Analyze_ab_test_results_notebook

December 19, 2018

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 below cell to find the number of rows in the dataset.

```
In [3]: df.count
```

Out[3]:

```
<bound method DataFrame.count of         user_id                   timestamp      group
```

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
5	936923	2017-01-10 15:20:49.083499	control	old_page	0
6	679687	2017-01-19 03:26:46.940749	treatment	new_page	1
7	719014	2017-01-17 01:48:29.539573	control	old_page	0
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
10	929503	2017-01-18 05:37:11.527370	treatment	new_page	0
11	834487	2017-01-21 22:37:47.774891	treatment	new_page	0
12	803683	2017-01-09 06:05:16.222706	treatment	new_page	0
13	944475	2017-01-22 01:31:09.573836	treatment	new_page	0
14	718956	2017-01-22 11:45:11.327945	treatment	new_page	0
15	644214	2017-01-22 02:05:21.719434	control	old_page	1
16	847721	2017-01-17 14:01:00.090575	control	old_page	0
17	888545	2017-01-08 06:37:26.332945	treatment	new_page	1
18	650559	2017-01-24 11:55:51.084801	control	old_page	0
19	935734	2017-01-17 20:33:37.428378	control	old_page	0
20	740805	2017-01-12 18:59:45.453277	treatment	new_page	0
21	759875	2017-01-09 16:11:58.806110	treatment	new_page	0
22	767017	2017-01-12 22:58:14.991443	control	new_page	0
23	793849	2017-01-23 22:36:10.742811	treatment	new_page	0
24	905617	2017-01-20 14:12:19.345499	treatment	new_page	0
25	746742	2017-01-23 11:38:29.592148	control	old_page	0
26	892356	2017-01-05 09:35:14.904865	treatment	new_page	1
27	773302	2017-01-12 08:29:49.810594	treatment	new_page	0
28	913579	2017-01-24 09:11:39.164256	control	old_page	1

```
29       736159  2017-01-06 01:50:21.318242  treatment     new_page          0
...         ...                         ...        ...          ...        ...
294448   776137  2017-01-12 05:53:12.386730  treatment     new_page          0
294449   883344  2017-01-22 23:15:58.645325  treatment     new_page          0
294450   825594  2017-01-06 12:37:08.897784  treatment     new_page          0
294451   875688  2017-01-14 07:19:49.042869    control     old_page          0
294452   927527  2017-01-12 10:52:11.084740    control     old_page          0
294453   789177  2017-01-17 18:17:56.215378    control     old_page          0
294454   937338  2017-01-19 03:23:22.236666  treatment     new_page          0
294455   733101  2017-01-23 12:52:58.711914  treatment     new_page          0
294456   679096  2017-01-02 16:43:49.237940  treatment     new_page          0
294457   691699  2017-01-09 23:42:35.963486  treatment     new_page          0
294458   807595  2017-01-22 10:43:09.285426  treatment     new_page          0
294459   924816  2017-01-20 10:59:03.481635    control     old_page          0
294460   846225  2017-01-16 15:24:46.705903  treatment     new_page          0
294461   740310  2017-01-10 17:22:19.762612    control     old_page          0
294462   677163  2017-01-03 19:41:51.902148  treatment     new_page          0
294463   832080  2017-01-19 13:18:27.352570    control     old_page          0
294464   834362  2017-01-17 01:51:56.106436    control     old_page          0
294465   925675  2017-01-07 20:38:26.346410  treatment     new_page          0
294466   923948  2017-01-09 16:33:41.104573    control     old_page          0
294467   857744  2017-01-05 08:00:56.024226    control     old_page          0
294468   643562  2017-01-02 19:20:05.460595  treatment     new_page          0
294469   755438  2017-01-18 17:35:06.149568    control     old_page          0
294470   908354  2017-01-11 02:42:21.195145    control     old_page          0
294471   718310  2017-01-21 22:44:20.378320    control     old_page          0
294472   822004  2017-01-04 03:36:46.071379  treatment     new_page          0
294473   751197  2017-01-03 22:28:38.630509    control     old_page          0
294474   945152  2017-01-12 00:51:57.078372    control     old_page          0
294475   734608  2017-01-22 11:45:03.439544    control     old_page          0
294476   697314  2017-01-15 01:20:28.957438    control     old_page          0
294477   715931  2017-01-16 12:40:24.467417  treatment     new_page          0

[294478 rows x 5 columns]>
```

c. The number of unique users in the dataset.

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

Out[4]: user_id         290584
        timestamp       294478
        group                2
        landing_page         2
        converted            2
        dtype: int64
```

d. The proportion of users converted.

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

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

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

```
In [6]: # looking for mismatch in group and landing page
        df_treat = df[(df['group'] == 'treatment') & (df['landing_page'] == 'old_page')]
        df_cont = df[(df['group'] == 'control') & (df['landing_page'] == 'new_page')]

        # Add lengths
        mismatch= len(df_treat) + len(df_cont)

        # Create one dataframe from it
        mismatch_df = pd.concat([df_treat, df_cont])

        mismatch

Out[6]: 3893
```

f. Do any of the rows have missing values?

```
In [7]: df.info()
        df.isnull().sum()

<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


Out[7]: user_id         0
        timestamp       0
        group           0
        landing_page    0
        converted       0
        dtype: int64
```

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

```
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.nunique()

Out[10]: 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 [11]: df2[df2.duplicated(['user_id'])]['user_id'].unique()

Out[11]: array([773192])
```

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

```
In [12]: df2[df2.duplicated(['user_id'],keep=False)]

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_duplicates(['user_id'], keep='first')
```

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]: df2['converted'].mean()

Out[14]: 0.11959708724499628
```

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

```
In [15]: df2[df2['group'] == 'control']['converted'].mean()

Out[15]: 0.1203863045004612
```

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

```
In [16]: df2[df2['group'] == 'treatment']['converted'].mean()
```

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

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

```
In [17]: len(df2.query("landing_page == 'new_page'")) / df2.shape[0]
```

```
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.

```
In [18]: len(df2.query("landing_page == 'old_page'"))
```

```
Out[18]: 145274
```

The average conversion is relatively high for control group with old page compared to treament group with new page. The difference is comparatively very small and in the order of 0.16%.

Approximately two groups (control and treatment) were divided equally to have better chance of predicting conversion rate. ### 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.

H1: $p_{new} > p_{old}$ 2. Assume under the null hypothesis, p_{new} and p_{old} both have "true" success rates equal to the **converted** success rate regardless of page - that is p_{new} and p_{old} are equal. Furthermore, assume they are equal to the **converted** rate in **ab_data.csv** regardless of the page.

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

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

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

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

```
In [19]: p_new = df2['converted'].mean()
         p_new
```

```
Out[19]: 0.11959708724499628
```

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

```
In [20]: p_old = df2['converted'].mean()
         p_old

Out[20]: 0.11959708724499628
```

c. What is n_{new}?

```
In [21]: n_new = df2[df2['group'] == 'treatment'].shape[0]
         n_new

Out[21]: 145310
```

d. What is n_{old}?

```
In [22]: n_old = df2[df2['group'] == 'control'].shape[0]
         n_old

Out[22]: 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 [23]: new_page_converted = np.random.binomial(n_new,p_new)
         new_page_converted

Out[23]: 17488
```

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

Out[24]: 17249
```

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

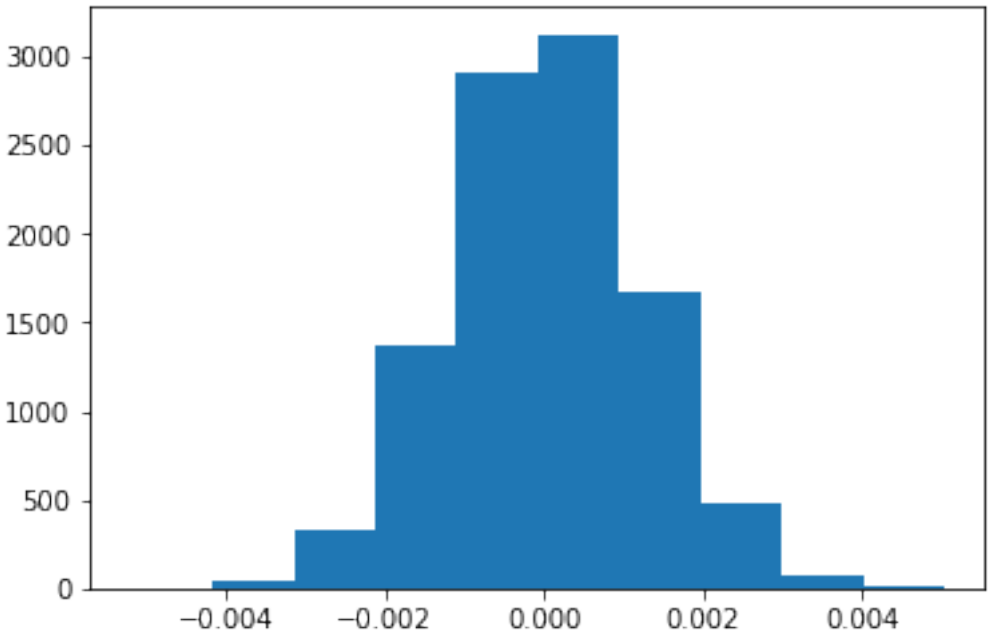
```
In [25]: diff = ((new_page_converted/n_new)-(old_page_converted/n_old))
         diff

Out[25]: 0.0016153435197843596
```

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 [26]: p_diffs = []
         for _ in range(10000):
             new_page_converted = np.random.binomial(n_new,p_new)
             old_page_converted = np.random.binomial(n_old, p_old)
             diff = ((new_page_converted/n_new) - (old_page_converted/n_old))
             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 [27]: plt.hist(p_diffs);
```



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

```
In [28]: act_diff = df2[df2['group'] == 'treatment']['converted'].mean() -  df2[df2['group'] ==
         print(act_diff)
         p_diffs = np.array(p_diffs)
         print(p_diffs)
         (act_diff < p_diffs).mean()
```

```
-0.00157823898536
[-0.00143358 -0.00104149  0.00362514 ...,  0.00023198  0.00021817
  0.00196643]
```

```
Out[28]: 0.90580000000000005
```

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?

Calculation of P value. It is the probability of estimating more extreme results given that the null hypothesis is true. Large p-value indicates weak evidence against the null hypothesis that means fail to reject the null hypothesis which suggest that the new page conversion rate is higher than the old rate.

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 [29]: import statsmodels.api as sm

         convert_old = df2.query(" landing_page == 'old_page' and converted == 1").shape[0]
         convert_new = df2.query(" landing_page == 'new_page' and converted == 1").shape[0]
         n_old = df2[df2['group'] == 'control'].shape[0]
         n_new = df2[df2['group'] == 'treatment'].shape[0]
         print(convert_old)
         print(convert_new)
```

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


17489
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 [30]: z_score, p_value = sm.stats.proportions_ztest([convert_old, convert_new], [n_old, n_new
         print(z_score, p_value)
```

```
1.31092419842 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 [31]: from scipy.stats import norm
         print(norm.cdf(z_score))

         print(norm.ppf(1-(0.05)))
```

```
0.905058312759
1.64485362695
```

Z score value is 1.31 and p-value is 1.644853. Z score is less than p value, hence we fail to reject null hypothesis. This indicates, new page conversion rate is higher than the old rate. Yes I Agree with findings in parts 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?

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 [32]: df2['intercept'] = 1
         df2[['control','treatment']] = pd.get_dummies(df2['group'])
```

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 [33]: import statsmodels.api as sm

         logit = sm.Logit(df2['converted'],df2[['intercept' ,'treatment']])
         results = logit.fit()

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

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

```
In [34]: results.summary()

Out[34]: <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:                Tue, 16 Oct 2018   Pseudo R-squ.:               8.077e-06
Time:                        00:12:46   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
treatment     -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**?

0.190
Part II was one sided test. This part deal with not equal case in our hypotheses.

$$H_0 : p_{new} = p_{old}$$
$$H_1 : p_{new} \neq p_{old}$$

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?

other factors like age, gender, type of program can influence individual converts to new page or not. There are some disadvantages like if we add high impactful factors or variables, leading to unreliable and unstable estimates of regression coefficients (Multicollinearity) can affect 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. 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 [35]: countries_df = pd.read_csv('./countries.csv')
         df_new = countries_df.set_index('user_id').join(df2.set_index('user_id'), how='inner')

In [36]: ### Create the necessary dummy variables
         df_new['intercept'] = 1
         df_new[['CA','US']] = pd.get_dummies(df_new['country'])[['CA','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.

Provide the summary results, and your conclusions based on the results.

```
In [37]: mod = sm.Logit(df_new['converted'], df_new[['intercept', 'US', 'CA']])

         results = mod.fit()

         results.summary()

Optimization terminated successfully.
         Current function value: 0.366116
         Iterations 6


Out[37]: <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:                Tue, 16 Oct 2018   Pseudo R-squ.:               1.521e-05
         Time:                        00:12:47   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     -1.9868      0.011   -174.174      0.000      -2.009      -1.964
         US            -0.0099      0.013     -0.746      0.456      -0.036       0.016
         CA            -0.0507      0.028     -1.786      0.074      -0.106       0.005
         ==============================================================================
         """
```

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

```
Out[38]: 0
```