Analyze_ab_test_results_notebook

July 28, 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 [93]: 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 [126]: df = pd.read_csv('ab_data.csv')
          df.head()
```

Out[126]:

	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 [95]: df.shape[0]

Out[95]: 294478
```

c. The number of unique users in the dataset.

```
In [96]: df['user_id'].nunique()

Out[96]: 290584
```

d. The proportion of users converted.

```
In [97]: df['converted'].mean()

Out[97]: 0.11965919355605512
```

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

```
In [98]: #first , check the unique values for landing_page and group columns
         print(df.group.unique())
         print(df.landing_page.unique())

['control' 'treatment']
['old_page' 'new_page']
```

```
In [99]: #So we need query the rows have the treatment group with old_page
         #or the control group with the new_page
         df_new = df.query('group == "treatment" and landing_page == "old_page" or group == "con
         df_new.shape[0]

Out[99]: 3893
```

f. Do any of the rows have missing values?

```
In [100]: df.isnull().any()

Out[100]: user_id         False
          timestamp       False
          group           False
          landing_page    False
          converted       False
          dtype: bool
```

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 [101]: # create new dataset that haven't any uncertain data
          df2 = df.query('group == "treatment" and landing_page == "new_page" or group == "contr

In [102]: # Double Check all of the correct rows were removed - this should be 0
          df2[((df2['group'] == 'treatment') == (df2['landing_page'] == 'new_page')) == False].s

Out[102]: 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 [103]: df2['user_id'].nunique()

Out[103]: 290584
```

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

```
In [104]: df2[df2.duplicated(['user_id'])]['user_id']

Out[104]: 2893    773192
          Name: user_id, dtype: int64
```

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

```
In [105]: df2[df2.duplicated(['user_id'], keep = False)]

Out[105]:       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 [106]: #drop one of the duplicated user_id rows from the datafra
          df2 = df2.drop_duplicates(['user_id'])
          #check the number of rows for the duplicated user_id
          df2.query('user_id == 773192')
```

```
Out[106]:       user_id                   timestamp      group landing_page  converted
          1899   773192  2017-01-09 05:37:58.781806  treatment     new_page          0
```

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

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

```
In [107]: df2['converted'].mean()
```

```
Out[107]: 0.11959708724499628
```

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

```
In [108]: p_control = df2[df2['group'] == 'control']['converted'].mean()
          p_control
```

```
Out[108]: 0.1203863045004612
```

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

```
In [109]: p_treatment = df2[df2['group'] == 'treatment']['converted'].mean()
          p_treatment
```

```
Out[109]: 0.11880806551510564
```

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

```
In [110]: df2[df2['landing_page'] == 'new_page'].shape[0] / df2.shape[0]
```

```
Out[110]: 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.

Evidence that one page leads to more conversions? 1.the convertion rate in the control group is higher than the conversion rate in the treatment group, but with very tiny diffrence.

2.We can't decide the old page leads more conversions because of this tiny diffrence, so these two pages have similar performance.

3.the probability of an indivisual recieved the new page is .5 which means that the diffrence in the conversion rate is between the same amount of traffic for each group .

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.

**

$$H_0 : P_{new} <= P_{old}$$

**

$$H_1 : P_{new} > P_{old}$$

Our alternative hypothesis is what we want to prove to be true, in this case, that the new page design has a higher converted rate than the old page. And the null hypothesis is what we assume to be true before analyzing data, which is that the new page has a converted rate that is less than or equal to that of the old page. we can rearrange our hypotheses to look like this:

$$H_0 : P_{new} - P_{old} <= 0$$

$$H_1 : P_{new} - P_{old} > 0$$

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

```
Out[111]: 0.11959708724499628
```

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

```
In [112]: p_old = df2['converted'].mean()
          p_old
```

```
Out[112]: 0.11959708724499628
```

c. What is n_{new}?

```
In [113]: n_new = df2.query('group == "treatment"').shape[0]
          n_new
```

```
Out[113]: 145310
```

d. What is n_{old}?

```
In [114]: n_old = df2.query('group == "control"').shape[0]
          n_old
```

```
Out[114]: 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 [115]: new_page_converted = np.random.choice([1,0],n_new,p=[p_new,(1-p_old)])
          new_page_converted
```

```
Out[115]: array([0, 0, 0, ..., 0, 0, 0])
```

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

```
Out[116]: array([0, 1, 0, ..., 0, 0, 0])
```

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

```
In [117]: new_page_converted.mean()/n_new  - old_page_converted.mean()/n_old
```

```
Out[117]: 5.3788231389931613e-10
```

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**.

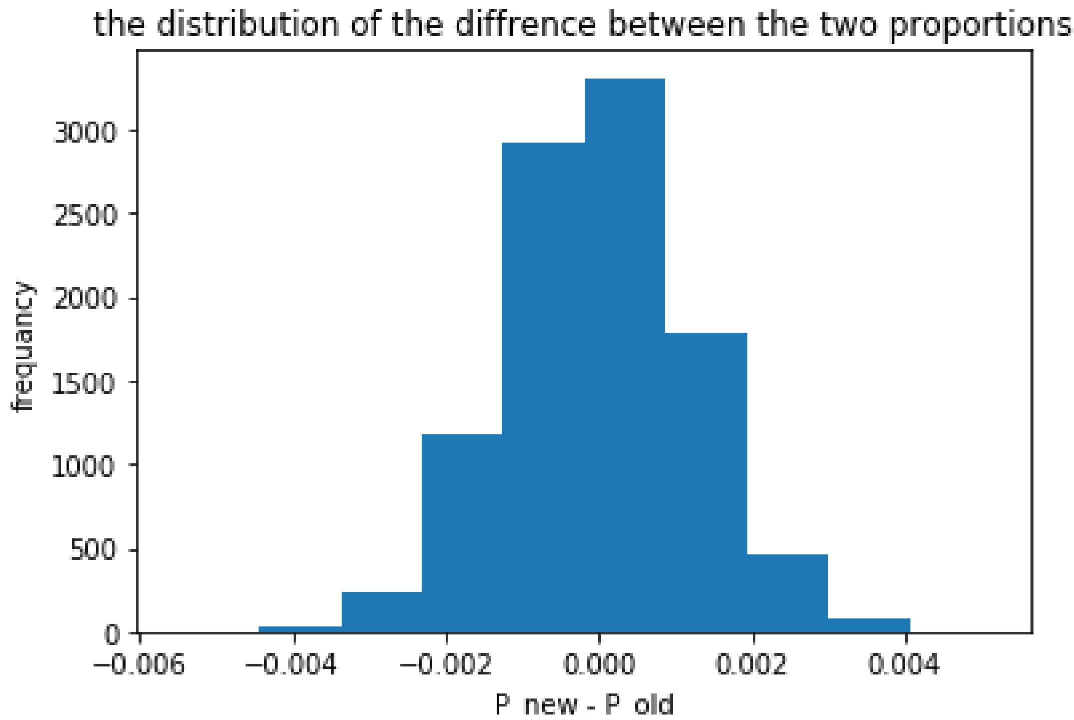
```
In [118]: p_diffs = []

          for _ in range(10000):
              old_page_converted = np.random.choice([1,0],n_old,p=[p_old,(1-p_old)])
              new_page_converted = np.random.choice([1,0],n_new,p=[p_new,(1-p_old)])
              diff = new_page_converted.mean() - old_page_converted.mean()
              p_diffs.append(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 [183]: plt.hist(p_diffs)
          plt.title('the distribution of the diffrence between the two proportions')
          plt.xlabel('P_new - P_old')
          plt.ylabel('frequancy')
```

```
Out[183]: Text(0,0.5,'frequancy')
```



In this plot, the distibution is norrmally distributed, here we plotted the diffrences in means between the converted rates for old page and new page, by generating random samples using sampling distribution

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

In [123]:
```
#calculate the actual diffrence in the data set
actual_diff = p_treatment - p_control
#fit the p-value which is the probability of the
#observing statistic if the null hypothesis is true
(p_diffs > actual_diff).mean()
```

Out[123]:
```
0.90849999999999997
```

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

In part j, I computed the probability of the observing statistic if the null hypothesis is true, this value is called the P-value**, this value means that if it is high ,the old page's performance is better than the new page or the same,and if it is very low (less than the type I error threshold), the new page's performance is better than the old one.**

we find that the p-value equals to .904 which is high enough and more than .05(which is the alpha value), so we fail to reject the null hypothesis, and make a decision that the old page's performance is better than or the same as the new page's performance

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 [124]: import statsmodels.api as sm

          convert_old = df.query('group == "control" and converted == 1').shape[0]
          convert_new = df.query('group == "treatment" and converted == 1').shape[0]
          n_old = df.query('group == "control"').shape[0]
          n_new = df.query('group == "treatment"').shape[0]
```

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 [125]: #the first parameter is the number of successes and the second is the number of trials
          z_score, p_value = sm.stats.proportions_ztest([convert_old, convert_new], [n_old, n_ne
          z_score,p_value

Out[125]: (1.2369217547321678, 0.89194193365121244)
```

in the built-in code above, i have used the alternative parameter equals to smaller, smaller means that the alternative hypothesis is p1<p2,where p1 is the proportion of the old page and p2 of the new page.

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.**?

we want first calculate the percentage of the z-score using cdf function which is the short of Cumulative density function , and then the critical value at 5% type I error rate using ppf function which is the short of Percent point function.

```
In [45]: from scipy.stats import norm

         print(norm.cdf(z_score))
          # Tells us the percentage of the significance of z-score is

         print(norm.ppf(1-0.05))
          # Tells us what our critical value at 95% confidence is,
          #that the type I error rate equal to 5%

0.891941933651
1.64485362695
```

In the z-test hypothesis testing, we calculate the critical value and the z-score to see wether the z-score is less than or more than the critical value,that if the z-score is less than the critical value means that we fail to reject the null hypothesis, anf if it is more than the critical value means we can reject the null hypothesis , in our status here we see that the z-score is less than the critical value, which means we fail to reject the null hypothesis and make a decision that the old page's converted rate is better than or equal to the new page's converted rate

We see that the findings in this part agree with the findings in parts j and k .

Part III - A regression approach

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

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

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 [67]:
```
import statsmodels.api as sm
#define the intercept column in the data frame
df2['intercept'] = 1
# create a dummy variable
df2['ab_page'] = pd.get_dummies(df['landing_page'])['new_page']
df2.tail()
```

Out[67]:
```
   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  ab_page
0          1        0
1          1        0
2          1        1
3          1        1
4          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 [68]:
```
#create the logistic regression model
logit_mod = sm.Logit(df2['converted'],df2[['intercept','ab_page']])
results_1 = logit_mod.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 [69]: results_1.summary()
```

```
Out[69]: <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, 27 Jul 2018   Pseudo R-squ.:               8.077e-06
Time:                        23:46:03   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
==============================================================================
"""
```

```
In [70]: np.exp(results_1.params)
```

```
Out[70]: intercept    0.136863
         ab_page      0.985123
         dtype: float64
```

the interpreting of this model is if the indivisual uses the new page, it is .985 times more likely to make a conversion than if he uses the old page. which means that there is a very tiny diffrence in the performance between the both pages, with more a little bit in the old page.

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**?

the P-value associated with ab_page equals to .19, it is diffrent from the p-value in the part II, that the hypothesises for null and alternative here are diffrent, here are look like this:

$$H_0 : P_{new} = P_{old}$$

$$H_1 : P_{new}! = P_{old}$$

Because we creates dummy variable ab_page that refers to new_page, then the baseline is the old page, and we can know the relashionsip comparing to the baseline.

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?

In general,when the customer arrived to the landing page, he was interested in the website's products, but may be there are other factors that influence whether he converted or not like the way of paying is suitable for him or not,the product was like what he wants or not, or the price was suitable for him or not. anyway, there are other factors that related to the customer himself, like the age, the range of his salary, or the time of opening the website.

It is a good idea to take these factors(age,salary,time) into our regression model, to know which of them is the most influenced in the conversion rate, but adding addtional terms to our regression model has disadvantages like Multicollinearity(that these factors may be correlated to one another) and if the linear relationship exists or not.

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 [79]: #read a csv file
         df3 = pd.read_csv("countries.csv")
         df3.head()
```

Out[79]:

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

```
In [80]: #knowing the unique values in the country columns
         df3.country.unique()
```

```
Out[80]: array(['UK', 'US', 'CA'], dtype=object)
```

```
In [81]: #join the country data frame with the converted rate data frame
         df_joined = df2.join(df3.set_index('user_id'),on ='user_id')
```

```
In [82]: #create dummy variables for the country column
         df_joined[['CA','UK','US']] = pd.get_dummies(df_joined['country'])
```

```
In [83]: #create a logistic reggresion model for the converted
         #column with the country dummy variables columns
         df_joined['intercept'] = 1
         logit_mod = sm.Logit(df_joined['converted'],df_joined[['intercept','UK','CA']])
         results_2 = logit_mod.fit()
         results_2.summary()
```

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

Out[83]:
```
<class 'statsmodels.iolib.summary.Summary'>
"""
                           Logit Regression Results
==============================================================================
Dep. Variable:              converted   No. Observations:               290584
Model:                          Logit   Df Residuals:                   290581
Method:                           MLE   Df Model:                            2
Date:                Sat, 28 Jul 2018   Pseudo R-squ.:               1.521e-05
Time:                        00:02:04   Log-Likelihood:            -1.0639e+05
converged:                       True   LL-Null:                   -1.0639e+05
                                        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
UK             0.0099      0.013      0.746      0.456      -0.016       0.036
CA            -0.0408      0.027     -1.518      0.129      -0.093       0.012
==============================================================================
"""
```

In [84]:
```
np.exp(results_2.params)
```

Out[84]:
```
intercept    0.135779
UK           1.009966
CA           0.960018
dtype: float64
```

In [85]:
```
1/np.exp(results_2.params)
```

Out[85]:
```
intercept    7.364925
UK           0.990133
CA           1.041647
dtype: float64
```

Interpretation of the previous logistic model

1.If an individual is from US, it is 0.9901 times more likely to make a conversion than if he is from UK , holding all other variables constant.

2.If an indivisual is from US , it is 1.04 more likely to make a conversion than if he is from CA, holding all other variables constant.

** from these values which is very close to 1 time, we can notice that there is no influence on the conversion rate comes from the country variable.**

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 [86]:
```
#showing the columns names
df_joined.columns
```

Out[86]:
```
Index(['user_id', 'timestamp', 'group', 'landing_page', 'converted',
       'intercept', 'ab_page', 'country', 'CA', 'UK', 'US'],
      dtype='object')
```

In [87]:
```
#create a logistic regression model for the ab_page and the dummy variables of country
logit_mod = sm.Logit(df_joined['converted'],df_joined[['intercept','ab_page','UK','CA']
results_3 = logit_mod.fit()
results_3.summary()
```

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

Out[87]:
```
<class 'statsmodels.iolib.summary.Summary'>
"""
                           Logit Regression Results
==============================================================================
Dep. Variable:              converted   No. Observations:               290584
Model:                          Logit   Df Residuals:                   290580
Method:                           MLE   Df Model:                            3
Date:                Sat, 28 Jul 2018   Pseudo R-squ.:               2.323e-05
Time:                        00:02:10   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     -1.9893      0.009   -223.763      0.000      -2.007      -1.972
ab_page       -0.0149      0.011     -1.307      0.191      -0.037       0.007
UK             0.0099      0.013      0.743      0.457      -0.016       0.036
CA            -0.0408      0.027     -1.516      0.130      -0.093       0.012
==============================================================================
"""
```

In [88]:
```
np.exp(results_3.params)
```

Out[88]:
```
intercept    0.136795
ab_page      0.985168
UK           1.009932
CA           0.960062
dtype: float64
```

We can interpret the result like this:

1. **if an indivisual uses the new page, it is .985 more likely to make a conversion than if he uses the old page, holding all other variables constant.**

2. **If an indivisual is from UK , it is 1,009 more likely to make a conversion than if he is from US, holding all other variables constant.**

3.**If an indivisual is from CA , it is 0.96 more likely to make a conversion than if he is from US, holding all other variables constant.**

In [74]:
```
#create the interaction model between the page and country using dmatrices

from patsy import dmatrices

# create dummy variables, and their interactions
y, X = dmatrices('converted ~ C(country)*C(landing_page)', df_joined, return_type="data
# flatten y into a 1-D array so scikit-learn can understand it
y = np.ravel(y)

#create a logistic model with X as independent variables, and y as dependent one.
logit_mod = sm.Logit(y,X)
results_4 = logit_mod.fit()
results_4.summary()
```

```
Optimization terminated successfully.
         Current function value: 0.366109
         Iterations 6
```

Out[74]:
```
<class 'statsmodels.iolib.summary.Summary'>
"""
                           Logit Regression Results
==============================================================================
Dep. Variable:                      y   No. Observations:               290584
Model:                          Logit   Df Residuals:                   290578
Method:                           MLE   Df Model:                            5
Date:                Fri, 27 Jul 2018   Pseudo R-squ.:               3.482e-05
Time:                        23:57:34   Log-Likelihood:            -1.0639e+05
converged:                       True   LL-Null:                   -1.0639e+05
                                        LLR p-value:                    0.1920
=====================================================================================
                                                    coef    std err          z      P>|z
-------------------------------------------------------------------------------------
Intercept                                        -2.0715      0.037    -55.798      0.00
C(country)[T.UK]                                  0.0901      0.040      2.225      0.02
C(country)[T.US]                                  0.0644      0.038      1.679      0.09
C(landing_page)[T.old_page]                       0.0674      0.052      1.297      0.19
C(country)[T.UK]:C(landing_page)[T.old_page]     -0.0783      0.057     -1.378      0.16
C(country)[T.US]:C(landing_page)[T.old_page]     -0.0469      0.054     -0.872      0.38
=====================================================================================
"""
```

In [75]:
```
np.exp(results_4.params)
```

Out[75]:
```
Intercept                                          0.126002
C(country)[T.UK]                                   1.094247
C(country)[T.US]                                   1.066532
C(landing_page)[T.old_page]                        1.069775
C(country)[T.UK]:C(landing_page)[T.old_page]       0.924703
C(country)[T.US]:C(landing_page)[T.old_page]       0.954198
dtype: float64
```

In [76]:
```
1/np.exp(results_4.params)
```

Out[76]:
```
Intercept                                          7.936353
C(country)[T.UK]                                   0.913871
C(country)[T.US]                                   0.937618
C(landing_page)[T.old_page]                        0.934776
C(country)[T.UK]:C(landing_page)[T.old_page]       1.081428
C(country)[T.US]:C(landing_page)[T.old_page]       1.048001
dtype: float64
```

interpreting the interaction model:

1.If an individual is from CA and use a new page, he is 0.913871 times more likely to make a conversion than if he is from UK and using the new page , holding all other variables constant.

2.If an individual is from CA and use a new page, he is 0.937618 times more likely to make a conversion than if he is from US and use a new page , holding all other variables constant.

3.If an individual is from CA and use a new page, he is 0.934776 times more likely to make a conversion than if he is from CA and use an old page, holding all other variables constant.

4.there is no influence in the conversion rate if an indivisual uses old page or new page, or if he is in a specific country or other country.

5.there is no diffrence in the conversion rate if an indivisual uses one of the pages and he is in a specific country.

0.3 Conclusion:

In this project, we showed three ways to know which page has the highest performace ,by the converted rate, they are the probability, hypothesis testing, and the regression models, all of these ways give us evidences that the performace of the old page is better than the new one but with a tiny diffrence, so we can make a decision that keep the old page and reject the new one.

0.4 resources:

1.http://knowledgetack.com/python/statsmodels/proportions_ztest/

2.http://www.statsmodels.org/dev/gettingstarted.html

3.https://www.theanalysisfactor.com/interaction-dummy-variables-in-linear-regression/

4.http://www.cantab.net/users/filimon/cursoFCDEF/will/logistic_interact.pdf

5.http://www.statsmodels.org/dev/generated/statsmodels.stats.proportion.proportions_ztest.html

In [1]:
```
from subprocess import call
call(['python', '-m', 'nbconvert', 'Analyze_ab_test_results_notebook.ipynb'])
```

Out[1]: 0