Analyze_ab_test_results_notebook

January 21, 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

- Section **??**
- Section **??**
- Section **??**
- Section **??**

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 cell below 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.user_id.nunique()

Out[4]: 290584
```

d. The proportion of users converted.

```
In [5]: df.converted.mean()

Out[5]: 0.11965919355605512
```

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

```
In [6]: # In this case we are looking for two situation:
        # first: landing_pages that have new_page value and the group is not treatment
        # Second: landing_pages that have not new_page value and the group is treatment
        # Finally we must add the two valuse to obtain the number of times the new_page and trea
        df_first= df[(df.landing_page == 'new_page') & (df.group != 'treatment')].user_id.count(
        df_second= df[(df.landing_page != 'new_page') & (df.group == 'treatment')].user_id.count
        df_final= df_first + df_second
        df_final

Out[6]: 3893
```

f. Do any of the rows have missing values?

```
In [7]: df.isnull().values.any()   # No rows have missing values.

Out[7]: False
```

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 [8]: df2 = df.drop(df[((df.landing_page == 'new_page') & (df.group != 'treatment')) | ((df.la
```

```
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.user_id.nunique()
```

```
Out[10]: 290584
```

we notice that f2.info() give us 290585 rows in the df2 dataset, otherwise df2.user_id.nunique() give us 290584 user_id which mean that we have one duplicated row in user_id

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

```
In [11]: df2[df2.duplicated('user_id')]
```

```
Out[11]:       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.user_id==773192]
```

```
Out[13]:       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 [14]: df2=df2.drop_duplicates("user_id")
```

```
In [15]: df2[df2.user_id == 773192]
```

```
Out[15]:       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 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 [16]: df2.converted.mean()
```

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

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

```
In [17]: df2.converted[df2.group=='control'].mean()
```

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

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

```
In [18]: df2.converted[df2.group=='treatment'].mean()
```

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

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

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

```
Out[19]: user_id         0.500062
         timestamp       0.500062
         group           0.500062
         landing_page    0.500062
         converted       0.500062
         dtype: float64
```

the probability that an individual received the new page is 0.500062

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.

Regarding to the information we obtained above, we can't judge that we have evidence that one page leads to more conversions because the portions is 0.5 so we have the same portion for the oppusite situation 0.5 which is that one page dos'nt leads to any conversions.

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.

$H_0 : p_{new} <= p_{old}$ ------- $H_1 : p_{new} > p_{old}$

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

```
Out[20]: 0.11959708724499628
```

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

```
In [21]: p_old = df2.converted.mean() # the same!!!
         p_old
```

```
Out[21]: 0.11959708724499628
```

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

```
In [22]: # number of users who got the new page (i.e. group = treatment)
         n_new = df2[df2.group == 'treatment'].count()[0]
         n_new
```

```
Out[22]: 145310
```

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

```
In [23]: # number of users who reamined with old page (i.e. group = control)
         n_old = df2[df2.group == 'control'].count()[0]
         n_old
```

```
Out[23]: 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 [64]: new_page_converted = np.random.binomial(1,p_new,n_new)
         new_page_converted.mean()
```

```
Out[64]: 0.12080379877503269
```

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 [65]: old_page_converted = np.random.binomial(1,p_old,n_old)
         old_page_converted.mean()

Out[65]: 0.1191679171771962
```

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

```
In [66]: # Calculate difference in p under the null hypothesis
         p_new = new_page_converted.mean()
         p_old = old_page_converted.mean()
         p_new - p_old

Out[66]: 0.0016358815978364943
```

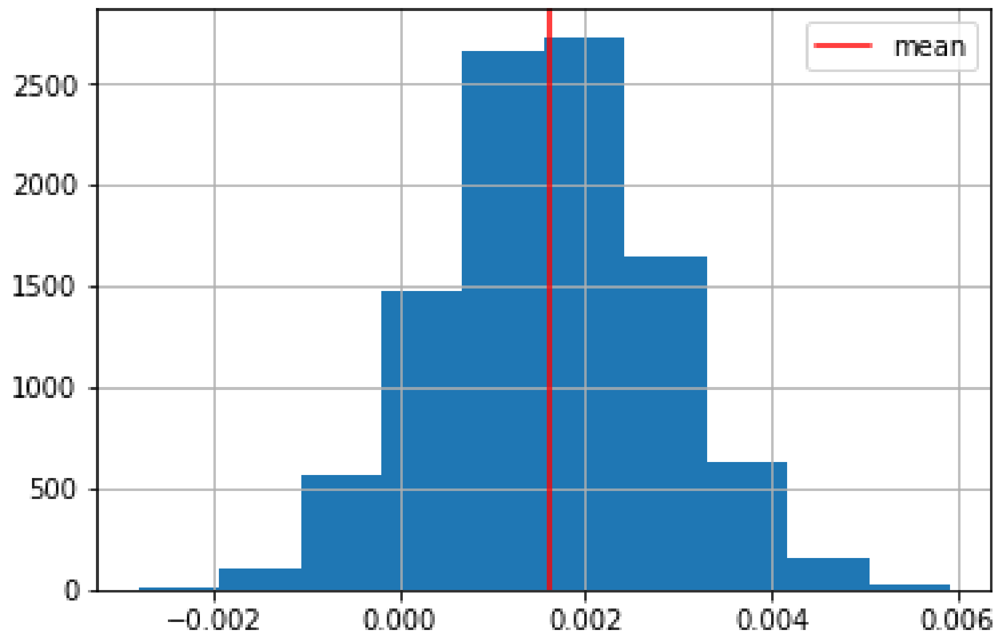
h. Create 10,000 p_{new} - p_{old} values using the same simulation process you used in parts (a) through (g) above. Store all 10,000 values in a NumPy array called **p_diffs**.

```
In [67]: p_diffs = []
         for _ in range(10000):
             new_page_converted = np.random.binomial(1,p_new,n_new)  # bootstrapping
             old_page_converted = np.random.binomial(1,p_old,n_old) # bootstrapping
             p_diffs.append(new_page_converted.mean() - old_page_converted.mean())
```

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 [68]: # convert p_diffs to numpy array for array based computations in the future
         p_diffs = np.array(p_diffs)

         # histogram
         plt.hist(p_diffs)
         plt.grid()
         plt.axvline(p_diffs.mean(), color='r', label='mean')
         plt.legend();
```



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

```
In [69]: actual_diff = df2.converted[df2.group == 'treatment'].mean() - df2.converted[df2.group
         (actual_diff < p_diffs).mean()
```

```
Out[69]: 0.9958000000000002
```

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?

What we computed in part j is called p-value in scientific studies.... p-value is the probability of observing your statistic (or one more extreme in favor of the alternative) if the null hypothesis is true.... In our case the p-value is big enough 99% that we can confidently say that we have evidence to reject 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 [47]: import statsmodels.api as sm
```

```
convert_old = df2[df2.group=='control'].converted.sum()
convert_new = df2[df2.group=='treatment'].converted.sum()
n_old = df2[df2.group=='control'].converted.count()
n_new = df2[df2.group=='treatment'].converted.count()
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[47]: (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 [48]: z_score, p_value = sm.stats.proportions_ztest([convert_old, convert_new], [n_old, n_new
         z_score, p_value
```

```
Out[48]: (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.**?

By searching in google about the definition of z-score we find that z-score is the number of standard deviations from the mean a data point is, But more technically it's a measure of how many standard deviations below or above the population mean a raw score is. So, The p-value is round 0.9 which is correspond to the p-value we calculated above in j which is round 0.9, so for this p-value (0.9) we can say we have a stronge evidence to reject the null hypothesis and our result here is nearly correspond from results in j.

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?

This is a case of Logistic Regression, In this scenario, we want to predict something that has only two possible outcomes.

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 [49]: df3 = df2 # copy dataframe in case of any mistakes
         df3['intercept'] = pd.Series(np.zeros(len(df3)), index=df3.index)
         df3['ab_page'] = pd.Series(np.zeros(len(df3)), index=df3.index)
         # Find indexes that need to be changed for treatment group
```

```
change_index = df3[df3['group']=='treatment'].index

# Change values
df3.set_value(index=change_index, col='ab_page', value=1)
df3.set_value(index=df3.index, col='intercept', value=1)

# Change datatype
df3[['intercept', 'ab_page']] = df3[['intercept', 'ab_page']].astype(int)

# Move "converted" to the end
df3 = df3[['user_id', 'timestamp', 'group', 'landing_page', 'ab_page', 'intercept', 'co
```

```
In [50]: # view the new datasat after adding the two columns ab_page and intercept
         df3[df3['group']=='treatment'].head()
```

```
Out[50]:    user_id                   timestamp      group landing_page  ab_page  \
         2   661590  2017-01-11 16:55:06.154213  treatment     new_page        1
         3   853541  2017-01-08 18:28:03.143765  treatment     new_page        1
         6   679687  2017-01-19 03:26:46.940749  treatment     new_page        1
         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

            intercept  converted
         2          1          0
         3          1          0
         6          1          1
         8          1          1
         9          1          1
```

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 [51]: # Set up logistic regression
         logit = sm.Logit(df3['converted'], df3[['ab_page', 'intercept']])

         # Calculate results
         result=logit.fit()
```

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

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

```
In [52]: result.summary2()
```

Out[52]: <class 'statsmodels.iolib.summary2.Summary'>

```
"""
                            Results: Logit
==================================================================
Model:              Logit            Pseudo R-squared: 0.000
Dependent Variable: converted        AIC:              212780.3502
Date:               2019-01-21 08:49 BIC:              212801.5095
No. Observations:   290584           Log-Likelihood:   -1.0639e+05
Df Model:           1                LL-Null:          -1.0639e+05
Df Residuals:       290582           LLR p-value:      0.18988
Converged:          1.0000           Scale:            1.0000
No. Iterations:     6.0000
-------------------------------------------------------------------
              Coef.   Std.Err.      z      P>|z|    [0.025   0.975]
-------------------------------------------------------------------
ab_page      -0.0150    0.0114    -1.3109  0.1899  -0.0374   0.0074
intercept    -1.9888    0.0081  -246.6690  0.0000  -2.0046  -1.9730
==================================================================

"""
```

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**?

As we saw above, the p-value associated with ab_page is 0.1899, which is slightly lower than the p-value we calculated using the z-score. It differ from the value we found in "Part II,j" I think adding intercept column which is meant to account the error is the reason why the p-value is lower, But this p-value is still low to reject the null hypothesis.

The null and alternative hypothesis associated with regression model will be as following: H_0 : p_{old} - p_{new}= 0 ------- H_1 : p_{old} - p_{new} != 0

f. Now, you are considering other things that might influence whether or not an individual converts. Discuss why it is a good idea to consider other factors to add into your regression model. Are there any disadvantages to adding additional terms into your regression model?

There can be many other factors that can be taken into consideration to add into our regression model.

- One of the first to consider would be the duration.if the duration is Too short, it would be advisable to increase the it.
- Geographic location is another important factor. If the page is available in multiple languages.
- Parameters like click through rate is another factor to consider.

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 [53]: # read file and join the dataframes
         countries_df = pd.read_csv('countries.csv')
         df_new = countries_df.set_index('user_id').join(df2.set_index('user_id'), how='inner')
```

```
In [54]: # check rows
         df_new.head()
```

```
Out[54]:         country                   timestamp      group landing_page  \
         user_id
         834778       UK  2017-01-14 23:08:43.304998    control     old_page
         928468       US  2017-01-23 14:44:16.387854  treatment     new_page
         822059       UK  2017-01-16 14:04:14.719771  treatment     new_page
         711597       UK  2017-01-22 03:14:24.763511    control     old_page
         710616       UK  2017-01-16 13:14:44.000513  treatment     new_page

                  converted  intercept  ab_page
         user_id
         834778           0          1        0
         928468           0          1        1
         822059           1          1        1
         711597           0          1        0
         710616           0          1        1
```

```
In [55]: # Create the necessary dummy variables
         df_new[['canada','uk','us']] = pd.get_dummies(df_new['country'])
```

```
In [56]: # let's consider canada being our baseline, therefore, we drop canada
         df_new.drop(['canada'], axis=1, inplace=True)
```

```
In [57]: df_new.head()
```

```
Out[57]:         country                   timestamp      group landing_page  \
         user_id
         834778       UK  2017-01-14 23:08:43.304998    control     old_page
         928468       US  2017-01-23 14:44:16.387854  treatment     new_page
         822059       UK  2017-01-16 14:04:14.719771  treatment     new_page
         711597       UK  2017-01-22 03:14:24.763511    control     old_page
         710616       UK  2017-01-16 13:14:44.000513  treatment     new_page

                  converted  intercept  ab_page  uk  us
         user_id
         834778           0          1        0   1   0
         928468           0          1        1   0   1
         822059           1          1        1   1   0
         711597           0          1        0   1   0
         710616           0          1        1   1   0
```

In [58]:
```
# Create logit_countries object
logit_countries = sm.Logit(df_new['converted'],
                           df_new[['uk', 'us', 'intercept']])

# Fit
result2 = logit_countries.fit()
```

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

In [59]:
```
# Show results
result2.summary2()
```

Out[59]:
```
<class 'statsmodels.iolib.summary2.Summary'>
"""
                          Results: Logit
=================================================================
Model:              Logit            Pseudo R-squared: 0.000
Dependent Variable: converted        AIC:              212780.8333
Date:               2019-01-21 08:54 BIC:              212812.5723
No. Observations:   290584           Log-Likelihood:   -1.0639e+05
Df Model:           2                LL-Null:          -1.0639e+05
Df Residuals:       290581           LLR p-value:      0.19835
Converged:          1.0000           Scale:            1.0000
No. Iterations:     6.0000
-------------------------------------------------------------------
               Coef.  Std.Err.     z     P>|z|    [0.025   0.975]
-------------------------------------------------------------------
uk             0.0507   0.0284   1.7863  0.0740  -0.0049   0.1064
us             0.0408   0.0269   1.5178  0.1291  -0.0119   0.0935
intercept     -2.0375   0.0260 -78.3639  0.0000  -2.0885  -1.9866
=================================================================

"""
```

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 [60]:
```
# Create logit_countries object
logit_countries = sm.Logit(df_new['converted'],
                           df_new[['ab_page','uk', 'us', 'intercept']])

# Fit
result2 = logit_countries.fit()
```

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

In [61]:
```
# Show results
result2.summary2()
```

Out[61]:
```
<class 'statsmodels.iolib.summary2.Summary'>
"""
                          Results: Logit
==================================================================
Model:              Logit            Pseudo R-squared: 0.000
Dependent Variable: converted        AIC:              212781.1253
Date:               2019-01-21 08:55 BIC:              212823.4439
No. Observations:   290584           Log-Likelihood:   -1.0639e+05
Df Model:           3                LL-Null:          -1.0639e+05
Df Residuals:       290580           LLR p-value:      0.17599
Converged:          1.0000           Scale:            1.0000
No. Iterations:     6.0000
-------------------------------------------------------------------
               Coef.  Std.Err.     z      P>|z|   [0.025   0.975]
-------------------------------------------------------------------
ab_page       -0.0149   0.0114   -1.3069  0.1912  -0.0374   0.0075
uk             0.0506   0.0284    1.7835  0.0745  -0.0050   0.1063
us             0.0408   0.0269    1.5161  0.1295  -0.0119   0.0934
intercept     -2.0300   0.0266  -76.2488  0.0000  -2.0822  -1.9778
==================================================================

"""
```

In [62]:
```
1/np.exp(0.0506), np.exp(0.0408)
```

Out[62]:
```
(0.9506588580330709З, 1.0416437559600236)
```

0.3 conclusion

Although it would seem from the outset that there is a difference between the conversion rates of new and old pages, there is just not enough evidence to reject the null hypothesis. From the histogram shown in this report, it seems that the new page does worse than the old pa.

It was also found that this was not dependent on countries with conversion rates being roughly the same in the UK as in the US. The test conditions were fairly good as well, users had a roughly 50% chance to recieve the new and old pages and the sample size of the initial dataframe is sufficiently big such that collecting data is likely not a good use of resources.

we would recommend that the e-commerce company spend their money on trying to improve their website before trying again.

- users from uk are 0.95 times more likely to convert.
- users from us are 1.04 times more likely to convert.

- So when adding everything together it seems that the p-values for all featues has increased.

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