Analyze_ab_test_results_notebook

May 7, 2020

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 [4]: df=pd.read_csv("ab_data.csv")
        df.head()
```

```
Out[4]:    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 [5]: df.shape[0]
```

```
Out[5]: 294478
```

c. The number of unique users in the dataset.

```
In [6]: df.nunique()
```

```
Out[6]: user_id         290584
        timestamp       294478
        group                2
        landing_page         2
        converted            2
        dtype: int64
```

d. The proportion of users converted.

```
In [7]: len(df.query("converted==1"))/df.shape[0]
```

```
Out[7]: 0.11965919355605512
```

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

```
In [8]: grp1=len(df.query('group!="treatment"and landing_page=="new_page"'))
        grp2=len(df.query('group!="control"and landing_page=="old_page"'))
        final_grp=grp1+grp2
        final_grp
```

```
Out[8]: 3893
```

f. Do any of the rows have missing values?

```
In [9]: 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 [10]: # Now we copying the dataframe
         df2=df
```

```
In [11]: # dataframe where where treatment is not aligned with new_page or control is not aligne
         df2 = df[((df.group=='treatment') & (df.landing_page=='new_page')) | ((df.group=='contr
```

```
In [12]: # 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[12]: 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 [13]: df2.nunique()
```

```
Out[13]: user_id         290584
         timestamp       290585
         group                2
         landing_page         2
         converted            2
         dtype: int64
```

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

```
In [14]: df2[df2.user_id.duplicated()]
```

```
Out[14]:       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 [15]: df2[df2.user_id .duplicated()]
```

```
Out[15]:       user_id                   timestamp      group landing_page  converted
         2893   773192  2017-01-14 02:55:59.590927  treatment     new_page          0
```

d. Remove **one** of the rows with a duplicate **user_id**, but keep your dataframe as **df2**.

```
In [16]: df2=df2.drop_duplicates()
```

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 [17]: df.converted.mean()
```

```
Out[17]: 0.11965919355605512
```

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

```
In [18]: control_group = len(df2.query('group=="control" and converted==1'))/len(df2.query('grou
         control_group
```

```
Out[18]: 0.1203863045004612
```

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

```
In [19]: treatment = len(df2.query('group=="treatment" and converted==1'))/len(df2.query('group=
         treatment
```

```
Out[19]: 0.11880724790277405
```

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

```
In [34]: len(df2.query('landing_page=="new_page"'))/df2.shape[0]
```

```
Out[34]: 0.5000636646764286
```

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.

Your answer goes here.
Probability of individual converting given individual is in control group is 0.1203863045004612. Probability of individual converting given individual is in treatment group is 0.11880724790277405. According to the analysis this is clear that there is no more conversion between new page and old page. As the converting rate is similar in both cases so it is important to consider other factors.
### 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.

```
H1:pnew-pold>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 **conversion rate** for p_{new} under the null?

```
In [35]: p_new=len(df2.query("converted==1"))/df2.shape[0]
         p_new

Out[35]: 0.11959667567149027
```

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

```
In [36]: p_old=len(df2.query("converted==1"))/df2.shape[0]
         p_old

Out[36]: 0.11959667567149027
```

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

```
In [37]: n_new=len(df.query('group=="treatment"'))
         n_new

Out[37]: 147276
```

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

```
In [38]: n_old=len(df.query('group=="control"'))
```

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 [39]: new_page_converted=np.random.choice([0,1],n_new,p = [p_new, 1-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 [26]: old_page_converted=np.random.choice([0,1],n_old,p = [p_new, 1-p_new])
```

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

```
In [27]: obs_dif=new_page_converted.mean()-old_page_converted.mean()
         obs_dif

Out[27]: -0.00066018707428794343
```

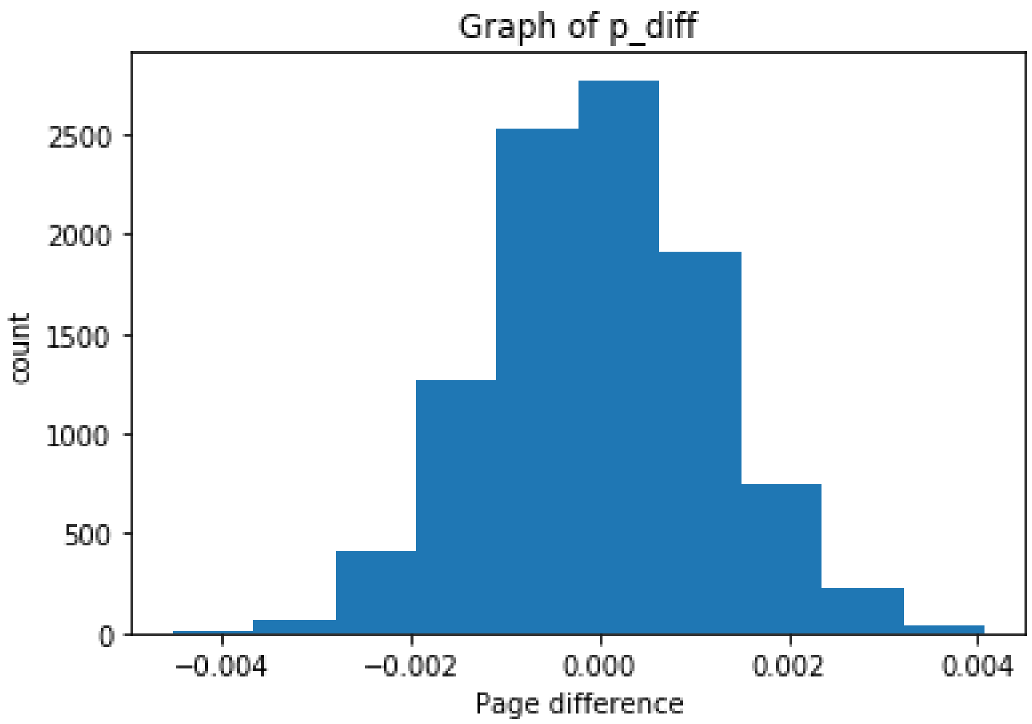
h. Create 10,000 p_{new} - p_{old} values using the same simulation process you used in parts (a) through (g) above. Store all 10,000 values in a NumPy array called **p_diffs**.

```
In [28]: diff=[]
         for i in range(10000):
             n_new1=np.random.choice([0,1],n_new,p = [p_new, 1-p_new])
             n_old1=np.random.choice([0,1],n_old,p = [p_new, 1-p_new])
             dif=n_new1.mean()-n_old1.mean()
             diff.append(dif)
```

```
In [29]: np.array(diff)

Out[29]: array([ -1.29792225e-03,  -1.58332638e-03,   1.15382447e-03, ...,
                  9.41142912e-05,   1.31363888e-05,  -8.77268220e-04])
```

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 [41]: plt.hist(diff)
         plt.title("Graph of p_diff")
         plt.xlabel("Page difference")
         plt.ylabel("count")

Out[41]: Text(0,0.5,'count')
```



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

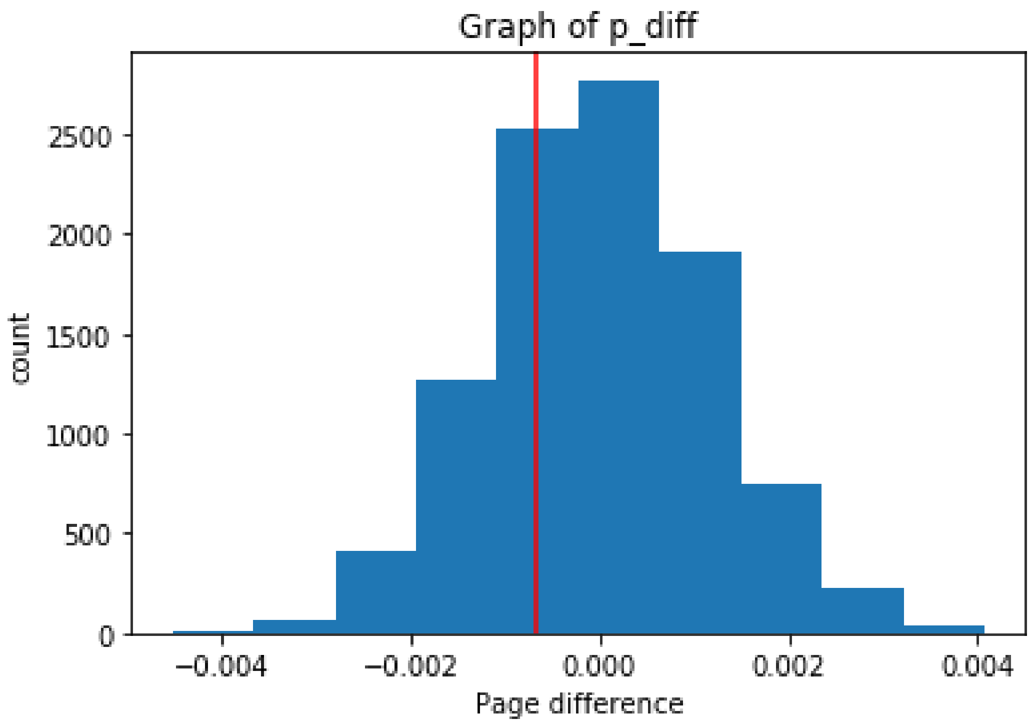
```
In [49]: var1 = df2[df2['landing_page'] == 'new_page']
         var1=var1['converted'].mean()
         var2 = df2[df2['landing_page'] == 'old_page']
         var2 = var2['converted'].mean()
         actual_diff = var1-var2
         count = 0
         for i in diff:
             if i> actual_diff:
                 count = count+1

         print (count/(len(diff)))
```

```
0.9098
```

```
In [ ]:
```

```
In [45]: plt.hist(diff)
         plt.axvline(x=obs_dif,color="r")
         plt.title("Graph of p_diff")
         plt.xlabel("Page difference")
         plt.ylabel("count")
```

```
Out[45]: Text(0,0.5,'count')
```



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?

Put your answer here.

The value calculated is called p-value. For accepting null hypothesis p-value should be greater than suggested p-value. We calculate that almost 91% of the population is above the real diffrence which suggested that new-page is not doing significantly better than the old page. New page is worse than old page, so we should stick to the null hyposthesis as p-value is large.

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 [53]: import statsmodels.api as sm

         convert_old =len(df2.query('converted==1 and landing_page=="old_page"'))
         convert_new = len(df2.query('converted==1 and landing_page=="new_page"'))
```

```
        n_old = len(df2.query('landing_page=="old_page"'))
        n_new = len(df2.query('landing_page=="new_page"'))
        n_new
```

```
Out[53]: 145311
```

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 [59]: z_score,p_val=sm.stats.proportions_ztest([convert_old,convert_new],[n_old,n_new],altern
         z_score,p_val
```

```
Out[59]: (1.3116075339133115, 0.90517370514059103)
```

```
In [63]: from scipy.stats import norm
         norm.cdf(z_score)#how significant our z_score is
```

```
Out[63]: 0.90517370514059103
```

```
In [64]: norm.ppf(1-(0.05)) #critical value of 95% confidence
```

```
Out[64]: 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.**?

Put your answer here.
z_score is less than critical value of 95% confidence. Hence we fail to reject null hypothesis. Therefore the conclusion is same as part j that we accept null hypothesis.
### 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.
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 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 [66]: df2['intercept']=1
```

```
In [69]: df2['ab_page']=pd.get_dummies(df2['group'])['treatment']
         df2.head()
```

```
Out[69]:    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  page  ab_page
        0          1     0        0
        1          1     0        0
        2          1     1        1
        3          1     1        1
        4          1     0        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 [75]: import statsmodels.api as sm
         lm=sm.Logit(df2['converted'],df2[['intercept','ab_page']])
         res=lm.fit()

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

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

```
In [76]: res.summary2()

Out[76]: <class 'statsmodels.iolib.summary2.Summary'>
         """
                                   Results: Logit
         ==================================================================
         Model:              Logit            No. Iterations:   6.0000
         Dependent Variable: converted        Pseudo R-squared: 0.000
         Date:               2020-05-07 05:31 AIC:              212780.6032
         No. Observations:   290585           BIC:              212801.7625
         Df Model:           1                Log-Likelihood:   -1.0639e+05
         Df Residuals:       290583           LL-Null:          -1.0639e+05
         Converged:          1.0000           Scale:            1.0000
         -------------------------------------------------------------------
                       Coef.   Std.Err.      z      P>|z|    [0.025   0.975]
         -------------------------------------------------------------------
         intercept    -1.9888    0.0081  -246.6690  0.0000  -2.0046  -1.9730
         ab_page      -0.0150    0.0114    -1.3116  0.1897  -0.0374   0.0074
         ==================================================================
"""
```

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**?

Put your answer here.
In Logistic regression H1:pnew-pold!=0
IN part2 H0:pnew-pold<=0 H1:pnew-pold>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?

Put your answer here.
Additional factors should be added into the regression models they may also influence the conversions also. The disadvantage is that we don't know that our additional factor will influence the result in which direction. As our additional factor changes every time on the basis of an additional factor.

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 [78]: df_3=pd.read_csv("countries.csv")
         df_3.head()

Out[78]:    user_id country
         0   834778      UK
         1   928468      US
         2   822059      UK
         3   711597      UK
         4   710616      UK

In [82]: df_new=df_3.set_index("user_id").join(df2.set_index("user_id"),how="inner")
         df_new.head()

Out[82]:         country                  timestamp      group landing_page  \
         user_id
         630000       US  2017-01-19 06:26:06.548941  treatment     new_page
         630001       US  2017-01-16 03:16:42.560309  treatment     new_page
         630002       US  2017-01-19 19:20:56.438330    control     old_page
         630003       US  2017-01-12 10:09:31.510471  treatment     new_page
```

```
630004       US  2017-01-18 20:23:58.824994  treatment     new_page

         converted  intercept  page  ab_page
user_id
630000           0          1     1        1
630001           1          1     1        1
630002           0          1     0        0
630003           0          1     1        1
630004           0          1     1        1
```

```
In [87]: df_new[['US','UK']]=pd.get_dummies(df_new['country'])[['US','UK']]
         df_new.head()
```

```
Out[87]:         country                   timestamp      group landing_page  \
         user_id
         630000       US  2017-01-19 06:26:06.548941  treatment     new_page
         630001       US  2017-01-16 03:16:42.560309  treatment     new_page
         630002       US  2017-01-19 19:20:56.438330    control     old_page
         630003       US  2017-01-12 10:09:31.510471  treatment     new_page
         630004       US  2017-01-18 20:23:58.824994  treatment     new_page

                  converted  intercept  page  ab_page  US  UK
         user_id
         630000           0          1     1        1   1   0
         630001           1          1     1        1   1   0
         630002           0          1     0        0   1   0
         630003           0          1     1        1   1   0
         630004           0          1     1        1   1   0
```

```
In [89]: df_new['us_ab_page']=df_new['ab_page']*df_new['US']
         df_new.head()
```

```
Out[89]:         country                   timestamp      group landing_page  \
         user_id
         630000       US  2017-01-19 06:26:06.548941  treatment     new_page
         630001       US  2017-01-16 03:16:42.560309  treatment     new_page
         630002       US  2017-01-19 19:20:56.438330    control     old_page
         630003       US  2017-01-12 10:09:31.510471  treatment     new_page
         630004       US  2017-01-18 20:23:58.824994  treatment     new_page

                  converted  intercept  page  ab_page  US  UK  us_ab_page
         user_id
         630000           0          1     1        1   1   0           1
         630001           1          1     1        1   1   0           1
         630002           0          1     0        0   1   0           0
         630003           0          1     1        1   1   0           1
         630004           0          1     1        1   1   0           1
```

```
In [90]: df_new['uk_ab_page']=df_new['ab_page']*df_new['UK']
         df_new.head()
```

```
Out[90]:         country                  timestamp      group landing_page  \
        user_id
        630000       US  2017-01-19 06:26:06.548941  treatment     new_page
        630001       US  2017-01-16 03:16:42.560309  treatment     new_page
        630002       US  2017-01-19 19:20:56.438330    control     old_page
        630003       US  2017-01-12 10:09:31.510471  treatment     new_page
        630004       US  2017-01-18 20:23:58.824994  treatment     new_page

                  converted  intercept  page  ab_page  US  UK  us_ab_page  uk_ab_page
        user_id
        630000            0          1     1        1   1   0           1           0
        630001            1          1     1        1   1   0           1           0
        630002            0          1     0        0   1   0           0           0
        630003            0          1     1        1   1   0           1           0
        630004            0          1     1        1   1   0           1           0
```

```
In [92]: df_new['intercept']=1
         import statsmodels.api as sm
         model=sm.Logit(df_new['converted'],df_new[['intercept','us_ab_page','uk_ab_page','US','
         result=model.fit()

Optimization terminated successfully.
         Current function value: 0.366111
         Iterations 6
```

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 [93]: result.summary2()

Out[93]: <class 'statsmodels.iolib.summary2.Summary'>
         """
                                  Results: Logit
         ==================================================================
         Model:              Logit            No. Iterations:   6.0000
         Dependent Variable: converted        Pseudo R-squared: 0.000
         Date:               2020-05-07 05:50 AIC:              212782.5944
         No. Observations:   290585           BIC:              212835.4926
         Df Model:           4                Log-Likelihood:   -1.0639e+05
         Df Residuals:       290580           LL-Null:          -1.0639e+05
         Converged:          1.0000           Scale:            1.0000
         -------------------------------------------------------------------
                         Coef.   Std.Err.     z      P>|z|    [0.025   0.975]
         -------------------------------------------------------------------
         intercept     -2.0375    0.0260  -78.3639  0.0000  -2.0885  -1.9866
```

```
us_ab_page    -0.0206    0.0137   -1.5060  0.1321  -0.0474   0.0062
uk_ab_page     0.0108    0.0228    0.4749  0.6349  -0.0339   0.0555
US             0.0511    0.0277    1.8414  0.0656  -0.0033   0.1054
UK             0.0453    0.0306    1.4806  0.1387  -0.0147   0.1053
==================================================================

"""
```

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