Analyze_ab_test_results_notebook

August 25, 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.info()
```

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

c. The number of unique users in the dataset.

```
In [4]: df.user_id.nunique()
```

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

d. The proportion of users converted.

```
In [5]: df[df['converted'] == 1].shape[0] / df.shape[0]
```

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

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

```
In [6]: df.query("group == 'control' and landing_page != 'old_page' or group == 'treatment' and
```

```
Out[6]: user_id         3893
        timestamp       3893
        group           3893
        landing_page    3893
        converted       3893
        dtype: int64
```

f. Do any of the rows have missing values?

```
In [7]: df.isnull().values.any()
```

```
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.query("group == 'control' and landing_page == 'old_page' or group == 'treatment
```

```
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
```

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

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

```
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 [ ]:
```

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

```
In [12]: df2.drop_duplicates(subset = "user_id",keep=False, inplace=True)
```

```
/opt/conda/lib/python3.6/site-packages/ipykernel_launcher.py:1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/indexing.html#
  """Entry point for launching an IPython kernel.
```

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

```
Out[13]: 0.11959749882133504
```

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

```
In [14]: df2.query("group == 'control' and converted == 1").shape[0] / df2.query("group == 'cont
```

```
Out[14]: 0.1203863045004612
```

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

```
In [15]: df2.query("group == 'treatment' and converted == 1").shape[0] / df2.query("group == 'tr
```

```
Out[15]: 0.11880888313869065
```

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

```
In [16]: df2.query("group == 'treatment'").shape[0] / df2.shape[0]
```

```
Out[16]: 0.5000602237570677
```

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.

```
In [17]: diff = df2.query("group == 'control' and converted == 1").shape[0] / df2.query("group =
         diff
```

```
Out[17]: 0.0015774213617705535
```

We can conclude that the treatment does not really have any significant impact on 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.

Null Hypothesis - The old page is same or better than the new page Alternative Hypothesis - The old page is not better than the new page

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 [18]: p_new = df2.query("group == 'treatment' and converted == 1").shape[0] / df2.query("grou
         p_new

Out[18]: 0.11880888313869065
```

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

```
In [19]: p_old = df2.query("group == 'control' and converted == 1").shape[0] / df2.query("group
         p_old

Out[19]: 0.1203863045004612
```

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

```
In [20]: n_new = df2.query("group == 'treatment'").shape[0]
         n_new

Out[20]: 145309
```

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

```
In [21]: n_old = df2.query("group == 'control'").shape[0]
         n_old

Out[21]: 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 [22]: new_page_converted = np.random.binomial(n_new,p_new)
         new_page_converted

Out[22]: 17185
```

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

```
Out[23]: 17121
```

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

```
In [24]: new_page_converted - old_page_converted
```

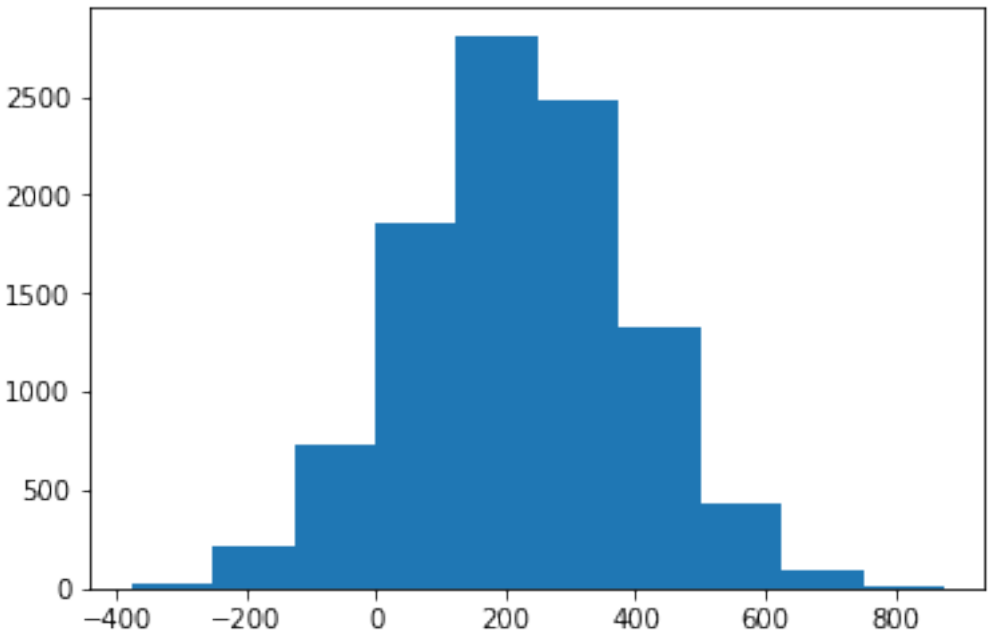
```
Out[24]: 64
```

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 [25]: p_diffs = []
         for _ in range(10000):
             new_convrt = np.random.binomial(n_new,p_new)
             old_convrt = np.random.binomial(n_old,p_old)
             diff = old_convrt - new_convrt
             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]: p_diffs = np.array(p_diffs)
         plt.hist(p_diffs);
```



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

```
In [27]: (p_diffs > (p_old - p_new)).mean()
```

```
Out[27]: 0.90100000000000002
```

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?

According to the experiments which we have run we - Fail to reject the null. The value which we have obtained is the p-value, ~89%-90% of the accepted pages are actually from the control web 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.query("group == 'control' and converted == 1").shape[0]
         convert_new = df2.query("group == 'treatment' and converted == 1").shape[0]
         n_old = df2.query("group == 'control'").shape[0]
         n_new = df2.query("group == 'treatment'").shape[0]
```

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

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]: t, p_score = sm.stats.proportions_ztest([convert_new, convert_old], [n_new, n_old], alt
         print('Test Statistic: ', t)
         print('P Value Score ', p_score)
```

```
Test Statistic:  -1.31024085793
P Value Score  0.904942816116
```

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.**?

Yes, these findings are similar to the ones from our manual method. Here, again, we fail to reject the null.

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?

Since the Y Values are binary we can use simple Linear Regression here. And that is what I have done. :)

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 [30]: df2['intercept'] = 1
         df2[['_', 'ab_page']] = pd.get_dummies(df2['group'])
         df2.head()
```

```
/opt/conda/lib/python3.6/site-packages/ipykernel_launcher.py:1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/indexing.html#
  """Entry point for launching an IPython kernel.
/opt/conda/lib/python3.6/site-packages/pandas/core/frame.py:3140: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/indexing.html#
  self[k1] = value[k2]
```

Out[30]:

	user_id	timestamp	group	landing_page	converted	intercept	_	ab_page
0	851104	2017-01-21 22:11:48.556739	control	old_page	0	1	1	0
1	804228	2017-01-12 08:01:45.159739	control	old_page	0	1	1	0
2	661590	2017-01-11 16:55:06.154213	treatment	new_page	0	1	0	1
3	853541	2017-01-08 18:28:03.143765	treatment	new_page	0	1	0	1
4	864975	2017-01-21 01:52:26.210827	control	old_page	1	1	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 [31]: lm = sm.Logit(df2['converted'], df2[['intercept', 'ab_page']])
         results = lm.fit()
```

```
Optimization terminated successfully.
         Current function value: 0.366119
         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:               290583
Model:                          Logit   Df Residuals:                   290581
Method:                           MLE   Df Model:                            1
Date:                Sun, 25 Aug 2019   Pseudo R-squ.:               8.068e-06
Time:                        15:20:05   Log-Likelihood:            -1.0639e+05
converged:                       True   LL-Null:                   -1.0639e+05
                                        LLR p-value:                    0.1901
==============================================================================
                 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     -1.310      0.190      -0.037       0.007
==============================================================================
"""
```

e. What is the p-value associated with **ab_page**? Why does it differ from the value you found in **Part II**? **Hint**: What are the null and alternative hypotheses associated with your regression model, and how do they compare to the null and alternative hypotheses in **Part II**?

The p-value is 0.190. Previously, we have tested with a different Null Hypothesis, and now we are testing with a different one, one which is basically conceptually opposite of the previous one. Hence, we are getting a p-value which is telling to Accept the Null.

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?

Hmm, here we have completely ignored the timestamps column, I would guess that it should be useful to determine if the user converted or not. Maybe, the new layout has more text and the users which actually went through it are more likely to convert?

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 [33]: df_countries = pd.read_csv('./countries.csv')
         df3 = df2.set_index('user_id').join(df_countries.set_index('user_id'), how='inner')
         df3.head()
```

Out[33]:

user_id	timestamp	group	landing_page	converted	intercept	_	ab_page	country
851104	2017-01-21 22:11:48.556739	control	old_page	0	1	1	0	US
804228	2017-01-12 08:01:45.159739	control	old_page	0	1	1	0	US
661590	2017-01-11 16:55:06.154213	treatment	new_page	0	1	0	1	US
853541	2017-01-08 18:28:03.143765	treatment	new_page	0	1	0	1	US
864975	2017-01-21 01:52:26.210827	control	old_page	1	1	1	0	US

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.

```
In [34]: df3[['CA','US','UK']] = pd.get_dummies(df3['country'])
         df3.head()
         df3 = df3.drop(['UK'], axis=1)
         lm = sm.Logit(df3['converted'], df3[['intercept', 'ab_page', 'US','CA']])
         results = lm.fit()
         results.summary()

Optimization terminated successfully.
         Current function value: 0.366114
         Iterations 6
```

Out[34]:

```
<class 'statsmodels.iolib.summary.Summary'>
"""
                           Logit Regression Results
==============================================================================
Dep. Variable:              converted   No. Observations:               290583
Model:                          Logit   Df Residuals:                   290579
Method:                           MLE   Df Model:                            3
Date:                Sun, 25 Aug 2019   Pseudo R-squ.:               2.322e-05
Time:                        15:20:06   Log-Likelihood:            -1.0639e+05
```

```
converged:                       True   LL-Null:                       -1.0639e+05
                                        LLR p-value:                       0.1761
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
intercept     -1.9893      0.009   -223.763      0.000      -2.007      -1.972
ab_page       -0.0149      0.011     -1.306      0.191      -0.037       0.007
US             0.0099      0.013      0.743      0.458      -0.016       0.036
CA            -0.0408      0.027     -1.516      0.129      -0.093       0.012
==============================================================================
"""
```

Based on the results of the results of the experiment we can conclude that the company should not implement the newer page, but should continue to use the older page.

```
In [35]: from subprocess import call
         call(['python', '-m', 'nbconvert', 'Analyze_ab_test_results_notebook.ipynb'])
```

```
Out[35]: 0
```