Analyze_ab_test_results_notebook

May 1, 2021

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 [2]: 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 [3]: df = pd.read_csv('ab_data.csv')
        df.head()
```

```
Out[3]:    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 [4]: df.shape
```

```
Out[4]: (294478, 5)
```

c. The number of unique users in the dataset.

```
In [5]: df['user_id'].nunique()
```

```
Out[5]: 290584
```

d. The proportion of users converted.

```
In [6]: converted_prop = df[df['converted']==1].shape[0]/df.shape[0]
        converted_prop
```

```
Out[6]: 0.11965919355605512
```

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

```
In [7]: df.query('group == "treatment" and landing_page != "new_page"').shape[0] + df.query('gro
```

```
Out[7]: 3893
```

f. Do any of the rows have missing values?

```
In [8]: df.info()
```

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

2. For the rows where **treatment** 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 [15]: x = df.query('group == "treatment" and landing_page == "new_page"')
         y = df.query('group == "control" and landing_page == "old_page"')
         df2 = x.append(y)
         df2.head()
```

```
Out[15]:    user_id                   timestamp      group landing_page  converted
         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
         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
```

```
In [17]: df2.shape
```

```
Out[17]: (290585, 5)
```

```
In [10]: df2.groupby(['landing_page', 'group']).count()
```

```
Out[10]:                         user_id  timestamp  converted
         landing_page group
         new_page     treatment   145311     145311     145311
         old_page     control     145274     145274     145274
```

```
In [20]: # Double Check all of the correct rows were removed - this should be 0
         df2[((df2['group'] == 'treatment') == (df2['landing_page'] == 'new_page')) == False].sh
```

```
Out[20]: 0
```

```
In [21]: df2.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 290585 entries, 2 to 294476
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
```

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

```
Out[22]: 290584
```

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

```
In [26]: df2[df2.duplicated('user_id')]
```

```
Out[26]:       user_id                   timestamp      group landing_page  converted
         2893   773192  2017-01-14 02:55:59.590927  treatment     new_page          0
```

```
In [27]: df2[df2['user_id']== 773192]
```

```
Out[27]:       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
```

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

```
In [32]: df2[df2['user_id']== 773192]
```

```
Out[32]:       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 [33]: df2.drop(index=2893, inplace=True)
```

`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 [34]: conversion_prob = df2[df2['converted']==1].shape[0]/df2.shape[0]
         conversion_prob
```

```
Out[34]: 0.11959708724499628
```

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

```
In [35]: control_conversion_prob = df2.query('group == "control" and converted == 1').shape[0]/\
                                   df2[df2['group']=='control'].shape[0]
         control_conversion_prob
```

```
Out[35]: 0.1203863045004612
```

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

```
In [36]: treatment_conversion_prob = df2.query('group == "treatment" and converted == 1').shape[
                                     df2[df2['group']=='treatment'].shape[0]
         treatment_conversion_prob
```

```
Out[36]: 0.11880806551510564
```

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

```
In [37]: new_page_prob = df2[df2['landing_page']=='new_page'].shape[0]/df2.shape[0]
         new_page_prob
```

```
Out[37]: 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.

No, because we can't conclude the new page leads to more conversation just based on probabilities. per answer in d, there is 50% of users can receive new page on the other side, 50% of users receive old page. The difference is not significant.

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.

Null Hypothesis : $p_{old} >= p_{new}$
Alternate Hypothesis: $p_{old} < p_{new}$

```
In [ ]:
```

```
In [ ]:
```

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

Out[40]: 0.11959708724499628
```

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

```
In [41]: p_old = df2['converted'].mean()
         p_old

Out[41]: 0.11959708724499628
```

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

```
In [42]: n_new = df2[df2['group']=='treatment'].shape[0]
         n_new

Out[42]: 145310
```

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

```
In [43]: n_old = df2[df2['group']=='control'].shape[0]
         n_old

Out[43]: 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 [87]: new_page_converted = np.random.binomial(n_new,p_new)
         p_new_page = new_page_converted/n_new
         p_new_page

Out[87]: 0.8843025256348497

In [ ]:
```

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 [141]: old_page_converted = np.random.binomial(n_old, p_old)
          p_old_page = old_page_converted/n_old
          p_old_page

Out[141]: 0.8811969106653634
```

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

```
In [142]: obs_diff = p_new_page - p_old_page
          obs_diff
```

```
Out[142]: 0.003105614969486248
```

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**.

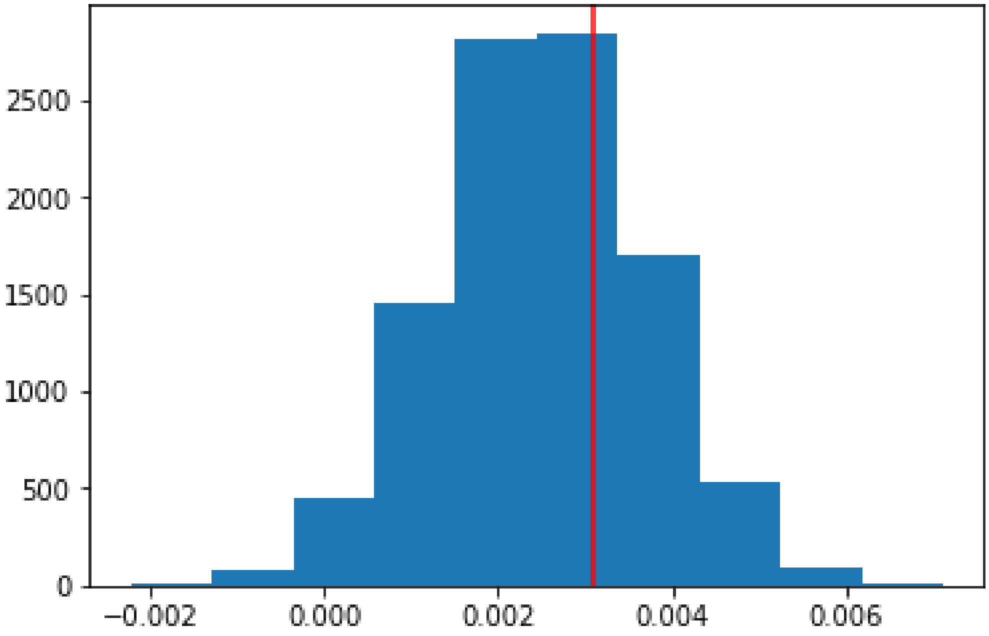
```
In [143]: p_diffs = []
          new_page_converted = np.random.binomial(n_new,p_new, 10000)/n_new
          old_page_converted = np.random.binomial(n_old, p_old, 10000)/n_old
          diff = new_page_converted - old_page_converted
          p_diffs.append(diff)
          p_diffs
```

```
Out[143]: [array([ 0.00396588,  0.0033122 ,  0.00119256, ...,  0.00416552,
                   0.00073806,  0.00117872])]
```

```
In [ ]:
```

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 [144]: np.array(p_diffs)
          plt.hist(p_diffs);
          plt.axvline(x=obs_diff, color = 'red')
          plt.show()
```



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

```
In [152]: # Calculate the actucl difference observed in ab_data

          old_mean_act = df.query('group =="control"').converted.mean()
          new_mean_act = df.query('group =="treatment"').converted.mean()
          act_diff = new_mean_act - old_mean_act
```

```
In [153]: (p_diffs > act_diff).mean()
```

```
Out[153]: 0.99950000000000006
```

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?

The P value is higher the type I error rate of 0.05 [95% confindence interval]. Hence, There is no evidence to reject the null hypothesis. i.e., we fail 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 [155]: import statsmodels.api as sm

          convert_old = len(df2[(df2['group']=='control') & (df2['converted']==1)])
          convert_new = len(df2[(df2['group']=='treatment') & (df2['converted']==1)])
          n_old = len(df2[df2['group']=='control'])
          n_new = len(df2[df2['group']=='treatment'])
          convert_old, convert_new, n_old, n_new
          #(17489, 17264, 145274, 145311)
```

```
Out[155]: (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 [161]: z_score, p_value = sm.stats.proportions_ztest([convert_old, convert_new], [n_old,n_new
          z_score, p_value
```

```
Out[161]: (1.3109241984234394, 0.90505831275902449)
```

```
In [164]: # testing the significance of z_score
          from scipy.stats import norm
          norm.cdf(z_score)
```

```
Out[164]: 0.90505831275902449
```

```
In [165]: #Assuming 95% CI for one-sided test, as stated in part II.1

          norm.ppf(1-0.05)

Out[165]: 1.6448536269514722
```

n. What do the z-score and p-value you computed in the previous question mean for the conversion rates of the old and new pages? Do they agree with the findings in parts **j.** and **k.**?

Z_score is less than 1.6448 and p value is 0.90, here indicates we do not have an evidence to reject the null hypothesis. So, it agrees with parts J and K
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?

Put your answer here.
Since there are only two outcomes, logistic regression is the best option to predict.

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 [166]: import statsmodels.api as sm

In [171]: df2 = df.copy()
          df2['intercept'] = 1
          df2[['old_page', 'new_page']] = pd.get_dummies(df2['landing_page'])
          df2['ab_page'] = pd.get_dummies(df2['group'])['treatment']
          df2.head()

Out[171]:    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  old_page  new_page  ab_page
          0          1         0         1        0
          1          1         0         1        0
          2          1         1         0        1
          3          1         1         0        1
          4          1         0         1        0
```

```
In [168]: df2 = df2.drop('control', axis=1)
          df2.head()
```

```
Out[168]:    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  old_page  new_page  ab_page
          0          1         0         1        0
          1          1         0         1        0
          2          1         1         0        1
          3          1         1         0        1
          4          1         0         1        0
```

c. Use **statsmodels** to 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 [177]: lm = sm.Logit(df['converted'], df2[['intercept',  'ab_page']])
          results = lm.fit()
          results.summary2()
```

```
Optimization terminated successfully.
         Current function value: 0.366243
         Iterations 6
```

```
Out[177]: <class 'statsmodels.iolib.summary2.Summary'>
          """
                                    Results: Logit
          ==================================================================
          Model:              Logit            No. Iterations:   6.0000
          Dependent Variable: converted        Pseudo R-squared: 0.000
          Date:               2021-04-28 08:13 AIC:              215704.9004
          No. Observations:   294478           BIC:              215726.0864
          Df Model:           1                Log-Likelihood:   -1.0785e+05
          Df Residuals:       294476           LL-Null:          -1.0785e+05
          Converged:          1.0000           Scale:            1.0000
          -------------------------------------------------------------------
                        Coef.   Std.Err.      z      P>|z|    [0.025   0.975]
          -------------------------------------------------------------------
          intercept    -1.9887    0.0080  -248.2967  0.0000  -2.0044  -1.9730
          ab_page      -0.0140    0.0114    -1.2369  0.2161  -0.0363   0.0082
          ==================================================================

          """
```

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

```
In [201]: np.exp(-0.0140)
```

```
Out[201]: 0.98609754426286189
```

Holding other variables constant, for every unit increase in the ab_page, there is 1.014 times changes, people like to be converted

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**?

H0: pnewpold=0 H1: pnewpold0
P-Value associated with ab_page is 0.2161 which is lower than the p-value found in Part II because, in part II, we performed a one-sided test, where in the logistic regression part, it is two-sided test.

```
In [ ]:
```

```
In [ ]:
```

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?

Considering other factors is a good idea as these factors may contribute to the significance of our test results and leads to more accurate decisions. One of the disadvantages of adding additional terms into the regression model is Simpson's paradox where the combined impact of different variables disappears or reverses when these variables are combined, but appears where these variables are tested individually.

Additionally, adding additional factors would also lead to multi-collinearity which can be found by using VIF's. VIF's provide a greater advantage in finding the high collinear factors to drop from the model.

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 [179]: df_country = pd.read_csv('countries.csv')
          df_country.head()
```

```
Out[179]:    user_id country
          0   834778      UK
          1   928468      US
          2   822059      UK
          3   711597      UK
          4   710616      UK
```

```
In [180]: df_new = df2.merge(df_country)
          df_new.head()
```

```
Out[180]:    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  old_page  new_page  ab_page country
          0          1         0         1        0      US
          1          1         0         1        0      US
          2          1         1         0        1      US
          3          1         1         0        1      US
          4          1         0         1        0      US
```

```
In [181]: df_new['country'].unique()
```

```
Out[181]: array(['US', 'CA', 'UK'], dtype=object)
```

```
In [182]: df_new[['CA', 'UK', 'US']] = pd.get_dummies(df_new['country'])
          df_new.head()
```

```
Out[182]:    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  old_page  new_page  ab_page country  CA  UK  US
          0          1         0         1        0      US   0   0   1
          1          1         0         1        0      US   0   0   1
          2          1         1         0        1      US   0   0   1
          3          1         1         0        1      US   0   0   1
          4          1         0         1        0      US   0   0   1
```

```
In [190]: df_new.query('CA == "1"').converted.mean(), df_new.query('UK == "1"').converted.mean()
```

```
Out[190]: (0.11588975842123171, 0.12058186572957953, 0.11959934872361458)
```

```
In [194]: lm = sm.Logit(df_new['converted'], df_new[['intercept', 'UK', 'US']])
          results2 = lm.fit()
          results2.summary2()

Optimization terminated successfully.
         Current function value: 0.366241
         Iterations 6


Out[194]: <class 'statsmodels.iolib.summary2.Summary'>
          """
                                    Results: Logit
          ==================================================================
          Model:              Logit            No. Iterations:   6.0000
          Dependent Variable: converted        Pseudo R-squared: 0.000
          Date:               2021-04-28 09:05 AIC:              215705.8310
          No. Observations:   294478           BIC:              215737.6099
          Df Model:           2                Log-Likelihood:   -1.0785e+05
          Df Residuals:       294475           LL-Null:          -1.0785e+05
          Converged:          1.0000           Scale:            1.0000
          -------------------------------------------------------------------
                         Coef.   Std.Err.     z      P>|z|    [0.025   0.975]
          -------------------------------------------------------------------
          intercept     -2.0319    0.0258  -78.8446  0.0000  -2.0825  -1.9814
          UK             0.0450    0.0282    1.5988  0.1099  -0.0102   0.1002
          US             0.0357    0.0266    1.3401  0.1802  -0.0165   0.0879
          ==================================================================

          """
```

```
In [196]: np.exp(0.0450), np.exp(0.0357)

Out[196]: (1.0460278599087169, 1.0363448963818249)
```

The P values associated with UK and US are low which indicates that these two factors are statisically significant. However, exponential values of the corresponding coeeficients say that there is not a significant difference in the conversion across counties.

```
In [205]: df_new['CA_ab_page'] = df_new['ab_page'] * df_new['CA']
          df_new['UK_ab_page'] = df_new['ab_page'] * df_new['UK']
          df_new['US_ab_page'] = df_new['ab_page'] * df_new['US']
          df_new.head()

Out[205]:    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  old_page  new_page  ab_page country  CA  UK  US  CA_ab_page  \
0          1         0         1        0      US   0   0   1           0
1          1         0         1        0      US   0   0   1           0
2          1         1         0        1      US   0   0   1           0
3          1         1         0        1      US   0   0   1           0
4          1         0         1        0      US   0   0   1           0

   UK_ab_page  US_ab_page
0           0           0
1           0           0
2           0           1
3           0           1
4           0           0
```

```
In [207]: lm = sm.Logit(df_new['converted'], df_new[['intercept', 'UK', 'US', 'UK_ab_page', 'US_
          results3 = lm.fit()
          results3.summary2()
```

```
Optimization terminated successfully.
         Current function value: 0.366238
         Iterations 6
```

```
Out[207]: <class 'statsmodels.iolib.summary2.Summary'>
          """
                                    Results: Logit
          ==================================================================
          Model:              Logit            No. Iterations:   6.0000
          Dependent Variable: converted        Pseudo R-squared: 0.000
          Date:               2021-04-28 09:31 AIC:              215707.9495
          No. Observations:   294478           BIC:              215760.9143
          Df Model:           4                Log-Likelihood:   -1.0785e+05
          Df Residuals:       294473           LL-Null:          -1.0785e+05
          Converged:          1.0000           Scale:            1.0000
          -------------------------------------------------------------------
                         Coef.   Std.Err.    z      P>|z|    [0.025   0.975]
          -------------------------------------------------------------------
          intercept     -2.0319    0.0258 -78.8446  0.0000  -2.0825  -1.9814
          UK             0.0413    0.0303   1.3620  0.1732  -0.0181   0.1008
          US             0.0447    0.0275   1.6272  0.1037  -0.0091   0.0986
          UK_ab_page     0.0074    0.0227   0.3254  0.7449  -0.0370   0.0518
          US_ab_page    -0.0181    0.0136  -1.3325  0.1827  -0.0447   0.0085
          ==================================================================

          """
```

```
In [208]: print (np.exp(0.0413), np.exp(0.0447), np.exp(0.0074), np.exp(-0.0181))
```

```
1.04216470807 1.04571409862 1.00742744766 0.982062821166
```

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 at country level as well as interactions.

Based on the above results, it appears that there is no significant differene in the conversion of users across the counties.

Conculusion:

Based on the statistica tests we have performed, the Z-test, Logistic Regression model and the actual difference abserved, the results showed that the old page and new page have equal probilities approximately of getting users converted. Hence, we fail to reject the null hypothesis. I would recommend the e-commece company to keep the old page since it will save huge money on development of new page.

```
In [ ]:
```

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.

0.3 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 [38]: from subprocess import call
         call(['python', '-m', 'nbconvert', 'Analyze_ab_test_results_notebook.ipynb'])

Out[38]: 0
```