	0
5	936923	2017-01-10 15:20:49.083499	control	old_page	0	1	0	1	0	US	0	0	1	0	0
6	679687	2017-01-19 03:26:46.940749	treatment	new_page	1	1	1	0	1	CA	1	0	0	0	0

In [54]:
```
#Create logistic regression for the intereaction variable between new page and country
lm4 = sm.Logit(df3['converted'], df3[['intercept','new_page','UK_new_page','US_new_page
result1 = lm4.fit()
result1.summary()
```

```
Optimization terminated successfully.
         Current function value: 0.366110
         Iterations 6
```

Out[54]:

```
<class 'statsmodels.iolib.summary.Summary'>
"""
                           Logit Regression Results
==============================================================================
Dep. Variable:              converted   No. Observations:               290582
Model:                          Logit   Df Residuals:                   290576
Method:                           MLE   Df Model:                            5
Date:                Fri, 06 Apr 2018   Pseudo R-squ.:               3.484e-05
Time:                        19:32:49   Log-Likelihood:            -1.0639e+05
converged:                       True   LL-Null:                   -1.0639e+05
                                        LLR p-value:                    0.1917
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
```

```
------------------------------------------------------------------------------
intercept      -2.0040      0.036    -55.008      0.000      -2.075      -1.933
new_page       -0.0674      0.052     -1.297      0.195      -0.169       0.034
UK_new_page     0.0783      0.057      1.378      0.168      -0.033       0.190
US_new_page     0.0469      0.054      0.871      0.384      -0.059       0.152
UK              0.0118      0.040      0.296      0.767      -0.066       0.090
US              0.0176      0.038      0.466      0.641      -0.056       0.091
==============================================================================
"""
```

```
In [55]: np.exp(result1.params)
```

```
Out[55]: intercept      0.134794
         new_page       0.934776
         UK_new_page    1.081428
         US_new_page    1.047978
         UK             1.011854
         US             1.017705
         dtype: float64
```

Conclusion:

From the above Logit Regression Results, we can see the coefficient of intereaction variable "UK_new_page" and "US_new_page" are different from the coefficient of new_page itself.

Also,only intercept's p-value is less than 0.05, which is statistically significant enough for converted rate. Other varable in the summary are not statistically significant. Additionally, Z-score for all X variables are not large enough to be significant for predicting converted rate. Therefore, the country a user lives is not significant on the converted rate considering the page the user land in.

For every unit for new_page decreases, convert will be 7.0% more likely to happen, holding all other varible constant. Convert is 1.08 times more likely to happen for UK and new page users than CA and new page users, holding all other varible constant.

Convert is 1.04 times more likely to happen for US and new page users than CA and new page users, holding all other varible constant.

Convert is 1.18 % more likely to happen for the users in UK than CA, holding all other varible constant.

Convert is 1.76 % more likely to happen for the users in US than CA, holding all other varible constant.

Conclusions

Congratulations on completing the project!

0.2.1 Gather Submission Materials

Once you are satisfied with the status of your Notebook, you should save it in a format that will make it easy for others to read. You can use the **File -> Download as -> HTML (.html)** menu to save your notebook as an .html file. If you are working locally and get an error about "No module name", then open a terminal and try installing the missing module using `pip install <module_name>` (don't include the "<" or ">" or any words following a period in the module name).

You will submit both your original Notebook and an HTML or PDF copy of the Notebook for review. There is no need for you to include any data files with your submission. If you made reference to other websites, books, and other resources to help you in solving tasks in the project, make sure that you document them. It is recommended that you either add a "Resources" section in a Markdown cell at the end of the Notebook report, or you can include a `readme.txt` file documenting your sources.

0.2.2 Submit the Project

When you're ready, click on the "Submit Project" button to go to the project submission page. You can submit your files as a .zip archive or you can link to a GitHub repository containing your project files. If you go with GitHub, note that your submission will be a snapshot of the linked repository at time of submission. It is recommended that you keep each project in a separate repository to avoid any potential confusion: if a reviewer gets multiple folders representing multiple projects, there might be confusion regarding what project is to be evaluated.

It can take us up to a week to grade the project, but in most cases it is much faster. You will get an email once your submission has been reviewed. If you are having any problems submitting your project or wish to check on the status of your submission, please email us at dataanalyst-project@udacity.com. In the meantime, you should feel free to continue on with your learning journey by continuing on to the next module in the program.