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.

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]:
```
data=pd.read_csv('ab_data.csv')
data.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]:
```
data.shape[0]
```

Out[3]:
```
294478
```

c. The number of unique users in the dataset.

In [4]:
```
# This function help me count all the unique values noting that I choose to use the 'us
data['user_id'].nunique()
```

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

d. The proportion of users converted.

In [5]:
```
(data.converted==1).mean()
```

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

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

In [6]:
```
#Here I tried to find out how many times "new_page" is lined up with "control" and this
data.query('landing_page == "new_page" and group == "control"').user_id.size + data.que
```

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

f. Do any of the rows have missing values?

In [7]:
```
# By using this function I figured that there is no missing values in any raw.
data.isnull().sum()
```

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 order to remove the raws that don't line up together for **'landing_page'** and **'treatment'** , I will

assigne the updated columns to list then append them to the new dataframe.

In [8]:
```
A = data.query('landing_page == "old_page" and group == "control"')
B = data.query('landing_page == "new_page" and group == "treatment"')
df2 =A.append(B, ignore_index=True)
df2.head()
```

Out[8]:

	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	864975	2017-01-21 01:52:26.210827	control	old_page	1
3	936923	2017-01-10 15:20:49.083499	control	old_page	0
4	719014	2017-01-17 01:48:29.539573	control	old_page	0

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].sh
```

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(keep=False)]
```

Out[11]:

	user_id	timestamp	group	landing_page	converted
146212	773192	2017-01-09 05:37:58.781806	treatment	new_page	0
146678	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]:
```
# This function will select certain raw(in this case the duplicated one) and show the i
df2.iloc[146678]
```

Out[12]:
```
user_id                            773192
timestamp       2017-01-14 02:55:59.590927
group                          treatment
landing_page                    new_page
converted                              0
Name: 146678, dtype: object
```

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

In [13]:
```
df2.drop(146212,inplace=True)
df2 = df2.drop_duplicates(subset='user_id', keep="first")
```

In [14]:
```
#The purpose of this stetement is to check if the drop has taken place and affected 'df
df2[df2.user_id.duplicated(keep=False)]
#We removed one record so we ended up with 290584 records
df2.shape[0]
```

Out[14]: 290584

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 [15]:
```
df2.converted.mean()
```

Out[15]: 0.11959708724499628

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

In [16]:
```
df2.query('group == "control"').converted.mean()
```

Out[16]: 0.1203863045004612

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

In [17]:
```
df2.query('group == "treatment"').converted.mean()
```

Out[17]: 0.11880806551510564

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

In [18]:
```
df2.query('landing_page == "new_page"').user_id.size / df2.user_id.size
```

Out[18]: 0.5000619442226688

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.

By looking at the results, it is noticeable that there is no big defferences when comparing the means of the control groups **(12%)** and treament group **(11%)** who converted, which is actually alomost the same if we round it (0.1188).In my opinion this is not solid evidence that we can relay on in order to make a decision saying that the new page is actually leading to more conversions.

Your answer goes here.

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

$$H_0 : p_{new} - p_{old} \leq 0$$
$$H_1 : p_{new} - p_{old} > 0$$

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

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

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

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

In [19]:
```
#Converted success rate
df2.converted.mean()
```

Out[19]: 0.11959708724499628

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

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

Out[20]: 0.11959708724499628

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

In [21]:
```
p_old=df2.converted.mean()
p_old
```

Out[21]: 0.11959708724499628

According to the previuos assumption " $pnew$ and $pold$ both have "true" success rates equal to the converted success rate regardless of page - that is $pnew$ and $pold$ are equal." in point 2 Part||, all values are the same. Approximately 12%

c. What is n_{new}?

In [22]:
```
#I decided to use "nunique()" function to make sure that I get the exact count of the p
n_new=df2.query('landing_page == "new_page"')['user_id'].nunique()
n_new
```

Out[22]: 145310

d. What is n_{old}?

In [23]:
```
n_old=df2.query('landing_page == "old_page"')['user_id'].nunique()
n_old
```

Out[23]: 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 [24]:
```
new_page_converted = np.random.choice([1, 0], size=n_new, p=[p_new, (1-p_new)])
new_page_converted.mean()
```

Out[24]: 0.11923473952240038

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 [25]:
```
old_page_converted= np.random.choice([1, 0], size=n_old, p=[p_old, (1-p_old)])
old_page_converted.mean()
```

Out[25]: 0.1197599019783306

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

In [26]:
```
obs_diff = new_page_converted.mean()-old_page_converted.mean()
obs_diff
```

Out[26]: -0.0005251624559302198

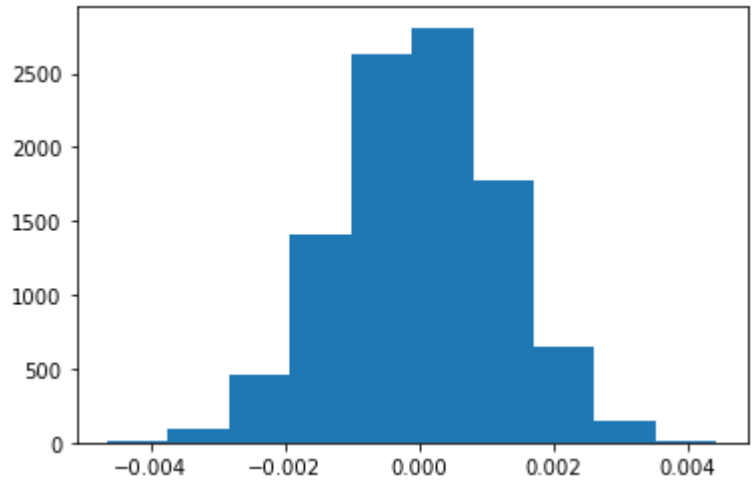
h. Simulate 10,000 p_{new} - p_{old} values using this same process similarly to the one you calculated in parts **a. through g.** above. Store all 10,000 values in a numpy array called **p_diffs**.

In [27]:
```
# After several attempts of bootstrapping using the for loop and np.random.choice this
p_diffs=[]
new_converted_simulation = np.random.binomial(n_new, p_new, 10000)/n_new
old_converted_simulation = np.random.binomial(n_old, p_old, 10000)/n_old
p_diffs = new_converted_simulation - old_converted_simulation
```

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.

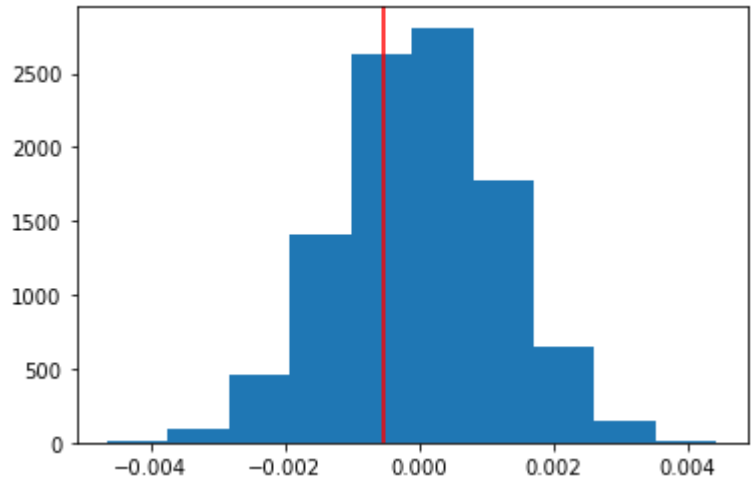
```
plt.hist(p_diffs);
```

In [28]:



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

In [29]:
```
plt.hist(p_diffs);
plt.axvline(x= obs_diff, color='r');
```



In [36]:
```
# Calculating the p-value noting that I decided to write the constent insted of the "ob
p_value=(p_diffs > -0.001576)
p_value.mean()
```

Out[36]: 0.9056

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

Answer In part j I have computed the proportion of the p_diffs (sampels differences) that are greater than the actual difference observed. In scientific studies this is called the p-value, which helps us determine the significance of the results in relation to the null hypothesis.
The difference between the new and old pages is not that significant as the calculated p-value is (0.9) which is grater than 0.05 and that wont allow us to reject the null hypothesis.

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 [32]:
```
import statsmodels.api as sm

converted_old = len(df2[df2.landing_page == 'old_page'][df2.converted == 1])
converted_new = len(df2[df2.landing_page == 'new_page'][df2.converted == 1])
n_old = len(df2[df2.landing_page == 'old_page'])
n_new = len(df2[df2.landing_page == 'new_page'])
```

```
<ipython-input-32-e28b0040c16f>:3: UserWarning: Boolean Series key will be reindexed to
match DataFrame index.
  converted_old = len(df2[df2.landing_page == 'old_page'][df2.converted == 1])
<ipython-input-32-e28b0040c16f>:4: UserWarning: Boolean Series key will be reindexed to
match DataFrame index.
  converted_new = len(df2[df2.landing_page == 'new_page'][df2.converted == 1])
```

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 [33]:
```
z_score, p_value = sm.stats.proportions_ztest([converted_old, converted_new], [n_old, n
z_score, p_value
# If we rounded (0.18988337448195103) we get 0.9056 which is the accurate value
```

Out[33]: `(1.3109241984234394, 0.18988337448195103)`

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 value is 1.31 and the p-value is 0.905 so, there is no difference from our findings in j. and k. . This leads us to similar conclusion that we can not reject the null hypothesis as 1.64is grater than the z-score .There is no difference between the old and new pages.

Part III - A regression approach

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

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

Answer

In this part we are trying to predict whether a user will convert or not depending on the page. The type of regression used is for similar cases is Logistic Regression as it models a binary dependent variable.

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 [37]:
```
#To add intercept column
df2['intercept'] = 1

#Create dummy variable column
df2['ab_page'] = pd.get_dummies(df2['group'])['treatment']

df2.head()
```

Out[37]:

	user_id	timestamp	group	landing_page	converted	intercept	ab_page
0	851104	2017-01-21 22:11:48.556739	control	old_page	0	1	0
1	804228	2017-01-12 08:01:45.159739	control	old_page	0	1	0
2	864975	2017-01-21 01:52:26.210827	control	old_page	1	1	0
3	936923	2017-01-10 15:20:49.083499	control	old_page	0	1	0
4	719014	2017-01-17 01:48:29.539573	control	old_page	0	1	0

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 [38]:
```
import statsmodels.api as sm
#Fitting regression line by following lesson 14 in Udacity.
lm = sm.OLS(df2['converted'], df2[['intercept', 'ab_page']])
results=lm.fit()
```

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

In [39]:
```
results.summary()
```

Out[39]:

OLS Regression Results

Dep. Variable:	converted	**R-squared:**	0.000
Model:	OLS	**Adj. R-squared:**	0.000
Method:	Least Squares	**F-statistic:**	1.719
Date:	Thu, 12 Nov 2020	**Prob (F-statistic):**	0.190
Time:	04:06:03	**Log-Likelihood:**	-85267.
No. Observations:	290584	**AIC:**	1.705e+05
Df Residuals:	290582	**BIC:**	1.706e+05
Df Model:	1		
Covariance Type:	nonrobust		

	coef	**std err**	**t**	**P>|t|**	**[0.025**	**0.975]**

intercept	0.1204	0.001	141.407	0.000	0.119	0.122
ab_page	-0.0016	0.001	-1.311	0.190	-0.004	0.001

Omnibus:	125553.456	**Durbin-Watson:**	2.000
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	414313.355
Skew:	2.345	**Prob(JB):**	0.00
Kurtosis:	6.497	**Cond. No.**	2.62

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.

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**?

Answer

The p-value associated with ab_page: 0.190
By looking at the hypotheses in **Part II** we are assuming that the new page will cause more conversion rate in the null hypotheses $H_1 : p_{new} - p_{old} > 0$. On the other hand the alternative $H_0 : p_{new} - p_{old} \leq 0$ hypotheses states that the old page will cause more or at least similar conversion rate to the new page.
In **Part III** we are using a linear model with binary values to calculate the p-value. The null hypothesis the difference between the pages is equal to 0, and the alternative hypothesis is the difference between the pages is greater or less than 0

f. Now, you are considering other things that might influence whether or not an individual converts. Discuss why it is a good idea to consider other factors to add into your regression model. Are there any disadvantages to adding additional terms into your regression model?

Answer

In my opinion there is one more feature we can consider according to the data we are working with which is **timestamp**, maybe this will give us an idea about certain times may witness more conversions compare to other times, however that may cause complexity if we tride to apply a regression model here as this is a linear model and works the best with a single predictor variable X used to model the response variable Y.It might be advised to use a Multiple Linear Regression models that describe how a single response variable Y depends linearly on a number of predictor variables.

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 variables.** Provide the statistical output as well as a written response to answer this question.

In [40]:
```
countries_df = pd.read_csv('countries.csv')
df_new = countries_df.set_index('user_id').join(df2.set_index('user_id'), how='inner')
countries_df.head()
```

Out[40]:

	user_id	country
0	834778	UK
1	928468	US
2	822059	UK
3	711597	UK
4	710616	UK

In [41]:
```
df_new.head()
```

Out[41]:

user_id	country	timestamp	group	landing_page	converted	intercept	ab_page
834778	UK	2017-01-14 23:08:43.304998	control	old_page	0	1	0
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	1	0
710616	UK	2017-01-16 13:14:44.000513	treatment	new_page	0	1	1

In [42]:
```
### Create the necessary dummy variables

country_dummies = pd.get_dummies(countries_df['country'])
df_new = countries_df.join(country_dummies)
df_new.head()
```

Out[42]:

	user_id	country	CA	UK	US
0	834778	UK	0	1	0
1	928468	US	0	0	1
2	822059	UK	0	1	0
3	711597	UK	0	1	0

	user_id	country	CA	UK	US
4	710616	UK	0	1	0

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 [43]:
```
#Create the necessary additional columns
Regdf = df2.set_index('user_id').join(df_new.set_index('user_id'))
Regdf.head()
```

Out[43]:

user_id	timestamp	group	landing_page	converted	intercept	ab_page	country	CA	UK	US
851104	2017-01-21 22:11:48.556739	control	old_page	0	1	0	US	0	0	1
804228	2017-01-12 08:01:45.159739	control	old_page	0	1	0	US	0	0	1
864975	2017-01-21 01:52:26.210827	control	old_page	1	1	0	US	0	0	1
936923	2017-01-10 15:20:49.083499	control	old_page	0	1	0	US	0	0	1
719014	2017-01-17 01:48:29.539573	control	old_page	0	1	0	US	0	0	1

In [81]:
```
### Fit Your Linear Model And Obtain the Results
lm = sm.OLS(Regdf['converted'],Regdf[['intercept', 'UK', 'US']])
results = lm.fit()
results.summary()
```

Out[81]:

OLS Regression Results

Dep. Variable:	converted	**R-squared:**	0.000
Model:	OLS	**Adj. R-squared:**	0.000
Method:	Least Squares	**F-statistic:**	1.605
Date:	Thu, 12 Nov 2020	**Prob (F-statistic):**	0.201
Time:	02:29:58	**Log-Likelihood:**	-85267.
No. Observations:	290584	**AIC:**	1.705e+05
Df Residuals:	290581	**BIC:**	1.706e+05
Df Model:	2		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
intercept	0.1153	0.003	42.792	0.000	0.110	0.121

UK	0.0053	0.003	1.787	0.074	-0.001	0.011
US	0.0042	0.003	1.516	0.130	-0.001	0.010

Omnibus:	125552.384	**Durbin-Watson:**	2.000
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	414306.036
Skew:	2.345	**Prob(JB):**	0.00
Kurtosis:	6.497	**Cond. No.**	9.94

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.

By looking at the result summery above, we notice that the correlation coefficient for UK is 0.0053 and 0.0042 for US and these values are very small which means that the relation between the countries and the conversion rate is insignificant and that will lead us to keep our initial null hypothesis.

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 [ ]: