Analyze_ab_test_results_notebook

November 3, 2019

0.1 Analyze A/B Test Results

Rahaf Alharbi

0.2 Table of Contents

Introduction
Part I - Probability
To get started, let's import our libraries.

```
In [48]: 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 [49]: #Read csv
         df = pd.read_csv('ab_data.csv')
         df.head()
```

```
Out[49]:    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 [50]: # Number of rows in the dataset
         df.shape[0]
```

```
Out[50]: 294478
```

c. The number of unique users in the dataset.

```
In [51]: #Number of unique users in the dataset
         df.user_id.nunique()
```

```
Out[51]: 290584
```

d. The proportion of users converted.

```
In [52]: # Proportion of users converted
         df.converted.mean()
```

```
Out[52]: 0.11965919355605512
```

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

```
In [53]: #Count the number of lines where new_page and control are aligned,old page and treatmen
         df.query('landing_page == "new_page" and group == "control"').count()[0] + df.query('la
```

```
Out[53]: 3893
```

f. Do any of the rows have missing values?

```
In [54]: # looking for missing values
         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
```

0.3 note : there are no missing values in the dataset

`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 [55]: #Filter on lines where new page and control are aligned
         npcontrol = df[(df.landing_page == "new_page") & (df.group == "control")]

         #Filter on lines where old page and treatment are aligned
         optreatment = df[(df.landing_page == "old_page") & (df.group == "treatment")]

         #Concatenate the inaccurate lines
         inaccurate = pd.concat([npcontrol, optreatment])

         #Assign  index for these lines
         inaccurate_index = inaccurate.index

         #Drop  lines with the indexes assigned above
         df2 = df.drop(inaccurate_index)
```

```
In [56]: # 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[56]: 0
```

```
In [57]: #check new data frame
         df2.head()
```

Out[57]:

	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

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 [58]: # Number of unique users
         df2['user_id'].nunique()
```

```
Out[58]: 290584
```

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

```
In [59]: #Search for unique users
         df2[df2['user_id'].duplicated()]['user_id']
```

```
Out[59]: 2893    773192
         Name: user_id, dtype: int64
```

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

```
In [60]: # view information of the repeated user_id
         df2[df2['user_id'].duplicated()]
```

Out[60]:

	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 [61]: # Number of rows before remove the duplicate
         df2.shape

Out[61]: (290585, 5)

In [62]: #drop the on row it's duplicate
         df2 = df2.drop_duplicates(subset='user_id');

In [63]: # Number of rows after remove the duplicate
         df2.shape

Out[63]: (290584, 5)
```

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 [64]: # The probability of an individual converting regardless of the page they receive
         df2['converted'].mean()

Out[64]: 0.11959708724499628
```

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

```
In [65]: df2[df2['group'] == "control"]['converted'].mean()

Out[65]: 0.1203863045004612
```

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

```
In [66]: df2[df2['group'] == "treatment"]['converted'].mean()

Out[66]: 0.11880806551510564
```

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

```
In [67]: (df2.landing_page == "new_page").mean()

Out[67]: 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.

Based on the results , it's like that the control group has a slightly higher conversion rate (0.1204) than the treatment group (0.1195), however, these results don't provide a strong evidence if one page leads to more conversions as we still don't know the significance of these results and the factors that might have contributed to the results, such as change resistence or test time duration. In order to provide a meaningful information to support the decision whether to implement the new page or keep the old page, we need to 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.

$$H_0: \ p_{new} - \ p_{old} <= 0$$

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

`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.

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

```
In [68]: #looking for the proportion of converted rate assuming p_new and p_old are equal
         P_new = df2['converted'].mean()
         P_new

Out[68]: 0.11959708724499628
```

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

```
In [69]: #looking for the proportion of converted rate assuming p_new and p_old are equal
         P_old = df2['converted'].mean()
         P_old

Out[69]: 0.11959708724499628
```

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

```
In [70]: # the number of individuals in the treatment group (users landing on new page)
         n_new = df2.query('group == "treatment"')['user_id'].count()
         n_new = int(n_new)
         n_new
```

```
Out[70]: 145310
```

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

```
In [71]: #the number of individuals in the control group(users landing on old page)
         n_old = df2.query('group == "control"')['user_id'].count()
         n_old = int(n_old)
         n_old
```

```
Out[71]: 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 [72]: #Draw samples from a binomial distribution
         new_page_converted = np.random.binomial(1, P_new, n_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 [73]: #Draw samples from a binomial distribution
         old_page_converted = np.random.binomial(1, P_old,n_old)
```

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

```
In [74]: #Number of rows from new page are higher than the ones on old page, so we will truncate
         #page and compute the difference
         new_page_converted = new_page_converted[:145274]
         new_page_converted.mean() - old_page_converted.mean()
```

```
Out[74]: -0.0015763316216253487
```

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**.

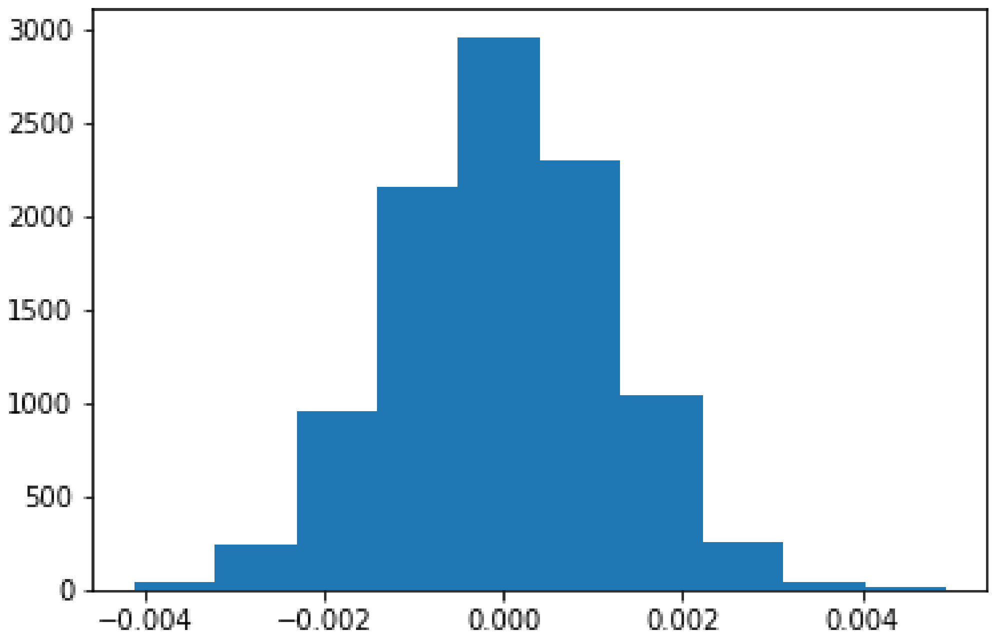
```
In [75]: #Simulate 10000 samples of the differences in conversion rates
         p_diffs = []

         for _ in range(10000):
             new_page_converted = np.random.binomial(1, P_new, n_new)
             old_page_converted = np.random.binomial(1, P_old, n_old)
             new_page_p = new_page_converted.mean()
             old_page_p = old_page_converted.mean()
             p_diffs.append(new_page_p - old_page_p)
```

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 [76]: #Show histogram

         plt.hist(p_diffs);
```



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

```
In [77]: #actual difference of converted rates

         actual_diff = (df2[df2['group'] == "treatment"]['converted'].mean()) - (df2[df2['group'
         actual_diff
```

```
Out[77]: -0.0015782389853555567
```

```
In [78]: #Convert to numpy array
         #calculate the p-value
         p_diffs = np.array(p_diffs)
         (p_diffs > actual_diff).mean()
```

```
Out[78]: 0.9019000000000003
```

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?

Actual difference represents the difference between converted rates of new page and old page, and based on our data.

p-diffs: Represent the simulated difference between converted rates of new page and old page, based on 10000 simulated samples.

The percentage of 90.5 is called scientifically p-value, which determines the probability of obtaining our observed statistic or one more extreme in favor of the alternative if the null hypothesis is true.

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 [79]: 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 [80]: #Two sample proportion hypothesis testing

         z_score, p_value = sm.stats.proportions_ztest([convert_new, convert_old], [n_new, n_old
         z_score

Out[80]: -1.3109241984234394

In [81]: p_value

Out[81]: 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.**?

A negative z-score suggests and the value of p-value suggests that we should fail to reject the null hypothesis.

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?

Since this is a Yes-No type of variable, the appropriate approach is 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 [87]:
```
#Create intercept column
df2['intercept']=1

#Create dummies
ab_page = ['treatment', 'control']
df2['ab_page'] = pd.get_dummies(df2.group)['treatment']
```

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 [88]:
```
logit = sm.Logit(df2['converted'], df2[['intercept','ab_page']])
```

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

In [89]:
```
results = logit.fit()
results.summary()
```

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

Out[89]:
```
<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:                Tue, 15 Oct 2019   Pseudo R-squ.:               8.077e-06
Time:                        12:03:43   Log-Likelihood:            -1.0639e+05
converged:                       True   LL-Null:                   -1.0639e+05
                                        LLR p-value:                    0.1899
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
intercept     -1.9888      0.008   -246.669      0.000      -2.005      -1.973
ab_page       -0.0150      0.011     -1.311      0.190      -0.037       0.007
==============================================================================
"""
```

e. What is the p-value associated with **ab_page**? Why does it differ from the value you found in **Part II**? **Hint**: What are the null and alternative hypotheses associated with your regression model, and how do they compare to the null and alternative hypotheses in **Part II**?

The p-value associated with ab_page column is 0.19 which is less than the p-value calculated using the z-score function. The reason why is different is due to the intercept added.

The logistic regression determines only two possible outcomes.

If the new page is equal to the old page or different

$$H_0 : p_{new} - p_{old} = 0$$

$$H_1 : p_{new} - p_{old} \neq 0$$

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?

We could consider introducing the timestamp metric to determine in which part of the day the individuals converted the most. For example, if we find that the morning is the period that users spend most of their time on the internet we might also take it into consideration.

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 [90]:
```
countries_df = pd.read_csv('./countries.csv')
df_new = countries_df.set_index('user_id').join(df2.set_index('user_id'), how='inner')
```

In [91]:
```
# Create the necessary dummy variables
df_new[['CA', 'US']] = pd.get_dummies(df_new['country'])[['CA','US']]
df_new.head()
```

Out[91]:
```
              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  ab_page  CA  US
user_id
834778           0          1        1          0        0   0   0
928468           0          1        0          1        1   0   1
```

822059	1	1	0	1	1	0	0
711597	0	1	1	0	0	0	0
710616	0	1	0	1	1	0	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 [92]:
```
df_new['intercept'] = 1
log_mod = sm.Logit(df_new['converted'], df_new[['CA', 'US', 'intercept', 'ab_page']])
results = log_mod.fit()
results.summary()
```

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

Out[92]:
```
<class 'statsmodels.iolib.summary.Summary'>
"""
                           Logit Regression Results
==============================================================================
Dep. Variable:              converted   No. Observations:               290584
Model:                          Logit   Df Residuals:                   290580
Method:                           MLE   Df Model:                            3
Date:                Tue, 15 Oct 2019   Pseudo R-squ.:               2.323e-05
Time:                        12:03:56   Log-Likelihood:            -1.0639e+05
converged:                       True   LL-Null:                   -1.0639e+05
                                        LLR p-value:                    0.1760
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
CA            -0.0506      0.028     -1.784      0.074      -0.106       0.005
US            -0.0099      0.013     -0.743      0.457      -0.036       0.016
intercept     -1.9794      0.013   -155.415      0.000      -2.004      -1.954
ab_page       -0.0149      0.011     -1.307      0.191      -0.037       0.007
==============================================================================
"""
```

Conclusions

Based on According to the analysis I found that the old page was better than the new page, for that I fail to reject the null hypothesis. Moreover, the histogram shows that the new page is not better than the old page.

From the regression we see that the p-value is higher in United States than in Canada, which means that users in the US are more likely to convert, but still not enough evidence to reject the null hypothesis.

In [93]:
```
from subprocess import call
call(['python', '-m', 'nbconvert', 'Analyze_ab_test_results_notebook.ipynb'])
```

Out[93]: 0

In [ ]: