Analyze_ab_test_results_notebook

June 27, 2021

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

- Section **??**
- Section **??**
- Section **??**
- Section **??**

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 [75]: 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 [76]: df=pd.read_csv('ab_data.csv')
         df.head()
```

Out[76]:

	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 [77]: df.shape[0]
```

```
Out[77]: 294478
```

c. The number of unique users in the dataset.

```
In [78]: df['user_id'].nunique()
```

```
Out[78]: 290584
```

d. The proportion of users converted.

```
In [79]: df.query('converted == 1').user_id.nunique()/df.shape[0]
```

```
Out[79]: 0.1194418598333322
```

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

```
In [80]: df.query('(group == "treatment" and landing_page!= "new_page" or group != "treatment" a
```

```
Out[80]: 3893
```

f. Do any of the rows have missing values?

```
In [81]: df.isnull().sum().sum()
```

```
Out[81]: 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 [82]: # Double Check all of the correct rows were removed - this should be 0
         df2=df.drop(df.query('(group == "treatment" and landing_page!= "new_page" or group != "
         df2.query('(group == "treatment" and landing_page!= "new_page" or group != "treatment"

Out[82]: 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 [83]: df2.user_id.nunique()

Out[83]: 290584
```

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

```
In [84]: df2[df2.user_id.duplicated()].user_id

Out[84]: 2893    773192
         Name: user_id, dtype: int64
```

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

```
In [85]: df2[df2.user_id.duplicated()]

Out[85]:       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 [86]: df2=df2.drop_duplicates('user_id',keep='first')
         df2.head(10)

Out[86]:    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
```

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 [87]: df2.query('converted ==1').user_id.nunique()/df2.shape[0]
```

```
Out[87]: 0.11959708724499628
```

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

```
In [88]: control_converted=df2.query('group=="control" and converted ==1').user_id.nunique()/df2
         control_converted

Out[88]: 0.1203863045004612
```

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

```
In [89]: treatment_converted=df2.query('group=="treatment" and converted ==1').user_id.nunique()
         treatment_converted

Out[89]: 0.11880806551510564
```

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

```
In [90]: new_page=df2.query('landing_page =="new_page"').user_id.nunique()/df2.shape[0]
         new_page

Out[90]: 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.

From what we saw previously that:

```
(a) Probability of who converted regardless of the page they receive= 11.96 %
(b) Probability of who converted - Given of an individual was in the control group = 12.04 %
(c) Probability of who converted - Given of an individual was in the treatment group = 11.88 %
(d) Probability of an individual received the new page = 50 %
```

That is no sufficient evidence to conclude that the new treatment page leads to more 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} <= 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 [91]: p_new=df2.query('converted ==1').user_id.nunique()/df2.user_id.nunique()
         p_new

Out[91]: 0.11959708724499628
```

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

```
In [92]: p_old=df2.query('converted ==1').user_id.nunique()/df2.user_id.nunique()
         p_old

Out[92]: 0.11959708724499628

In [93]: n_new=df2.query('landing_page =="new_page"').user_id.nunique()
         n_old=df2.query('landing_page =="old_page"').user_id.nunique()
         n_new,n_old

Out[93]: (145310, 145274)
```

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

```
In [94]: n_new1=df2.query('group == "treatment" and converted ==1').user_id.nunique()
         n_new1

Out[94]: 17264
```

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

```
In [95]: n_old1=df2.query('group == "control" and converted ==1').user_id.nunique()
         n_old1

Out[95]: 17489
```

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 [96]: new_page_converted=np.random.binomial(1,p_new,n_new)
         new_page_converted.mean()
```

```
Out[96]: 0.12049411602780263
```

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 [97]: old_page_converted=np.random.binomial(1,p_old,n_old)
         old_page_converted.mean()
```

```
Out[97]: 0.1182455222544984
```

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

```
In [98]: new_page_converted.mean() - old_page_converted.mean()
```

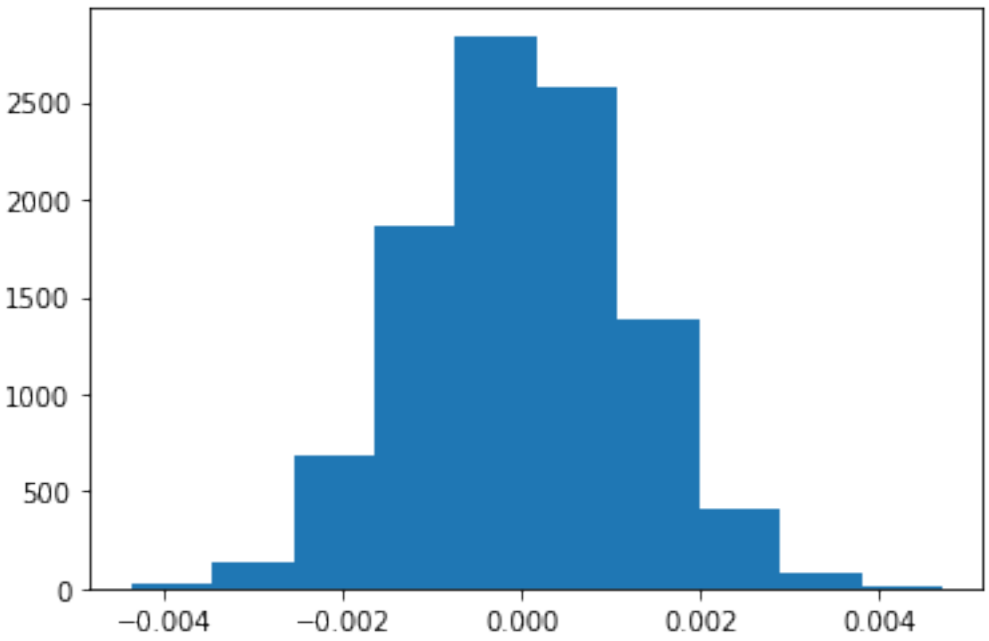
```
Out[98]: 0.0022485937733042333
```

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 [99]: p_diffs=[]
         new_converted_sample=np.random.binomial(n_new, p_new, 10000)/n_new
         old_converted_sample=np.random.binomial(n_old, p_old, 10000)/n_old
         p_diffs.append(new_converted_sample-old_converted_sample)
```

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 [100]: plt.hist(p_diffs);
```



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

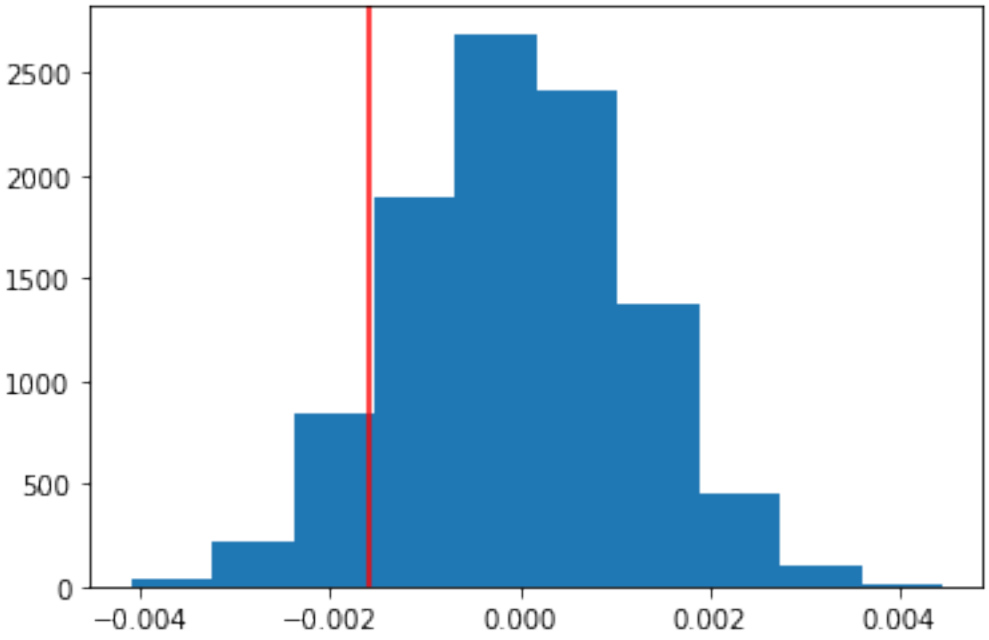
```
In [101]: p_diffs=np.array(p_diffs)
          actual_diff=treatment_converted-control_converted
          (p_diffs>actual_diff).mean()
```

```
Out[101]: 0.9034
```

```
In [102]: null_vals = np.random.normal(0, p_diffs.std(), p_diffs.size)
          plt.hist(null_vals)

          plt.axvline(actual_diff, c='red')
```

```
Out[102]: <matplotlib.lines.Line2D at 0x19c4d5b8af0>
```




```
In [103]: (null_vals > actual_diff).mean()
```

```
Out[103]: 0.8985
```

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 computed the P-value in j, About 90 % and it is larger than type I error α 5%, So we fail to reject the null hypothesis H_0.

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 [104]: import statsmodels.api as sm
          convert_old = df2.query('landing_page=="old_page"').converted.sum()
          convert_new = df2.query('landing_page=="new_page"').converted.sum()
          n_old = df2.query('landing_page=="old_page"').user_id.nunique()
          n_new = df2.query('landing_page=="new_page"').user_id.nunique()
          convert_old,convert_new,n_old,n_new

Out[104]: (17489, 17264, 145274, 145310)
```

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 [105]: z_score,p_value=sm.stats.proportions_ztest(np.array([convert_new,convert_old]),np.arra
          z_score,p_value

Out[105]: (-1.3109241984234394, 0.9050583127590245)
```

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

The Z_score and P_value from the previous test doesn't reject the null hypothesis, the z-score about -1.31 less than critical value 1.96, and the p-value about 90% larger than type I error α 5%, from both that means there is statistical significance to affirm the previous results in J that fail to reject the null hypothesis H_0.

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?

This is a logistic regression, since we want to know the odds of conversion, rather than a linear figure.

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 [106]: df2.head()
```

```
Out[106]:    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 [107]: df2['intercept']=1
          df2['ab_page']=pd.get_dummies(df2['group'])['treatment']
          df2.head()
```

```
Out[107]:    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 [108]: lm=sm.Logit(df2.converted,df2[['intercept','ab_page']])
          log=lm.fit()

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

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

```
In [109]: log.summary()

Out[109]: <class 'statsmodels.iolib.summary.Summary'>
          """
                                  Logit Regression Results
          ==============================================================================
          Dep. Variable:              converted   No. Observations:               290584
          Model:                          Logit   Df Residuals:                   290582
          Method:                           MLE   Df Model:                            1
```