Analyze_ab_test_results_notebook (NEW)

August 26, 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

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 [2]: 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 [3]: ab_test = pd.read_csv('ab_data.csv')
        ab_test.head()
```

```
Out[3]:    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 [4]: ab_test.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
```

c. The number of unique users in the dataset.

```
In [5]: ab_test.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]: ab_test.converted.mean()
```

```
Out[6]: 0.11965919355605512
```

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

```
In [7]: ab_test.groupby(['group', 'landing_page'])['landing_page'].count()

Out[7]: group      landing_page
        control    new_page          1928
                   old_page        145274
        treatment  new_page        145311
                   old_page          1965
        Name: landing_page, dtype: int64
```

1928 + 1965 = 3893

```
In [8]: # number of times when group is not treatment but langing page is new page
        groupA = len(ab_test.query('group!="treatment" and landing_page=="new_page"'))# number o

        # number of times when group is not control but langing page is old page
        groupB = len(ab_test.query('group!="control" and landing_page=="old_page"'))# number of

        group=groupA+groupB
        group

Out[8]: 3893
```

f. Do any of the rows have missing values?

```
In [9]: #check for missing values
        ab_test.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
```

COMMENT

There are no missing values

`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]: df2 = ab_test[((ab_test['group'] == 'treatment') & (ab_test['landing_page'] == 'new_pag
```

```
In [11]: # 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[11]: 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 [12]: #Find user_ids that are unique
         df2['user_id'].nunique()
```

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

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

```
In [13]: #No. of duplicate user_ids
         df2['user_id'].duplicated().sum()
```

```
Out[13]: 1
```

```
In [14]: df2.user_id[df2.user_id.duplicated()]
```

```
Out[14]: 2893    773192
         Name: user_id, dtype: int64
```

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

```
In [15]: #Row information for the duplicated user_id
         df2[df2['user_id'].duplicated()]
```

```
Out[15]:       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 [16]: #drop duplicate rows
         df2 = df2.drop(index=2893)
```

```
In [17]: #double check for duplicates
         df2['user_id'].duplicated().sum()
```

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

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 [18]: # Probability of an individual converting regardless of the page they receive
         df2['converted'].mean()
```

```
Out[18]: 0.11959708724499628
```

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

```
In [19]: # The probability of an individual converting given that an individual was in the contr
         control_group = df2[df2['group'] == 'control']['converted'].sum() / len(df2[df2['group'
         control_group
```

```
Out[19]: 0.1203863045004612
```

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

```
In [20]: # The probability of an individual converting given that an individual was in the treat
         treatment_group = df2[df2['group'] == 'treatment']['converted'].sum() / len(df2[df2['gr
         treatment_group
```

```
Out[20]: 0.11880806551510564
```

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

```
In [21]: # The probability of individual received new page
         df2[df2['landing_page'] == 'new_page']['group'].count() / len(df2)
```

```
Out[21]: 0.50006194422266881
```

e. Consider your results from parts (a) through (d) above, and explain below whether you think there is sufficient evidence to conclude that the new treatment page leads to more conversions.

ANSWER:

Based on the analysis so far, conversion from control group is just a bit higher than conversion from the treatment group.

Probability of individual converting given individual is in control group is 0.1203 or 12.03% while

Probability of individual converting given individual is in treatment group is 0.1188 or 11.88%

The difference between 'control' and 'treatment' is not significant enough to conclude evidence 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.

H0 :   0

H1 : > 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 [22]: p_new = df2['converted'].sum() / len(df2)
         p_new
```

```
Out[22]: 0.11959708724499628
```

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

```
In [23]: # probablity under null
         p_old = df2['converted'].sum() / len(df2)
         p_old
```

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

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

```
In [24]: #calculating number of queries when landing_page is equal to new_page
         n_new = df2[df2['landing_page'] == 'new_page']['landing_page'].count()
         #print n_new
         n_new
```

```
Out[24]: 145310
```

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

```
In [25]: #calculating number of queries when landing_page is equal to old_page
         n_old = df2[df2['landing_page'] == 'old_page']['landing_page'].count()
         #print n_old
         n_old
```

```
Out[25]: 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 [26]: new_page_converted = np.random.choice([0, 1], size=n_new, p=[1-p_new, p_new])
         new_page_converted

Out[26]: array([0, 0, 0, ..., 0, 0, 0])
```

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

Out[27]: array([1, 0, 0, ..., 0, 0, 1])
```

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

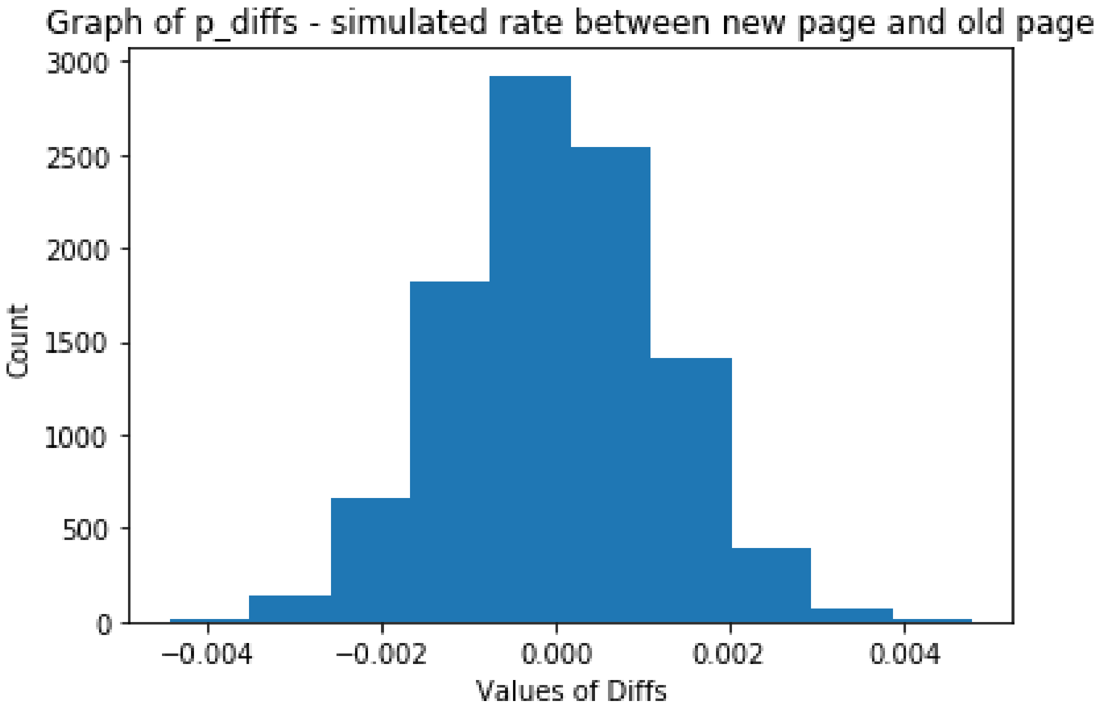
```
In [28]: #difference of p_new and p_old
         new_page_converted.mean() - old_page_converted.mean()

Out[28]: -0.00057340524796956061
```

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 [29]: # Creating sampling distribution for difference in p_new-p_old simulated values
         p_diffs = []
         for _ in range(10000):
             new_page_converted = np.random.choice([0, 1], size=n_new, p=[1-p_new, p_new])
             old_page_converted = np.random.choice([0, 1], size=n_old, p=[1-p_old, p_old])
             p_diffs.append(new_page_converted.mean() - old_page_converted.mean())
         p_diffs = np.array(p_diffs)
```

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 [30]: # The sampling distribution should be normal by the Central Limit Theorem
         plt.xlabel('Values of Diffs')
         plt.ylabel('Count')
         plt.title('Graph of p_diffs - simulated rate between new page and old page');
         plt.hist(p_diffs);
```



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

```
In [31]: actual_diffs = df2[df2['group'] == 'treatment']['converted'].mean() -  df2[df2['group']
         (p_diffs > actual_diffs).mean()
```

```
Out[31]: 0.9038000000000005
```

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?

0.2.1 ANSWER

The value calculated is called p-value

The p-value is the probability of obtaining results at least as extreme as the observed results of a statistical hypothesis test, assuming that the null hypothesis is correct.

We computed the actual difference versus observed difference (which we stored in p_diffs) in means of converted old page and converted new page.

Our p-value of 0.9 exceeds the critical value of 0.05 in this case and so we fail to reject the null hypothesis, we cannot assume the new page is doing significantly better than 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.

```
In [32]: import statsmodels.api as sm

         #rows converted with old_page
         convert_old = len(df2.query('converted==1 and landing_page=="old_page"'))

         #rows converted with new_page
         convert_new = len(df2.query('converted==1 and landing_page=="new_page"'))

         #rows_associated with old_page
         n_old = len(df2.query('landing_page=="old_page"'))

         #rows associated with new_page
         n_new = len(df2.query('landing_page=="new_page"'))
         convert_old, convert_new, n_old, n_new
```

```
/opt/conda/lib/python3.6/site-packages/statsmodels/compat/pandas.py:56: FutureWarning: The panda
  from pandas.core import datetools
```

```
Out[32]: (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 [33]: #Computing z_score and p_value
         z_stat, p_value = sm.stats.proportions_ztest([convert_old, convert_new], [n_old, n_new]

         #display z_score and p_value
         (z_stat, p_value)
```

```
Out[33]: (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 [34]: # Critical value of 95% confidence
         from scipy.stats import norm

         norm.ppf(1-(0.05))
```

```
Out[34]: 1.6448536269514722
```

ANSWER The z-score of 1.311 is less than critical value at 95% confidence interval, 1.960. Hence we fail to reject null hypothesis.

These values agree with the findings in parts j. and k that we fail to reject the null hypothesis

Part III - A regression approach

1. In this final part, you will see that the result you achieved in the A/B test in Part II above can also be achieved by performing regression.

a. Since each row is either a conversion or no conversion, what type of regression should you be performing in this case?

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 [35]: #adding an intercept column
         df2['intercept'] = 1

         #Create dummy variable column
         df2['ab_page'] = pd.get_dummies(df2['group'])['treatment']

         df2.head()
```

```
Out[35]:    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 [36]: log_mod = sm.Logit(df2['converted'], df2[['intercept', 'ab_page']])
         results = log_mod.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 [37]: results.summary2()
```

```
Out[37]: <class 'statsmodels.iolib.summary2.Summary'>
         """
                                  Results: Logit
         =================================================================
         Model:              Logit            No. Iterations:   6.0000
         Dependent Variable: converted        Pseudo R-squared: 0.000
         Date:               2021-08-26 10:47 AIC:              212780.3502
         No. Observations:   290584           BIC:              212801.5095
         Df Model:           1                Log-Likelihood:   -1.0639e+05
         Df Residuals:       290582           LL-Null:          -1.0639e+05
         Converged:          1.0000           Scale:            1.0000
         ------------------------------------------------------------------
                      Coef.   Std.Err.      z      P>|z|    [0.025   0.975]
         ------------------------------------------------------------------
         intercept    -1.9888    0.0081  -246.6690  0.0000  -2.0046  -1.9730
         ab_page      -0.0150    0.0114    -1.3109  0.1899  -0.0374   0.0074
         =================================================================

         """
```

e. What is the p-value associated with **ab_page**? Why does it differ from the value you found in **Part II**?

ANSWER

The p-value associated with ab_page is 0.1899 It is different from what was found in parts j and k because the null and alternative hypothesis model assumed that there is an equal probability of the old and new page converting users.

This is not the case in the logistic regression model The Logistic Regression performed a two-tailed test, whereas the computation done in Part II is a one-tailed test.

COMMENT

In Logistic regression H1 : Pnew - Pold! = 0

Part 2 H0 : Pnew - Pold < = 0 H1 : Pnew - Pold > 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

It's a good idea to consider new factors for the model. Other new explanatory variables that can influence whether an individual converts could be age, income or location.

For example, older users may prefer more information on the conversion pages as opposed to a younger folks, where they may prefer a more attractive page layout

However, we must be careful in adding new variables to our model. The disadvantage here is that we don't know in what direction will the additional variables influence the result.

Adding more factors into the regression model will increase or decrease confidence intervals.

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 [38]: # Store Countries.csv data in dataframe
         countries = pd.read_csv('countries.csv')
         countries.head()
```

```
Out[38]:    user_id country
         0   834778      UK
         1   928468      US
         2   822059      UK
         3   711597      UK
         4   710616      UK
```

```
In [39]: #Inner join two datas
         new = countries.set_index('user_id').join(df2.set_index('user_id'), how = 'inner')
         new.head()
```

```
Out[39]:         country                   timestamp      group landing_page  \
         user_id
         834778       UK  2017-01-14 23:08:43.304998    control     old_page
         928468       US  2017-01-23 14:44:16.387854  treatment     new_page
         822059       UK  2017-01-16 14:04:14.719771  treatment     new_page
         711597       UK  2017-01-22 03:14:24.763511    control     old_page
         710616       UK  2017-01-16 13:14:44.000513  treatment     new_page

                  converted  intercept  ab_page
         user_id
         834778           0          1        0
```

```
928468          0          1        1
822059          1          1        1
711597          0          1        0
710616          0          1        1
```

```
In [45]: #adding dummy variables country column
         new[['UK', 'US']] = pd.get_dummies(new['country'])[['UK', 'US']]
         new.head()
```

```
Out[45]:         country                  timestamp      group landing_page  \
         user_id
         834778       UK  2017-01-14 23:08:43.304998    control     old_page
         928468       US  2017-01-23 14:44:16.387854  treatment     new_page
         822059       UK  2017-01-16 14:04:14.719771  treatment     new_page
         711597       UK  2017-01-22 03:14:24.763511    control     old_page
         710616       UK  2017-01-16 13:14:44.000513  treatment     new_page

                  converted  intercept  ab_page  UK  US  CA  ab_US  ab_UK  ab_CA
         user_id
         834778           0          1        0   1   0   0      0      0      0
         928468           0          1        1   0   1   0      1      0      0
         822059           1          1        1   1   0   0      0      1      0
         711597           0          1        0   1   0   0      0      0      0
         710616           0          1        1   1   0   0      0      1      0
```

```
In [46]: log_mod = sm.Logit(new['converted'], new[['intercept', 'UK', 'US']])
         results = log_mod.fit()
         results.summary2()
```

```
Optimization terminated successfully.
         Current function value: 0.366116
         Iterations 6
```

```
Out[46]: <class 'statsmodels.iolib.summary2.Summary'>
         """
                                  Results: Logit
         ==================================================================
         Model:              Logit            No. Iterations:   6.0000
         Dependent Variable: converted        Pseudo R-squared: 0.000
         Date:               2021-08-26 10:50 AIC:              212780.8333
         No. Observations:   290584           BIC:              212812.5723
         Df Model:           2                Log-Likelihood:   -1.0639e+05
         Df Residuals:       290581           LL-Null:          -1.0639e+05
         Converged:          1.0000           Scale:            1.0000
         -------------------------------------------------------------------
                        Coef.   Std.Err.     z      P>|z|    [0.025   0.975]
         -------------------------------------------------------------------
         intercept     -2.0375    0.0260  -78.3639  0.0000  -2.0885  -1.9866
```

```
UK              0.0507    0.0284    1.7863  0.0740  -0.0049   0.1064
US              0.0408    0.0269    1.5178  0.1291  -0.0119   0.0935
==================================================================

"""
```

Our p-values exceed the critical value of 0.05, indicating that we should fail to reject the null hypothesis and conclude that there is not sufficient evidence to suggest that there is an interaction between country and page received that will predict whether a user converts or not.

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 [47]: new['ab_US'] = new['US'] * new['ab_page']
         new['ab_UK'] = new['UK'] * new['ab_page']
```

```
In [49]: logit_mod = sm.Logit(new['converted'], new[['intercept','ab_page','US', 'UK','ab_US', '
         results = logit_mod.fit()
         results.summary2()
```

```
Optimization terminated successfully.
         Current function value: 0.366109
         Iterations 6
```

```
Out[49]: <class 'statsmodels.iolib.summary2.Summary'>
         """
                                  Results: Logit
         ==================================================================
         Model:              Logit            No. Iterations:   6.0000
         Dependent Variable: converted        Pseudo R-squared: 0.000
         Date:               2021-08-26 10:51 AIC:              212782.6602
         No. Observations:   290584           BIC:              212846.1381
         Df Model:           5                Log-Likelihood:   -1.0639e+05
         Df Residuals:       290578           LL-Null:          -1.0639e+05
         Converged:          1.0000           Scale:            1.0000
         -------------------------------------------------------------------
                         Coef.   Std.Err.     z      P>|z|    [0.025   0.975]
         -------------------------------------------------------------------
         intercept     -2.0040    0.0364  -55.0077  0.0000  -2.0754  -1.9326
         ab_page       -0.0674    0.0520   -1.2967  0.1947  -0.1694   0.0345
         US             0.0175    0.0377    0.4652  0.6418  -0.0563   0.0914
         UK             0.0118    0.0398    0.2957  0.7674  -0.0663   0.0899
         ab_US          0.0469    0.0538    0.8718  0.3833  -0.0585   0.1523
         ab_UK          0.0783    0.0568    1.3783  0.1681  -0.0330   0.1896
         ==================================================================

         """
```

0.2.2 CONCLUSION

Once again we couldn't find anything significant to reject our null hypothesis.

Hence, based on the available information, we do not have sufficient evidence to suggest that the new page results in more conversions than the old page.

RECOMMENDATION: The E-commerce company is advised to keep the old page. This will help the company save money and other resources that will go into creating a new page

0.2.3 SOURCES

https://www.investopedia.com/terms/p/p-value.asp https://stackoverflow.com/questions/49814258/s
attributeerror-module-scipy-stats-has-no-attribute-chisqprob course : 2020 Data Science & Machine Learning Bootcamp

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

Out[44]: 0
```