Analyze A/B Test Results

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!

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 (https://review.udacity.com/#!/projects/37e27304-ad47-4eb0-a1ab-8c12f60e43d0/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 s
et 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.tail()
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
```

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

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

Out[3]:
```
(294478, 5)
```

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]:
```
df1=df.loc[df['converted'] == 1]
df1.nunique()
35173/290584
```

Out[5]:
```
0.12104245244060237
```

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

In [6]:
```
df[((df['group'] == 'treatment') == (df['landing_page'] == 'new_page'))==False
].shape
```

Out[6]:
```
(3893, 5)
```

f. Do any of the rows have missing values?

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

Out[7]:
```
0
```

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.drop(df[((df['group'] == 'treatment') == (df['landing_page'] == 'new_page'))==False].index)
```

In [9]:
```
df2.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 290585 entries, 0 to 294477
Data columns (total 5 columns):
user_id         290585 non-null int64
timestamp       290585 non-null object
group           290585 non-null object
landing_page    290585 non-null object
converted       290585 non-null int64
dtypes: int64(2), object(3)
memory usage: 13.3+ MB
```

In [10]:
```
df[((df['group'] == 'control') == (df['landing_page'] == 'old_page'))==False].shape
```

Out[10]:
```
(3893, 5)
```

In [11]:
```
df2.nunique()
```

Out[11]:
```
user_id         290584
timestamp       290585
group                2
landing_page         2
converted            2
dtype: int64
```

In [12]:
```
# Double Check all of the correct rows were removed - this should be 0
df2[((df2['group'] == 'treatment') == (df2['landing_page'] == 'new_page')) == False].shape[0]
```

Out[12]:
```
0
```

In [13]:
```
df2[((df2['group'] == 'control') == (df2['landing_page'] == 'old_page')) == False].shape[0]
```

Out[13]: 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 [14]:
```
df2.nunique()
df2.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 290585 entries, 0 to 294477
Data columns (total 5 columns):
user_id         290585 non-null int64
timestamp       290585 non-null object
group           290585 non-null object
landing_page    290585 non-null object
converted       290585 non-null int64
dtypes: int64(2), object(3)
memory usage: 13.3+ MB
```

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

In [15]:
```
df2.info()
pd.concat(g for _, g in df2.groupby("user_id") if len(g) > 1)
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 290585 entries, 0 to 294477
Data columns (total 5 columns):
user_id         290585 non-null int64
timestamp       290585 non-null object
group           290585 non-null object
landing_page    290585 non-null object
converted       290585 non-null int64
dtypes: int64(2), object(3)
memory usage: 13.3+ MB
```

Out[15]:

	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 [16]:
```
df.iloc[2893]
```

Out[16]:
```
user_id                         773192
timestamp       2017-01-14 02:55:59.590927
group                          treatment
landing_page                    new_page
converted                              0
Name: 2893, dtype: object
```

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

In [17]:
```
#df2.drop(df2.index[1899])
#pd.concat(g for _, g in df2.groupby("user_id") if len(g) > 1)
df2=df2[df2.timestamp != '2017-01-14 02:55:59.590927']
#pd.concat(g for _, g in df2.groupby("user_id") if len(g) > 1)
df2.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 290584 entries, 0 to 294477
Data columns (total 5 columns):
user_id         290584 non-null int64
timestamp       290584 non-null object
group           290584 non-null object
landing_page    290584 non-null object
converted       290584 non-null int64
dtypes: int64(2), object(3)
memory usage: 13.3+ MB
```

In [18]:
```
df2.nunique()
```

Out[18]:
```
user_id         290584
timestamp       290584
group                2
landing_page         2
converted            2
dtype: int64
```

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 [19]:
```
#df2.loc[df2['converted'] =='1'].count
df2.loc[df['converted'] == 1].nunique()
```

Out[19]:
```
user_id         34753
timestamp       34753
group               2
landing_page        2
converted           1
dtype: int64
```

In [20]:
```
34753/290584
```

Out[20]:
```
0.11959708724499628
```

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

In [21]:
```
df2.loc[df['group'] == 'control'].nunique()
```

Out[21]:
```
user_id         145274
timestamp       145274
group                1
landing_page         1
converted            2
dtype: int64
```

In [22]:
```
df3=df2.loc[df['group'] == 'control']
df3.loc[df['converted']==1].count()
```

Out[22]:
```
user_id         17489
timestamp       17489
group           17489
landing_page    17489
converted       17489
dtype: int64
```

In [23]:
```
17489/145272
```

Out[23]:
```
0.12038796189217468
```

In [24]:
```
0.12038796189217468-0.11880806551510564
```

Out[24]:
```
0.00157989637706904
```

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

In [25]:
```
df2.loc[df['group'] == 'treatment'].nunique()
```

Out[25]:
```
user_id         145310
timestamp       145310
group                1
landing_page         1
converted            2
dtype: int64
```

In [26]:
```
df4=df2.loc[df['group'] == 'treatment']
df4.loc[df['converted']==1].count()
```

Out[26]:
```
user_id         17264
timestamp       17264
group           17264
landing_page    17264
converted       17264
dtype: int64
```

In [27]:
```
17264/145310
```

Out[27]:
```
0.11880806551510564
```

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

In [28]:
```
df2.loc[df['landing_page'] == 'new_page'].nunique()
145310/290582
```

Out[28]:
```
0.5000653860184044
```

e. Consider your results from a. through d. above, and explain below whether you think there is sufficient evidence to say that the new treatment page leads to more conversions.

Actually, the conversion rate for the new page is slightly less than that of old page. So there is no evidence that the new treatment page leads to more 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: Old page is better or as good as the new page Alternative hypothesis: New page is better than the old 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 **convert rate** for p_{new} under the null?

In [29]:
```
p_new=0.1196
```

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

In [30]:
```
p_old=0.1196
```

c. What is n_{new}?

In [31]:
```
n_new=145310
```

d. What is n_{old}?

In [32]:
```
n_old=145272
```

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 [33]:
```
new_page_converted=np.random.binomial(n_new,p_new)
```

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 [34]:
```
old_page_converted=np.random.binomial(n_old,p_old)
```

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

In [35]:
```
new_page_converted/n_new-old_page_converted/n_old
```

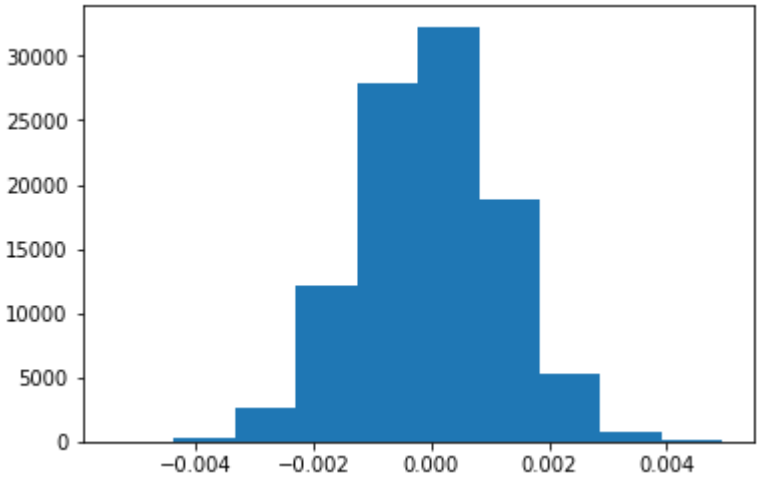
Out[35]: -0.00013447440504525676

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 a numpy array called **p_diffs**.

In [36]:
```
p_diffs=[]
for _ in range(100000):
    old_page_converted=np.random.binomial(n_old,p_old)
    new_page_converted=np.random.binomial(n_new,p_new)
    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 [37]:
```
plt.hist(p_diffs);
```



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

In [38]:
```
p_diff_actual= 0.12038796189217468-0.11880806551510564
(np.array(p_diffs) > -0.00158).mean()
```

Out[38]: 0.90442

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?

1. We have calculated the proportion of values that that are greater than actual difference in probablities of coonverting. This is a p-value.
2. The probability of observing our statistic or a more extreme value in favour of the alternative, given there was actually no difference in conversion rates for old and new pages.
3. Our p-value is very large which means that observing the data from the null is not unlikely. So we fail to reject the null in favour of the alternative hypothesis. So we conclude that the new page is no better 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 [39]:
```
import statsmodels.api as sm

convert_old = 17489
convert_new = 17264
n_old = 145274
n_new = 145310
```

```
C:\Users\maddhikunta\Anaconda3\lib\site-packages\statsmodels\compat\pandas.p
y:56: FutureWarning: The pandas.core.datetools module is deprecated and will
be removed in a future version. Please use the pandas.tseries module instead.
  from pandas.core import datetools
```

m. Now use `stats.proportions_ztest` to compute your test statistic and p-value. Here (http://knowledgetack.com/python/statsmodels/proportions_ztest/) is a helpful link on using the built in.

In [40]:
```
import statsmodels.api as sm
z_score, p_value = sm.stats.proportions_ztest(np.array([17264, 17489]), np.arr
ay([145310, 145274]), alternative='larger')
z_score
```

Out[40]:
```
-1.3109241984234394
```

In [41]:
```
p_value
```

Out[41]:
```
0.90505831275902449
```

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 order to change the default assumption of alternative hypothesis here to 'new page is bettere than old', the parameter alternative='larger' has been added to the ztest

The z value is -1.31 which is greater than -1.96. So we fail to reject the null hypothesis. So there is no difference in conversion rates of the old and new pages. And the p-value is 0.91 which is almost close to the p-value obtained in part j.

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 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 [42]:
```
df2[['ab_page','control']] = pd.get_dummies(df2['landing_page'])
df2.head()
```

Out[42]:

	user_id	timestamp	group	landing_page	converted	ab_page	control
0	851104	2017-01-21 22:11:48.556739	control	old_page	0	0	1
1	804228	2017-01-12 08:01:45.159739	control	old_page	0	0	1
2	661590	2017-01-11 16:55:06.154213	treatment	new_page	0	1	0
3	853541	2017-01-08 18:28:03.143765	treatment	new_page	0	1	0
4	864975	2017-01-21 01:52:26.210827	control	old_page	1	0	1

In [43]:
```
df2['intercept']=1
```

In [44]:
```
del df2['control']
```

In [45]:
```
df2.head()
```

Out[45]:

	user_id	timestamp	group	landing_page	converted	ab_page	intercept
0	851104	2017-01-21 22:11:48.556739	control	old_page	0	0	1
1	804228	2017-01-12 08:01:45.159739	control	old_page	0	0	1
2	661590	2017-01-11 16:55:06.154213	treatment	new_page	0	1	1
3	853541	2017-01-08 18:28:03.143765	treatment	new_page	0	1	1
4	864975	2017-01-21 01:52:26.210827	control	old_page	1	0	1

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 [46]:
```
import statsmodels.api as sm
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 [47]:
```
results.summary()
```

Out[47]:

Logit Regression Results

Dep. Variable:	converted	**No. Observations:**	290584
Model:	Logit	**Df Residuals:**	290582
Method:	MLE	**Df Model:**	1
Date:	Wed, 24 Jan 2018	**Pseudo R-squ.:**	8.077e-06
Time:	20:31:18	**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 **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**?.

The p-value associated with the ab_page is 0.19. It differs from the p-value obtained in part-2 because the null hypothesis test of regression model checks for the probability that there is no relationship between the ab_page and the conversion. But the null hypothesis in Part II tests for the probability of old page being better or as good as the new page.

The p-value here suggests that we to accept the null hypothesis. So for the regression model, we can safely conclude that the version of the page is statistically insignificant. The version of the page is not associated with changes in conversion factor.

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?

1. It is always a good idea to consider other factors into the regression model as we do not want to end up with wrong conclusions by ignoring significant factors that relate to the response variable.
2. No. Adding additional factors lead to misleading results in the summary only for datasets where there are not enough samples. We do have large number of samples for this case. So there are no disadvantages of adding additional terms.

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 (https://pandas.pydata.org/pandas-docs/stable/generated/pandas.DataFrame.join.html) 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 [48]:
```
countries_df = pd.read_csv('./countries.csv')
df_new = countries_df.set_index('user_id').join(df2.set_index('user_id'), how=
'inner')
df_new.head()
```

Out[48]:

	country	timestamp	group	landing_page	converted	ab_page	intercep
user_id							
834778	UK	2017-01-14 23:08:43.304998	control	old_page	0	0	1
928468	US	2017-01-23 14:44:16.387854	treatment	new_page	0	1	1
822059	UK	2017-01-16 14:04:14.719771	treatment	new_page	1	1	1
711597	UK	2017-01-22 03:14:24.763511	control	old_page	0	0	1
710616	UK	2017-01-16 13:14:44.000513	treatment	new_page	0	1	1

In [49]:
```
### Create the necessary dummy variables
df_new[['US','CA']] = pd.get_dummies(df_new['country'])[['US', 'CA']]

#del df_new['US']
#df_new.tail()
```

In [50]:
```
df_new['intercept']=1
df_new.tail()
```

Out[50]:

	country	timestamp	group	landing_page	converted	ab_page	intercep
user_id							
653118	US	2017-01-09 03:12:31.034796	control	old_page	0	0	1
878226	UK	2017-01-05 15:02:50.334962	control	old_page	0	0	1
799368	UK	2017-01-09 18:07:34.253935	control	old_page	0	0	1
655535	CA	2017-01-09 13:30:47.524512	treatment	new_page	0	1	1
934996	UK	2017-01-09 00:30:08.377677	control	old_page	0	0	1

In [51]:
```
import statsmodels.api as sm
logit_mod = sm.Logit(df_new['converted'], df_new[['intercept','ab_page','CA',
'US']])
results=logit_mod.fit()
results.summary()
```

```
Optimization terminated successfully.
         Current function value: 0.366113
         Iterations 6
```

Out[51]:

Logit Regression Results

Dep. Variable:	converted	**No. Observations:**	290584
Model:	Logit	**Df Residuals:**	290580
Method:	MLE	**Df Model:**	3
Date:	Wed, 24 Jan 2018	**Pseudo R-squ.:**	2.323e-05
Time:	20:31:20	**Log-Likelihood:**	-1.0639e+05
converged:	True	**LL-Null:**	-1.0639e+05
		LLR p-value:	0.1760

	coef	**std err**	**z**	**P>|z|**	**[0.025**	**0.975]**
intercept	-1.9794	0.013	-155.415	0.000	-2.004	-1.954
ab_page	-0.0149	0.011	-1.307	0.191	-0.037	0.007
CA	-0.0506	0.028	-1.784	0.074	-0.106	0.005
US	-0.0099	0.013	-0.743	0.457	-0.036	0.016

The pvalues in the summary suggest that country the user belongs to, has no impact on conversion.

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 the below 3 cells, separate datasets have been created for each of the countries, one in each cell, in order to analyse whether the way page influences conversion is different from one country to another.

In [ ]:
```
df_US=df_new.loc[df_new['country'] == 'US']
df_US['intercept']=1
df_US.tail()
```

In [ ]:
```
### Fit Your Linear Model And Obtain the Results
import statsmodels.api as sm
logit_mod = sm.Logit(df_US['converted'], df_US[['intercept','ab_page']])
results=logit_mod.fit()
results.summary()
```

In [ ]:
```
df_UK=df_new.loc[df_new['country'] == 'UK']
df_UK['intercept']=1
logit_mod = sm.Logit(df_UK['converted'], df_UK[['intercept','ab_page']])
results=logit_mod.fit()
results.summary()
```

In [ ]:
```
df_CA=df_new.loc[df_new['country'] == 'CA']
df_CA['intercept']=1
logit_mod = sm.Logit(df_CA['converted'], df_CA[['intercept','ab_page']])
results=logit_mod.fit()
results.summary()
```

In [ ]:
```
And looking at the p-values of ab_page in all 3 models, it can be concluded th
at there is no significant effect of page version
on conversion, when looking at the interaction between page and country.
```

Conclusions

Congratulations on completing the project!

Gather Submission Materials

Once you are satisfied with the status of your Notebook, you should save it in a format that will make it easy for others to read. You can use the **File -> Download as -> HTML (.html)** menu to save your notebook as an .html file. If you are working locally and get an error about "No module name", then open a terminal and try installing the missing module using `pip install <module_name>` (don't include the "<" or ">" or any words following a period in the module name).

You will submit both your original Notebook and an HTML or PDF copy of the Notebook for review. There is no need for you to include any data files with your submission. If you made reference to other websites, books, and other resources to help you in solving tasks in the project, make sure that you document them. It is recommended that you either add a "Resources" section in a Markdown cell at the end of the Notebook report, or you can include a `readme.txt` file documenting your sources.

Submit the Project

When you're ready, click on the "Submit Project" button to go to the project submission page. You can submit your files as a .zip archive or you can link to a GitHub repository containing your project files. If you go with GitHub, note that your submission will be a snapshot of the linked repository at time of submission. It is recommended that you keep each project in a separate repository to avoid any potential confusion: if a reviewer gets multiple folders representing multiple projects, there might be confusion regarding what project is to be evaluated.

It can take us up to a week to grade the project, but in most cases it is much faster. You will get an email once your submission has been reviewed. If you are having any problems submitting your project or wish to check on the status of your submission, please email us at dataanalyst-project@udacity.com. In the meantime, you should feel free to continue on with your learning journey by beginning the next module in the program.

In [ ]:
```
Based on the above analyses, we can conclude that it is best for the website t
o keep the old page, as all the different methods
of analysis tried out here prove that the old page is as good as the new page
and the conversion rate for new page is slightly
less than that of the old page.
```