Analyze_ab_test_results_notebook

September 28, 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)
        import warnings
        warnings.filterwarnings('ignore')
```

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(55)
```

```
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
        5   936923  2017-01-10 15:20:49.083499    control     old_page          0
        6   679687  2017-01-19 03:26:46.940749  treatment     new_page          1
        7   719014  2017-01-17 01:48:29.539573    control     old_page          0
        8   817355  2017-01-04 17:58:08.979471  treatment     new_page          1
        9   839785  2017-01-15 18:11:06.610965  treatment     new_page          1
        10  929503  2017-01-18 05:37:11.527370  treatment     new_page          0
        11  834487  2017-01-21 22:37:47.774891  treatment     new_page          0
        12  803683  2017-01-09 06:05:16.222706  treatment     new_page          0
        13  944475  2017-01-22 01:31:09.573836  treatment     new_page          0
        14  718956  2017-01-22 11:45:11.327945  treatment     new_page          0
        15  644214  2017-01-22 02:05:21.719434    control     old_page          1
        16  847721  2017-01-17 14:01:00.090575    control     old_page          0
        17  888545  2017-01-08 06:37:26.332945  treatment     new_page          1
        18  650559  2017-01-24 11:55:51.084801    control     old_page          0
        19  935734  2017-01-17 20:33:37.428378    control     old_page          0
        20  740805  2017-01-12 18:59:45.453277  treatment     new_page          0
        21  759875  2017-01-09 16:11:58.806110  treatment     new_page          0
        22  767017  2017-01-12 22:58:14.991443    control     new_page          0
        23  793849  2017-01-23 22:36:10.742811  treatment     new_page          0
        24  905617  2017-01-20 14:12:19.345499  treatment     new_page          0
        25  746742  2017-01-23 11:38:29.592148    control     old_page          0
        26  892356  2017-01-05 09:35:14.904865  treatment     new_page          1
        27  773302  2017-01-12 08:29:49.810594  treatment     new_page          0
        28  913579  2017-01-24 09:11:39.164256    control     old_page          1
        29  736159  2017-01-06 01:50:21.318242  treatment     new_page          0
        30  690284  2017-01-13 17:22:57.182769    control     old_page          0
        31  826115  2017-01-05 11:27:16.756633  treatment     new_page          0
        32  875124  2017-01-05 15:39:25.439906  treatment     new_page          1
        33  931013  2017-01-07 03:23:57.932344  treatment     new_page          0
        34  710349  2017-01-11 22:24:44.226492    control     old_page          0
        35  677533  2017-01-23 17:48:50.491821    control     old_page          0
        36  831737  2017-01-11 21:18:20.911015    control     old_page          1
        37  648583  2017-01-19 09:03:05.545308  treatment     new_page          0
        38  728086  2017-01-03 17:07:00.837852  treatment     new_page          0
```

```
39   870163  2017-01-02 21:33:49.325594  treatment     new_page          0
40   771087  2017-01-16 00:05:29.983919    control     old_page          0
41   739414  2017-01-03 13:25:55.139705  treatment     new_page          0
42   896163  2017-01-22 09:10:20.753218    control     old_page          0
43   862225  2017-01-08 14:49:37.335432    control     old_page          1
44   939593  2017-01-05 09:15:31.984283    control     old_page          0
45   702260  2017-01-18 13:55:31.488221    control     old_page          0
46   943635  2017-01-22 13:37:39.722775  treatment     new_page          0
47   800436  2017-01-20 07:47:47.224386  treatment     new_page          0
48   698590  2017-01-23 11:51:59.925413  treatment     new_page          0
49   830513  2017-01-12 00:50:01.470557  treatment     new_page          0
50   670941  2017-01-05 08:16:41.306478    control     old_page          0
51   850231  2017-01-18 17:18:04.790584    control     old_page          1
52   916511  2017-01-22 06:20:04.691382  treatment     new_page          0
53   897174  2017-01-17 02:03:25.962173  treatment     new_page          0
54   906999  2017-01-22 19:16:44.715266  treatment     new_page          0
```

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

```
In [3]: len(df.index)
```

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

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

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

c. The number of unique users in the dataset.

```
In [12]: df['user_id'].nunique()
```

```
Out[12]: 290584
```

d. The proportion of users converted.

```
In [5]: df.query('converted == 1').user_id.nunique() / df.user_id.nunique()
```

```
Out[5]: 0.12104245244060237
```

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

```
In [6]: df.query("(group != 'treatment' and landing_page == 'new_page') or (group == 'treatment'
```

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

f. Do any of the rows have missing values?

```
In [7]: df.shape[0] - df.dropna().shape[0]
```

```
Out[7]: 0
```

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 [ ]:
```

```
In [44]: df2 = df.query("(group == 'control' and landing_page == 'old_page') or (group == 'treat
         len(df2)
```

```
Out[44]: 290585
```

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 [45]: df2.user_id.nunique()
```

```
Out[45]: 290584
```

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

```
In [10]: print(df2[df2['user_id'].duplicated()]['user_id'].to_string(index=False))
```

```
773192
```

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

```
In [11]: df2[df2.duplicated(['user_id'])]
```

```
Out[11]:       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 [12]: df2.drop_duplicates(['user_id'], keep='first',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 [13]: df2['converted'].mean()
```

```
Out[13]: 0.11959708724499628
```

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

```
In [14]: df2[df2['group'] == 'control']['converted'].mean()
```

```
Out[14]: 0.1203863045004612
```

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

```
In [15]: df2[df2['group'] == 'treatment']['converted'].mean()
```

```
Out[15]: 0.11880806551510564
```

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

```
In [16]: len(df2[df2['landing_page'] == 'new_page'].index)/len(df2.index)
```

```
Out[16]: 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.

Your answer goes here.
### Part II - A/B Test
Notice that because of the time stamp associated with each event, you could technically run a hypothesis test continuously as each observation was observed.

However, then the hard question is do you stop as soon as one page is considered significantly better than another or does it need to happen consistently for a certain amount of time? How long do you run to render a decision that neither page is better than another?

These questions are the difficult parts associated with A/B tests in general.

`1.` For now, consider you need to make the decision just based on all the data provided. If you want to assume that the old page is better unless the new page proves to be definitely better at a Type I error rate of 5%, what should your null and alternative hypotheses be? You can state your hypothesis in terms of words or in terms of p_{old} and p_{new}, which are the converted rates for the old and new pages.

Put your answer here.

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

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

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

```
In [18]: p_old = df2['converted'].mean()
         p_old

Out[18]: 0.11959708724499628
```

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

```
In [19]: n_new = len(df2.query('landing_page == "new_page"'))
         n_new

Out[19]: 145310
```

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

```
In [20]: n_old = len(df2.query('landing_page == "old_page"'))
         n_old

Out[20]: 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 [21]: new_page_converted = np.random.binomial(1, p=p_new, size=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 [22]: old_page_converted = np.random.binomial(1, p=p_old, size=n_old)
```

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

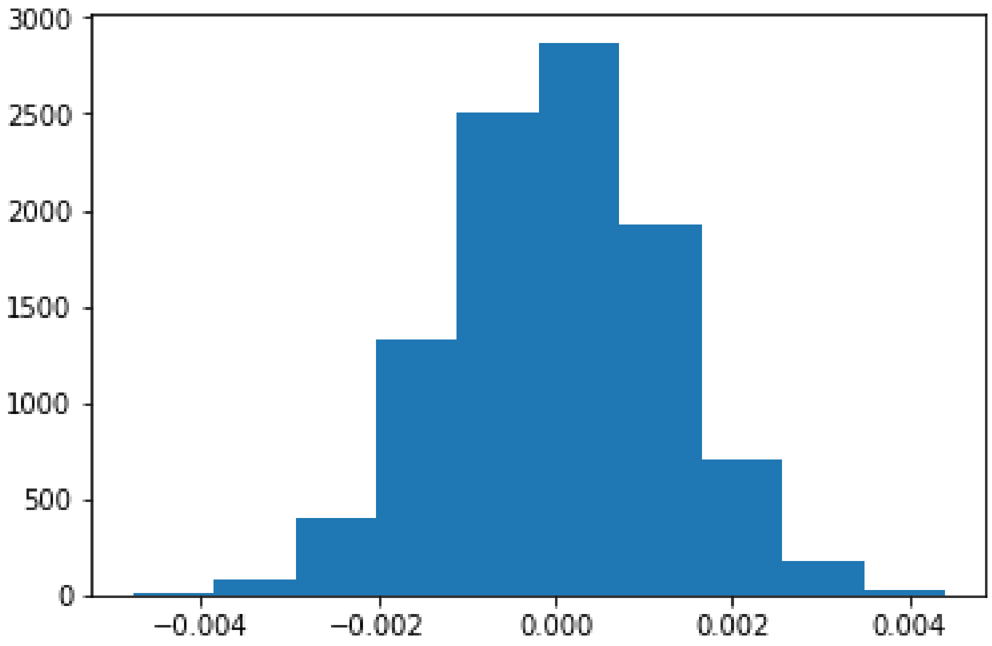
```
In [23]: new_page_converted.mean()-old_page_converted.mean()

Out[23]: -0.000945079116546523888
```

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 [24]: new_converted_simulation = np.random.binomial(n_new, p_new,  10000)/n_new
         old_converted_simulation = np.random.binomial(n_old, p_old,  10000)/n_old
         p_diffs = new_converted_simulation - old_converted_simulation
```

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

```
In [25]: plt.hist(p_diffs);
```



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

```
In [26]: treatment_converted = df2.query('group =="treatment"').converted.mean()
         control_converted = df2.query('group =="control"').converted.mean()
         act_diff = treatment_converted - control_converted

         p_diffs = np.array(p_diffs)

         (p_diffs > act_diff).mean()

Out[26]: 0.90400000000000003
```

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

Answer: We calculated the p value. P value helps us determine the significance of our results, whether we accept the null hypothesis or reject it. Since the value p is not small, we could not reject the null hypothesis and do not prefer the new page and keep the old page.

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.

len(df2.query(" landing_page == 'old_page' and converted == 1").index)

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 [28]: import statsmodels.api as sm

         convert_old = len(df2.query(" landing_page == 'old_page' and converted == 1").index)
         convert_new = len(df2.query(" landing_page == 'new_page' and converted == 1").index)
         n_old = len(df2[df2['group'] == 'control'].index)
         n_new = len(df2[df2['group'] == 'treatment'].index)
```

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

         z_score, p_value

Out[29]: (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.**?

```
In [30]: from scipy.stats import norm
         print(norm.cdf(z_score))
         print(norm.ppf(1-(0.05)))

0.905058312759
1.64485362695
```

### 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 [54]: df2['intercept']=1
         df2[['control', 'ab_page']]=pd.get_dummies(df2['group'])
         df2.drop(labels=['control'], axis=1, inplace=True)
         df2.head()
```

```
Out[54]:    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  ab_page
         0          1        0
         1          1        0
         2          1        1
         3          1        1
         4          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 [55]: import statsmodels.api as sm
         m = sm.Logit(df2.converted, df2[['intercept', 'ab_page']])
         results =  m.fit()
```

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

```
In [60]: from scipy import stats
         stats.chisqprob = lambda chisq, df: stats.chi2.sf(chisq, df)
```

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

```
In [61]: results.summary()
```

```
Out[61]: <class 'statsmodels.iolib.summary.Summary'>
         """
                                    Logit Regression Results
         ==============================================================================
         Dep. Variable:              converted   No. Observations:               290585
         Model:                          Logit   Df Residuals:                   290583
         Method:                           MLE   Df Model:                            1
         Date:                Mon, 28 Sep 2020   Pseudo R-squ.:               8.085e-06
         Time:                        04:43:21   Log-Likelihood:            -1.0639e+05
```

```
converged:                       True   LL-Null:                   -1.0639e+05
                                        LLR p-value:                    0.1897
==============================================================================
                 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     -1.312      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 is 0.19. It is not statistifically significant as it is not less than 0.05. The reason why p-value is different in Part 2 because in the previous part we performed a one-sided test, where in this part, logistic regression, we performed two-sided test. This implies that p_new is equal to p_old. which is the null hypothesis of a two tailed test.
Ho: p_new = p_old
H1: p_new != p_old**

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 things that might influence the conversions and taking them into account is a good idea. These new features can contribute to the significance of the results of our tests and lead to more precise decisions. One of the drawbacks of adding additional terms in the regression model is Simpson's paradox. It is that a trend can appear in several different groups of data but disappears or reverses when these groups are combined.

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

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 [63]: df_new[['UK', 'US']] = pd.get_dummies(df_new['country'])[['UK','US']]
         df_new.head()

Out[63]:         country                   timestamp      group landing_page  \
         user_id
         630000       US  2017-01-19 06:26:06.548941  treatment     new_page
         630001       US  2017-01-16 03:16:42.560309  treatment     new_page
         630002       US  2017-01-19 19:20:56.438330    control     old_page
         630003       US  2017-01-12 10:09:31.510471  treatment     new_page
         630004       US  2017-01-18 20:23:58.824994  treatment     new_page

                  converted  intercept  ab_page  UK  US
         user_id
         630000           0          1        1   0   1
         630001           1          1        1   0   1
         630002           0          1        0   0   1
         630003           0          1        1   0   1
         630004           0          1        1   0   1

In [64]: model = sm.Logit(df_new.converted, df_new[['UK', 'US']])
         results = model.fit()
         results.summary()

Optimization terminated successfully.
         Current function value: 0.382863
         Iterations 6


Out[64]: <class 'statsmodels.iolib.summary.Summary'>
         """
                                    Logit Regression Results
         ==============================================================================
         Dep. Variable:              converted   No. Observations:               290585
         Model:                          Logit   Df Residuals:                   290583
         Method:                           MLE   Df Model:                            1
         Date:                Mon, 28 Sep 2020   Pseudo R-squ.:                -0.04573
         Time:                        04:44:50   Log-Likelihood:            -1.1125e+05
         converged:                       True   LL-Null:                   -1.0639e+05
                                                 LLR p-value:                     1.000
         ==============================================================================
                          coef    std err          z      P>|z|      [0.025      0.975]
         ------------------------------------------------------------------------------
         UK            -1.9868      0.011   -174.174      0.000      -2.009      -1.964
         US            -1.9967      0.007   -292.315      0.000      -2.010      -1.983
         ==============================================================================
         """
```

## Finishing Up

Congratulations! You have reached the end of the A/B Test Results project! You should be very proud of all you have accomplished!

Tip: Once you are satisfied with your work here, check over your report to make sure that it is satisfies all the areas of the rubric (found on the project submission page at the end of the lesson). You should also probably remove all of the "Tips" like this one so that the presentation is as polished as possible.

0.3 Directions to Submit

Before you submit your project, you need to create a .html or .pdf version of this notebook in the workspace here. To do that, run the code cell below. If it worked correctly, you should get a return code of 0, and you should see the generated .html file in the workspace directory (click on the orange Jupyter icon in the upper left).

Alternatively, you can download this report as .html via the **File** > **Download as** submenu, and then manually upload it into the workspace directory by clicking on the orange Jupyter icon in the upper left, then using the Upload button.

Once you've done this, you can submit your project by clicking on the "Submit Project" button in the lower right here. This will create and submit a zip file with this .ipynb doc and the .html or .pdf version you created. Congratulations!

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