Analyze_ab_test_results_notebook

February 11, 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 [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.shape[0]
```

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

c. The number of unique users in the dataset.

```
In [4]: df.groupby('user_id').nunique().shape[0]
```

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

d. The proportion of users converted.

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

```
Out[5]: 0.11965919355605512
```

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

```
In [6]: df.query("group == 'treatment' and landing_page != 'new_page'").shape[0]
```

```
Out[6]: 1965
```

f. Do any of the rows have missing values?

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

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

        inaccurate_rows = (df[(((df['group']=='treatment') ==(df['landing_page']== 'new_page'))

        df2= df.drop(inaccurate_rows)
        df2.head()
```

```
Out[8]:    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
```

```
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]: df2.groupby('user_id').nunique().shape[0]
```

```
Out[10]: 290584
```

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

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

```
Out[11]:       user_id                   timestamp      group landing_page  converted
         2893   773192  2017-01-14 02:55:59.590927  treatment     new_page          0
```

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

```
In [12]: df2[df2['user_id']== 773192]
```

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

```
In [13]: # Remove one of the rows with a duplicate user_id..
         df2.drop_duplicates(subset= 'user_id', keep= 'first', inplace= True)

         # Check again if the row with a duplicate user_id is deleted or not
         print(df2[df2['user_id']==773192])
```

```
      user_id                   timestamp      group landing_page  converted
1899   773192  2017-01-09 05:37:58.781806  treatment     new_page          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 = df2.converted.mean()
         P
```

```
Out[14]: 0.11959708724499628
```

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

```
In [15]: P_control_converted= (df2.query("group=='control'")['converted']== 1).mean()
         P_control_converted
```

```
Out[15]: 0.1203863045004612
```

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

```
In [16]: P_treatment_converted = (df2.query("group=='treatment'")['converted']== 1).mean()
         P_treatment_converted
```

Out[16]: 0.11880806551510564

```
In [17]: # Calculate the actual difference (obs_diff) between the conversion rates for the two g
         obs_diff = P_treatment_converted - P_control_converted
         obs_diff
```

Out[17]: -0.0015782389853555567

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

```
In [18]: (df2['landing_page']=='new_page').mean()
```

Out[18]: 0.50006194422266881

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

from part(d) ,it appears that probability of receiving new page and old page is almost the same ,50% for each group .

control group has slightly higher conversion rate than treatment group(12.03 % vs 11.88 %) so the new page has no significant effect on convertion rate and do not lead to more conversions ## 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.

(0 : Pnew - Pold <= 0)

(H1 : pnew - Pold > 0)

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]: Pnew = df2.converted.mean()
         Pnew

Out[19]: 0.11959708724499628
```

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

```
In [20]: Pold = df2.converted.mean()
         Pold

Out[20]: 0.11959708724499628
```

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.shape[0]
         n_new

Out[21]: 145310
```

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

```
In [22]: n_old = df2.query("group== 'control'").user_id.shape[0]
         n_old

Out[22]: 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
         # pnew = df2.converted.mean()
         new_page_converted = np.random.choice([0,1], size = n_new ,p = (1-Pnew, Pnew), replace=
         new_page_converted

Out[23]: array([0, 0, 0, ..., 0, 0, 0])
```

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([0,1], size = n_old, p = (1-Pold, Pold), replace=
         old_page_converted
```

```
Out[24]: array([1, 0, 0, ..., 0, 0, 0])
```

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

```
In [26]:  = new_page_converted.mean()
          = old_page_converted.mean()
         simulated_diff = new_page_converted.mean() - old_page_converted.mean()
         simulated_diff
```

```
Out[26]: -0.00038060461700242798
```

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 [27]: # Sampling distribution
         p_diffs = []
         for _ in range(10000):
             New_page_converted = np.random.choice([0,1], size = n_new ,p = (1-Pnew, Pnew), repl
             Old_page_converted = np.random.choice([0,1], size = n_old, p = (1-Pold, Pold), repl
             p_diffs.append(New_page_converted.mean() - Old_page_converted.mean())
```

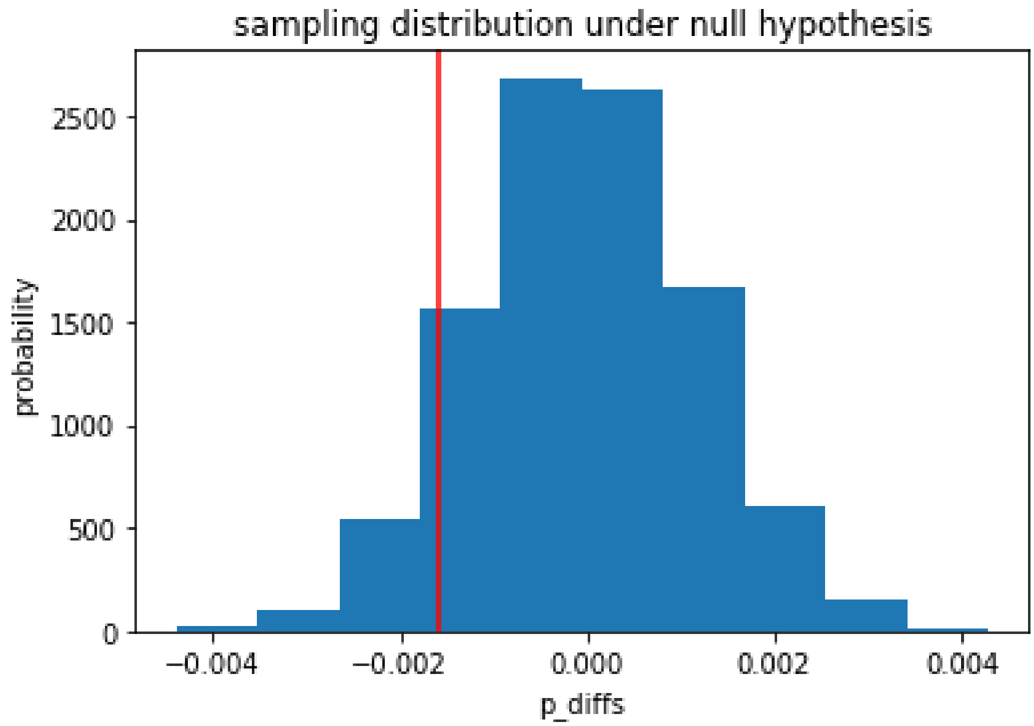
i. Histogram Plot a histogram of the **p_diffs**. Does this plot look like what you expected? Use the matching problem in the classroom to assure you fully understand what was computed here.

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 [28]: p_diffs =np.array(p_diffs)
         plt.hist(p_diffs)
         plt.axvline(x= obs_diff, color= 'r', label = 'actual difference')
         plt.xlabel('p_diffs')
         plt.ylabel('probability')
         plt.title('sampling distribution under null hypothesis')
```

```
Out[28]: Text(0.5,1,'sampling distribution under null hypothesis')
```



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

```
In [29]: (p_diffs > obs_diff).mean()
```

```
Out[29]: 0.9061000000000002
```

1.1 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)".

this value is called p-value (the probabilty of obtaining our observed statistic if the null hypothesis is true) we use it to determine the stastical significance of our observed diff as our p-value is large compered to type 1 error rate(0.05), we have no eivdance that new page can convert more users and we fail to reject null hypothesis

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 walkthrough 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 [30]: import statsmodels.api as sm

         # number of conversions with the old_page
         convert_old = df2.query("group == 'control' and converted == 1").shape[0]
         # number of conversions with the new_page
         convert_new = df2.query("group=='treatment' and converted == 1 ").shape[0]

         # number of individuals who were shown the old_page
         n_old = df2.query("group== 'control'").user_id.shape[0]

         # number of individuals who received new_page
         n_new = df2.query("group== 'treatment'").user_id.shape[0]

         ( 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[30]: (17489, 17264, 145274, 145310)
```

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

```
Out[31]: (-1.3109241984234394, 0.905058312759024 49)
```

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.

z-score is smaller than the critcal z score (1.645)for right tailed test , p-value calculated fron two-sample ztest is also much larger than type 1 error rate (.05) so we fail to reject null hypothesis and there no evidance that changing to the new page can lead to more conversions

Part III - A regression approach

1.1.1 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?

** 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 [32]: df2['intercept'] = 1
         df2[['ab_page', 'old_page']] = pd.get_dummies(df2['landing_page'])
         df2.drop(['old_page'], axis = 1, inplace = True)
         df2.head()
```

```
Out[32]:    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). above, then fit the model to predict whether or not an individual converts.

```
In [33]: import statsmodels.api as sm
         log_mod = sm.Logit(df2['converted'], df2[['intercept','ab_page']])
         results= log_mod.fit()
         results.summary2()
```

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

```
Out[33]: <class 'statsmodels.iolib.summary2.Summary'>
"""
                          Results: Logit
==================================================================
Model:              Logit            No. Iterations:   6.0000
Dependent Variable: converted        Pseudo R-squared: 0.000
Date:               2022-02-11 21:22 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
==================================================================

"""
```

```
In [34]: np.exp(-0.0150)
```

```
Out[34]: 0.98511193960306265
```

```
In [35]: 1/np.exp(-.0150)
```

```
Out[35]: 1.0151130646157189
```

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

```
In [ ]: conversion rate is 1.015 times as likely on using old page than new page so old and new
        approximately equal in thier probabilites to convert users
        so it appears that coefficient(change in ab_page) is not significant related to our
        response(convertion rate)
        we fail to reject null hypothsis and we fail to prove that changing to new page
        can lead to more conversions
```

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

null hypothesis associated with this logistic regression model is that there is no statistically signifcant relationship between the predicator (change in ab-page) and our response (converted) so null is changing from old page to new page has no significant effect in our response conversion rate null : coefficient = 0 and alternative : coefficient != 0, this is two-sided hypothesis

part II(A/B TEST) our null hypothesis is new page has conversion rate equal or smaller than old page and alternative is new page has more conversions than old page null: Pnew <= Pold, alternative : Pnew > Pold , this is right-sided hypothesis

so p_value calculated in two-sided regression model will be different from one calculated in one-sided A/B test in our regression model p_value is larger than type 1 error (.05) so it give us the same result that coefficient is not significant related to conversion rate and we fail to reject null hyporhesis.

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?

we may consider other things as they can influence significance of our test and they may interact with ab_page and have significant effect in our response,
disadvantage is that they increase complexity of our model and lead to multicolinearity when x-variables are correlatted with one anothor ,this may lead to negative effects on our model like flipping relationship to our response

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 [36]:
```
# Read the countries.csv
countries_df = pd.read_csv('countries.csv')
countries_df.head()
```

Out[36]:

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

In [37]:
```
# Join with the df2 dataframe
df_merged = df2.set_index('user_id').join(countries_df.set_index('user_id'))
df_merged.head()
```

Out[37]:

```
                          timestamp      group landing_page  converted  \
user_id
851104   2017-01-21 22:11:48.556739    control     old_page          0
804228   2017-01-12 08:01:45.159739    control     old_page          0
661590   2017-01-11 16:55:06.154213  treatment     new_page          0
853541   2017-01-08 18:28:03.143765  treatment     new_page          0
864975   2017-01-21 01:52:26.210827    control     old_page          1

         intercept  ab_page country
user_id
```

```
851104          1         0      US
804228          1         0      US
661590          1         1      US
853541          1         1      US
864975          1         0      US
```

```
In [12]: # Create the necessary dummy variables
         df_merged[['CA', 'UK','US']] = pd.get_dummies(df_merged['country'])
         log_mod = sm.Logit(df_merged['converted'], df_merged[['intercept','ab_page', 'US', 'UK'
         results= log_mod.fit()
         results.summary2()
```

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

```
Out[12]: <class 'statsmodels.iolib.summary2.Summary'>
         """
                                  Results: Logit
         ==================================================================
         Model:              Logit            No. Iterations:   6.0000
         Dependent Variable: converted        Pseudo R-squared: 0.000
         Date:               2022-02-10 12:19 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     -2.0300    0.0266  -76.2488  0.0000  -2.0822  -1.9778
         ab_page       -0.0149    0.0114   -1.3069  0.1912  -0.0374   0.0075
         US             0.0408    0.0269    1.5161  0.1295  -0.0119   0.0934
         UK             0.0506    0.0284    1.7835  0.0745  -0.0050   0.1063
         ==================================================================

         """
```

```
In [38]: np.exp(.0408) , np.exp(.0506)
```

```
Out[38]: (1.0416437559600236, 1.0519020483004984)
```

again comparing to CA as basline , each country has approximately the same convertion rate ,also p-values are > type 1 error rate (0.05) so changing country has no significant effect on convertion artes and again we reject null hypothesis

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 [39]:
```
# Fit your model, and summarize the results
df_merged[['CA', 'UK','US']] = pd.get_dummies(df_merged['country'])
df_merged['ab_page_US']= df_merged['US'] * df_merged['ab_page']
df_merged['ab_page_UK']= df_merged['UK'] * df_merged['ab_page']
df_merged['ab_page_CA']= df_merged['CA'] * df_merged['ab_page']
#df_merged.head(10)
log_mod = sm.Logit(df_merged['converted'], df_merged[['intercept','ab_page', 'US', 'UK'
results= log_mod.fit()
results.summary2()
```

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

Out[39]:
```
<class 'statsmodels.iolib.summary2.Summary'>
"""
                          Results: Logit
=================================================================
Model:              Logit            No. Iterations:   6.0000
Dependent Variable: converted        Pseudo R-squared: 0.000
Date:               2022-02-11 21:27 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_page_US     0.0469   0.0538    0.8718 0.3833 -0.0585   0.1523
ab_page_UK     0.0783   0.0568    1.3783 0.1681 -0.0330   0.1896
=================================================================

"""
```

```
In [40]: np.exp(results.params)

Out[40]: intercept     0.134794
         ab_page       0.934776
         US            1.017682
         UK            1.011854
         ab_page_US    1.048001
         ab_page_UK    1.081428
         dtype: float64
```

we choose ab_page_CA as basline ,converation rate is 1.04 times as likely in US than in canda holding other variables constant , also conversion rate is 1.08 times as likely in UK than canda holding other variables constant,

so it appear that each country has the same chance in converting users, no great diference between conversion rates from interaction

and this interactions between ab_page and countries has no significant effect on convertion rates

also p-values are higher than type 1 error rate (.05) so we fail to reject null hypothesis

Conculsion

statistically based on our tests probability test , A/B test, regression model , there is no statistical significant difference between old and new page on converting users, so we can not reject null hypothesis

practically there are many factos or difficulties to consider like

1- change aversion effect when existing users first experience a change .

2-simpson's paradox due to imbalance in populations in each group (existing customers vs new ones, using phone /tablet vs pc, age groups, gender), we may segemnt our A/B test to evalute these meterics and use bonferroni correction to fix the prolblem of icreasing false positive results testing multiple meterics at the same time

also we may run A/B test for Long enough time to account for changes in behavior based on time of day/week and to protect against change aversion for existing users to acclimate to the change

Final Check!

Congratulations! You have reached the end of the A/B Test Results project! You should be very proud of all you have accomplished!

> **Tip**: Once you are satisfied with your work here, check over your notebook to make sure that it satisfies all the specifications mentioned in the rubric. You should also probably remove all of the "Hints" and "Tips" like this one so that the presentation is as polished as possible.

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 [41]:
```
from subprocess import call
call(['python', '-m', 'nbconvert', 'Analyze_ab_test_results_notebook.ipynb'])
```

Out[41]: 0

In [ ]: