Analyze_ab_test_results_notebook

November 11, 2019

0.1 Analyze A/B Test Results

0.2 Table of Contents

Introduction

This A/B test run by an e-commerce website. we will 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.

Part I - Probability

importing 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)
```

a. Read in the dataset

```
In [2]: df = pd.read_csv('ab_data.csv')
```

```
In [3]: df.head()
```

```
Out[3]:    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 [4]: df.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 294478 entries, 0 to 294477
Data columns (total 5 columns):
user_id         294478 non-null int64
timestamp       294478 non-null object
group           294478 non-null object
landing_page    294478 non-null object
converted       294478 non-null int64
dtypes: int64(2), object(3)
memory usage: 11.2+ MB
```

c. The number of unique users in the dataset.

```
In [5]: df['user_id'].nunique()

Out[5]: 290584
```

d. The proportion of users converted.

```
In [6]: df['converted'].mean()

Out[6]: 0.11965919355605512
```

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

```
In [7]: df[df['landing_page'] == "new_page"].query("group != 'treatment'").count() + df[df['grou

Out[7]: user_id         3893
        timestamp       3893
        group           3893
        landing_page    3893
        converted       3893
        dtype: int64
```

f. Do any of the rows have missing values?

```
In [8]: df.isna().sum()

Out[8]: user_id         0
        timestamp       0
        group           0
        landing_page    0
        converted       0
        dtype: int64
```

2. For the rows where **treatment** does not match with **new_page** or **control** does not match with **old_page**, we cannot be sure if this row truly received the new or old page. Use **Quiz 2** in the classroom to figure out how we should handle these rows.

a. Now use the answer to the quiz to create a new dataset that meets the specifications from the quiz. Store your new dataframe in **df2**.

```
In [9]: df2 = df

In [10]: df2.drop(df[df['landing_page'] == "new_page"].query("group != 'treatment'").index, inpl
         df2.drop(df[df['group'] == "treatment"].query("landing_page != 'new_page'").index, inpl

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

Out[11]:

	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

```
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].sh
```

```
Out[12]: 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 [13]: df2['user_id'].nunique()

Out[13]: 290584
```

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

```
In [14]: df2[df2['user_id'].duplicated()]['user_id']

Out[14]: 2893    773192
         Name: user_id, dtype: int64
```

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

```
In [15]: df2[df2['user_id'].duplicated()]
```

Out[15]:

	user_id	timestamp	group	landing_page	converted
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 [16]: df2.drop(2893, inplace= True)
```

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 [17]: df['converted'].mean()
```

```
Out[17]: 0.11959708724499628
```

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

```
In [18]: df2[df2['group'] == 'control'].query('converted == 1')['converted'].count()/df2[df2['gr
```

```
Out[18]: 0.1203863045004612
```

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

```
In [19]: df2[df2['group'] == 'treatment'].query('converted == 1')['converted'].count()/df2[df2['
```

```
Out[19]: 0.11880806551510564
```

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

```
In [20]: df2[df2['landing_page'] == 'new_page']['landing_page'].count()/df2['landing_page'].coun
```

```
Out[20]: 0.50006194422266881
```

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 probability of Conversion from new treatment page is nearly similart to the old page, Therfore we have no enough evidence to prove that the treatment page lead to higher 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.

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

$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.

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

```
In [21]: p_new1= df2.query('converted == 1')['converted'].count()/df2['landing_page'].count()
         p_new1

Out[21]: 0.11959708724499628
```

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

```
In [22]: p_old1= df2.query('converted == 1')['converted'].count()/df2['landing_page'].count()
         p_old1

Out[22]: 0.11959708724499628
```

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

```
In [23]: n_new1= df2[df2['group'] == 'treatment']['group'].count()
         n_new1

Out[23]: 145310
```

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

```
In [24]: n_old1 = df2[df2['group'] == 'control']['group'].count()
         n_old1

Out[24]: 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 [25]: new_page_converted  = np.random.choice(2, size= n_new1, p=[(1-p_new1), p_new1])
         new_page_converted

Out[25]: array([0, 0, 1, ..., 0, 0, 0])
```

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 [26]: old_page_converted = np.random.choice(2, size= n_old1, p=[(1-p_new1), p_new1])
         old_page_converted

Out[26]: array([0, 0, 0, ..., 0, 0, 0])
```

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

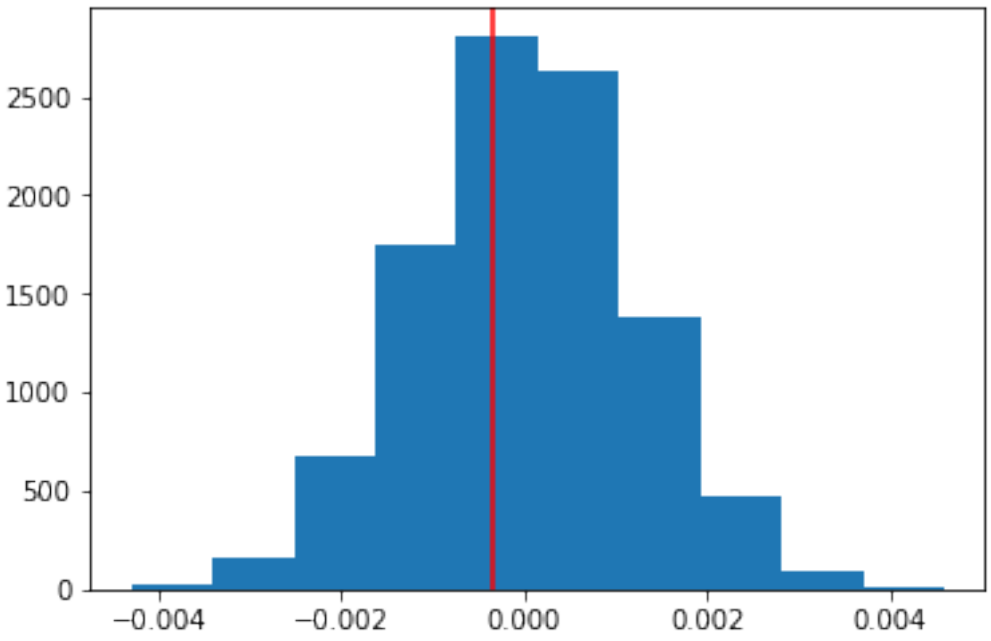
```
In [27]: actual_diffs= new_page_converted.mean() - old_page_converted.mean()
         actual_diffs

Out[27]: -0.00035306361871786929
```

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 [28]: p_diffs = []

         for i in range(10000):

             new_page_converted2  = np.random.choice(2, size= n_new1, p=[(1-p_new1), p_new1])
             old_page_converted2 = np.random.choice(2, size= n_old1, p=[(1-p_old1), p_old1])

             p_diffs.append(new_page_converted2.mean() - old_page_converted2.mean())
```

i. Plot a histogram of the **p_diffs**. Does this plot look like what you expected? Use the matching problem in the classroom to assure you fully understand what was computed here.

```
In [29]: p_diffs =np.asarray(p_diffs)
         plt.hist(p_diffs)
         plt.axvline(actual_diffs, color='r')

Out[29]: <matplotlib.lines.Line2D at 0x7f6e470fc160>
```



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

```
In [30]: (p_diffs > actual_diffs).mean()
```

```
Out[30]: 0.62190000000000001
```

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?

This means that the P-Value equals to 0.621 which is a high value compared to the alpha value (0.05% error limit) and depending on that we don't have enough prove to reject the null.

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 [31]: import statsmodels.api as sm

         convert_old = df2.query('converted == 1 and group == "control"')['converted'].count()
         convert_new = df2.query('converted == 1 and group == "treatment"')['converted'].count()
         n_old = df2[df2['group'] == 'control']['group'].count()
         n_new = df2[df2['group'] == 'treatment']['group'].count()
```

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

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 [32]: sm.stats.proportions_ztest([convert_new, convert_old], [n_new, n_old], alternative='lar
```

```
Out[32]: (-1.3109241984234394, 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.**?

The values doesn't agrees with the previous finding on 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 [33]: df2['intercept']= 1
         df2[['page','ab_page']]= pd.get_dummies(df2['group'])
```

```
In [34]: df2.head()
```

```
Out[34]:    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

            intercept  page  ab_page
         0          1     1        0
         1          1     1        0
         2          1     0        1
         3          1     0        1
         4          1     1        0
```

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 [35]: mod = sm.OLS(df2['converted'], df2[['intercept','ab_page']])
         res= mod.fit()
```

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

```
In [36]: res.summary()
```

```
Out[36]: <class 'statsmodels.iolib.summary.Summary'>
         """
                                    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:                Mon, 11 Nov 2019   Prob (F-statistic):              0.190
         Time:                        01:31:08   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:                   1.995
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 specif
"""
```

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**?

p-value associated with ab_page is 0.190 and differs because this is a two tailed test and depending on that we will reject the null since there no enought prove to reject the null

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?

considering other factors into the regression model will make our data clear about the pure conversions coming only from the change of the new page and not including other factors for better decision making

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 [37]: countries = pd.read_csv('countries.csv')
         countries.head()
```

```
Out[37]:    user_id country
         0   834778      UK
         1   928468      US
```

```
2   822059      UK
3   711597      UK
4   710616      UK
```

In [38]:
```
df2 =df2.set_index('user_id').join(countries.set_index('user_id'))
df2.head()
```

Out[38]:
```
                         timestamp      group landing_page  converted  \
user_id
851104   2017-01-21 22:11:48.556739    control     old_page          0
804228   2017-01-12 08:01:45.159739    control     old_page          0
661590   2017-01-11 16:55:06.154213  treatment     new_page          0
853541   2017-01-08 18:28:03.143765  treatment     new_page          0
864975   2017-01-21 01:52:26.210827    control     old_page          1

         intercept  page  ab_page country
user_id
851104           1     1        0      US
804228           1     1        0      US
661590           1     0        1      US
853541           1     0        1      US
864975           1     1        0      US
```

In [39]:
```
countries.tail()
```

Out[39]:
```
        user_id country
290579   653118      US
290580   878226      UK
290581   799368      UK
290582   655535      CA
290583   934996      UK
```

In [40]:
```
df2[['CA','UK','US']]= pd.get_dummies(df2['country'])
```

In [41]:
```
mod = sm.OLS(df2['converted'], df2[['intercept','CA','UK']])
res= mod.fit()
res.summary()
```

Out[41]:
```
<class 'statsmodels.iolib.summary.Summary'>
"""
                            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:                Mon, 11 Nov 2019   Prob (F-statistic):              0.201
Time:                        01:31:13   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      0.1195      0.001    166.244      0.000       0.118       0.121
CA            -0.0042      0.003     -1.516      0.130      -0.010       0.001
UK             0.0010      0.001      0.746      0.455      -0.002       0.004
==============================================================================
Omnibus:                   125552.384   Durbin-Watson:                   1.995
Prob(Omnibus):                  0.000   Jarque-Bera (JB):           414306.036
Skew:                           2.345   Prob(JB):                         0.00
Kurtosis:                       6.497   Cond. No.                         4.84
==============================================================================

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specif
"""
```

results notes: p-values of countries are higher than the limitation error and it means that the country factor has no significat change on individuals conversions, therefore we fail to reject the null.

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 [42]: df2['CA_p'] = df2['CA'] * df2['ab_page']
         df2['UK_p'] = df2['UK'] * df2['ab_page']
         df2.tail()
```

Out[42]:

user_id	timestamp	group	landing_page	converted	intercept	page	ab_page	country	CA	UK	US	CA_p	UK_p
751197	2017-01-03 22:28:38.630509	control	old_page	0	1	1	0	US	0	0	1	0	0
945152	2017-01-12 00:51:57.078372	control	old_page	0	1	1	0	US	0	0	1	0	0
734608	2017-01-22 11:45:03.439544	control	old_page	0	1	1	0	US	0	0	1	0	0
697314	2017-01-15 01:20:28.957438	control	old_page	0	1	1	0	US	0	0	1	0	0
715931	2017-01-16 12:40:24.467417	treatment	new_page	0	1	0	1	UK	0	1	0	0	1

In [43]:
```
mod = sm.OLS(df2['converted'], df2[['intercept','ab_page', 'CA','UK', 'CA_p','UK_p']])
res= mod.fit()
res.summary()
```

Out[43]:
```
<class 'statsmodels.iolib.summary.Summary'>
"""
                            OLS Regression Results
==============================================================================
Dep. Variable:              converted   R-squared:                       0.000
Model:                            OLS   Adj. R-squared:                  0.000
Method:                 Least Squares   F-statistic:                     1.466
Date:                Mon, 11 Nov 2019   Prob (F-statistic):              0.197
Time:                        01:31:14   Log-Likelihood:                -85265.
No. Observations:              290584   AIC:                         1.705e+05
Df Residuals:                  290578   BIC:                         1.706e+05
Df Model:                           5
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
intercept      0.1206      0.001    118.563      0.000       0.119       0.123
ab_page       -0.0022      0.001     -1.505      0.132      -0.005       0.001
CA            -0.0018      0.004     -0.467      0.641      -0.010       0.006
UK            -0.0006      0.002     -0.307      0.759      -0.004       0.003
CA_p          -0.0047      0.006     -0.845      0.398      -0.016       0.006
UK_p           0.0033      0.003      1.180      0.238      -0.002       0.009
==============================================================================
Omnibus:                   125549.436   Durbin-Watson:                   1.995
Prob(Omnibus):                  0.000   Jarque-Bera (JB):           414285.945
Skew:                           2.345   Prob(JB):                         0.00
Kurtosis:                       6.497   Cond. No.                         12.7
==============================================================================

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specif
"""
```

Results and Conclusion: all p-values of countries interaction with page conversions are higher than the limitation error alpha and it means that the country of the use interacting with the old or the new page has no significat change on individuals conversions, therefore we fail to reject the null.

Conclusion The data suggests that we has no enough evidence to prove that using the new page will lead to more conversions than using the old page.

In [44]:
```
from subprocess import call
call(['python', '-m', 'nbconvert', 'Analyze_ab_test_results_notebook.ipynb'])
```

Out[44]: 0

In [ ]: