Analyze_ab_test_results_notebook

May 15, 2020

0.1 Analyze A/B Test Results

You may either submit your notebook through the workspace here, or you may work from your local machine and submit through the next page. Either way assure that your code passes the project RUBRIC. **Please save regularly.**

This project will assure you have mastered the subjects covered in the statistics lessons. The hope is to have this project be as comprehensive of these topics as possible. Good luck!

0.2 Table of Contents

Introduction

A/B tests are very commonly performed by data analysts and data scientists. It is important that you get some practice working with the difficulties of these

For this project, you will be working to understand the results of an A/B test run by an e-commerce website. Your goal is to work through this notebook to help the company understand if they should implement the new page, keep the old page, or perhaps run the experiment longer to make their decision.

As you work through this notebook, follow along in the classroom and answer the corresponding quiz questions associated with each question. The labels for each classroom concept are provided for each question. This will assure you are on the right track as you work through the project, and you can feel more confident in your final submission meeting the criteria. As a final check, assure you meet all the criteria on the RUBRIC.

Part I - Probability

To get started, let's import our libraries.

```
In [1]: import pandas as pd
        import numpy as np
        import random
        import matplotlib.pyplot as plt
        %matplotlib inline
        #We are setting the seed to assure you get the same answers on quizzes as we set up
        random.seed(42)
```

1. Now, read in the ab_data.csv data. Store it in df. **Use your dataframe to answer the questions in Quiz 1 of the classroom.**

a. Read in the dataset and take a look at the top few rows here:

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

```
Out[2]:    user_id                   timestamp      group landing_page  converted
        0   851104  2017-01-21 22:11:48.556739    control     old_page          0
        1   804228  2017-01-12 08:01:45.159739    control     old_page          0
        2   661590  2017-01-11 16:55:06.154213  treatment     new_page          0
        3   853541  2017-01-08 18:28:03.143765  treatment     new_page          0
        4   864975  2017-01-21 01:52:26.210827    control     old_page          1
```

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

```
In [3]: # use describe or shape to find number of rows
        df.describe()
```

```
Out[3]:              user_id      converted
        count  294478.000000  294478.000000
        mean   787974.124733       0.119659
        std     91210.823776       0.324563
        min    630000.000000       0.000000
        25%    709032.250000       0.000000
        50%    787933.500000       0.000000
        75%    866911.750000       0.000000
        max    945999.000000       1.000000
```

```
In [4]: df.shape
```

```
Out[4]: (294478, 5)
```

c. The number of unique users in the dataset.

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

```
Out[5]: user_id         290584
        timestamp       294478
        group                2
        landing_page         2
        converted            2
        dtype: int64
```

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.query ('group == "treatment" and landing_page != "new_page"')
```

```
Out[7]:        user_id                   timestamp      group landing_page  converted
        308     857184  2017-01-20 07:34:59.832626  treatment     old_page          0
        327     686623  2017-01-09 14:26:40.734775  treatment     old_page          0
        357     856078  2017-01-12 12:29:30.354835  treatment     old_page          0
        685     666385  2017-01-23 08:11:54.823806  treatment     old_page          0
        713     748761  2017-01-10 15:47:44.445196  treatment     old_page          0
        776     820951  2017-01-04 02:42:54.770627  treatment     old_page          0
        889     839954  2017-01-06 20:58:22.280929  treatment     old_page          0
        1037    880442  2017-01-07 21:42:39.026815  treatment     old_page          0
        1106    817911  2017-01-17 21:51:43.220160  treatment     old_page          0
        1376    844475  2017-01-20 14:25:37.359614  treatment     old_page          0
        1551    838336  2017-01-14 22:05:24.310302  treatment     old_page          0
        1706    916207  2017-01-20 11:53:39.683012  treatment     old_page          0
        1762    690127  2017-01-11 16:02:57.551297  treatment     old_page          1
        2233    869707  2017-01-02 18:36:28.222510  treatment     old_page          0
        2422    853156  2017-01-15 23:19:45.427866  treatment     old_page          0
        2689    793494  2017-01-09 02:09:08.534282  treatment     old_page          0
        3262    710871  2017-01-15 13:58:39.846106  treatment     old_page          0
        3306    809229  2017-01-17 22:37:26.403828  treatment     old_page          0
        3364    915093  2017-01-16 18:02:59.006193  treatment     old_page          0
        3689    878413  2017-01-03 13:41:19.090123  treatment     old_page          0
        3869    792890  2017-01-12 21:42:36.159299  treatment     old_page          0
        4000    706721  2017-01-04 00:32:24.564711  treatment     old_page          0
        4043    846754  2017-01-24 01:27:40.512402  treatment     old_page          0
        4074    768200  2017-01-21 15:48:44.216867  treatment     old_page          0
        4475    706878  2017-01-09 20:33:39.727111  treatment     old_page          0
        4537    761716  2017-01-23 20:32:13.298444  treatment     old_page          0
        4961    844946  2017-01-04 07:20:58.924520  treatment     old_page          1
        5418    926559  2017-01-16 00:59:10.283392  treatment     old_page          0
        5492    662456  2017-01-07 19:48:48.540429  treatment     old_page          0
        5800    709280  2017-01-19 22:05:06.906174  treatment     old_page          1
        ...        ...                         ...        ...          ...        ...
        288375  631156  2017-01-04 03:05:13.816388  treatment     old_page          0
        288465  767964  2017-01-19 09:41:32.875795  treatment     old_page          1
        289242  698366  2017-01-04 00:22:43.306653  treatment     old_page          0
        289665  693835  2017-01-20 11:44:50.517253  treatment     old_page          0
        289799  909162  2017-01-09 17:12:38.910965  treatment     old_page          0
        289846  934943  2017-01-04 18:45:15.921776  treatment     old_page          0
        290062  928175  2017-01-05 03:51:08.933502  treatment     old_page          1
        290149  858910  2017-01-10 05:20:37.997730  treatment     old_page          1
        290328  658911  2017-01-05 15:14:40.331200  treatment     old_page          0
        290360  714840  2017-01-10 23:35:22.559510  treatment     old_page          1
```

```
290647   904581  2017-01-17 11:35:54.031953  treatment     old_page          0
291313   807667  2017-01-15 19:11:59.976235  treatment     old_page          0
291754   795252  2017-01-19 02:43:07.343575  treatment     old_page          1
291922   634098  2017-01-07 23:45:07.976016  treatment     old_page          0
292412   693843  2017-01-09 06:31:48.749071  treatment     old_page          1
292521   689329  2017-01-06 03:58:15.546309  treatment     old_page          0
292607   699462  2017-01-17 23:54:08.826755  treatment     old_page          0
292800   712112  2017-01-14 23:33:41.083796  treatment     old_page          0
292963   742202  2017-01-12 04:34:20.344485  treatment     old_page          0
292977   638460  2017-01-22 13:38:30.677806  treatment     old_page          0
293240   861420  2017-01-04 20:34:09.065070  treatment     old_page          0
293302   825937  2017-01-04 20:56:48.825875  treatment     old_page          0
293391   934444  2017-01-12 19:49:35.581289  treatment     old_page          0
293443   738761  2017-01-04 15:20:52.694440  treatment     old_page          0
293530   934040  2017-01-04 20:52:26.981566  treatment     old_page          0
293773   688144  2017-01-16 20:34:50.450528  treatment     old_page          1
293817   876037  2017-01-17 16:15:08.957152  treatment     old_page          1
293917   738357  2017-01-05 15:37:55.729133  treatment     old_page          0
294014   813406  2017-01-09 06:25:33.223301  treatment     old_page          0
294252   892498  2017-01-22 01:11:10.463211  treatment     old_page          0

[1965 rows x 5 columns]
```

```
In [8]: df.query ('group == "control" and landing_page != "old_page"')
```

```
Out[8]:         user_id                   timestamp    group landing_page  converted
        22       767017  2017-01-12 22:58:14.991443  control      new_page          0
        240      733976  2017-01-11 15:11:16.407599  control      new_page          0
        490      808613  2017-01-10 21:44:01.292755  control      new_page          0
        846      637639  2017-01-11 23:09:52.682329  control      new_page          1
        850      793580  2017-01-08 03:25:33.723712  control      new_page          1
        988      698120  2017-01-22 07:09:37.540970  control      new_page          0
        1198     646342  2017-01-06 18:39:23.484797  control      new_page          0
        1354     735021  2017-01-16 09:51:29.349493  control      new_page          0
        1474     678638  2017-01-18 06:36:42.515395  control      new_page          0
        1877     717682  2017-01-07 03:05:39.891873  control      new_page          0
        2023     937692  2017-01-19 01:29:42.739007  control      new_page          0
        2214     649781  2017-01-20 03:50:20.837704  control      new_page          0
        2745     872666  2017-01-05 07:44:32.050781  control      new_page          0
        2759     639817  2017-01-06 23:39:11.754971  control      new_page          0
        2857     738999  2017-01-08 15:21:55.309961  control      new_page          0
        2947     847673  2017-01-07 18:45:04.253063  control      new_page          1
        3362     858458  2017-01-06 04:51:33.183576  control      new_page          1
        3421     638068  2017-01-20 01:57:00.012096  control      new_page          0
        3548     807355  2017-01-21 11:10:28.793058  control      new_page          0
        3817     832098  2017-01-15 06:06:26.163307  control      new_page          0
        3903     855630  2017-01-10 16:24:01.119709  control      new_page          1
        3913     937090  2017-01-22 07:38:49.397402  control      new_page          0
```

```
4038     919582  2017-01-04 12:24:28.755065  control     new_page          0
4282     866677  2017-01-24 05:04:14.004157  control     new_page          0
4284     847508  2017-01-03 19:31:14.396402  control     new_page          0
4311     924330  2017-01-23 07:08:56.964247  control     new_page          0
4485     838568  2017-01-15 04:02:13.337797  control     new_page          0
4693     799659  2017-01-22 09:50:16.421384  control     new_page          0
4748     872738  2017-01-08 02:16:03.976589  control     new_page          0
4962     729859  2017-01-19 14:17:09.976523  control     new_page          0
...         ...                         ...      ...          ...        ...
290811   931254  2017-01-19 03:56:48.943007  control     new_page          0
291093   922957  2017-01-12 00:58:45.303371  control     new_page          0
291100   810979  2017-01-07 18:48:46.403714  control     new_page          0
291240   807517  2017-01-22 10:07:39.903169  control     new_page          0
291358   929094  2017-01-11 03:52:10.013362  control     new_page          0
291423   848305  2017-01-19 07:30:03.635089  control     new_page          0
291728   828985  2017-01-02 13:55:08.790046  control     new_page          0
291839   740434  2017-01-13 07:04:11.067609  control     new_page          0
291876   766031  2017-01-03 22:49:27.025028  control     new_page          0
291946   861129  2017-01-12 19:00:59.118294  control     new_page          1
292147   746367  2017-01-10 04:37:37.933511  control     new_page          0
292178   645830  2017-01-14 11:12:33.289733  control     new_page          0
292235   679326  2017-01-07 07:27:46.910711  control     new_page          0
292239   908003  2017-01-22 15:17:03.083738  control     new_page          0
292405   819974  2017-01-03 05:58:44.734645  control     new_page          0
292570   778969  2017-01-21 12:59:42.740399  control     new_page          1
292748   684361  2017-01-19 03:59:57.656614  control     new_page          0
292845   893018  2017-01-10 15:05:37.522921  control     new_page          0
293017   792268  2017-01-06 09:21:58.341063  control     new_page          0
293085   884635  2017-01-19 14:19:48.484389  control     new_page          0
293393   636565  2017-01-12 07:26:31.103374  control     new_page          0
293480   638376  2017-01-18 15:41:02.395882  control     new_page          0
293568   704024  2017-01-15 17:06:09.309987  control     new_page          0
293662   927109  2017-01-04 09:14:33.647192  control     new_page          0
293888   865405  2017-01-12 08:38:50.511434  control     new_page          0
293894   741581  2017-01-09 20:49:03.391764  control     new_page          0
293996   942612  2017-01-08 13:52:28.182648  control     new_page          0
294200   928506  2017-01-13 21:32:10.491309  control     new_page          0
294253   886135  2017-01-06 12:49:20.509403  control     new_page          0
294331   689637  2017-01-13 11:34:28.339532  control     new_page          0

[1928 rows x 5 columns]
```

f. Do any of the rows have missing values?

```
In [9]: # check if any null 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
```

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 [10]: # put all treatment AND new_page into one dataframe
         df2t = df.query('group == "treatment" and landing_page == "new_page"')
```

```
In [12]: # put all control AND old_page into one dataframe
         df2c = df.query('group == "control" and landing_page == "old_page"')
```

```
In [13]: # merge two properly aligned dataframes together
         df2 = df2t.merge(df2c, how='outer')
```

```
In [14]: df2.shape
```

```
Out[14]: (290585, 5)
```

```
In [15]: df2.describe()
```

```
Out[15]:              user_id      converted
         count  290585.000000  290585.000000
         mean   788004.825246       0.119597
         std     91224.582639       0.324490
         min    630000.000000       0.000000
         25%    709035.000000       0.000000
         50%    787995.000000       0.000000
         75%    866956.000000       0.000000
         max    945999.000000       1.000000
```

```
In [16]: df2.head()
```

```
Out[16]:    user_id                   timestamp      group landing_page  converted
         0   661590  2017-01-11 16:55:06.154213  treatment     new_page          0
         1   853541  2017-01-08 18:28:03.143765  treatment     new_page          0
         2   679687  2017-01-19 03:26:46.940749  treatment     new_page          1
         3   817355  2017-01-04 17:58:08.979471  treatment     new_page          1
         4   839785  2017-01-15 18:11:06.610965  treatment     new_page          1
```

```
In [17]: # 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[17]: 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 [18]: df2.nunique()
```

```
Out[18]: user_id         290584
         timestamp       290585
         group                2
         landing_page         2
         converted            2
         dtype: int64
```

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

```
In [19]: sum(df2.user_id.duplicated())
```

```
Out[19]: 1
```

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

```
In [20]: df2[df2.duplicated(['user_id'], keep=False)]
```

```
Out[20]:       user_id                   timestamp      group landing_page  converted
         938    773192  2017-01-09 05:37:58.781806  treatment     new_page          0
         1404   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 [21]: df2 = df2[~df2.user_id.duplicated(keep='first')]
         # https://stackoverflow.com/questions/13035764/remove-rows-with-duplicate-indices-panda
```

```
In [22]: df2.shape
```

```
Out[22]: (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 [23]: df2.converted.mean()
```

```
Out[23]: 0.11959708724499628
```

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

```
In [24]: df2_control = df2.query('group == "control"')
         df2_control.converted.mean()
```

```
Out[24]: 0.1203863045004612
```

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

```
In [25]: df2_treatment = df2.query('group == "treatment"')
         df2_treatment.converted.mean()
```

```
Out[25]: 0.11880806551510564
```

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

```
In [26]: len(df2_treatment.index)/len(df2.index)
```

```
Out[26]: 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.

Answer: It appears that individuals in the treatment group had a conversion rate of 11.88% and individuals in the control grounp had a conversion rate of 12.04%. This leads us to think that the treatment group does not lead to more conversions than the treatment group. However, it remains to be seen if this is true, or due to some bias.

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 [ ]:

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

In [ ]:

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

In [ ]:

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

In [ ]:

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 [ ]:

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 [ ]:

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

In [ ]:

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 [ ]:

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 [ ]:

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

In [ ]:

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?

Put your answer here.

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 [ ]: import statsmodels.api as sm

        convert_old =
        convert_new =
        n_old =
        n_new =
```

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 [ ]:
```

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.**?

Put your answer here.
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?

Put your answer here.

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 [ ]:
```

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 [ ]:
```

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

```
In [ ]:
```

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**?

Put your answer here.

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?

Put your answer here.

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 [ ]:
```

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 [ ]:
```

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'])
```