Analyze_ab_test_results_notebook

January 23, 2021

0.1 Analyze A/B Test Results

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, I will be working to analyse results of an A/B test run by an e-commerce website. my 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.

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]: #Read the dataset
        abo_df = pd.read_csv('ab_data.csv')
        abo_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]: #call teh shape to knoe the rows and cloumns.
        abo_df.shape
```

```
Out[3]: (294478, 5)
```

c. The number of unique users in the dataset.

```
In [4]: # call unique to extract the number of unique ids
        abo_df.user_id.nunique()
```

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

d. The proportion of users converted.

```
In [5]: #find the proportion by taking the mean multiplie it by 100
        print("Proportion of user converted is:")
        abo_df.converted.mean() *100
```

```
Proportion of user converted is:


Out[5]: 11.965919355605511
```

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

```
In [6]: #Extract the total where newpage and treatment dont match
        nepg = abo_df.query("group == 'treatment' and landing_page == 'old_page'").shape[0]
        treat = abo_df.query("group == 'control' and landing_page == 'new_page'").shape[0]
        notmatch = nepg + treat
        notmatch
```

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

f. Do any of the rows have missing values?

```
In [7]: #Check missing values by dislpying the info and check for any missing data.
        abo_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 [8]: #drop rows that dont match and create new dataset

        drpod = abo_df.query('group == "treatment"').query('landing_page == "new_page"')
        df2 = drpod.append(abo_df.query('group == "control"').query('landing_page == "old_page"'

In [9]: # Double Check all of the correct rows were removed - this should be 0
        df2[((df2['group'] == 'treatment') == (df2['landing_page'] == 'new_page')) == False].sha

Out[9]: 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 [10]: # call unique to extract the number of unique ids
         df2.user_id.nunique()

Out[10]: 290584
```

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

```
In [11]: #Check duplicat!
         df2[df2.duplicated(['user_id'])].user_id

Out[11]: 2893    773192
         Name: user_id, dtype: int64
```

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

```
In [12]: df2[df2.duplicated(['user_id'],keep=False)]

Out[12]:       user_id                   timestamp      group landing_page  converted
         1899   773192  2017-01-09 05:37:58.781806  treatment     new_page          0
         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 [13]: #remove the repated id using drop
         df2.drop_duplicates('user_id', inplace=True)

In [14]: #Check the data sahpe rows
         df2.shape

Out[14]: (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 [15]: #calculate the mean to get probability of converting
         df2.converted.mean()

Out[15]: 0.11959708724499628
```

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

```
In [16]: #groupby converted control

         df2.query('group == "control"')['converted'].mean()

Out[16]: 0.1203863045004612
```

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

```
In [17]: #groupby converted treatment

         df2.query('group == "treatment"')['converted'].mean()

Out[17]: 0.11880806551510564
```

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

```
In [18]: df2.query('landing_page == "new_page"').shape[0]/df2.landing_page.shape[0]

Out[18]: 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.

There is not enough difference between the control group and treatment 12% and 11.8%, so it is hard to prove that the old page had more success.

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

```
In [19]: p_new = df2['converted'].mean()
         print("conversion rate for p_new  is: ",p_new)
```

```
conversion rate for p_new  is:  0.119597087245
```

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

```
In [20]: p_old = df2['converted'].mean()
         print("conversion rate for p_old  is: ",p_old)
```

```
conversion rate for p_old  is:  0.119597087245
```

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

```
In [21]: n_new = df2.query("group == 'treatment'").user_id.nunique()
         print("uumber of users in tratment group: ",n_new)
```

```
uumber of users in tratment group:  145310
```

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

```
In [22]: n_old=df2.query('group=="control"').shape[0]
         print("number of users in control page: ", n_old)
```

```
number of users in control page:  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 [23]: new_page_converted = np.random.binomial(n_new, p_new)
```

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

```
In [24]: old_page_converted = np.random.binomial(n_old, p_old)
```

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

```
In [25]: difr_pd = new_page_converted/n_new - old_page_converted/n_old
         print("diffrence is ", difr_pd)
```

```
diffrence is  0.0008581690062153813
```

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**.

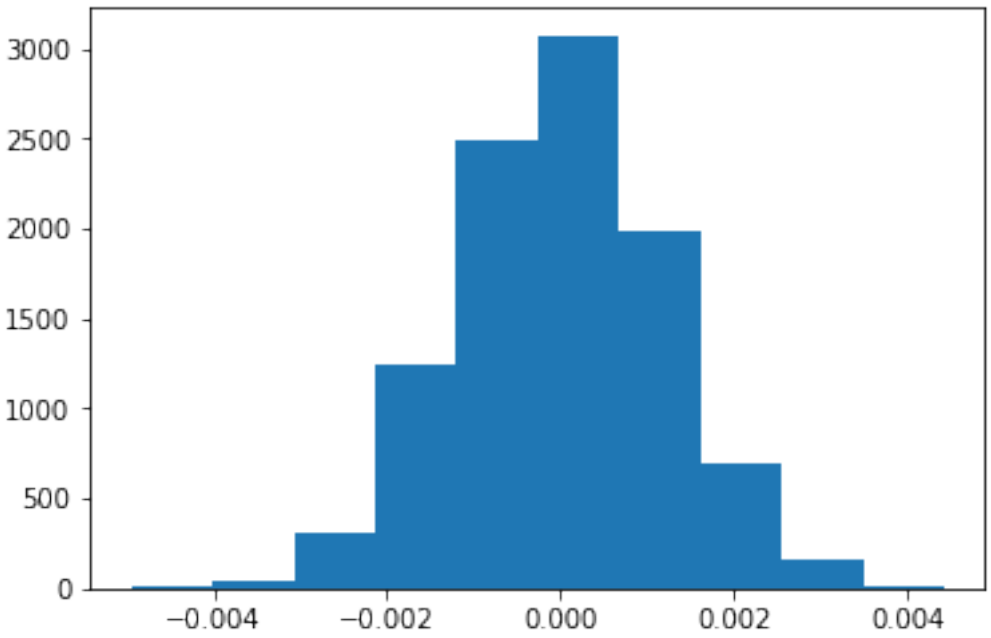
```
In [26]: p_diffs =[]

         for _ in range(1,10000):
             new_page_converted = np.random.binomial(n_new, p_new)
             old_page_converted = np.random.binomial(n_old, p_old)
             difr_pd = new_page_converted / n_new - old_page_converted / n_old
             p_diffs.append(difr_pd)
```

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 [44]: #plot line for observed:
         plt.hist(p_diffs)
```

```
Out[44]: (array([   6.,   37.,  307., 1237., 2490., 3071., 1981.,  694.,
                  157.,   19.]),
          array([-0.00495085, -0.00401344, -0.00307603, -0.00213862, -0.00120121,
                 -0.0002638 ,  0.00067361,  0.00161102,  0.00254842,  0.00348583,
                  0.00442324]),
          <a list of 10 Patch objects>)
```



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

```
In [28]: # The actual difference  :
         a_diff = df2.query("group == 'treatment'")['converted'].mean() - df2.query("group == 'c
         print("diffrence = ",a_diff)
```

```
diffrence =  -0.00157823898536
```

```
In [29]: #calculate P-value
         p_v = (p_diffs>a_diff).mean()
         print("p_value = ",p_v)
```

```
p_value =  0.905390539054
```

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?

Since we calculte the p-value is 0.9 we faild to reject the null hypothesis Which mean that the Null is true, so the old page should remain

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 [30]: import statsmodels.api as sm

         convert_old = df2.query("landing_page == 'old_page' and converted == 1").shape[0]
         print("old page with conversion: ",convert_old)
         convert_new = df2.query("landing_page == 'new_page' and converted == 1").shape[0]
         print("new page with conversion: ",convert_new)
         n_old = df2[df2['group'] == 'control'].shape[0]
         print("The rows associated with old page: ",n_old)
         n_new = df2[df2['group'] == 'treatment'].shape[0]
         print("The rows associated with new page: ",n_new)
```

```
old page with conversion:  17489
new page with conversion:  17264
The rows associated with old page:  145274
The rows associated with new page:  145310
```

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

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 [31]: z_score, p_value = sm.stats.proportions_ztest([convert_old, convert_new], [n_old, n_new
         print("z_score:", z_score, "\np_value:", p_value)
         #z_score, p_value = sm.stats.proportions_ztest([convert_new, convert_old], [n_new, n_ol
         #z_score, p_value
```

```
z_score: 1.31092419842
p_value: 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.**?

The result is simlier to the finding above in j,k, we fail to rejct the null hypothiss .

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?

as there are only two outcome i will be using logstic 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 [32]: #create intercept column
         df2['intercept']=1
         #create dummies
         df2[['control','treatment']] = pd.get_dummies(df2['group'])

         df2.head()
```

```
Out[32]:    user_id                   timestamp      group landing_page  converted  \
         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
         6   679687  2017-01-19 03:26:46.940749  treatment     new_page          1
         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
```

```
   intercept  control  treatment
2          1        0          1
3          1        0          1
6          1        0          1
8          1        0          1
9          1        0          1
```

```
In [33]: df_ab = df2.rename(columns={'treatment': 'ab_page'})
         df_ab.head()
```

```
Out[33]:    user_id                   timestamp      group landing_page  converted  \
         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
         6   679687  2017-01-19 03:26:46.940749  treatment     new_page          1
         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

            intercept  control  ab_page
         2          1        0        1
         3          1        0        1
         6          1        0        1
         8          1        0        1
         9          1        0        1
```

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 [34]: import statsmodels.api as sm

         log_b = sm.Logit(df_ab['converted'],df_ab[['intercept' ,'ab_page']])
         results = log_b.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 [35]: results.summary2()
```

```
Out[35]: <class 'statsmodels.iolib.summary2.Summary'>
         """
                                  Results: Logit
         =================================================================
         Model:              Logit            No. Iterations:   6.0000
```

```
Dependent Variable: converted        Pseudo R-squared: 0.000
Date:               2021-01-23 09:57 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**? **Hint**: What are the null and alternative hypotheses associated with your regression model, and how do they compare to the null and alternative hypotheses in **Part II**?

The P-value is 0.1899 Hence,the new landing page is not significant in customers decision whether to convert or not.

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?

If we added another factor it would increase the accuracy of our result. Factor such as time can make a difference in our analysis to understand user behaviour. On the other hand and if we keep adding other factor to our sample it will increase the chance of multicollinearity which consider as a disadvantage

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 [36]: countryfile_df = pd.read_csv('./countries.csv')
         updt_df = countryfile_df.set_index('user_id').join(df_ab.set_index('user_id'), how='inn
         updt_df.head()
```

```
Out[36]:         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  control  ab_page
user_id
834778           0          1        1        0
928468           0          1        0        1
822059           1          1        0        1
711597           0          1        1        0
710616           0          1        0        1
```

In [37]:
```
#get the new unique value
updt_df['country'].unique()
```

Out[37]:
```
array(['UK', 'US', 'CA'], dtype=object)
```

In [38]:
```
#create dummies variable with result of uniqe value
updt_df[['UK', 'US', 'CA']] = pd.get_dummies(updt_df['country'])[['UK', 'US', 'CA']]


updt_df.head()
```

Out[38]:
```
         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  control  ab_page  UK  US  CA
user_id
834778           0          1        1        0   1   0   0
928468           0          1        0        1   0   1   0
822059           1          1        0        1   1   0   0
711597           0          1        1        0   1   0   0
710616           0          1        0        1   1   0   0
```

In [39]:
```
log_b = sm.Logit(updt_df['converted'], updt_df[['intercept','UK','US']]).fit()
```

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

In [40]:
```
log_b.summary2()
```

Out[40]:
```
<class 'statsmodels.iolib.summary2.Summary'>
"""
```

```
                           Results: Logit
=================================================================
Model:              Logit            No. Iterations:   6.0000
Dependent Variable: converted        Pseudo R-squared: 0.000
Date:               2021-01-23 09:57 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
=================================================================

"""
```

p_value of the dummies variables coclude that they are not

The influence of landing_page in Uk & US is not different to the influence of landing_page in the other countries.

we fail to reject the null hypothesis.

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 [41]:
```
#Create interacton between  UK and US
updt_df['UK_ab_page'] = updt_df['UK'] * updt_df['ab_page']
updt_df['US_ab_page'] = updt_df['US'] * updt_df['ab_page']
updt_df.head()
```

Out[41]:

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

```
710616           0          1        0        1   1   0   0           1

         US_ab_page
user_id
834778            0
928468            1
822059            0
711597            0
710616            0
```

In [42]:
```
#Create logistic regression for the intereaction variable between ab_page and country u
abo_df['intercept'] = 1 # add column for interpret
log_b = sm.Logit(updt_df['converted'], updt_df[['intercept', 'ab_page','UK', 'US', 'UK_
results = log_b.fit()
results.summary2()
```

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

Out[42]:
```
<class 'statsmodels.iolib.summary2.Summary'>
"""
                          Results: Logit
=================================================================
Model:              Logit            No. Iterations:   6.0000
Dependent Variable: converted        Pseudo R-squared: 0.000
Date:               2021-01-23 09:57 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
UK             0.0118    0.0398    0.2957  0.7674  -0.0663   0.0899
US             0.0175    0.0377    0.4652  0.6418  -0.0563   0.0914
UK_ab_page     0.0783    0.0568    1.3783  0.1681  -0.0330   0.1896
US_ab_page     0.0469    0.0538    0.8718  0.3833  -0.0585   0.1523
=================================================================

"""
```

Again the result proves that there is no evidince intraction between the page and countries

0.3 Conclusion

To conclude, There is no not suffint evidence that the new page add any value than the old page beside all countries did not affect the the conversion rate so once again we failed to rejct the hypothesis using many ways amd methods ro ptove that,

Final word that the old page should remained based on the A/B test analysis have been prefomed.

0.4 Refrence

Github the resource posted on the notebbok

```
In [43]: from subprocess import call
         call(['python', '-m', 'nbconvert', 'Analyze_ab_test_results_notebook.ipynb'])
```

```
Out[43]: 0
```