Analyze_ab_test_project

January 14, 2019

0.1 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!

0.2 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 [218]: #importing libraries
          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 [219]: #Reading in the dataset
          df = pd.read_csv('ab_data.csv')
          df.head(3)
```

Out[219]:

	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

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

```
In [220]: # number of rows in the dataset

          df.shape[0]
```

```
Out[220]: 294478
```

c. The number of unique users in the dataset.

```
In [221]: # The number unique users in the dataset

          user_total = df.nunique()['user_id']
          print("Number of unique users  : {}".format(user_total))
```

```
Number of unique users  : 290584
```

d. The proportion of users converted.

```
In [222]: # The proportion of users converted

          df.converted.mean()
```

```
Out[222]: 0.11965919355605512
```

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

```
In [223]: # Looking for rows where treatment/control doesn't line up
          df_t_not_n = df[(df['group'] == 'treatment') & (df['landing_page'] == 'old_page')]
          df_not_t_n = df[(df['group'] == 'control') & (df['landing_page'] == 'new_page')]

          # Add lengths
          mismatch= len(df_t_not_n) + len(df_not_t_n)

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

          mismatch
```

Out[223]: 3893

f. Do any of the rows have missing values?

```
In [224]: # Check for missing values?
          df.isnull().values.any()
```

Out[224]: False

Based on the cell above, there are no missing values in the dataset

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 [225]: # Copy dataframe
          df2 = df

          # Remove incriminating rows
          mismatch_index = mismatch_df.index
          df2 = df2.drop(mismatch_index)
```

```
In [226]: # Double Check all of the correct rows were removed - this should be 0
          df2[((df2['group'] == 'treatment') == (df2['landing_page'] == 'new_page')) == False]
```

Out[226]: 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 [227]: # No. of unique user_ids after cleaning our dataset.

          df2['user_id'].nunique()
```

Out[227]: 290584

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

```
In [228]: # Find duplicated user
          df2[df2.duplicated('user_id')]
```

Out[228]:

	user_id	timestamp	group	landing_page	converted
2893	773192	2017-01-14 02:55:59.590927	treatment	new_page	0

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

```
In [229]: #investigate details of rows with duplicate user ids
          df2[df2.duplicated(['user_id'], keep=False)]
```

Out[229]:

	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 [230]:
```
# No. of rows before removing the duplicate

df2.shape
```

Out[230]:
```
(290585, 5)
```

In [231]:
```
# Drop one of the rows which belongs to the repeated user_id

df2 = df2.drop_duplicates(subset='user_id');
```

In [232]:
```
# No. of rows after removing the duplicate

df2.shape
```

Out[232]:
```
(290584, 5)
```

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 [233]:
```
# The probability of user converting
print("Probability of user converted:", df2.converted.mean())
```

```
Probability of user converted: 0.11959708724499628
```

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

In [234]:
```
# Probability of control group converting
print("Probability of control group converting:",
      df2[df2['group']=='control']['converted'].mean())
```

```
Probability of control group converting: 0.1203863045004612
```

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

In [235]:
```
df2.query('group =="treatment"').converted.mean()
```

Out[235]:
```
0.11880806551510564
```

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

```
In [236]: # Probability an individual recieved new page
          print("Probability an individual recieved new page:",
                df2['landing_page'].value_counts()[0]/len(df2))
```

```
Probability an individual recieved new page: 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.

Based on the above output, it seems that the control group has a slightly higher conversion rate (0.1204) than the treatment group (0.1189). These results don't provide a solid evidence if one page leads to more conversions as we still don't know the significance of these results and the factors. We shall need to continue and define our test hypothesis and calculate p-value for the new and old pages.

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.

Put your answer here.

$$H_0 : p_{new} \leq p_{old}$$

$$H_1 : p_{new} > p_{old}$$

`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 [237]: # As per the instruction above, p_old = p_new = converted rate in ab_data.csv regard
          p_new = df2.converted.mean()
          p_new
```

```
Out[237]: 0.11959708724499628
```

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

```
In [238]: # As per the instruction above, p_old = p_new = converted rate in ab_data.csv regard
          p_old = df2.converted.mean()
          p_old

Out[238]: 0.11959708724499628
```

c. What is n_{new}?

```
In [239]: # Create a dataframe with all new page records from df2

          newPage_df = df2.query('landing_page == "new_page"')
          n_new = newPage_df.shape[0]
          n_new

Out[239]: 145310
```

d. What is n_{old}?

```
In [240]: # Create a dataframe with all old page records from df2

          oldPage_df = df2.query('landing_page == "old_page"')
          n_old = oldPage_df.shape[0]
          n_old

Out[240]: 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 [241]: new_page_converted = np.random.binomial(n_new,p_new)
          new_page_converted

Out[241]: 17307
```

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 [242]: 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 [243]: p_diff = (new_page_converted/n_new) - (old_page_converted/n_old)
          p_diff

Out[243]: -0.0003530414488831374
```

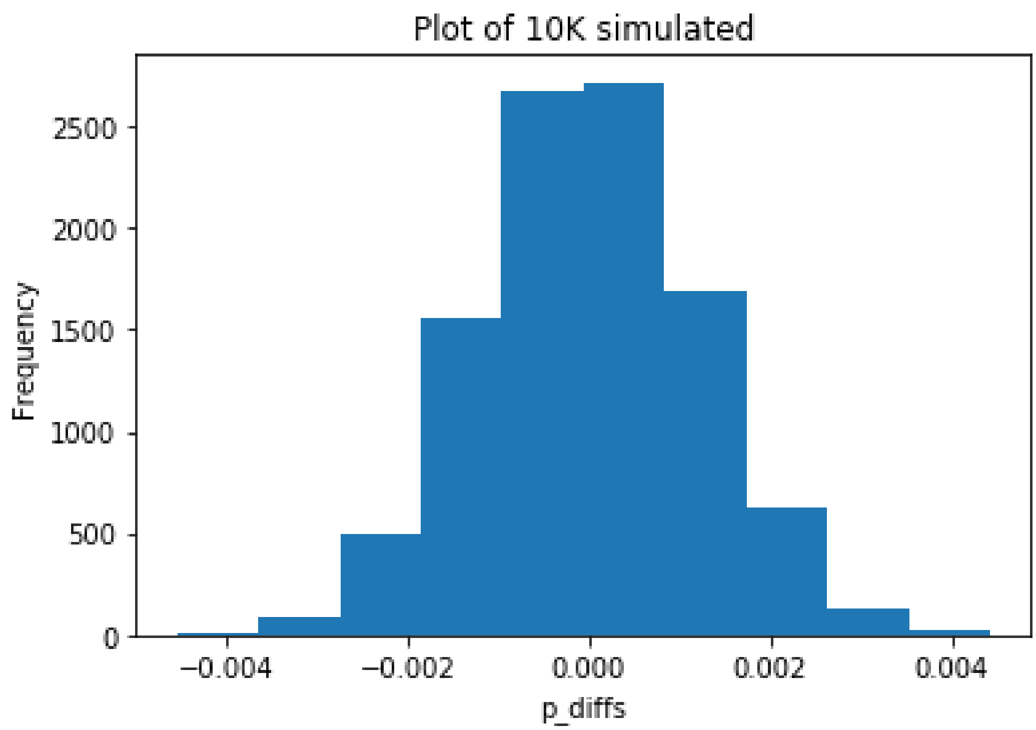
h. Simulate 10,000 p_{new} - p_{old} values using this same process similarly to the one you calculated in parts **a. through g.** above. Store all 10,000 values in a numpy array called **p_diffs**.

```
In [244]: p_diffs = []

          for _ in range(10000):
              new_converted_simulation = np.random.binomial(n_new,p_new)/n_new
              old_converted_simulation = np.random.binomial(n_old,p_old)/n_old
              diff = new_converted_simulation - old_converted_simulation
              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 [245]: plt.hist(p_diffs)
          plt.xlabel('p_diffs')
          plt.ylabel('Frequency')
          plt.title('Plot of 10K simulated');
```



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

```
In [246]: # Calculate the actual observed difference

          org_old_mean = df.query('group =="control"').converted.mean()
          org_new_mean = df.query('group =="treatment"').converted.mean()
          org_diff = org_new_mean - org_old_mean
```

```
# Convert p_diffs to array

p_diffs = np.array(p_diffs)

# Calculate the propotion of the p_diffs are greater than the actual difference obse
(p_diffs > org_diff).mean()
```

Out[246]: 0.8931

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?

Put your answer here. What we computed in part j. is called p-value in scientific studies. p-value is the probability of observing your statistic (or one more extreme in favor of the alternative) if the null hypothesis is true. In our case the p-value is so big that we can confidently say that we fail to reject 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 [247]:
```
import statsmodels.api as sm

convert_old = sum(df2.query("group == 'control'")['converted'])
convert_new = sum(df2.query("group == 'treatment'")['converted'])
n_old = len(df2.query("group == 'control'"))
n_new = len(df2.query("group == 'treatment'"))
```

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 [248]:
```
z_score, p_value = sm.stats.proportions_ztest([convert_old, convert_new], [n_old, n_

z_score, p_value
```

Out[248]: (1.3109241984234394, 0.9050583127590245)

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.**?

A z-score represents how many standard deviations away our data point is from the mean.
-A positive z-score suggests that our data point is on the right side of the mean line on the bell curve -p-value of 0.9050 is very close to the p-value we computed earlier. -With this computation, we can confidently say we fail to reject null hypothesis

```
In [249]: # Shows the significance of the z_score
          from scipy.stats import norm
          print(norm.cdf(z_score))
```

```
0.9050583127590245
```

```
In [250]: # Assuming 95% CI for one-sided test, as stated in part II.1

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

```
1.6448536269514722
```

Put your answer here. z_score is less than 1.6448, therefore, we would fail to reject the Null; which is consistent with the results in parts j & 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?

Put your answer here. As this is a Yes-No type of variable, the good approach would be 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 [251]: 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 [252]: import statsmodels.api as sm

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

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

```
In [253]: results = logit.fit()
          results.summary()
```

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

Out[253]:
```
<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:                Mon, 14 Jan 2019   Pseudo R-squ.:               8.077e-06
Time:                        15:50:02   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 **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**?

$$H_0 : p_{new} = p_{old}$$

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

The difference is, in part II, we performed a one-sided test, where in the logistic regression part, it is two-sided 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?

Put your answer here. Considering other factors is a good idea as these factors may contribute to the significance of the test results and leads to more accurate decisions. One of the disadvantages for adding additional terms into the regression model is Simpson's paradox where the combined impact of different variables disappears or reverses when these variables are combined, but appears where these variables are tested individually.

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 [254]: #importing countries data set
          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[254]:         country                   timestamp      group landing_page  \
          user_id
          834778       UK  2017-01-14 23:08:43.304998    control     old_page
          928468       US  2017-01-23 14:44:16.387854  treatment     new_page
          822059       UK  2017-01-16 14:04:14.719771  treatment     new_page
          711597       UK  2017-01-22 03:14:24.763511    control     old_page
          710616       UK  2017-01-16 13:14:44.000513  treatment     new_page

                   converted  intercept  control  treatment
          user_id
          834778           0          1        1          0
          928468           0          1        0          1
          822059           1          1        0          1
          711597           0          1        1          0
          710616           0          1        0          1
```

```
In [255]: # Create the necessary dummy variables
          df_new[['CA','UK','US']] = pd.get_dummies(df_new['country'])[['CA','UK', 'US']]
```

```
In [256]: # let's consider US being our baseline, therefore, we drop US
          df_new.drop(['US'], axis=1, inplace=True)
```

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 [257]: df_new.head()
```

```
Out[257]:         country                   timestamp      group landing_page  \
          user_id
          834778       UK  2017-01-14 23:08:43.304998    control     old_page
          928468       US  2017-01-23 14:44:16.387854  treatment     new_page
          822059       UK  2017-01-16 14:04:14.719771  treatment     new_page
          711597       UK  2017-01-22 03:14:24.763511    control     old_page
          710616       UK  2017-01-16 13:14:44.000513  treatment     new_page

                   converted  intercept  control  treatment  CA  UK
          user_id
```

```
834778            0            1         1            0    0    1
928468            0            1         0            1    0    0
822059            1            1         0            1    0    1
711597            0            1         1            0    0    1
710616            0            1         0            1    0    1
```

In [258]:
```python
### Fit Your Linear Model And Obtain the Results
df_new['intercept'] = 1

logit_mod = sm.Logit(df_new['converted'], df_new[['intercept','CA','UK']])
results = logit_mod.fit()
results.summary()
```

```
Optimization terminated successfully.
         Current function value: 0.366116
         Iterations 6
```

Out[258]:
```
<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:                Mon, 14 Jan 2019   Pseudo R-squ.:               1.521e-05
Time:                        15:50:12   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     -1.9967      0.007   -292.314      0.000      -2.010      -1.983
CA            -0.0408      0.027     -1.518      0.129      -0.093       0.012
UK             0.0099      0.013      0.746      0.456      -0.016       0.036
==============================================================================
"""
```

0.2.1 Conclusions

Within the framework this project, we tried to understand whether the company should implement a new page or keep the old page:

Probability based approach: -We found that probability of an individual receiving the new page is 0.5001 -Meaning, there is almost the same chance that an individual received the old page

A/B test: -In A/B test we set up our hypothesis to test if new page results in better conversion or not -We simulated our user groups with respect to conversions -We found the p_value -With such a p-value, we failed to reject null hypothesis -By using the built-in stats.proportions_ztest we computed z-score and p-value which confirmed our earlier p-value and failure to reject null hypothesis

Regression Approach: -By further adding geographic location of the users, Looking at the results above, we may conclude there is no significant effect on the convertion based on the country.

```
In [ ]:
```