Analyze_ab_test_results_notebook

June 6, 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

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.shape[0]
```

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

c. The number of unique users in the dataset.

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

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

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.

There are two times that 'new_page' and 'treatment' don't match:

- Control group, new_page
- Treatment group, old_page

```
In [6]: # Find the amount of times treatment group lands incorrectly on old_page
        mismatch_grp1 = df.query("group == 'treatment' and landing_page == 'old_page'")
        print("The number of times that the Treatment Group lands incorrectly on the old_page is

        # Find the amount of times where the control group lands incorrectly on new_page
        mismatch_grp2 = df.query("group == 'control' and landing_page == 'new_page'")
        print("The number of times the Control Group incorrectly lands on the new_page is: {}".f

        print("The number of times the new_page and Treatment Group don't line up is: {}".format
```

```
The number of times that the Treatment Group lands incorrectly on the old_page is: 1965
The number of times the Control Group incorrectly lands on the new_page is: 1928
The number of times the new_page and Treatment Group don't line up is: 3893
```

f. Do any of the rows have missing values?

- There are no missing values (294,478 values for each column)

```
In [7]: 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
```

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]: df2 = df[((df['group'] == 'treatment') == (df['landing_page'] == 'new_page'))]
        # check to make sure there are 290,585 rows
        df2.shape[0]
```

```
Out[8]: 290585
```

```
In [9]: # Double Check all of the correct rows were removed - this should be 0
        df2[((df2['group'] == 'treatment') == (df2['landing_page'] == 'new_page')) == False].sha
```

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

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

a. How many unique **user_id**s are in **df2**?

```
In [10]: df2.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()]
```

```
Out[11]:       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 [12]: df2[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  2017-01-14 02:55:59.590927  treatment     new_page          0
```

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

```
In [13]: df2 = df2.drop(1899)
```

```
In [14]: #check and confirm entry
         len(df['user_id'].unique())
```

```
Out[14]: 290584
```

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 [15]: df_grp = df.groupby('group')
         df_grp.describe()
```

```
Out[15]:           converted                                                 user_id  \
                       count      mean       std  min  25%  50%  75%  max     count
         group
         control    147202.0  0.120399  0.325429  0.0  0.0  0.0  0.0  1.0  147202.0
         treatment  147276.0  0.118920  0.323695  0.0  0.0  0.0  0.0  1.0  147276.0

                                                                                \
                              mean           std       min       25%       50%
         group
         control    788123.098035  91278.896888  630002.0  709287.0  788053.5
         treatment  787825.226283  91142.800641  630000.0  708729.5  787837.5


                         75%       max
         group
         control    867155.50  945998.0
         treatment  866693.75  945999.0
```

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

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

```
Out[16]: 0.1203863045004612
```

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

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

```
Out[17]: 0.11880806551510564
```

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

```
In [18]: df2.query('landing_page == "new_page"').shape[0]/df2.shape[0]
```

```
Out[18]: 0.5000619442226688
```

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.

The following overall conversions, control and treatment as follows:

- The overall conversions: 11.96%
- The control group (old page) conversions: 12.04%
- The treatment group (new page) conversions: 11.88%
- We can see that the old page does slightly better.

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.

p_{old} and p_{new},

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 [26]: p_new = df2['converted'].mean()
         print(p_new)
```

```
0.119597087245
```

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

```
In [27]: p_old = df2['converted'].mean()
         print(p_old)
```

```
0.119597087245
```

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

```
In [28]: n_new = df2.query("group == 'treatment'").shape[0]
         n_new
```

```
Out[28]: 145310
```

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

```
In [29]: n_old = df2.query("group == 'control'").shape[0]
         n_old
```

```
Out[29]: 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 [30]: new_page_converted = np.random.choice([1, 0], size=n_new, p=[p_new, (1-p_new)])
```

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

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

```
In [33]: new_page_converted.mean() - old_page_converted.mean()
```

```
Out[33]: 0.0003974081200735502
```

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**.

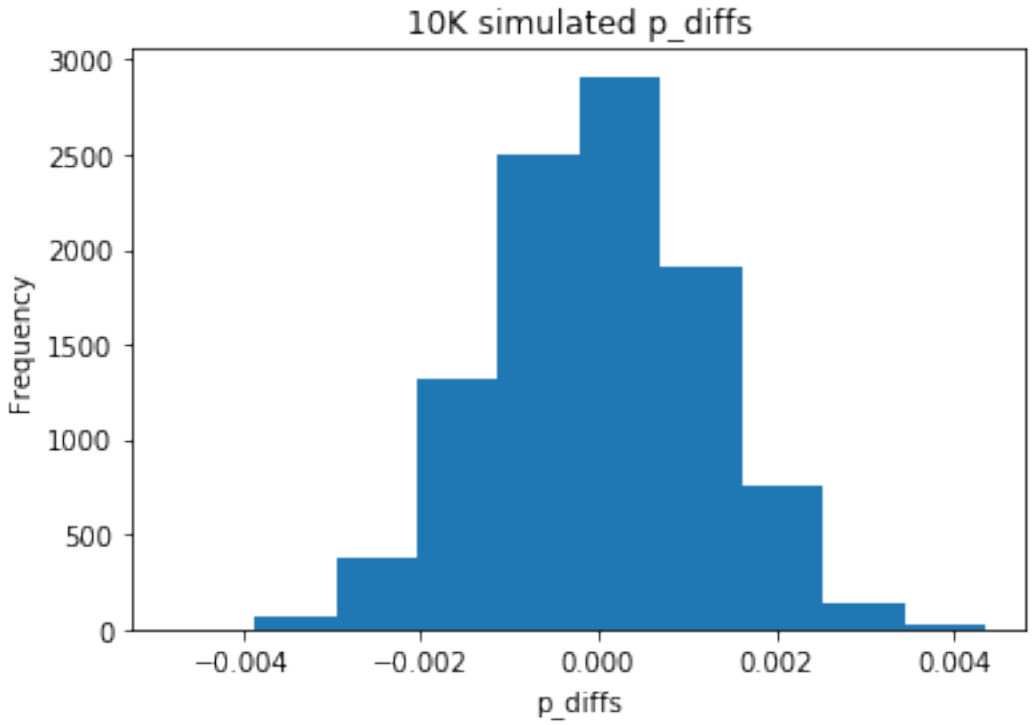
In [37]:
```
p_diffs = []

for _ in range(10000):
    new_page_converted = np.random.choice([1, 0], size=n_new, p=[p_new, (1-p_new)]).mea
    old_page_converted = np.random.choice([1, 0], size=n_old, p=[p_old, (1-p_old)]).mea
    diff = new_page_converted - old_page_converted
    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 [36]:
```
#Plot historgram

plt.hist(p_diffs)
plt.xlabel('p_diffs')
plt.ylabel('Frequency')
plt.title('10K simulated p_diffs');
```



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

In [38]:
```
# compute difference from original dataset ab_data.csv
act_diff = df[df['group'] == 'treatment']['converted'].mean() -  df[df['group'] == 'con
act_diff
```

```
Out[38]: -0.0014795997940775518
```

```
In [39]: p_diffs = np.array(p_diffs)
         p_diffs
```

```
Out[39]: array([ -5.80261506e-04,  -7.08655925e-05,  -1.75055270e-03, ...,
                  2.55364284e-05,  -4.22168509e-04,  -1.06893346e-03])
```

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?

- We are computing p-values.
- The formula in part j computed the p-value, which is the probability that we will observe this statistic, given the null hypothesis is true.
- Type I error rate of 5%, and Pold > Alpha, we fail to reject the null.
- In conclusion, old pages perform better than new pages.

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 [40]: import statsmodels.api as sm
         df2.head(6)
```

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

```
Out[40]:    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
         5   936923  2017-01-10 15:20:49.083499    control     old_page          0
```

```
In [41]: 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 [42]: z_score, p_value = sm.stats.proportions_ztest([convert_old, convert_new], [n_old, n_new
         print(z_score, p_value)
```

```
1.31092419842 0.905058312759
```

n. What do the z-score and p-value you computed in the previous question mean for the conversion rates of the old and new pages? Do they agree with the findings in parts **j.** and **k.**?

```
In [43]: from scipy.stats import norm

         #What is our z-score
         print(norm.cdf(z_score))
         print(norm.ppf(1-(0.05)))
```

```
0.905058312759
1.64485362695
```

Put your answer here.

- As per the above output, the z-score and p-value computations differ. This is because the conversion rates for the old and new pages are not statistically different.
- The z-score of 1.31092419842 means that the test statistic is less than the cirtical value of 1.64485362695. - Therefore, this does not reject the null hypothesis, so we will accept the null hypothesis.
- These values agree with the findings mention in parts j and k.

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?

- 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 [51]: df['intercept']=1
         df[['control', 'treatment']] = pd.get_dummies(df['group'])
```

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 [52]: import statsmodels.api as sm
         logit = sm.Logit(df['converted'],df[['intercept','treatment']])
         results = logit.fit()
```

```
Optimization terminated successfully.
         Current function value: 0.366243
         Iterations 6
```

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

```
In [53]: results.summary()
```

```
Out[53]: <class 'statsmodels.iolib.summary.Summary'>
"""
                           Logit Regression Results
==============================================================================
Dep. Variable:              converted   No. Observations:               294478
Model:                          Logit   Df Residuals:                   294476
Method:                           MLE   Df Model:                            1
Date:                Sat, 06 Jun 2020   Pseudo R-squ.:               7.093e-06
Time:                        10:30:56   Log-Likelihood:            -1.0785e+05
converged:                       True   LL-Null:                   -1.0785e+05
                                        LLR p-value:                    0.2161
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
intercept     -1.9887      0.008   -248.297      0.000      -2.004      -1.973
treatment     -0.0140      0.011     -1.237      0.216      -0.036       0.008
==============================================================================
"""
```

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**?

The p-value associated with the ab_page is 0.19 which is significantly lower than the one in Part II which was 0.9. The p-value here suggests that that new page is not statistically significant as 0.19 > 0.05.

- H_0 : p_{new} - p_{old} = 0
- H_1 : p_{new} - p_{old} != 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?

- There are a few factors we should take in consideration when using the regression model, as this can influence data conversions. For example demographics such as age, ethnicity, gender can influence the conversions.
- We can find new trends.
- Some Disadvantages is that it may produce inaccurate results due to correlated error.

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.

```
In [54]: countries_df = pd.read_csv('./countries.csv')
         countries_df.head()
```

Out[54]:

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

```
In [55]: df_new = countries_df.set_index('user_id').join(df2.set_index('user_id'), how='inner')
         df_new.head()
```

Out[55]:

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

```
In [56]: df_new['country'].value_counts()
```

```
Out[56]: US    203619
         UK     72466
         CA     14499
         Name: country, dtype: int64
```

```
In [57]: ### Create the necessary dummy variables
         df_new[['CA','UK','US']]=pd.get_dummies(df_new['country'])
         df_new.head()
```

Out[57]:

user_id	country	timestamp	group	landing_page	\
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  CA  UK  US
user_id
834778           0   0   1   0
928468           0   0   0   1
822059           1   0   1   0
711597           0   0   1   0
710616           0   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 [58]:
```
df['intercept'] = 1
```

```
log_mod = sm.Logit(df_new['converted'], df_new[['CA', 'US']])
results = log_mod.fit()
results.summary()
```

```
Optimization terminated successfully.
         Current function value: 0.447174
         Iterations 6
```

Out[58]:
```
<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:                Sat, 06 Jun 2020   Pseudo R-squ.:                -0.2214
Time:                        10:31:13   Log-Likelihood:            -1.2994e+05
converged:                       True   LL-Null:                   -1.0639e+05
                                        LLR p-value:                     1.000
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
CA            -2.0375      0.026    -78.364      0.000      -2.088      -1.987
US            -1.9967      0.007   -292.314      0.000      -2.010      -1.983
==============================================================================
"""
```

In [59]:
```
np.exp(results.params)
```

Out[59]:
```
CA    0.130350
US    0.135779
dtype: float64
```

```
In [60]: df.groupby('group').mean()['converted']
```

```
Out[60]: group
         control      0.120399
         treatment    0.118920
         Name: converted, dtype: float64
```

Conclusion

- In this analysis and given by the results in this data set, there are indications that we can accept the nulll hypothesis as there is not a significant difference in the conversion rates between the control group and the treatment group.
- This indicates that that the null hypothesis can be accepted and keep the exisiting page. Therefore, the recommendation is to remain with the old version.
- There are some limitations of this analysis due to the results are limited to the available dataset and effects of change aversion and novelty effects may influence the results.

0.3 References

- Stackoverflow: https://thispointer.com/pandas-find-duplicate-rows-in-a-dataframe-based-on-all-or-selected-columns-using-dataframe-duplicated-in-python/
- Stackoverflow: https://stackoverflow.com/questions/11587782/creating-dummy-variables-in-pandas-for-python
- Stackoverflow:https://stackoverflow.com/questions/14657241/how-do-i-get-a-list-of-all-the-duplicate-items-using-pandas-in-python
- Github Kaledimad https://github.com/khaledimad
- Stackoveflow: https://stackoverflow.com/questions/18172851/deleting-dataframe-row-in-pandas-based-on-column-value
- Udacity Videos- Github Kaledimad https://github.com/khaledimad
- Medium: https://towardsdatascience.com/a-summary-of-udacity-a-b-testing


```
In [61]: from subprocess import call
         call(['python', '-m', 'nbconvert', 'Analyze_ab_test_results_notebook.ipynb'])
```

```
Out[61]: 0
```