Analyze_ab_test_results_notebook

November 16, 2021

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 **??**

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.

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 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
```

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

c. The number of unique users in the dataset.

```
In [4]: n_uniq = df.user_id.nunique()
        n_uniq
```

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

d. The proportion of users converted.

```
In [5]: df.converted.mean()*100
```

```
Out[5]: 11.965919355605511
```

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

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

        treat + control
```

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

f. Do any of the rows have missing values?

```
In [7]: df.shape[0] - df.dropna().shape[0]
```

```
Out[7]: 0
```

1.0.2 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
```

In [9]:
```
df2 = df.query("group == 'control' and landing_page == 'old_page'")
df2 = df2.append(df.query("group == 'treatment' and landing_page == 'new_page'"))
```

In [10]:
```
# 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[10]:
```
0
```

1.0.3 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 [11]:
```
df2.user_id.nunique()
```

Out[11]:
```
290584
```

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

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

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

In [13]:
```
df2[df2['user_id'] == 773192]
```

Out[13]:

	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 [14]:
```
# Remove one of the rows with a duplicate user_id..

# Check again if the row with a duplicate user_id is deleted or not
df2 = df2.drop(2893)
```

1.0.4 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?

```
In [15]: df2.converted.mean()
```

```
Out[15]: 0.11959708724499628
```

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

```
In [16]: prob_c = df2.query("group == 'control'")['converted'].mean()
         prob_c
```

```
Out[16]: 0.1203863045004612
```

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

```
In [17]: prob_t = df2.query("group == 'treatment'")['converted'].mean()
         prob_t
```

```
Out[17]: 0.11880806551510564
```

```
In [18]: # Calculate the actual difference (obs_diff) between the conversion rates for the two g
         obs_diff = prob_t - prob_c
         obs_diff
```

```
Out[18]: -0.0015782389853555567
```

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

```
In [19]: prob_new_page =df2.query('landing_page == "new_page"').shape[0]/df2.shape[0]
         prob_new_page
```

```
Out[19]: 0.5000619442226688
```

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

> **It doesn't seems that one page leads to more conversions than other. The new page led to a lower conversion rate, than the old page with a negligible difference .**

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 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 H_0: $p_{new} = p_{old}$

Alternative-hypotheses H_1: $p_{new} > p_{old}$

1.0.6 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 [20]: p_new = df2['converted'].mean()
         p_new
```

```
Out[20]: 0.11959708724499628
```

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

```
In [21]: p_old = len(df2.query('converted==1'))/len(df2.index)
         p_old
```

```
Out[21]: 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 [22]: n_new = df2.query("landing_page == 'new_page'").shape[0]
         n_new
```

```
Out[22]: 145310
```

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

```
In [23]: n_old = df2.query("landing_page == 'old_page'").shape[0]
         n_old
```

```
Out[23]: 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 [24]: # Simulate a Sample for the treatment Group
         # np.random.choice don't work by me so i get binomial function

         new_page_converted = np.random.choice([1,0], size=n_new, p=[p_new,1 - p_new])
         new_page_converted.sum()
```

```
Out[24]: 17293
```

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 [25]: # Simulate a Sample for the control Group

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

```
Out[25]: 17273
```

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]: #p_new_conv = (new_page_converted/n_new)
         #p_old_conv = (old_page_converted/n_old)
         p_diff = old_page_converted.mean()-new_page_converted.mean()
         p_diff
```

```
Out[26]: -0.00010817988767240772
```

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`.

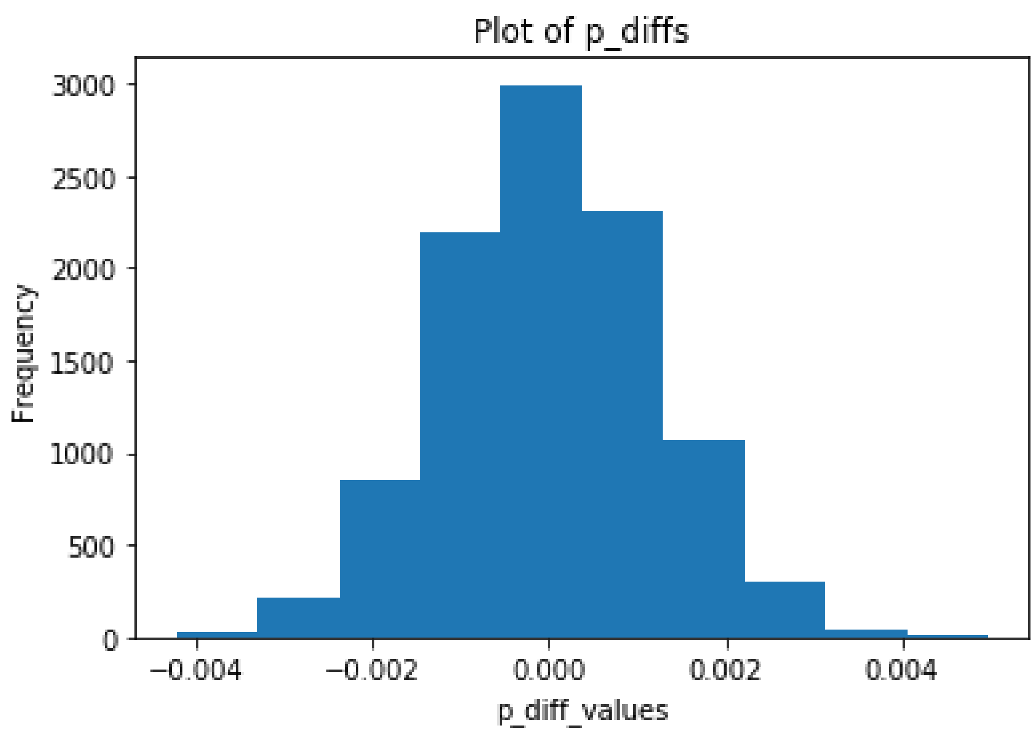
```
In [27]: # Sampling distribution

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

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.

```
In [28]: plt.hist(p_diffs)
         plt.xlabel('p_diff_values')
         plt.ylabel('Frequency')
         plt.title('Plot of p_diffs');
```



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

```
In [29]: obs_diff = prob_t - prob_c

         control_mean = df2.query("group == 'control'").converted.mean()
         treatment_mean = df2.query("group == 'treatment'").converted.mean()
```

```
In [30]: act_diffs = treatment_mean - control_mean
         act_diffs
```

```
Out[30]: -0.0015782389853555567
```

```
In [31]: #p-value
         (p_diffs > act_diffs).mean()
```

```
Out[31]: 0.90749999999999997
```

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)".

> **Bellow we examined the basic AB test step:** - finding sample distribution - distribution under the null

If p-value were lower than 5% indicate a very low and probability, assuming the null hypothesis were true. But our value signify that our p value around 90% that means P_new doesn't equal to P_{old} and that the new page does not significantly better than the old page. Therefore, we failed to reject null hypothesis. **Summarising, the old page is better than new page for the company.**

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 [32]: import statsmodels.api as sm

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

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

         # number of individuals who were shown the old_page
         n_old = df2[df2['landing_page'] == 'old_page'].count()[0]

         # number of individuals who received new_page
         n_new = df2[df2['landing_page'] == 'new_page'].count()[0]

         print("convert_old:", convert_old,
               "\nconvert_new:", convert_new,
               "\nn_old:", n_old,
               "\nn_new:", n_new)
```

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

```
convert_old: 17489
convert_new: 17264
n_old: 145274
n_new: 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.

1.0.7 About the two-sample z-test

Recall that you have plotted a distribution `p_diffs` representing the difference in the "converted" probability $(p'_{new} - p'_{old})$ for your two simulated samples 10,000 times.

Another way for comparing the mean of two independent and normal distribution is a **two-sample z-test**. You can perform the Z-test to calculate the Z_score, as shown in the equation below:

$$Z_{score} = \frac{(p'_{new} - p'_{old}) - (p_{new} - p_{old})}{\sqrt{\frac{\sigma^2_{new}}{n_{new}} + \frac{\sigma^2_{old}}{n_{old}}}}$$

where, - p' is the "converted" success rate in the sample - p_{new} and p_{old} are the "converted" success rate for the two groups in the population. - σ_{new} and σ_{new} are the standard deviation for the two groups in the population. - n_{new} and n_{old} represent the size of the two groups or samples (it's same in our case)

> Z-test is performed when the sample size is large, and the population variance is known. The z-score represents the distance between the two "converted" success rates in terms of the standard error.

Next step is to make a decision to reject or fail to reject the null hypothesis based on comparing these two values: - Z_{score} - Z_{α} or $Z_{0.05}$, also known as critical value at 95% confidence interval. $Z_{0.05}$ is 1.645 for one-tailed tests, and 1.960 for two-tailed test. You can determine the Z_{α} from the z-table manually.

Decide if your hypothesis is either a two-tailed, left-tailed, or right-tailed test. Accordingly, reject OR fail to reject the null based on the comparison between Z_{score} and Z_{α}. We determine whether or not the Z_{score} lies in the "rejection region" in the distribution. In other words, a "rejection region" is an interval where the null hypothesis is rejected iff the Z_{score} lies in that region.

Hint: For a right-tailed test, reject null if $Z_{score} > Z_{\alpha}$. For a left-tailed test, reject null if $Z_{score} < Z_{\alpha}$.

Reference: - Example 9.1.2 on this page, courtesy www.stats.libretexts.org

```
In [33]: import statsmodels.api as sm
         # ToDo: Complete the sm.stats.proportions_ztest() method arguments

         z_score, p_value = sm.stats.proportions_ztest(count=[convert_new, convert_old], nobs=[n

         print(z_score, p_value)

-1.31092419842 0.189883374482
```

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.**?

> *Z-score is the number of standard deviations from the mean a data point. The given definition, it would seem that the differences between the lines shown in the histogram above is -1.31 standard deviations. The p-value is roughly 19.0% which is the probability that this result is due to random chance.
>
> As we see above our model p value and z value are greater than alpha level () =.05. It means that the null hypothesis rejected again and conversion rate of old page is better than new page.*

Part III - A regression approach
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?

> **The logistic regression is a regression approach used to predict only two possible outcomes. And we are predicting if the new page conversion is better or not. Briefly, I want to predict one of two possible outcomes.**

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`.
Or first need to create in df2 a column for the intercept, and create a dummy variable column for which page each user received. Next add an **intercept** and **ab_page** column, which is 1 when an individual receives the **treatment** and 0 if **control**.

```
In [34]: df2[['no_ab_page', 'ab_page']] = pd.get_dummies(df2['group'])
         df2.drop(['no_ab_page'], axis=1, inplace=True)
         df2['intercept'] = 1
         df2.head()
```

```
Out[34]:    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

            ab_page  intercept
         0        0          1
         1        0          1
         4        0          1
         5        0          1
         7        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 [35]: logit_mod = sm.Logit(df2['converted'], df2[['intercept', 'ab_page']])
         result = logit_mod.fit()
         result.summary2()
```

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

```
Out[35]: <class 'statsmodels.iolib.summary2.Summary'>
         """
                                  Results: Logit
         ==================================================================
         Model:              Logit            No. Iterations:   6.0000
         Dependent Variable: converted        Pseudo R-squared: 0.000
         Date:               2021-11-16 09:40 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
         ==================================================================

         """
```

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

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

The p-value for me associated with ab_page = 0.1899, which is slightly lower than the p-value, which used the z-test. The reason why the value is lower - addeded an intercept which is meant to account for bias. Hence follows that this value is more accurate becouse closer to the true p-value. However, this p-value is still too high to reject the null hypothesis.

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?

Adding an appropriate variable can prevent bias in the estimate of another regression coefficient, but it can also increase the variance of another regression coefficient. Adding an irrelevant variable may increase the variance of the estimate for another correlation coefficient and will not provide any benefit. There are certainly disadvantages to adding too many features. When we do regression analysis we want to have functions that influence the result, small influences usually do not matter and should be left to intercept.

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_new = countries_df.set_index('user_id').join(df2.set_index('user_id'), how='inner')
         df_new[['CA', 'UK', 'US']] = pd.get_dummies(df_new['country'])
         df_new.head()
```

```
Out[37]:         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  ab_page  intercept  CA  UK  US
user_id
834778           0        0          1   0   1   0
928468           0        1          1   0   0   1
822059           1        1          1   0   1   0
711597           0        0          1   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 [38]: df_new['US_ab_page'] = df_new['US'] * df_new['ab_page']
         df_new['UK_ab_page'] = df_new['UK'] * df_new['ab_page']
         df_new['CA_ab_page'] = df_new['CA'] * df_new['ab_page']
         df_new.head()
```

```
Out[38]:         country                   timestamp      group landing_page  \
user_id
834778       UK  2017-01-14 23:08:43.304998    control     old_page
928468       US  2017-01-23 14:44:16.387854  treatment     new_page
822059       UK  2017-01-16 14:04:14.719771  treatment     new_page
711597       UK  2017-01-22 03:14:24.763511    control     old_page
710616       UK  2017-01-16 13:14:44.000513  treatment     new_page

         converted  ab_page  intercept  CA  UK  US  US_ab_page  UK_ab_page  \
user_id
834778           0        0          1   0   1   0           0           0
928468           0        1          1   0   0   1           1           0
822059           1        1          1   0   1   0           0           1
711597           0        0          1   0   1   0           0           0
710616           0        1          1   0   1   0           0           1

         CA_ab_page
user_id
834778            0
928468            0
```

```
822059            0
711597            0
710616            0
```

```
In [42]: # drop US (now baseline)
         import statsmodels.api as sm
         from scipy import stats
         stats.chisqprob = lambda chisq, df: stats.chi2.sf(chisq,df)

         df_new['intercept'] = 1
         lm = sm.Logit(df_new['converted'], df_new[['intercept', 'UK', 'CA', 'ab_page', 'CA_ab_p
         results2 = lm.fit()
         results2.summary()
```

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

```
Out[42]: <class 'statsmodels.iolib.summary.Summary'>
         """
                                    Logit Regression Results
         ==============================================================================
         Dep. Variable:              converted   No. Observations:               290584
         Model:                          Logit   Df Residuals:                   290578
         Method:                           MLE   Df Model:                            5
         Date:                Tue, 16 Nov 2021   Pseudo R-squ.:               3.482e-05
         Time:                        09:59:41   Log-Likelihood:            -1.0639e+05
         converged:                       True   LL-Null:                   -1.0639e+05
                                                 LLR p-value:                    0.1920
         ==============================================================================
                          coef    std err          z      P>|z|      [0.025      0.975]
         ------------------------------------------------------------------------------
         intercept     -1.9865      0.010   -206.344      0.000      -2.005      -1.968
         UK            -0.0057      0.019     -0.306      0.760      -0.043       0.031
         CA            -0.0175      0.038     -0.465      0.642      -0.091       0.056
         ab_page       -0.0206      0.014     -1.505      0.132      -0.047       0.006
         CA_ab_page    -0.0469      0.054     -0.872      0.383      -0.152       0.059
         UK_ab_page     0.0314      0.027      1.181      0.238      -0.021       0.084
         ==============================================================================
         """
```

Conclusion:

1. Adding the ab_page variable and the interaction variables UK_ab_page and CA_ab_page degrades the p-value of the UK and CA variables. Also, the p-values of ab_page and the interaction variables are greater than 0.05, so they are not matter. Thus, the null hypothesis is not refused.

2. As we can see from the results obtained in our A / B testing analysis as well as in our regression model, we can conclude that we cannot reject the null hypothesis. Therefore, the company should maintain the old page.

3. It may also happen that the result is that we cannot reject the null hypothesis, but we need more data to actually confirm. These data are of a short-term nature and such tests need to be performed longer than 5 days to obtain a reliable result.

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!

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 [40]:
```
from subprocess import call
call(['python', '-m', 'nbconvert', 'Analyze_ab_test_results_notebook.ipynb'])
```

Out[40]: 0

In [ ]: