Analyze_ab_test_results_notebook-Copy1

October 28, 2019

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 [107]: 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 [108]: df = pd.read_csv("ab_data.csv")
```

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

```
In [109]: 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
```

c. The number of unique users in the dataset.

```
In [110]: df.user_id.nunique()

Out[110]: 290584
```

d. The proportion of users converted.

```
In [111]: df['converted'].describe()

Out[111]: count    294478.000000
          mean          0.119659
          std           0.324563
          min           0.000000
          25%           0.000000
          50%           0.000000
          75%           0.000000
          max           1.000000
          Name: converted, dtype: float64

In [112]: df.head(10)
```

Out[112]:

	user_id	timestamp	group	landing_page	converted
0	851104	2017-01-21 22:11:48.556739	control	old_page	0
1	804228	2017-01-12 08:01:45.159739	control	old_page	0
2	661590	2017-01-11 16:55:06.154213	treatment	new_page	0
3	853541	2017-01-08 18:28:03.143765	treatment	new_page	0

4	864975	2017-01-21 01:52:26.210827	control	old_page	1
5	936923	2017-01-10 15:20:49.083499	control	old_page	0
6	679687	2017-01-19 03:26:46.940749	treatment	new_page	1
7	719014	2017-01-17 01:48:29.539573	control	old_page	0
8	817355	2017-01-04 17:58:08.979471	treatment	new_page	1
9	839785	2017-01-15 18:11:06.610965	treatment	new_page	1

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

In [113]:
```
treatment_old_page = df.query("group == 'treatment' and landing_page == 'old_page'")
control_new_page = df.query("group == 'control' and landing_page == 'new_page'")
times = len(treatment_old_page) + len(control_new_page)
print("The number of times new_page and treatment don't match is {}".format(times))
```

```
The number of times new_page and treatment don't match is 3893
```

f. Do any of the rows have missing values?

In [114]:
```
# Using info() or df.isnull().sum()
df.isnull().sum()
```

Out[114]:
```
user_id         0
timestamp       0
group           0
landing_page    0
converted       0
dtype: int64
```

According to the cell above, no values are missing

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

a. Now use the answer to the quiz to create a new dataset that meets the specifications from the quiz. Store your new dataframe in **df2**.

In [115]:
```
# Find the indeces of the rows where treatment does not match with
# new_page. Then using the indeces, drop these rows:
df.drop(df.query("group == 'treatment' and landing_page == 'old_page'").index, inplace

# Find the indeces of the rows where control group does not match with
# new_page. Using the indeces, drop these rows
df.drop(df.query("group == 'control' and landing_page == 'new_page'").index, inplace=T
```

In [116]:
```
# Check df after removing rows above:
df.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 [117]:
```
# Save df into a new csv file
df.to_csv('ab_data_clean.csv', index=False)
```

In [118]:
```
# Read clean csv file into a new dataframe df2:
df2 = pd.read_csv('ab_data_clean.csv')
```

In [119]:
```
# 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[119]: 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 [120]:
```
len(df2['user_id'].unique())
```

Out[120]: 290584

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

In [121]:
```
df2[df2.user_id.duplicated(keep=False)].user_id
```

Out[121]:
```
1876    773192
2862    773192
Name: user_id, dtype: int64
```

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

In [122]:
```
df2[df2.user_id.duplicated(keep=False)]
```

Out[122]:

	user_id	timestamp	group	landing_page	converted
1876	773192	2017-01-09 05:37:58.781806	treatment	new_page	0
2862	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 [123]:
```
# Let's remove the first index of the duplicated rows, and check result
df2.drop(df2.index[[1876]], inplace=True)
```

```
In [124]: # Check df2 for duplicated user_id
          df2[df2.user_id.duplicated(keep=False)]
```

```
Out[124]: Empty DataFrame
          Columns: [user_id, timestamp, group, landing_page, converted]
          Index: []
```

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

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

```
In [125]: df2.converted.mean()
```

```
Out[125]: 0.11959708724499628
```

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

```
In [126]: # Select the control group of the converted and find the mean
          df2.query('group=="control"').converted.mean()
```

```
Out[126]: 0.1203863045004612
```

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

```
In [127]: # Select the control group of the converted and find the mean
          df2.query('group=="treatment"').converted.mean()
```

```
Out[127]: 0.11880806551510564
```

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

```
In [128]: # The number of individuals receive the new_page
          df2.query('landing_page=="new_page"').count()
```

```
Out[128]: user_id         145310
          timestamp       145310
          group           145310
          landing_page    145310
          converted       145310
          dtype: int64
```

```
In [129]: # The probability of an indicidual receive the new page is:
          # individuals with new_page divided by number of users
          indiv_new_page = df2.query('landing_page=="new_page"').user_id.count()/df2.shape[0]
          print("The probability that an individual received the new page is {}".format(indiv_ne
```

```
The probability that an individual received the new page is 0.5000619442226688
```

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

I don't think there is sufficient evidence to conclude that the new treatment page leads to more conversions specially that the two probabilities are very close. Also, not to forget that the probability of the control group of the converted individuals is still slightly higher

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.

`2.` Assume under the null hypothesis, p_{new} and p_{old} both have "true" success rates equal to the **converted** success rate regardless of page - that is p_{new} and p_{old} are equal. Furthermore, assume they are equal to the **converted** rate in **ab_data.csv** regardless of the page.

Use a sample size for each page equal to the ones in **ab_data.csv**.

Perform the sampling distribution for the difference in **converted** between the two pages over 10,000 iterations of calculating an estimate from the null.

Use the cells below to provide the necessary parts of this simulation. If this doesn't make complete sense right now, don't worry - you are going to work through the problems below to complete this problem. You can use **Quiz 5** in the classroom to make sure you are on the right track.

a. What is the **conversion rate** for p_{new} under the null?

```
In [130]: # The conversion rate is the mean
          p_new = df2['converted'].mean()
          p_new

Out[130]: 0.11959708724499628
```

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

```
In [131]: # The conversion rate is the mean
          p_old = df2['converted'].mean()
          p_old

Out[131]: 0.11959708724499628
```

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

```
In [132]: n_new=df2.query('group=="treatment"').user_id.nunique()
          n_new
```

```
Out[132]: 145310
```

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

```
In [133]: n_old=df2.query('group=="control"').user_id.nunique()
          n_old
```

```
Out[133]: 145274
```

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

```
In [134]: new_page_converted = np.random.binomial(n_new, p_new)
```

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

```
In [135]: old_page_converted = np.random.binomial(n_old, p_old)
```

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

```
In [136]: p_new = new_page_converted/n_new
          p_old = old_page_converted/n_old
          p_new - p_old
```

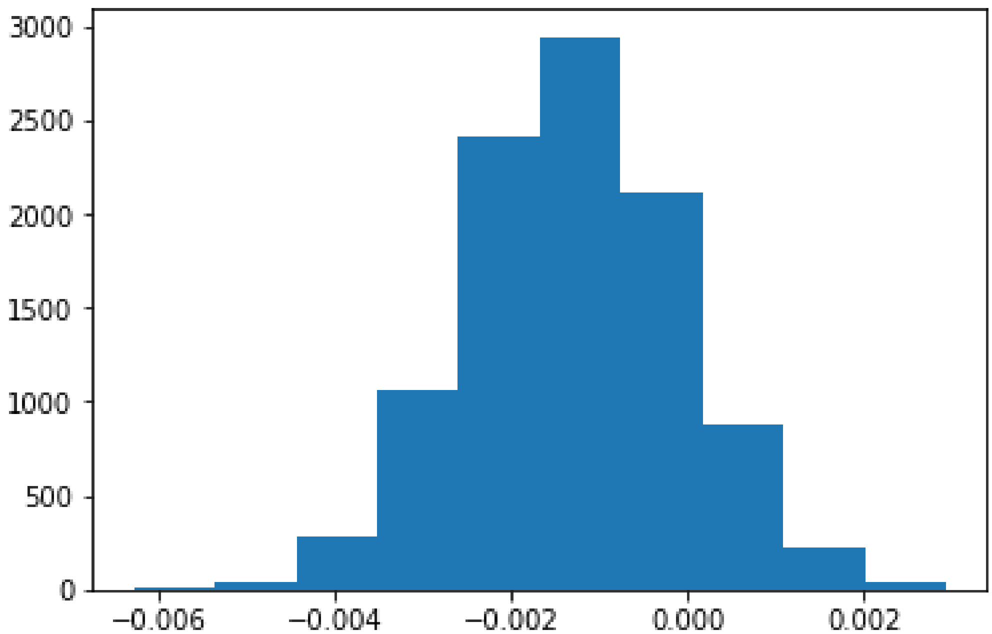
```
Out[136]: -0.0012962029694680З
```

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

```
In [137]: 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)
              diff = new_page_converted/n_new - old_page_converted/n_old
              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 [138]: plt.hist(p_diffs);
```



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

In [139]:
```
actual_difference = df2.query('group == "treatment"')['converted'].mean() - df2.query(
actual_difference
```

Out[139]: -0.0015782389853555567

In [140]:
```
p_diffs = np.array(p_diffs)
(p_diffs > actual_difference).mean()
```

Out[140]: 0.58679999999999999

k. Please explain using the vocabulary you've learned in this course what you just computed in part **j.** What is this value called in scientific studies? What does this value mean in terms of whether or not there is a difference between the new and old pages?

This is called the p-value. It is quite high (> 0.05), which implies that the difference between the old pages and new pages is not significant statistically. As a result, we failed to reject 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 [141]:
```
import statsmodels.api as sm

convert_old = sum(df2.query('group == "control"')['converted'])
convert_new = sum(df2.query('group == "treatment"')['converted'])
n_old = len(df2.query('group == "control"'))
n_new = len(df2.query('group == "treatment"'))
convert_old, convert_new, n_old, n_new
```

Out[141]:
```
(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 [142]:
```
z_score, p_value = sm.stats.proportions_ztest([convert_old, convert_new], [n_old, n_ne
z_score, p_value
```

Out[142]:
```
(1.3109241984234394, 0.094941687240975514)
```

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.**?

z_score and p_value > 0.05 mean the difference between the old pages and new ones is not statistically significant. Yes, I agree with the findings.
Part III - A regression approach
`1.` In this final part, you will see that the result you achieved in the A/B test in Part II above can also be achieved by performing regression.

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

Logistic regression should be performed in this case.

b. The goal is to use **statsmodels** to fit the regression model you specified in part **a.** to see if there is a significant difference in conversion based on which page a customer receives. However, you first need to create in df2 a column for the intercept, and create a dummy variable column for which page each user received. Add an **intercept** column, as well as an **ab_page** column, which is 1 when an individual receives the **treatment** and 0 if **control**.

In [143]:
```
df2['intercept'] = 1
df2[['control', 'treatment']] = pd.get_dummies(df2['group'])
df2.rename({'treatment': 'ab_page'}, axis=1, inplace=True)
df2.drop('control', axis=1, inplace=True)
df2.head()
```

Out[143]:
```
   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 instantiate your regression model on the two columns you created in part b., then fit the model using the two columns you created in part **b.** to predict whether or not an individual converts.

In [144]:
```
logistic_model = sm.Logit(df2['converted'],df2[['intercept','ab_page']])
results = logistic_model.fit()
```

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

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

In [145]:
```
results.summary()
```

Out[145]:
```
<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:                Sat, 26 Oct 2019   Pseudo R-squ.:               8.077e-06
Time:                        19:27:33   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 **Part II**?

0.3 Answer:

In part III, the p-value associated with ab_page is 0.190. In part II the p-value was 0.932.

Regression model / Null hypothesis: P_new is equal P_old. Regression model / Alternative hypothesis: P_new is not equal P_old.

In Part II: Null hypothesis: P_new is equal or worse P_old. Alternative hypothesis: P_new is better than P_old

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?

For example, age and gender can influence individual converts. The advantages of adding other factors into the regression model can improve prediction. The disadvantages of adding other factors into the regression model is that the model can be overfitted.

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

Does it appear that country had an impact on conversion? Don't forget to create dummy variables for these country columns - **Hint: You will need two columns for the three dummy variables.** Provide the statistical output as well as a written response to answer this question.

In [146]:
```
df_countries=pd.read_csv('countries.csv')
df_countries.head()
```

Out[146]:

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

In [147]:
```
len(df2.user_id.unique()) == len(df_countries.user_id.unique())
```

Out[147]: True

In [148]:
```
df3=df2.join(df_countries.set_index('user_id'),on='user_id')
df3.head()
```

Out[148]:

	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 country
```

```
0          1        0      US
1          1        0      US
2          1        1      US
3          1        1      US
4          1        0      US
```

In [149]:
```python
df3.country.unique()
```

Out[149]:
```
array(['US', 'CA', 'UK'], dtype=object)
```

In [150]:
```python
df3[['UK','US']] = pd.get_dummies(df3['country'],drop_first=True)
df3.head()
```

Out[150]:
```
   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 country  UK  US
0          1        0      US   0   1
1          1        0      US   0   1
2          1        1      US   0   1
3          1        1      US   0   1
4          1        0      US   0   1
```

In [151]:
```python
logistic_model = sm.Logit(df3['converted'],df3[['intercept','ab_page','UK','US']])
results = logistic_model.fit()
```

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

In [152]:
```python
results.summary()
```

Out[152]:
```
<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, 26 Oct 2019   Pseudo R-squ.:               2.323e-05
Time:                        19:27:34   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     -2.0300      0.027    -76.249      0.000      -2.082      -1.978
ab_page       -0.0149      0.011     -1.307      0.191      -0.037       0.007
UK             0.0506      0.028      1.784      0.074      -0.005       0.106
US             0.0408      0.027      1.516      0.130      -0.012       0.093
==============================================================================
"""
```

The P-values in the summary show that landing page and countries are not statistically significant

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 [153]:
```python
df3['ab_page_UK']=df3['ab_page'] * df3['UK']
df3['ab_page_US']=df3['ab_page'] * df3['US']
df3.head()
```

Out[153]:
```
   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 country  UK  US  ab_page_UK  ab_page_US
0          1        0      US   0   1           0           0
1          1        0      US   0   1           0           0
2          1        1      US   0   1           0           1
3          1        1      US   0   1           0           1
4          1        0      US   0   1           0           0
```

In [154]:
```python
logistic_model = sm.Logit(df3['converted'],df3[['intercept','ab_page','UK','US','ab_pa
results=logistic_model.fit()
```

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

In [155]:
```python
results.summary()
```

Out[155]:
```
<class 'statsmodels.iolib.summary.Summary'>
"""
                           Logit Regression Results
```

```
==============================================================================
Dep. Variable:              converted   No. Observations:               290584
Model:                          Logit   Df Residuals:                   290578
Method:                           MLE   Df Model:                            5
Date:                Sat, 26 Oct 2019   Pseudo R-squ.:               3.482e-05
Time:                        19:27:35   Log-Likelihood:            -1.0639e+05
converged:                       True   LL-Null:                   -1.0639e+05
                                        LLR p-value:                    0.1920
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
intercept     -2.0040      0.036    -55.008      0.000      -2.075      -1.933
ab_page       -0.0674      0.052     -1.297      0.195      -0.169       0.034
UK             0.0118      0.040      0.296      0.767      -0.066       0.090
US             0.0175      0.038      0.465      0.642      -0.056       0.091
ab_page_UK     0.0783      0.057      1.378      0.168      -0.033       0.190
ab_page_US     0.0469      0.054      0.872      0.383      -0.059       0.152
==============================================================================
"""
```

The P-values in the summary show that none of the variables are statistically significant.
Conclusion
We can't conclude that the difference in conversion rates of control and treatment groups are considerable enough. In addition, the result of the A/B test shows that we fail to reject the null hypothesis. Therefore, changing to a new page will not add significant value.
Finishing Up

Congratulations! You have reached the end of the A/B Test Results project! You should be very proud of all you have accomplished!

Tip: Once you are satisfied with your work here, check over your report to make sure that it is satisfies all the areas of the rubric (found on the project submission page at the end of the lesson). You should also probably remove all of the "Tips" like this one so that the presentation is as polished as possible.

0.4 Directions to Submit

Before you submit your project, you need to create a .html or .pdf version of this notebook in the workspace here. To do that, run the code cell below. If it worked correctly, you should get a return code of 0, and you should see the generated .html file in the workspace directory (click on the orange Jupyter icon in the upper left).

Alternatively, you can download this report as .html via the **File** > **Download as** submenu, and then manually upload it into the workspace directory by clicking on the orange Jupyter icon in the upper left, then using the Upload button.

Once you've done this, you can submit your project by clicking on the "Submit Project" button in the lower right here. This will create and submit a zip file with this .ipynb doc and the .html or .pdf version you created. Congratulations!

In [156]:
```
from subprocess import call
call(['python', '-m', 'nbconvert', 'Analyze_ab_test_results_notebook.ipynb'])
```

Out[156]:
```
0
```