

Analyze_ab_test_results_notebook

June 12, 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

- 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.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 match.

```
In [6]: question_e = len(df.query('group == "treatment" and landing_page != "new_page")) + \
                    len(df.query('group != "treatment" and landing_page == "new_page"))
question_e
```

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

f. Do any of the rows have missing values?

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

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

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]: df_01 = df[(df.group == 'treatment') & (df.landing_page == 'new_page')]
        df_02 = df[(df.group == 'control') & (df.landing_page == 'old_page')]
        frames = [df_01, df_02]
        df2 = pd.concat(frames)
```

```
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].shape
```

```
Out[9]: 0
```

3. Use **df2** and the cells below to answer questions for **Quiz3** in the classroom.

a. How many unique **user_ids** 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'], keep=False)]
```

```
Out[11]:
```

	user_id	timestamp	group	landing_page	converted
1899	773192	2017-01-09 05:37:58.781806	treatment	new_page	0
2893	773192	2017-01-14 02:55:59.590927	treatment	new_page	0

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

```
In [12]: """
For user_id = 773192
all things are similar except timestamp so by making timestamp equal to each other and
row
reference: https://stackoverflow.com/questions/26882499/reset-time-part-of-a-pandas-ti
"""

df2['timestamp'].replace('2017-01-14 02:55:59.590927', '2017-01-09 05:37:58.781806', in
df2.loc[df2['user_id'] == 773192]
```

```
Out[12]:
```

	user_id	timestamp	group	landing_page	converted
1899	773192	2017-01-09 05:37:58.781806	treatment	new_page	0
2893	773192	2017-01-09 05:37:58.781806	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(keep = 'first')
```

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 [14]: df.converted.value_counts()[1]/len(df.index)
```

```
Out[14]: 0.11965919355605512
```

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

```
In [15]: df2.groupby(["group", "converted"]).size()[1] / df2.group.value_counts()[1]
```

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

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

```
In [16]: df2.groupby(["group", "converted"]).size()[3] / df2.group.value_counts()[0]
```

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

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

```
In [17]: df2.landing_page.value_counts()[0]/len(df.index)
```

```
Out[17]: 0.49344942576355449
```

- 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.

Answer: * Based on the total users that converted, there is a slightly higher number of people that converted in the control group as opposed to the treatment group. * The percentage of converted in the control group is 12.04% and converted in the treatment group is 11.88%. * The difference is very subtle so I do not believe there is a significant enough difference for a certain page that will convert more users.

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: * Null Hypotheses: if p-value is less than 5%, the old page has a higher chance of converting users. * Alternative Hypotheses: if p-value is equal to or greater than 5%, then the new page has a higher chance of converting users.

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.converted.mean()  
p_new
```

```
Out[18]: 0.11959708724499628
```

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

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

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

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

```
In [20]: n_new = df2.landing_page.value_counts()[0]
         n_new
```

```
Out[20]: 145310
```

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

```
In [21]: n_old = df2.landing_page.value_counts()[1]
         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.choice(2, size=n_new, p=[p_new, 1 - p_new])
         new_page_converted.mean()
```

```
Out[22]: 0.88000825820659279
```

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.choice(2, size=n_old, p=[p_old, 1 - p_old])
         old_page_converted.mean()
```

```
Out[23]: 0.8798202018255159
```

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

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

```
Out[24]: 0.00018805638107688427
```

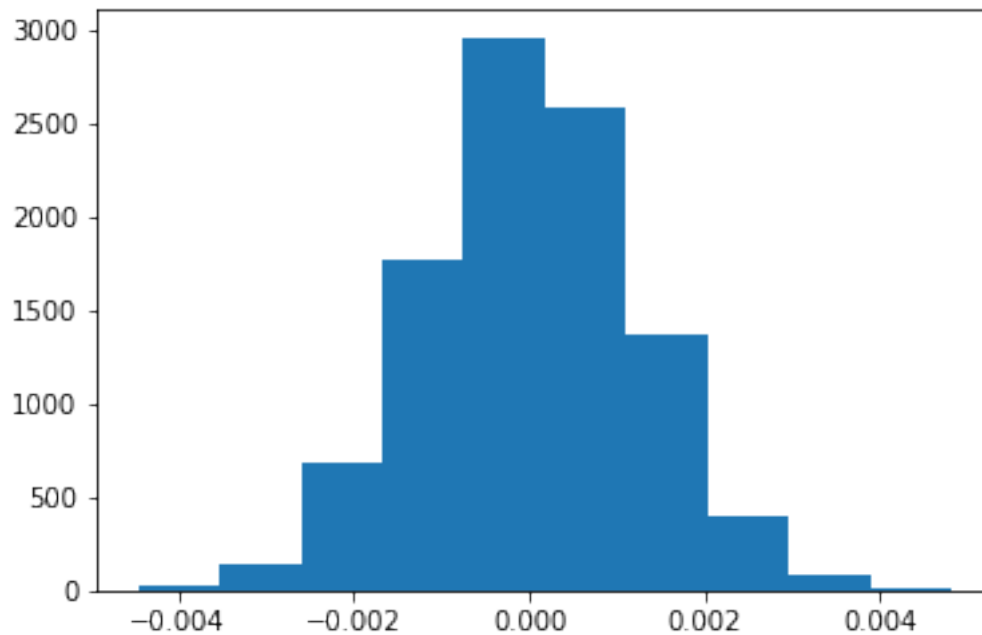
h. Create 10,000 $p_{new} - p_{old}$ values using the same simulation process you used in parts (a) through (g) above. Store all 10,000 values in a NumPy array called **p_diffs**.

```
In [25]: p_diffs, bs_new_means, bs_old_means = [], [], []

         for _ in range(10000):
             bs_new = np.random.choice(2, size=n_new, p=[p_new, 1 - p_new])
             bs_old = np.random.choice(2, size=n_old, p=[p_old, 1 - p_old])
             bs_new_means.append(bs_new.mean())
             bs_old_means.append(bs_old.mean())
             p_diffs.append(bs_new.mean() - bs_old.mean())
```

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 [28]: actual_diff = df2.query('landing_page == "new_page").converted.mean() - \
          df2.query('landing_page == "old_page").converted.mean()
          (p_diffs > actual_diff).mean()
```

```
Out[28]: 0.9042999999999999
```

- 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?

Answer: * What we computed is the actual versus observed difference in means of converted old page and converted new page. * This means that we had random choices of these mean converted values for the observed difference, which is stored in p_diffs. * The actual difference was calculated from the dataset ab_data.csv. This value that we calculated, difference in means, is the p-value. * Our p-value exceeds the critical value of 0.05 in this case and so we fail to reject the null hypothesis, we cannot assume the new page converts more users than the old 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 [30]: import statsmodels.api as sm
```

```
convert_old = df2.query('landing_page == "old_page" & converted == "1"').count()[0]
convert_new = df2.query('landing_page == "new_page" & converted == "1"').count()[0]
n_old = df2.landing_page.value_counts()[1]
n_new = df2.landing_page.value_counts()[0]
convert_old, convert_new, n_old, n_new
```

```
Out[30]: (17489, 17264, 145274, 145310)
```

m. Now use `stats.proportions_ztest` to compute your test statistic and p-value. [Here](#) is a helpful link on using the built in.

```
In [31]: z_score, p_value = sm.stats.proportions_ztest([convert_old, convert_new], [n_old, n_new],
              z_score, p_value
```

```
Out[31]: (1.3109241984234394, 0.18988337448195103)
```

```
In [32]: from scipy.stats import norm
          norm.cdf(z_score) # this tells us how significant our z-score is
```

```
Out[32]: 0.90505831275902449
```

```
In [33]: norm.ppf(1-(0.05/2)) # this tells us our critical value at 95% confidence interval
```

```
Out[33]: 1.959963984540054
```

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.?

Answer: * The z-score of 1.311 is less than critical value at 95% confidence interval, 1.960, so it falls within the confidence interval range of 95%. We fail to reject the null hypothesis; the old page converted more users than the new page. * Also, our p-value(0.189) is different than what we computed in parts j and k(0.898) which is expected because the null and alternative hypothesis assumed that the `p_new` and `p_old` are the same and that there is no difference in converting users from the old page and new page. However, that is not the case since the old page does convert more users than the old page based on the lower p-value(0.189).

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?

Answer: In this case, we will use 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 [34]: # create dummy variables from the group column
df2[['ab_page', 'treatment']] = pd.get_dummies(df2['group'])
# drop the treatment column since this is not necessary
df2 = df2.drop('treatment', 1)
# create intercept column
df2['intercept'] = 1
# check to make sure the columns are added and to compare with old df2
# dataframe above
df2.head()
```

```
Out[34]:
```

	user_id	timestamp	group	landing_page	converted	\
2	661590	2017-01-11 16:55:06.154213	treatment	new_page	0	
3	853541	2017-01-08 18:28:03.143765	treatment	new_page	0	
6	679687	2017-01-19 03:26:46.940749	treatment	new_page	1	
8	817355	2017-01-04 17:58:08.979471	treatment	new_page	1	
9	839785	2017-01-15 18:11:06.610965	treatment	new_page	1	

	ab_page	intercept
2	0	1
3	0	1
6	0	1
8	0	1
9	0	1

- 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 [42]: log_mod = sm.Logit(df2['converted'], df2[['intercept', 'ab_page']])
results = log_mod.fit()
results.summary2()
"""
reference :https://stackoverflow.com/questions/49814258/statsmodel-attributeerror-module
for results.summary2()
"""
```

Optimization terminated successfully.

Current function value: 0.366118

Iterations 6

```

Out[42]: <class 'statsmodels.iolib.summary2.Summary'>
        """
                Results: Logit
        =====
Model:                Logit                No. Iterations:    6.0000
Dependent Variable: converted                Pseudo R-squared: 0.000
Date:                2020-06-12 16:27 AIC:                212780.3502
No. Observations:    290584                BIC:                212801.5095
Df Model:            1                    Log-Likelihood:    -1.0639e+05
Df Residuals:        290582                LL-Null:            -1.0639e+05
Converged:            1.0000                Scale:            1.0000
-----
                Coef.    Std.Err.    z        P>|z|    [0.025    0.975]
-----
intercept    -2.0038    0.0081   -247.1457   0.0000   -2.0197   -1.9879
ab_page       0.0150    0.0114    1.3109   0.1899   -0.0074    0.0374
=====
        """

```

```
In [43]: np.exp(0.0150)
```

```
Out[43]: 1.0151130646157189
```

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

Summary: Holding all other variables constant, the number of converted is 1.015 times more likely to be converted than those that are not converted. This means that the old page and new page are both equal in chance of converting users. We should not assume that the new page is better than the old page.

- 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**?

Answer: * The p-value found in the logistic regression model(0.19) is different than what we found in parts j and k because our null and alternative hypothesis model assumed that there is an equal probability of the old and new page converting users. * In the logistic regression model, this is not the case. Also, the Logistic Regression performed is a two-tailed test, whereas the computation done in Part II is a one-tailed test.

- 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: * Other factors that influence whether an individual converts could be age. Older users may prefer more information on the pages as opposed to a kid, where they may prefer more pictures and a more casual theme. * Adding more factors into the regression model will increase or decrease confidence intervals. * A disadvantage of multiple factors in a logistic regression model is that it reduces the power of analysis.

- 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.

Answer: The country that a user is in does not affect the conversion rate. We can determine this by figuring out the conversion rate of each of the countries. US: 0.1195 UK: 0.1206 CA: 0.1153 These are approximately all the same. We fail to reject the null hypothesis.

```
In [44]: df3 = pd.read_csv('countries.csv')
df3.head()
```

```
Out[44]:   user_id country
0    834778      UK
1    928468      US
2    822059      UK
3    711597      UK
4    710616      UK
```

```
In [45]: # create dummy variables for country column
df3[['CA', 'UK', 'US']] = pd.get_dummies(df3['country'])
# drop the country column since this is not necessary
df3 = df3.drop('country', 1)
df3.head()
```

```
Out[45]:   user_id  CA  UK  US
0    834778    0   1   0
1    928468    0   0   1
2    822059    0   1   0
3    711597    0   1   0
4    710616    0   1   0
```

```
In [46]: new_df = df2.join(df3.set_index('user_id'), on='user_id')
new_df.head()
```

```
Out[46]:   user_id      timestamp      group landing_page  converted  \
2    661590  2017-01-11 16:55:06.154213  treatment    new_page         0
3    853541  2017-01-08 18:28:03.143765  treatment    new_page         0
6    679687  2017-01-19 03:26:46.940749  treatment    new_page         1
8    817355  2017-01-04 17:58:08.979471  treatment    new_page         1
9    839785  2017-01-15 18:11:06.610965  treatment    new_page         1

   ab_page  intercept  CA  UK  US
2         0          1   0   0   1
3         0          1   0   0   1
6         0          1   1   0   0
8         0          1   0   1   0
9         0          1   1   0   0
```

```
In [48]: new_df.query('US == "1"').converted.mean(), new_df.query('UK == "1"').converted.mean(), \
        new_df.query('CA == "1"').converted.mean()
```

```
Out[48]: (0.1195468006423762, 0.12059448568984076, 0.11531829781364232)
```

- 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 [49]: new_df['US_ab_page'] = new_df['US'] * new_df['ab_page']
        new_df['UK_ab_page'] = new_df['UK'] * new_df['ab_page']
        new_df['CA_ab_page'] = new_df['CA'] * new_df['ab_page']
```

```
In [51]: log_mod = sm.Logit(new_df['converted'], new_df[['intercept', 'US_ab_page', 'UK_ab_page']]
        results = log_mod.fit()
        results.summary2()
```

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

```
Out[51]: <class 'statsmodels.iolib.summary2.Summary'>
        """
```

```

                                Results: Logit
=====
Model:                Logit                No. Iterations:    6.0000
Dependent Variable:    converted            Pseudo R-squared:    0.000
Date:                 2020-06-12 17:27      AIC:                212784.0722
No. Observations:     290584                BIC:                212826.3908
Df Model:              3                    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    -2.0038    0.0081  -247.1457  0.0000   -2.0197  -1.9879
US_ab_page    0.0173    0.0126   1.3736  0.1696   -0.0074   0.0420
UK_ab_page    0.0115    0.0181   0.6392  0.5227   -0.0239   0.0469
CA_ab_page   -0.0002    0.0373  -0.0064  0.9949   -0.0734   0.0729
=====
        """
```

```
In [52]: np.exp(results.params)
```

```
Out[52]: intercept      0.134827
         US_ab_page     1.017439
         UK_ab_page     1.011611
         CA_ab_page     0.999761
         dtype: float64
```

Summary: By looking at the odds ratio of the interaction between country and ab_page, we have an equal chance of the ab_page converting users in each country(1.017, 1.012, 1.00). We thus fail to reject the null hypothesis; the new page does not convert more than the old page.

0.2.1 Conclusions

Based on the statistical tests we used, the Z-test, logistic regression model, and actual difference observed, the results have shown that the new and old page have an approximately equal chance of converting users. We fail to reject the null hypothesis. I recommend to the e-commerce company to keep the old page. This will save time and money on creating a new page.

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 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** sub-menu, 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 [53]: from subprocess import call
         call(['python', '-m', 'nbconvert', 'Analyze_ab_test_results_notebook.ipynb'])
```

```
Out[53]: 0
```

```
In [ ]:
```