Analyze_ab_test_results_notebook

January 14, 2018

0.1 Analyze A/B Test Results

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 that we get the same answers on quizzes as we set u
        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]: 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 below cell to find the number of rows in the dataset.

```
In [3]: df.shape[0]
```

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

number of rows in the dataset: 294478

c. The number of unique users in the dataset.

```
In [7]: df['user_id'].nunique()
```

```
Out[7]: 290584
```

number of unique users in the dataset: 290584

d. The proportion of users converted.

```
In [8]: df['converted'].mean() * 100
```

```
Out[8]: 11.96591935560551
```

The proportion of users converted: 11.97

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

```
In [9]: mismatch_g1 = df.query('group == "treatment" and landing_page == "old_page"')
        len(mismatch_g1)
```

```
Out[9]: 1965
```

```
In [10]: mismatch_g2 = df.query("group == 'control' and landing_page == 'new_page'")
         len(mismatch_g2)
```

```
Out[10]: 1928
```

```
In [11]: len(mismatch_g1) + len(mismatch_g2)
```

```
Out[11]: 3893
```

The number of times the new_page and treatment don't line up: 3893

f. Do any of the rows have missing values?

```
In [12]: # we check number of values in each rows using info function
         # entry values denote if any column has 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
```

All seen from above figures, no values are missing.

2. For the rows where **treatment** is not aligned with **new_page** or **control** is not aligned with **old_page**, we cannot be sure if this row truly received the new or old page. Use **Quiz 2** in the classroom to provide 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 [13]: df.drop(df.query("group == 'treatment' and landing_page == 'old_page'").index, inplac

         df.drop(df.query("group == 'control' and landing_page == 'new_page'").index, inplace=
```

```
In [14]: df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 290585 entries, 0 to 294477
Data columns (total 5 columns):
user_id         290585 non-null int64
timestamp       290585 non-null object
group           290585 non-null object
landing_page    290585 non-null object
converted       290585 non-null int64
dtypes: int64(2), object(3)
memory usage: 13.3+ MB
```

```
In [15]: df.to_csv('ab_edited.csv', index=False)

In [16]: df2 = pd.read_csv('ab_edited.csv')

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].
```

```
Out[17]: 0
```

3. Use **df2** and the cells below to answer questions for **Quiz3** in the classroom.

```
In [18]: df2.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 290585 entries, 0 to 290584
Data columns (total 5 columns):
user_id         290585 non-null int64
timestamp       290585 non-null object
group           290585 non-null object
landing_page    290585 non-null object
converted       290585 non-null int64
dtypes: int64(2), object(3)
memory usage: 11.1+ MB
```

a. How many unique **user_id**s are in **df2**?

```
In [19]: len(df2['user_id'].unique())
```

```
Out[19]: 290584
```

unique user_ids: 290584

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

```
In [20]: sum(df2['user_id'].duplicated())
```

```
Out[20]: 1
```

```
In [21]: df2[df2.duplicated(['user_id'], keep=False)]['user_id']
```

```
Out[21]: 1876    773192
         2862    773192
         Name: user_id, dtype: int64
```

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

```
In [22]: df2[df2.duplicated(['user_id'], keep=False)]
```

```
Out[22]:       user_id                   timestamp      group landing_page  converted
         1876   773192  2017-01-09 05:37:58.781806  treatment     new_page          0
         2862   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 [23]: time_dup = "2017-01-09 05:37:58.781806"
         df2 = df2[df2.timestamp != time_dup]
```

```
In [24]: df2.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 290584 entries, 0 to 290584
Data columns (total 5 columns):
user_id         290584 non-null int64
timestamp       290584 non-null object
group           290584 non-null object
landing_page    290584 non-null object
converted       290584 non-null int64
dtypes: int64(2), object(3)
memory usage: 13.3+ MB
```

```
In [25]: len(df['user_id'].unique())
```

```
Out[25]: 290584
```

4. Use **df2** in the below cells 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 [26]: df['converted'].mean()
```

```
Out[26]: 0.11959667567149027
```

probability of an individual converting regardless of the page they receive: 0.11959667567149027

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

```
In [27]: df_grp = df.groupby('group')
         df_grp.describe()
```

```
Out[27]:          converted                                                     user_id  \
                      count      mean       std  min  25%  50%  75%  max     count
         group
         control   145274.0  0.120386  0.325414  0.0  0.0  0.0  0.0  1.0  145274.0
         treatment 145311.0  0.118807  0.323563  0.0  0.0  0.0  0.0  1.0  145311.0

                                                                             \
                            mean           std       min       25%       50%
         group
         control    788164.072594  91287.914601  630002.0  709279.5  788128.5
         treatment  787845.618446  91161.258854  630000.0  708746.5  787874.0


                          75%       max
         group
         control    867208.25  945998.0
         treatment  866718.50  945999.0
```

Thus, given that an individual was in the `control` group, the probability they converted is 0.120386

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

Thus, given that an individual was in the `treatment` group, the probability they converted is 0.118807

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

```
In [28]: new_user = len(df.query("group == 'treatment'"))
         users=df.shape[0]
         new_user_p = new_user/users
         print(new_user_p)
```

```
0.5000636646764286
```

probability that an individual received the new page: 0.5000636646764286

e. Use the results in the previous two portions of this question to suggest if you think there is evidence that one page leads to more conversions? Write your response below.

Evidence that one page leads to more conversions? - given that an individual was in the treatment group, the probability they have converted is 0.118807 - given that an individual was in the control group, the probability they have converted is 0.120386 - we are able to find that old page does better , but by a very small margin. - changed aversion, the test span duration and other potentially influencing factors have not been acoounted for. So,we cannot state that one page leads to more conversions.This is very important as both pages show similar performance

### 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.

Hypothesis

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

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

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 **convert rate** for p_{new} under the null?

```
In [29]: p_new = df2['converted'].mean()
         print(p_new)
```

```
0.11959708724499628
```

convert rate for pnewpnew under the null: 0.11959708724499628

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

```
In [30]: p_old = df2['converted'].mean()
         print(p_old)
```

```
0.11959708724499628
```

convert rate for poldpold under the nul: 0.11959708724499628

c. What is n_{new}?

```
In [31]: n_new = len(df2.query("group == 'treatment'"))
         print(n_new)
```

```
145310
```

$n_{new} = 145310$

d. What is n_{old}?

```
In [32]: n_old = len(df2.query("group == 'control'"))
         print(n_old)
```

```
145274
```

$n_{old} = 145274$

e. Simulate n_{new} transactions with a convert rate of p_{new} under the null. Store these n_{new} 1's and 0's in **new_page_converted**.

```
In [40]: new_page_converted = np.random.choice([1, 0], size=n_new, p=[p_new, (1-p_new)])
```

f. Simulate n_{old} transactions with a convert rate of p_{old} under the null. Store these n_{old} 1's and 0's in **old_page_converted**.

```
In [41]: old_page_converted = np.random.choice([1, 0], size=n_old, p=[p_old, (1-p_old)])
```

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

```
In [42]: new_page_converted = new_page_converted[:145274]
```

```
In [43]: p_diff = (new_page_converted/n_new) - (old_page_converted/n_old)
```

h. Simulate 10,000 p_{new} - p_{old} values using this same process similarly to the one you calculated in parts **a. through g.** above. Store all 10,000 values in **p_diffs**.

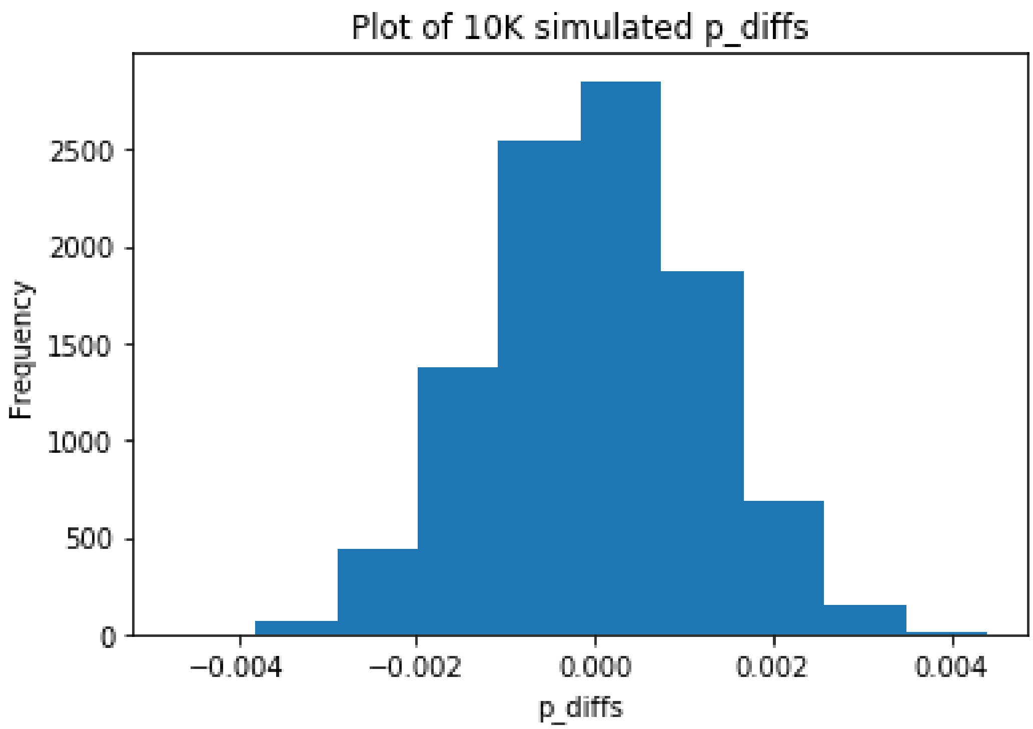
Here, value of size is different for n_new and n_old. So, computing difference will throw an error. Hence, we use mean function for both old and new page conversion simulations to overcome this problem of shape difference. We are still using probabilities as previous case.

```
In [46]: p_diffs = []

         for _ in range(10000):
             new_page_converted = np.random.choice([1, 0], size=n_new, p=[p_new, (1-p_new)]).me
             old_page_converted = np.random.choice([1, 0], size=n_old, p=[p_old, (1-p_old)]).me
             diff = new_page_converted - old_page_converted
             p_diffs.append(diff)
```

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 [34]: plt.hist(p_diffs)
         plt.xlabel('p_diffs')
         plt.ylabel('Frequency')
         plt.title('Plot of 10K simulated p_diffs');
```



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

```
In [35]: act_diff = df[df['group'] == 'treatment']['converted'].mean() -  df[df['group'] == 'c
         act_diff

Out[35]: -0.0015790565976871451

In [36]: p_diffs = np.array(p_diffs)
         p_diffs

Out[36]: array([ -9.15656809e-05,   8.30529096e-04,   4.17623883e-04, ...,
                 -9.79669080e-04,  -1.49568932e-03,   2.24847951e-03])

In [37]: (act_diff < p_diffs).mean()

Out[37]: 0.90290000000000004
```

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?

0.3 Answer:

- we are computing p values here
- This is the probability of observing our statistic,if the null hypothesis is true or not
- The most extreme in favor of the alternative portion of this statement determines the shading associated with your p-value
- we find that there is no conversion advantage in the new page.So we can conclude that null hypothesis is true as old and new perform almolst the same.As the number shows the old page performed slightly better

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 [39]: import statsmodels.api as sm
```

```
C:\Users\hi\Anaconda3\lib\site-packages\statsmodels\compat\pandas.py:56: FutureWarning: The par
  from pandas.core import datetools
```

```
In [39]: 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 [53]: z_score, p_value = sm.stats.proportions_ztest([convert_old, convert_new], [n_old, n_ne
         print(z_score, p_value)
```

```
1.31092419842 0.905058312759
```

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 [54]: from scipy.stats import norm

         print(norm.cdf(z_score))
         # Tells us how significant our z-score is

         # for our single-sides test, assumed at 95% confidence level, we calculate:
         print(norm.ppf(1-(0.05)))
         # Tells us what our critical value at 95% confidence is
         # Here, we take the 95% values as specified in PartII.1
```

```
0.905058312759
1.64485362695
```

Answer: - we found that z-score of 1.31092419842 is less than the critical value of 1.64485362695. So, we accept the null hypothesis. - we find that old pages are only minutely better than new pages - These values agree with the findings in parts j. and k.

### Part III - A regression approach

`1.` In this final part, you will see that the result you acheived in the previous A/B test can also be acheived 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 a colun 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 [42]: df['intercept']=1
         df[['control', 'treatment']] = pd.get_dummies(df['group'])
```

c. Use **statsmodels** to import your regression model. Instantiate the model, and fit the model using the two columns you created in part **b.** to predict whether or not an individual converts.

```
In [43]: import statsmodels.api as sm
         logit = sm.Logit(df['converted'],df[['intercept','treatment']])
```

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

```
In [44]: results = logit.fit()
         results.summary()
```

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

```
Out[44]: <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:                Tue, 12 Dec 2017   Pseudo R-squ.:               8.085e-06
Time:                        14:04:02   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
treatment     -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 the **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 the **Part II**?

Answer: - Our hypothesis here is: - $H_0 : p_{new}$ - $p_{old} = 0$ - $H_1 : p_{new}$ - p_{old} != 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?

Answer: - We should consider other factors into the regression model as they might influence the conversions too. For instance student segments [new v/s returning candidates] might create change aversion or even, the opposite as a predisposition to conversion. Seasonality like new terms or New years might mean more interest in new skills/ resolutions. Timestamps are inlcuded but without regionality, they do not indicate if seasonality was a factor or not. [as different countries follow different term and weather patterns. - Factors like device on which tests were taken or course which was looked at, prior academic background, age, might alter experience and ultimately, conversions. These are limitations which should be at least kept in mind while making the final decision. - The disadvantages to adding additional terms into the regression model is that even with additional factors we can never account for all influencing factors or accomodate them. Plus, small pilots and pivots sometimes work better in practice than long-drawn research without execution.

g. Now along with testing if the conversion rate changes for different pages, also add an effect based on which country a user lives. You will need to read in the **countries.csv** dataset and merge together your datasets on the approporiate 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 varaibles.** Provide the statistical output as well as a written response to answer this question.

```
In [45]: countries_df = pd.read_csv('./countries.csv')
         countries_df.head()
```

```
Out[45]:    user_id country
         0   834778      UK
```

```
        1   928468      US
        2   822059      UK
        3   711597      UK
        4   710616      UK
```

```
In [46]: df_new = countries_df.set_index('user_id').join(df2.set_index('user_id'), how='inner'
         df_new.head()
```

```
Out[46]:         country                   timestamp      group landing_page  converted
         user_id
         834778       UK  2017-01-14 23:08:43.304998    control     old_page          0
         928468       US  2017-01-23 14:44:16.387854  treatment     new_page          0
         822059       UK  2017-01-16 14:04:14.719771  treatment     new_page          1
         711597       UK  2017-01-22 03:14:24.763511    control     old_page          0
         710616       UK  2017-01-16 13:14:44.000513  treatment     new_page          0
```

```
In [47]: df_new['country'].value_counts()
```

```
Out[47]: US    203619
         UK     72466
         CA     14499
         Name: country, dtype: int64
```

```
In [48]: ### Create the necessary dummy variables
         df_new[['CA', 'US']] = pd.get_dummies(df_new['country'])[['CA','US']]

         df_new['country'].astype(str).value_counts()
```

```
Out[48]: US    203619
         UK     72466
         CA     14499
         Name: country, dtype: int64
```

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 [49]: ### Fit Your Linear Model And Obtain the Results
         df['intercept'] = 1


         log_mod = sm.Logit(df_new['converted'], df_new[['CA', 'US']])
         results = log_mod.fit()
         results.summary()
```

```
Optimization terminated successfully.
         Current function value: 0.447174
         Iterations 6
```

```
Out[49]: <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, 12 Dec 2017   Pseudo R-squ.:                 -0.2214
         Time:                        14:04:05   Log-Likelihood:            -1.2994e+05
         converged:                       True   LL-Null:                   -1.0639e+05
                                                 LLR p-value:                     1.000
         ==============================================================================
                          coef    std err          z      P>|z|      [0.025      0.975]
         ------------------------------------------------------------------------------
         CA            -2.0375      0.026    -78.364      0.000      -2.088      -1.987
         US            -1.9967      0.007   -292.314      0.000      -2.010      -1.983
         ==============================================================================
         """
```

```
In [50]: np.exp(results.params)
```

```
Out[50]: CA    0.130350
         US    0.135779
         dtype: float64
```

```
In [51]: 1/_
```

```
Out[51]: 0.00010001000100010001
```

```
In [52]: df.groupby('group').mean()['converted']
```

```
Out[52]: group
         control      0.120386
         treatment    0.118807
         Name: converted, dtype: float64
```

0.3.1 Conclusions from Regression:

- As in this logistic regression model too, we find that the values do not show a substantial difference in teh conversion rates for control group and treatment group.
- This indicates that we can acceot the Null Hypothesis and keep the existing page as is.

## Conclusions

- The performance of the old page was found better (by miniscule values only) as computed by different techniques.
- Hence, we accept the Null Hypothesis and Reject the Alternate Hypothesis.
- These inferences are strictly based on data on hand. This analysis acknowledges its limitations due to factors not included in the data. [see part III.f]

0.3.2 Resources

- Udacity Nanodegree Videos and Resources, including Links in this .ipynb
- https://pandas.pydata.org/pandas-docs/stable/generated/pandas.DataFrame.duplicated.html
- https://stackoverflow.com/questions/14657241/how-do-i-get-a-list-of-all-the-duplicate-items-using-pandas-in-python
- https://stackoverflow.com/questions/18172851/deleting-dataframe-row-in-pandas-based-on-column-value
- https://youtu.be/7FTp9JJ5DfE : Project Walkthrough Link on Slack's Project Thread

1 Thank You