Analyze_ab_test_results_notebook

February 19, 2023

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 [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)
```

1.0.1 ToDo 1.1

Now, read in the `ab_data.csv` data. Store it in `df`. Below is the description of the data, there are a total of 5 columns:

Data columns	Purpose	Valid values
user_id	Unique ID	Int64 values
timestamp	Time stamp when the user visited the webpage	-
group	In the current A/B experiment, the users are categorized into two broad groups. The `control` group users are expected to be served with `old_page`; and `treatment` group users are matched with the `new_page`. However, **some inaccurate rows** are present in the initial data, such as a `control` group user is matched with a `new_page`.	`['control', 'treatment']`
landing_page	It denotes whether the user visited the old or new webpage.	`['old_page', 'new_page']`
converted	It denotes whether the user decided to pay for the company's product. Here, `1` means yes, the user bought the product.	`[0, 1]`

Use your dataframe to answer the questions in Quiz 1 of the classroom.

Tip: Please save your work regularly.

a. Read in the dataset from the `ab_data.csv` file 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 cell below to find the number of rows in the dataset.

```
In [3]: df.count()
```

```
Out[3]: user_id         294478
        timestamp       294478
        group           294478
        landing_page    294478
        converted       294478
        dtype: int64
```

c. The number of unique users in the dataset.

```
In [4]: print("Unique Users = ", df['user_id'].nunique())
```

```
Unique Users =  290584
```

d. The proportion of users converted.

```
In [5]: print("Proportion of users converted = ",df['converted'].mean().round(2)*100,"%")
```

```
Proportion of users converted =  12.0 %
```

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

```
In [6]: df_treat_not_new=df.query('group == "treatment"').query('landing_page != "new_page"')
        df_not_treat_new=df.query('group != "treatment"').query('landing_page == "new_page"')
        print(df_treat_not_new.user_id.count())
```

```
1965
```

f. Do any of the rows have missing values?

```
In [7]: df.isnull().sum()
```

```
Out[7]: user_id         0
        timestamp       0
        group           0
        landing_page    0
        converted       0
        dtype: int64
```

1.0.2 ToDo 1.2

In a particular row, the **group** and **landing_page** columns should have either of the following acceptable values:

user_id	timestamp	group	landing_page	converted
XXXX	XXXX	`control`	`old_page`	X
XXXX	XXXX	`treatment`	`new_page`	X

It means, the `control` group users should match with `old_page`; and `treatment` group users should matched with the `new_page`.

However, 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.

Use **Quiz 2** in the classroom to figure out how should we handle the rows where the group and landing_page columns don't match?

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]: # Remove the inaccurate rows, and store the result in a new dataframe df2
        df_treat_new=df.query('group == "treatment"').query('landing_page == "new_page"')
        df_control_old=df.query('group == "control"').query('landing_page == "old_page"')
        df2=df_treat_new.append(df_control_old)
        df2.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 290585 entries, 2 to 294476
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 [9]: # 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].sha
```

```
Out[9]: 0
```

1.0.3 ToDo 1.3

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 [10]: print("Unique Users = ", df2['user_id'].nunique())
```

```
Unique Users =  290584
```

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

```
In [11]: duplicate_id=df2[df2['user_id'].duplicated()].user_id
         print(duplicate_id)
```

```
2893    773192
Name: user_id, dtype: int64
```

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

```
In [12]: df2[df2['user_id'].duplicated()]
```

Out[12]:

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

```
In [13]: # Remove one of the rows with a duplicate user_id..
         # Hint: The dataframe.drop_duplicates() may not work in this case because the rows with
         df2 = df2.drop_duplicates('user_id')
         # Check again if the row with a duplicate user_id is deleted or not
         df2.info()
         print("Duplicate Users = ", df2['user_id'].count()-df2['user_id'].nunique())
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 290584 entries, 2 to 294476
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
Duplicate Users =  0
```

1.0.4 ToDo 1.4

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?

Tip: The probability you'll compute represents the overall "converted" success rate in the population and you may call it $p_{population}$.

```
In [14]: p_pop=df2['converted'].mean()
         print("Ppop = ",p_pop)
```

```
Ppop =  0.119597087245
```

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

```
In [15]: control_prop=df2.query('group == "control"').converted.mean()
         print("Control Conversion Prop. = ",control_prop)
```

```
Control Conversion Prop. =  0.1203863045
```

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

```
In [16]: treatment_prop=df2.query('group == "treatment"').converted.mean()
         print("Treatment Conversion Prop. = ",treatment_prop)
```

```
Treatment Conversion Prop. =  0.118808065515
```

> **Tip**: The probabilities you've computed in the points (b). and (c). above can also be treated as conversion rate. Calculate the actual difference (`obs_diff`) between the conversion rates for the two groups. You will need that later.

```
In [17]: # Calculate the actual difference (obs_diff) between the conversion rates for the two g
         obs_diff=treatment_prop-control_prop
         print("Conversion diff. = ",obs_diff)
```

```
Conversion diff. =  -0.00157823898536
```

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

```
In [18]: new_page=df2.query('landing_page == "new_page"')
         print("New page Prop. = ",new_page['landing_page'].count()/df2['landing_page'].count())
```

```
New page Prop. =  0.500061944223
```

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

> ***Observations:*** - From calculating the Ppop its seems that small part of the population converted to the new page. - Although its fair distribution for the test that both groups has equal propabilities for receving either the new page or the old page; and this proves that the test is not biased to new or old page. - The propability of those who converted form the whole population was a little higher for the control group.

Part II - A/B Test

Since a timestamp is associated with each event, you could run a hypothesis test continuously as long as you observe the events.

However, then the hard questions would be: - 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.

1.0.5 ToDo 2.1

For now, consider you need to make the decision just based on all the data provided.

> Recall that you just calculated that the "converted" probability (or rate) for the old page is *slightly* higher than that of the new page (ToDo 1.4.c).

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 be your null and alternative hypotheses (H_0 and H_1)?

You can state your hypothesis in terms of words or in terms of p_{old} and p_{new}, which are the "converted" probability (or rate) for the old and new pages respectively.

- Null Hypothesis is that ***H0: p_old p_new***
- Alternative hypothesis is that ***H1: p_new > p_old***

1.0.6 ToDo 2.2 - Null Hypothesis H_0 Testing

Under the null hypothesis H_0, assume that p_{new} and p_{old} are equal. Furthermore, assume that p_{new} and p_{old} both are equal to the **converted** success rate in the `df2` data regardless of the page. So, our assumption is:

$p_{new} = p_{old} = p_{population}$

In this section, you will:

- Simulate (bootstrap) sample data set for both groups, and compute the "converted" probability p for those samples.
- Use a sample size for each group equal to the ones in the `df2` data.
- Compute the difference in the "converted" probability for the two samples above.
- Perform the sampling distribution for the "difference in the converted probability" between the two simulated-samples over 10,000 iterations; and calculate an estimate.

Use the cells below to provide the necessary parts of this simulation. 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 hypothesis?

```
In [19]: p_new=p_pop
         print("Pnew = ",p_new)
```

```
Pnew =  0.119597087245
```

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

```
In [20]: p_old=p_pop
         print("Pold = ",p_old)
```

```
Pold =  0.119597087245
```

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

```
In [21]: n_new=df2.query('group=="treatment"').user_id.count()
         print("Nnew = ",n_new)
```

```
Nnew =  145310
```

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

```
In [22]: n_old=df2.query('group=="control"').user_id.count()
         print("Nold = ",n_old)
```

```
Nold =  145274
```

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 [23]: # Simulate a Sample for the treatment Group
         new_page_converted=np.random.choice(2, size=n_new ,p=[p_new,1 - p_new])
         p_new1=new_page_converted.mean()
         print("Pnew 01 =", p_new1)
```

```
Pnew 01 = 0.88104741587
```

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 [24]: # Simulate a Sample for the control Group
         old_page_converted=np.random.choice(2, size=n_old ,p=[p_old,1 - p_old])
         p_old1=old_page_converted.mean()
         print("Pold 01 =",p_old1)
```

```
Pold 01 = 0.879276401834
```

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

```
In [25]: P_diff=p_new1-p_old1
         print("Pdiff=",P_diff)
```

```
Pdiff= 0.00177101403574
```

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 [26]: # Sampling distribution
         p_diffs = []
         p_new2=np.random.binomial(n_new, p_new, 10000)/n_new
         p_old2=np.random.binomial(n_old, p_old, 10000)/n_old
         p_diffs.append(p_new2-p_old2)
```

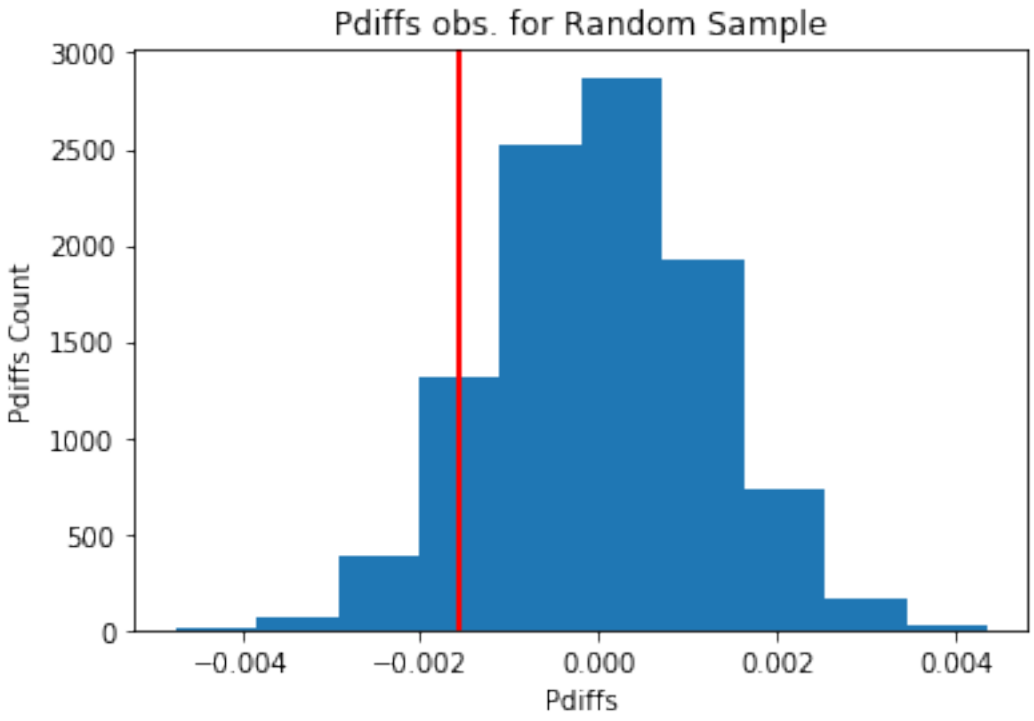
i. Histogram Plot a histogram of the **p_diffs**. Does this plot look like what you expected? Use the matching problem in the classroom to assure you fully understand what was computed here.

Also, use `plt.axvline()` method to mark the actual difference observed in the `df2` data (recall `obs_diff`), in the chart.

Tip: Display title, x-label, and y-label in the chart.

```
In [27]: plt.hist(p_diffs)
         plt.title('Pdiffs obs. for Random Sample')
         plt.xlabel('Pdiffs')
         plt.ylabel('Pdiffs Count')
         plt.axvline(obs_diff,color='r', linewidth=2)
```

```
Out[27]: <matplotlib.lines.Line2D at 0x7f6261a599e8>
```



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

```
In [28]: null_p=(p_diffs>obs_diff).mean()
         print("null P=",null_p.round(3)*100,"%")
```

```
null P= 90.6 %
```

k. Please explain in words what you have just computed in part **j** above.
- What is this value called in scientific studies?
- What does this value signify in terms of whether or not there is a difference between the new and old pages? *Hint*: Compare the value above with the "Type I error rate (0.05)".

> ***Observations:*** - This Value is called the P-Value for Null Hypothesis & Since it's greater than our threshold which is type 1 error rate = 5%, we cannot accept the alternative Hypothesis in other terms we fail to reject the null hypothesis and we can't for sure say that new page has better conversion rate than old page.

l. Using Built-in Methods for Hypothesis Testing 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 walk-through of the ideas that are critical to correctly thinking about statistical significance.

Fill in the statements below to calculate the: - `convert_old`: number of conversions with the old_page - `convert_new`: number of conversions with the new_page - `n_old`: number of individuals who were shown the old_page - `n_new`: number of individuals who were shown the new_page

```
In [29]: import statsmodels.api as sm

         # number of conversions with the old_page
         convert_old = df2.query('landing_page == "old_page"').query('converted == 1').converted

         # number of conversions with the new_page
         convert_new =df2.query('landing_page == "new_page"').query('converted == 1').converted.

         # number of individuals who were shown the old_page
         n_old = df2.query('landing_page == "old_page"').landing_page.count()

         # number of individuals who received new_page
         n_new = df2.query('landing_page == "new_page"').landing_page.count()
         print(convert_old,convert_new,n_old,n_new)
```

```
17489 17264 145274 145310
```

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

m. Now use `sm.stats.proportions_ztest()` to compute your test statistic and p-value. Here is a helpful link on using the built in.

The syntax is:

```
proportions_ztest(count_array, nobs_array, alternative='larger')
```

where, - `count_array` = represents the number of "converted" for each group - `nobs_array` = represents the total number of observations (rows) in each group - `alternative` = choose one of the values from `[two-sided, smaller, larger]` depending upon two-tailed, left-tailed, or right-tailed respectively. >**Hint**: It's a two-tailed if you defined H_1 as $(p_{new} = p_{old})$. It's a left-tailed if you defined H_1 as $(p_{new} < p_{old})$. It's a right-tailed if you defined H_1 as $(p_{new} > p_{old})$.

The built-in function above will return the z_score, p_value.

> **Tip**: You don't have to dive deeper into z-test for this exercise. **Try having an overview of what does z-score signify in general.**

```
In [30]: import statsmodels.api as sm
         # ToDo: Complete the sm.stats.proportions_ztest() method arguments
         count_array=np.array([convert_new,convert_old])
         nobs_array=np.array([n_new,n_old])
         z_score, p_value = sm.stats.proportions_ztest(count_array, nobs_array,alternative='larg
         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.**?

> **Tip**: Notice whether the p-value is similar to the one computed earlier. Accordingly, can you reject/fail to reject the null hypothesis? It is important to correctly interpret the test statistic and p-value.

> **Observations:** we fail to reject the null hypothesis since we chose a right tailed ***Z-score<1.645*** & these findings agrees with findings in *j* & *k*.

Part III - A regression approach

1.0.7 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?

- ***since converted values are represented by 1 & 0 we should use logistic regression***

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 [31]: df2[['treatment', 'control']]= pd.get_dummies(df2['group'])
         df3= df2.drop('control', axis=1)
         df3['intercept']=1
         df3.rename(columns = {'treatment':'ab_page'}, inplace = True)
         df3.head()
```

Out[31]:

	user_id	timestamp	group	landing_page	converted	ab_page	intercept
2	661590	2017-01-11 16:55:06.154213	treatment	new_page	0	0	1
3	853541	2017-01-08 18:28:03.143765	treatment	new_page	0	0	1
6	679687	2017-01-19 03:26:46.940749	treatment	new_page	1	0	1
8	817355	2017-01-04 17:58:08.979471	treatment	new_page	1	0	1
9	839785	2017-01-15 18:11:06.610965	treatment	new_page	1	0	1

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 [32]: logit_mod = sm.Logit(df3['converted'], df3[['intercept','ab_page']])
         results=logit_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 [33]: #there is an error when using result.summary as provide in the lesson showing the error
         #error: module 'scipy.stats' has no attribute 'chisqprob'
         #found a soultion to the error by replacing 'summary' syntax with 'summary2' on stack o
         #https://stackoverflow.com/questions/49814258/statsmodel-attributeerror-module-scipy-st
         results.summary2()
```

```
Out[33]: <class 'statsmodels.iolib.summary2.Summary'>
         """
                                 Results: Logit
         =================================================================
         Model:              Logit            No. Iterations:   6.0000
         Dependent Variable: converted        Pseudo R-squared: 0.000
         Date:               2023-02-19 15:28 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    -2.0038    0.0081  -247.1457  0.0000  -2.0197  -1.9879
ab_page       0.0150    0.0114     1.3109  0.1899  -0.0074   0.0374
======================================================================

"""
```

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).

> **Observations:** - ***P-value*=** 0.189 - in this part our null/alternative hypothesis is studying the difference of individual groups on the conversion rate to the new page which is two sided while in Part II the null & alternative hypotheses are one sided, each comparing wether receiving a new/old page has higher conversion rates

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?

- yes there a lot of factors that can be taken into consideration like age groups, devices they are using, internet speed & location
- but adding this factors will limit the model flexibility & simplicity

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 [34]:
```
# Read the countries.csv
df_countries=pd.read_csv('./countries.csv')
df_countries.head()
```

Out[34]:

	user_id	country
0	834778	UK
1	928468	US
2	822059	UK
3	711597	UK
4	710616	UK

In [35]:
```
# Join with the df2 dataframe
df_merged= df_countries.merge(df3, left_on='user_id', right_on='user_id', how='inner')
df_merged.head()
```

Out[35]:

	user_id	country	timestamp	group	landing_page	converted	ab_page	intercept
0	834778	UK	2017-01-14 23:08:43.304998	control	old_page	0	1	1
1	928468	US	2017-01-23 14:44:16.387854	treatment	new_page	0	0	1
2	822059	UK	2017-01-16 14:04:14.719771	treatment	new_page	1	0	1
3	711597	UK	2017-01-22 03:14:24.763511	control	old_page	0	1	1
4	710616	UK	2017-01-16 13:14:44.000513	treatment	new_page	0	0	1

In [36]:
```
# Create the necessary dummy variables
df_merged[['UK','US','CA']]= pd.get_dummies(df_merged['country'])
df_merged= df_merged.drop('UK', axis=1)
df_merged.head()
logit_mod2 = sm.Logit(df_merged['converted'], df_merged[['intercept','ab_page','US','CA
results2=logit_mod2.fit()
results2.summary2()
```

```
Optimization terminated successfully.
         Current function value: 0.366113
         Iterations 6
```

Out[36]:
```
<class 'statsmodels.iolib.summary2.Summary'>
"""
                           Results: Logit
=================================================================
Model:              Logit            No. Iterations:   6.0000
Dependent Variable: converted        Pseudo R-squared: 0.000
Date:               2023-02-19 15:28 AIC:              212781.1253
No. Observations:   290584           BIC:              212823.4439
Df Model:           3                Log-Likelihood:   -1.0639e+05
Df Residuals:       290580           LL-Null:          -1.0639e+05
Converged:          1.0000           Scale:            1.0000
```

```
----------------------------------------------------------------------
               Coef.    Std.Err.      z      P>|z|     [0.025   0.975]
----------------------------------------------------------------------
intercept     -2.0450     0.0266  -76.8197  0.0000   -2.0971  -1.9928
ab_page        0.0149     0.0114    1.3069  0.1912   -0.0075   0.0374
US             0.0506     0.0284    1.7835  0.0745   -0.0050   0.1063
CA             0.0408     0.0269    1.5161  0.1295   -0.0119   0.0934
======================================================================

"""
```

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.

> **Tip**: Conclusions should include both statistical reasoning, and practical reasoning for the situation.
>
> **Hints**: - Look at all of p-values in the summary, and compare against the Type I error rate (0.05). - Can you reject/fail to reject the null hypotheses (regression model)? - Comment on the effect of page and country to predict the conversion.

In [37]:
```
# Fit your model, and summarize the results
df_merged['ab_US'] = df_merged['ab_page'] * df_merged['US']
df_merged['ab_CA'] = df_merged['ab_page'] * df_merged['CA']
logit_mod3 = sm.Logit(df_merged['converted'], df_merged[['intercept','ab_page','US','CA
results3=logit_mod3.fit()
results3.summary2()
```

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

Out[37]:
```
<class 'statsmodels.iolib.summary2.Summary'>
"""
                          Results: Logit
=================================================================
Model:              Logit            No. Iterations:   6.0000
Dependent Variable: converted        Pseudo R-squared: 0.000
Date:               2023-02-19 15:28 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      -2.0715     0.0371  -55.7977  0.0000  -2.1442  -1.9987
ab_page         0.0674     0.0520    1.2967  0.1947  -0.0345   0.1694
US              0.0901     0.0405    2.2252  0.0261   0.0107   0.1694
CA              0.0644     0.0384    1.6788  0.0932  -0.0108   0.1396
ab_US          -0.0783     0.0568   -1.3783  0.1681  -0.1896   0.0330
ab_CA          -0.0469     0.0538   -0.8718  0.3833  -0.1523   0.0585
========================================================================

"""
```

Observations: - There is no effect on the conversion rate for individuals inwether in US or CA receivnig eitheer new or old page despite the small P-value of US but it is still >0.05 so we fail to reject the null hypothesis - while adding another two parameters like the landing page in US & CA has so signfacnt impact on the previous test since their P-values >0.05 but it has an impact on the relation of the US to Conversion rate where Pvalue<0.05, on which we can say that controlling the treatment groups receive the landing page in US, we will most likely be able to reject the null Hypothesis & more users will convert to new page which is kind of a biased test since we control which group receive the page.

Final Conclusion

- Through out this testing case we wanted to prove that more users will convert to the new page through many tests which we failed to prove through Hypothesis Testing, Z-test & logistic regression model where every Test either single tailed or two sided always failed to reject the null hypothesis.

Submission You may either submit your notebook through the "SUBMIT PROJECT" button at the bottom of this workspace, or you may work from your local machine and submit on the last page of this project lesson.

1. 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).

2. 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.

3. 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 [38]:
```
from subprocess import call
call(['python', '-m', 'nbconvert', 'Analyze_ab_test_results_notebook.ipynb'])
```

Out[38]: 0

In [ ]: