Analyze_ab_test_results_notebook

March 27, 2022

1 Analyze A/B Test Results

This project will assure you have mastered the subjects covered in the statistics lessons. We have organized the current notebook into the following sections:

- Section ??
- Section ??
- Section ??
- Section ??
- Section ??
- Section ??

Specific programming tasks are marked with a **ToDo** tag.

Introduction

A/B tests are very commonly performed by data analysts and data scientists. 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 webpage, - Keep the old webpage, or - Perhaps run the experiment longer to make their decision.

Each **ToDo** task below has an associated quiz present in the classroom. Though the classroom quizzes are **not necessary** to complete the project, they help ensure you are on the right track as you work through the project, and you can feel more confident in your final submission meeting the rubric specification.

> **Tip**: Though it's not a mandate, students can attempt the classroom quizzes to ensure statistical numeric values are calculated correctly in many cases.

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)
```

```
In [3]: df = pd.read_csv("ab_data.csv")

In [4]: df.head()

Out[4]:    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 [5]: df.shape[0]

Out[5]: 294478
```

c. The number of unique users in the dataset.

```
In [6]: df.user_id.nunique()

Out[6]: 290584

In [7]: #Number of not unique
        df.shape[0] -df.user_id.nunique()

Out[7]: 3894

In [8]:  df['converted'].sum()

Out[8]: 35237
```

d. The proportion of users converted.

```
In [9]: df['converted'].sum()/df.user_id.nunique()

Out[9]: 0.12126269856564711
```

e. The number of times when the "group" is `treatment` but "landing_page" is not a `new_page`.

```
In [10]: treatment_pag_new  = df.query("group =='control' and landing_page == 'new_page'")
         treatment_pag_od  = df.query("group =='treatment' and landing_page == 'old_page'")
         treatment_pag_new.shape[0] + treatment_pag_od.shape[0]

Out[10]: 3893
```

f. Do any of the rows have missing values?

```
In [11]: 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
```

for the rows where `treatment` does not match with `new_page` or `control` does not match with `old_page`, we cannot be sure if such rows truly received the new or old wepage.

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 [66]: # Remove the inaccurate rows, and store the result in a new dataframe df2
         df2 = df[(df['group'] ==  'control') & (df['landing_page'] == 'old_page')]
         df2 = df2.append(df[(df['group'] ==  'treatment') & (df['landing_page'] == 'new_page')]
```

```
In [67]: # Double Check all of the incorrect rows were removed from df2 -
         # Output of the statement below should be 0
         df2[((df2['group'] == 'treatment') == (df2['landing_page'] == 'new_page')) == False].sh

Out[67]: 0
```

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

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

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

Out[14]: 290584
```

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

```
In [15]: 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
```

```
In [52]: id_rep_value = df2[df2.user_id.duplicated()]['user_id'].values
         id_rep_value

Out[52]: array([773192])
```

c. Display the rows for the duplicate **user_id**?

```
In [55]: df2[df2['user_id'] == id_rep_value[0]]
```

Out[55]:

	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**, from the **df2** dataframe.

In [62]:
```
# df.drop(id_rep_index)
df2.drop(df2[df2.user_id.duplicated()]['user_id'].index, inplace = True)
```

In [63]:
```
df2[df2.user_id.duplicated()]
```

Out[63]:
```
Empty DataFrame
Columns: [user_id, timestamp, group, landing_page, converted]
Index: []
```

a. What is the probability of an individual converting regardless of the page they receive?

In [78]:
```
df2['converted'].mean()
```

Out[78]: 0.11959749882133504

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

In [79]:
```
c_prob = df2[df2["group"] == 'control']['converted'].mean()
c_prob
```

Out[79]: 0.1203863045004612

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

In [80]:
```
t_prob = df2[df2["group"] == 'treatment']['converted'].mean()
t_prob
```

Out[80]: 0.11880888313869065

In [81]:
```
# Calculate the actual difference (obs_diff) between the conversion rates for the two g
obs_diff =  c_prob - t_prob
obs_diff
```

Out[81]: 0.0015774213617705535

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

In [85]:
```
df2[df2["landing_page"] == 'new_page'].shape[0]/df2.shape[0]
```

Out[85]: 0.5000602237570677

e. Consider your results from parts (a) through (d) above, and explain below whether the new `treatment` group users lead to more conversions.

**Probability of individual converting given individual is in control group is 0.1203863045004612. Probability of individual converting given individual is in treatment group is 0.11880724790277405. According to the analysis this is clear that there is no more conversion between new page and old page. As the converting rate is similar in both cases so it is important to consider other factors.

```
In [ ]: <a id='ab_test'></a>
        ## Part II - A/B Test

        Since a timestamp is associated with each event, you could run a hypothesis test continu

        However, then the hard questions would be:
        - Do you stop as soon as one page is considered significantly better than another or doe
        - How long do you run to render a decision that neither page is better than another?

        ANSWER: If we assume the old page is better unless the new page proves to be better, the
        Null Hypotheses: polypoid is equal greater or equal to p new
        Alternative Hypothesis: polypoid is less than new

        ### ToDo 2.1
        For now, consider you need to make the decision just based on all the data provided.If y

        H0:Pold=Pnew
        H1:Pnew>Pold
        or...

        H0:Pold-Pnew=0
        H1:Pnew-Pold>0
```

1.0.1 ToDo 2.2 - Null Hypothesis H_0 Testing

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 pold and pnew , which are the converted rates for the old and new pages.

Assume: Null hypothesis: the conversion rate of the old_page is greater or the same than the conversion rate of the newpage. H0: Pold >= Pnew

Alternative hypothesis: the conversion rate of the old_page is less than the conversion rate of the newpage. H1: Pnew > Pold

Assume under the null hypothesis, pnew and pold both have "true" success rates equal to the converted success rate regardless of page - that is pnew and pold 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.

a. What is the **conversion rate** for p_{new} under the null hypothesis?

```
In [23]: P_null = df2[df2['landing_page'] == 'new_page']['converted'].sum() / df2[df2['landing_p
         P_null
```

```
Out[23]: 0.11880806551510564
```

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

```
In [24]: P_old = P_null
         P_old
```

```
Out[24]: 0.11880806551510564
```

c. What is n_{new}, the number of individuals in the treatment group? *Hint*: The treatment group users are shown the new page.

```
In [25]: n_new = df2.query("landing_page == 'new_page'").shape[0]
         n_new
```

```
Out[25]: 145310
```

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

```
In [26]: n_old = df2.query("landing_page == 'old_page'").shape[0]
         n_old
```

```
Out[26]: 145274
```

```
In [27]: P_null - P_old
```

```
Out[27]: 0.0
```

2 e. Simulate Sample for the `treatment` Group

Simulate n_{new} transactions with a conversion rate of p_{new} under the null hypothesis. *Hint*: Use `numpy.random.choice()` method to randomly generate n_{new} number of values. Store these n_{new} 1's and 0's in the `new_page_converted` numpy array.

```
In [28]: # Simulate a Sample for the treatment Group
         new_page_converted = np.random.binomial(1, P_null, n_new)
```

f. Simulate Sample for the `control` Group Simulate n_{old} transactions with a conversion rate of p_{old} under the null hypothesis. Store these n_{old} 1's and 0's in the `old_page_converted` numpy array.

```
In [29]: # Simulate a Sample for the control Group
         old_page_converted = np.random.binomial(1, P_null, n_old)
```

g. Find the difference in the "converted" probability $(p'_{new}$ - $p'_{old})$ for your simulated samples from the parts (e) and (f) above.

```
In [31]: diff_prob = new_page_converted.mean() - old_page_converted.mean()
         diff_prob

Out[31]: -0.00044930021849878821
```

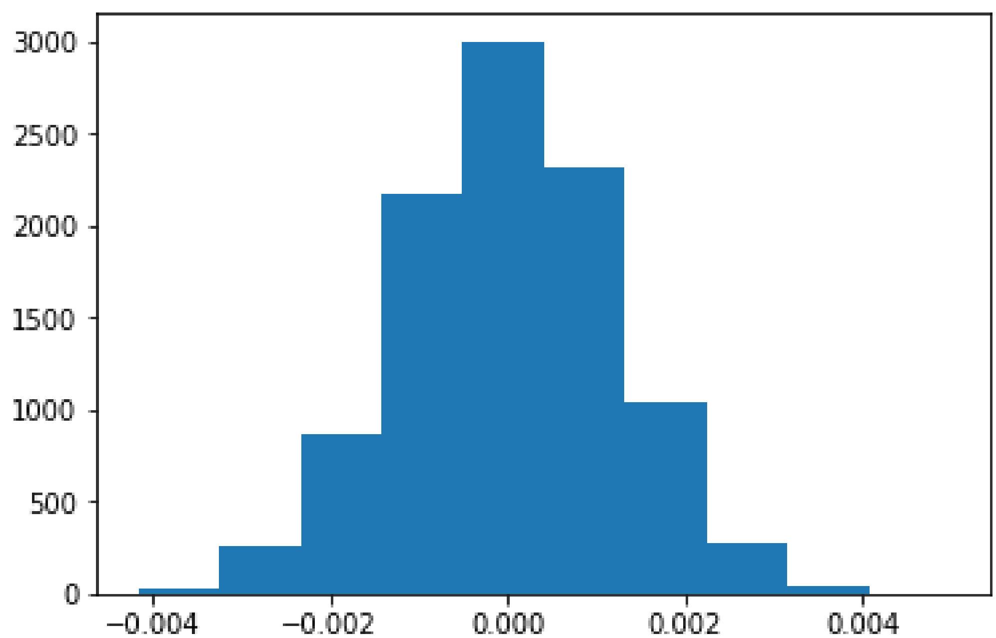
h. Sampling distribution Re-create `new_page_converted` and `old_page_converted` and find the $(p'_{new}$ - $p'_{old})$ value 10,000 times using the same simulation process you used in parts (a) through (g) above.

Store all $(p'_{new}$ - $p'_{old})$ values in a NumPy array called `p_diffs`.

```
In [32]: # Sampling distribution
         p_diffs = []
         new_converted_simulation = np.random.binomial(n_new, df2['converted'].mean(), 10000)/n_
         old_converted_simulation = np.random.binomial(n_old, df2['converted'].mean(), 10000)/n_
         p_diffs = new_converted_simulation - old_converted_simulation
```

i. Histogram 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 [33]: plt.hist(p_diffs);
```



j. What proportion of the **p_diffs** are greater than the actual difference observed in the `df2` data?

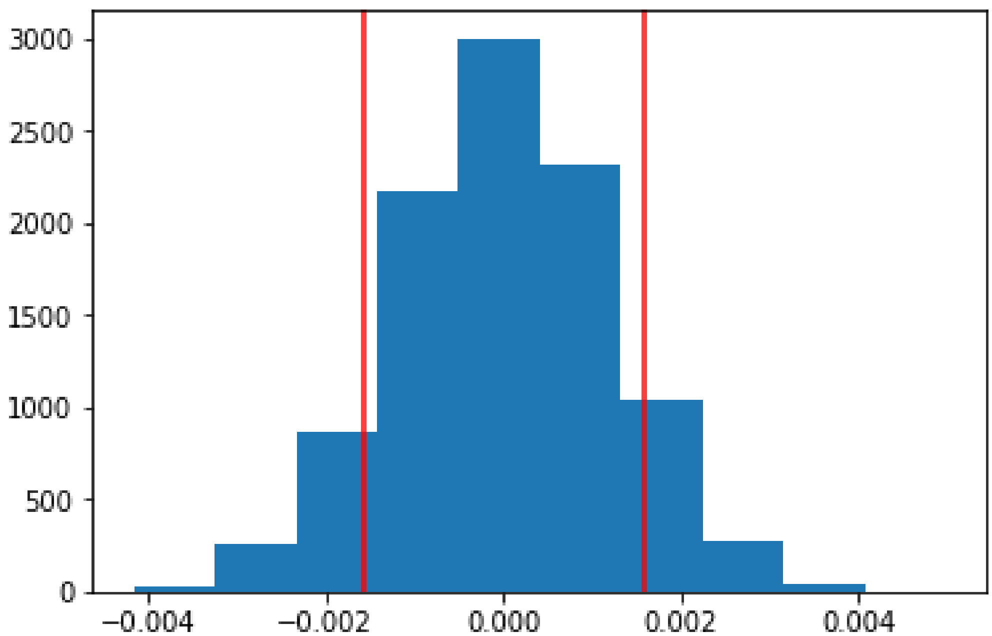
In [34]:
```
obs_diff = t_prob - c_prob

low_prob = (p_diffs < obs_diff).mean()
high_prob = (p_diffs.mean() + (p_diffs.mean() - obs_diff) < p_diffs).mean()

plt.hist(p_diffs);
plt.axvline(obs_diff, color='red');
plt.axvline(p_diffs.mean() + (p_diffs.mean() - obs_diff), color='red');

p_val = low_prob + high_prob
print(p_val)
```

```
0.1883
```



In [35]:
```
#calculate the proportion of p_diffs greater than the observe difference
(p_diffs > obs_diff).mean()
```

Out[35]: 0.9068000000000005

**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?

90.69% is the proportion of the p_diffs that are greater than the actual difference observed in ab_data.csv. In scientific studies this value is also called p-value. This value means that we cannot reject the null hypothesis and that we do not have sufficient evidence that the new_page has a higher conversion rate 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 [73]: import statsmodels.api as sm

         # number of conversions with the old_page
         convert_old = df2[df2['landing_page'] == "old_page"]['converted'].sum()

         # number of conversions with the new_page
         convert_new = df2[df2['landing_page'] == "new_page"]['converted'].sum()

         # number of individuals who were shown the old_page
         n_old = df2[df2['landing_page'] == "old_page"].shape[0]

         # number of individuals who received new_page
         n_new = df2[df2['landing_page'] == "new_page"].shape[0]
         n_new
```

```
Out[73]: 145309
```

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 [74]: import statsmodels.api as sm
         # ToDo: Complete the sm.stats.proportions_ztest() method arguments
         z_score, p_value = sm.stats.proportions_ztest([convert_old, convert_new], [n_old, n_new

         z_score, p_value
```

```
Out[74]: (1.3102408579271012, 0.90494281611597494)
```

```
In [ ]: The z -score and p_value indicate that one does not reject a Null. This means that the o
```

Part III - A regression approach

2.0.1 ToDo 3.1

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 in the `df2` data is either a conversion or no conversion, what type of regression should you be performing in this case?

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

b. The goal is to use **statsmodels** library to fit the regression model you specified in part **a.** above to see if there is a significant difference in conversion based on the page-type a customer receives. However, you first need to create the following two columns in the `df2` dataframe: 1. `intercept` - It should be `1` in the entire column. 2. `ab_page` - It's a dummy variable column, having a value `1` when an individual receives the **treatment**, otherwise `0`.

```
In [75]: df2['intercept'] = 1
         df2[['temporary_page', 'ab_page']] = pd.get_dummies(df2['group'])
         df2 = df2.drop('temporary_page', axis=1)
         df2.head()
```

```
Out[75]:    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
         4   864975  2017-01-21 01:52:26.210827  control     old_page          1
         5   936923  2017-01-10 15:20:49.083499  control     old_page          0
         7   719014  2017-01-17 01:48:29.539573  control     old_page          0

            intercept  ab_page
         0          1        0
         1          1        0
         4          1        0
         5          1        0
         7          1        0
```

c. Use **statsmodels** to instantiate your regression model on the two columns you created in part (b). above, then fit the model to predict whether or not an individual converts.

```
In [76]: lm = sm.Logit(df2['converted'], df2[['intercept', 'ab_page']])
         results = lm.fit()
```

```
Optimization terminated successfully.
         Current function value: 0.366119
         Iterations 6
```

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

```
In [77]: results.summary2()
```

```
Out[77]: <class 'statsmodels.iolib.summary2.Summary'>
         """
                                   Results: Logit
         ==================================================================
         Model:              Logit            No. Iterations:   6.0000
         Dependent Variable: converted        Pseudo R-squared: 0.000
         Date:               2022-03-27 20:59 AIC:              212780.0972
         No. Observations:   290583           BIC:              212801.2565
```

```
Df Model:           1              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    -1.9888    0.0081  -246.6690  0.0000  -2.0046  -1.9730
ab_page      -0.0150    0.0114    -1.3102  0.1901  -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**?

Hints: - 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**? - You may comment on if these hypothesis (Part II vs. Part III) are one-sided or two-sided. - You may also compare the current p-value with the Type I error rate (0.05).

The probability value related with ab_page is 0.190 here, which is like to the previous values, but slightly upper. I tried to predict whether it will convert based on their page. The null hypothesis occur when ab_page = 1, transform = 0; The alternative hypothesis occur when ab_page = 1, the transformation is Approaching to be 1.

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?

it's good to work! when considering Additional factors into the regression models they may influence the conversions also. The disadvantage is complexity because we don't know that our additional factor will influence the result in which direction when our additional factor changes every time based on an additional factor.

g. Adding countries Now along with testing if the conversion rate changes for different pages, also add an effect based on which country a user lives in.

1. You will need to read in the **countries.csv** dataset and merge together your `df2` datasets on the appropriate rows. You call the resulting dataframe `df_merged`. Here are the docs for joining tables.

2. Does it appear that country had an impact on conversion? To answer this question, consider the three unique values, `['UK', 'US', 'CA']`, in the `country` column. Create dummy variables for these country columns. >**Hint:** Use `pandas.get_dummies()` to create dummy variables. **You will utilize two columns for the three dummy variables.**

Provide the statistical output as well as a written response to answer this question.

```
In [45]: # Read the countries.csv
         countries_df = pd.read_csv('countries.csv')
```

```
In [46]: countries_df.head()
```

```
Out[46]:    user_id country
        0   834778      UK
        1   928468      US
        2   822059      UK
        3   711597      UK
        4   710616      UK
```

```
In [47]: # Join with the df2 dataframe
         df_new = countries_df.set_index('user_id').join(df2.set_index('user_id'), how='inner')
         # Create the necessary dummy variables
         df_new[['CA', 'UK', 'US']] = pd.get_dummies(df_new['country'])
         df_new.head()
```

```
Out[47]:         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  CA  UK  US
         user_id
         834778           0          1        0   0   1   0
         928468           0          1        1   0   0   1
         822059           1          1        1   0   1   0
         711597           0          1        0   0   1   0
         710616           0          1        1   0   1   0
```

h. Fit your model and obtain the results 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 are there significant effects on conversion. **Create the necessary additional columns, and fit the new model.**

Provide the summary results (statistical output), and your conclusions (written response) based on the results.

```
In [48]: df_new['CA_ab']=df_new['CA']*df_new['ab_page']
         df_new['UK_ab']=df_new['UK']*df_new['ab_page']
```

```
In [49]: # Fit your model, and summarize the results
         logit_mod = sm.Logit(df_new['converted'],df_new[['intercept', 'ab_page', 'CA', 'UK', 'C
         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:               2022-03-27 14:24 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    -1.9865    0.0096  -206.3440  0.0000  -2.0053  -1.9676
ab_page      -0.0206    0.0137    -1.5052  0.1323  -0.0473   0.0062
CA           -0.0175    0.0377    -0.4652  0.6418  -0.0914   0.0563
UK           -0.0057    0.0188    -0.3057  0.7598  -0.0426   0.0311
CA_ab        -0.0469    0.0538    -0.8718  0.3833  -0.1523   0.0585
UK_ab         0.0314    0.0266     1.1807  0.2377  -0.0207   0.0835
==================================================================

"""
```

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

Out[50]:
```
intercept    0.137178
ab_page      0.979646
CA           0.982625
UK           0.994272
CA_ab        0.954198
UK_ab        1.031896
dtype: float64
```

Conclusions: None of the variables have significant p-values. Therefore, we will fail to reject the null 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.

In the larger picture, 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.>**#country has a significant impact on the conversion. because last p_value show that.**

In [86]:
```python
from subprocess import call
call(['python', '-m', 'nbconvert', 'Analyze_ab_test_results_notebook.ipynb'])
```

Out[86]: 0

In [ ]: