Analyze_ab_test_results_notebook

January 5, 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 [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(3)
```

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

b. Use the below cell 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]: len(df['user_id'].unique())
```

```
Out[4]: 290584
```

d. The proportion of users converted.

```
In [5]: len(df[df['converted'] == 1])/len(df)
```

```
Out[5]: 0.11965919355605512
```

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

```
In [6]: dfn = df[df['landing_page'] == 'new_page']
        dfo = df[df['group'] == 'treatment']
        len(dfn[dfn['group'] != 'treatment'] + dfo[dfo['landing_page'] != 'new_page'])
```

```
Out[6]: 3893
```

f. Do any of the rows have missing values?

```
In [7]: # there is No row with missing data
        print(df[df.isnull().any(axis=1)])
```

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

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

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

```
In [8]: df2 = df[((df['group'] == 'treatment') == (df['landing_page'] == 'new_page'))]
        df2 = df2[((df2['group'] == 'control') == (df2['landing_page'] == 'old_page'))]
```

```
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]: print(len(df2['user_id'].unique()))
```

```
290584
```

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

```
In [11]: df2['user_id'].duplicated()
         df2.groupby('user_id').count()
         df2['user_id'].value_counts()
         #df2['user_id'].repeat
```

```
Out[11]: 773192    2
         630732    1
         811737    1
         797392    1
         795345    1
         801490    1
         799443    1
         787157    1
         793302    1
         817882    1
         842446    1
         815835    1
         805596    1
         803549    1
         809694    1
         807647    1
         895712    1
         840399    1
         836301    1
         899810    1
         834242    1
         936604    1
```

```
934557    1
940702    1
938655    1
830144    1
828097    1
832195    1
838348    1
821956    1
          ..
734668    1
736717    1
730574    1
775632    1
771538    1
642451    1
773587    1
783828    1
785877    1
779734    1
781783    1
759256    1
726472    1
748999    1
746950    1
753093    1
751044    1
740803    1
738754    1
744897    1
742848    1
634271    1
632222    1
636316    1
630169    1
650647    1
648598    1
654741    1
652692    1
630836    1
Name: user_id, Length: 290584, dtype: int64
```

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

```
In [12]: df2[df2['user_id'] == 773192]
         #print(len(df2))
```

Out[12]:

	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 [13]: df2 = df2.drop([1899])
         #print(len(df2))
```

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

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

```
In [14]: len(df2[df['converted'] == True])/len(df2)
```

```
/opt/conda/lib/python3.6/site-packages/ipykernel_launcher.py:1: UserWarning: Boolean Series key
  """Entry point for launching an IPython kernel.
```

```
Out[14]: 0.11959708724499628
```

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

```
In [15]: dfc = df2[df2['group'] == 'control']
         len(dfc[dfc['converted'] == True])/len(dfc)
```

```
Out[15]: 0.1203863045004612
```

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

```
In [16]: dft = df2[df2['group'] == 'treatment']
         len(dft[dft['converted'] == True])/len(dft)
```

```
Out[16]: 0.11880806551510564
```

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

```
In [17]: len(df2[df2['landing_page'] == 'new_page'])/len(df2)
```

```
Out[17]: 0.5000619442226688
```

e. Use the results in the previous two portions of this question to suggest if you think there is evidence that one page leads to more conversions? Write your response below.

Answer: The propability of indevedual to receive the new_page is 50%, however there is no significant difference between the two groups when it comes to the average of 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.

Answer: ### H0:p_{new} - p_{old} <= 0

0.2.1 H1:p_{new} - p_{old} > 0

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

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

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

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

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

```
In [18]: pnew=len(df2[df2['converted'] == True])/len(df2)
         print(pnew)
         #print(df2[df2['group'] == 'treatment'].converted.mean())
```

```
0.11959708724499628
```

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

```
In [19]: pold=len(df2[df2['converted'] == True])/len(df2)
         print(pold)
         #print(df2[df2['group'] == 'control'].converted.mean())
```

```
0.11959708724499628
```

c. What is n_{new}?

```
In [20]: nnew = len(df2[df2['group'] == 'treatment'])
         print(nnew)
```

```
145310
```

d. What is n_{old}?

```
In [21]: nold = len(df2[df2['group'] == 'control'])
         print(nold)
```

```
145274
```

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

```
In [22]: new_page_converted = np.random.binomial(1,pnew,nnew)
         print(new_page_converted.mean())
```

```
0.120397770284
```

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

```
In [23]: old_page_converted = np.random.binomial(1,pold,nold)
         print(old_page_converted.mean())
```

```
0.119580929829
```

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

```
In [24]: pdiff=new_page_converted.mean()-old_page_converted.mean()
         print(pdiff)
```

```
0.00081684045507
```

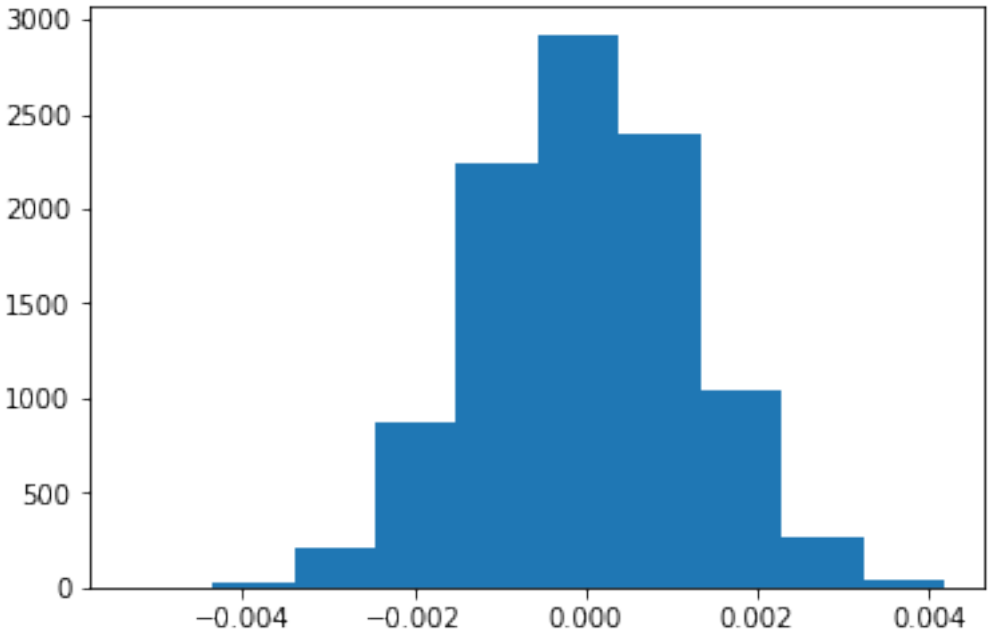
h. Simulate 10,000 p_{new} - p_{old} values using this same process similarly to the one you calculated in parts **a. through g.** above. Store all 10,000 values in **p_diffs**.

```
In [25]: p_diffs = []
         for i in range(10000):
             new_page_converted = np.random.binomial(1,pnew,nnew)
             old_page_converted = np.random.binomial(1,pold,nold)
             diff=new_page_converted.mean()-old_page_converted.mean()
             p_diffs.append(diff)
```

i. Plot a histogram of the **p_diffs**. Does this plot look like what you expected? Use the matching problem in the classroom to assure you fully understand what was computed here.

```
In [26]: plt.hist(p_diffs)
```

```
Out[26]: (array([    3.,    20.,   208.,   869.,  2234.,  2921.,  2397.,  1044.,
                   264.,    40.]),
          array([-0.00530155, -0.00435316, -0.00340477, -0.00245638, -0.00150799,
                 -0.0005596 ,  0.00038879,  0.00133718,  0.00228557,  0.00323396,
                  0.00418235]),
          <a list of 10 Patch objects>)
```



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

```
In [27]: #(p_diffs-pdiff > 0).sum()/10000
         (p_diffs > pdiff).mean()
```

```
Out[27]: 0.25190000000000001
```

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

Answer: The P-value calculted here shows a very high value that indicates a weak evidence against the null hypothsis which means that we fail to reject the null hypothsis suggesting that the existing page is better or equal to the new page

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 [28]: import statsmodels.api as sm

         convert_old = df2[(df2.landing_page == 'old_page') & (df2.converted == True)].shape[0]
```

```
convert_new = df2[(df2.landing_page == 'new_page') & (df2.converted == True)].shape[0]
n_old = df2[df2.group == 'control'].shape[0]
n_new = df2[df2.group == 'treatment'].shape[0]
#df2[df2.group == 'treatment'].shape[0]
convert_new
```

```
/opt/conda/lib/python3.6/site-packages/statsmodels/compat/pandas.py:56: FutureWarning: The panda
  from pandas.core import datetools
```

```
Out[28]: 17264
```

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 [29]: z_score, p_value = sm.stats.proportions_ztest([convert_old,convert_new],[n_old,n_new],a
         print('Z-score:', z_score,'\nP-value:', p_value)
```

```
Z-score: 1.31092419842
P-value: 0.905058312759
```

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.**?

```
In [36]: from scipy.stats import norm

         norm.cdf(z_score)
         # 0.9999999383005862 # Tells us how significant our z-score is

         norm.ppf(1-(0.05/2))
         # 1.959963984540054 # Tells us what our critical value at 95% confidence is
```

```
Out[36]: 1.959963984540054
```

Answer: Since the z-score of 1.31 doesn't exceeds the critical value of 1.95, and the p-value is very high, we fail to reject the null hypothesis that suggests that the old page coversion rate is equal or greater than it for new page, and yes it does agree with our previus finding in j and k

Part III - A regression approach

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

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

Answer: Logistic Regression

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

```
In [30]: df2[['control','ab_page']] = pd.get_dummies(df2.group)
         df2['intercept'] = 1
         df2.drop(['control'], axis=1)
         df2.ab_page.value_counts()
         #df2.group.value_counts()
```

```
Out[30]: 1    145310
         0    145274
         Name: ab_page, dtype: int64
```

c. Use **statsmodels** to import your regression model. Instantiate the model, and fit the model using the two columns you created in part **b.** to predict whether or not an individual converts.

```
In [31]: logit_mod = sm.Logit(df2.converted, df2[['intercept', 'ab_page']])
         results = logit_mod.fit()
```

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

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

```
In [32]: results.summary()
```

```
Out[32]: <class 'statsmodels.iolib.summary.Summary'>
         """
                                  Logit Regression Results
         ==============================================================================
         Dep. Variable:              converted   No. Observations:               290584
         Model:                          Logit   Df Residuals:                   290582
         Method:                           MLE   Df Model:                            1
         Date:                Fri, 04 Jan 2019   Pseudo R-squ.:              8.077e-06
         Time:                        17:46:38   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 the **Part II**? **Hint**: What are the null and alternative hypotheses associated with your regression model, and how do they compare to the null and alternative hypotheses in the **Part II**?

Answer: the way we built our hypothsis test here is as two-tailed test where the hypothsis goes as following: ### H_0 : p_{new} - p_{old} = 0 ### H_1 : p_{new} - p_{old} != 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?

Answer: The parameters we used in our test here can be enhanced to better understand the potential we have in our new page, customer properties like gender, age or origin might be helpful, however, one thing we need to be aware of that is involving too many parameters might cause a case of model overfitting.

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

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

```
In [33]: df_cntry = pd.read_csv('countries.csv')
         df_mrg = df_cntry.set_index('user_id').join(df2.set_index('user_id'), how='inner')
         df_mrg.country.value_counts()
         df_mrg['intercept'] = 1
         df_mrg[['US','UK']] = pd.get_dummies(df_mrg['country'])[['US','UK']]
```

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 [34]: Logit_mod = sm.Logit(df_mrg['converted'], df_mrg[['intercept', 'US', 'UK']])
         results = Logit_mod.fit()
         results.summary()

Optimization terminated successfully.
         Current function value: 0.366116
         Iterations 6


Out[34]: <class 'statsmodels.iolib.summary.Summary'>
         """
                                    Logit Regression Results
         ==============================================================================
```

```
Dep. Variable:              converted   No. Observations:              290584
Model:                         Logit   Df Residuals:                  290581
Method:                          MLE   Df Model:                           2
Date:               Fri, 04 Jan 2019   Pseudo R-squ.:               1.521e-05
Time:                       17:46:39   Log-Likelihood:            -1.0639e+05
converged:                      True   LL-Null:                   -1.0639e+05
                                       LLR p-value:                    0.1984
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
intercept     -2.0375      0.026    -78.364      0.000      -2.088      -1.987
US             0.0408      0.027      1.518      0.129      -0.012       0.093
UK             0.0507      0.028      1.786      0.074      -0.005       0.106
==============================================================================
"""
```

Answer: based on the previus test and as the p value is for both countries are higher that the significant level 0.05 we don't see any significance based on the origin country

Finishing Up

Congratulations! You have reached the end of the A/B Test Results project! This is the final project in Term 1. 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 [35]:
```
from subprocess import call
call(['python', '-m', 'nbconvert', 'Analyze_ab_test_results_notebook.ipynb'])
```

Out[35]: 0

In [ ]:

In [ ]: