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:

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 [5]:
```
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)
```

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]`

</center> Use your dataframe to answer the questions in Quiz 1 of the classroom.

a. Read in the dataset from the `ab_data.csv` file and take a look at the top few rows here:

In [6]:
```
df=pd.read_csv('ab_data.csv')
```

b. Use the cell below to find the number of rows in the dataset.

In [7]:
```
print(df.shape)
```

```
(294478, 5)
```

. The number of unique users in the dataset.

In [8]:
```
df_user=df.user_id.unique()
print(df_user.shape)
```

```
(290584,)
```

d. The proportion of users converted.

In [9]:
```
df.query('converted=="1"').user_id.nunique()/df.query('converted').count()
```

Out[9]:
```
user_id         0.0
timestamp       0.0
group           0.0
landing_page    0.0
converted       0.0
dtype: float64
```

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

In [10]:
```
df2=df.dropna()
df2[((df2['group'] == 'treatment') == (df2['landing_page'] == 'new_page')) == False].shape[0]
```

Out[10]:
```
3893
```

f. Do any of the rows have missing values?

In [11]:
```
df.isnull().sum()
```

Out[11]:
```
user_id         0
timestamp       0
group           0
landing_page    0
converted       0
dtype: int64
```

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 [12]:
```
#remove the mismatch rows
df1 = df.drop(df[(df.group =="treatment") & (df.landing_page == "old_page")].index)

df2 = df1.drop(df1[(df.group =="control") & (df1.landing_page == "new_page")].index)
```

```
C:\Users\micheal\AppData\Local\Temp/ipykernel_2684/1562854388.py:4: UserWarning: Boolean Series key will be reindexed to match DataFrame index.
  df2 = df1.drop(df1[(df.group =="control") & (df1.landing_page == "new_page")].index)
```

In [13]:
```
# 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].shape[0]
```

Out[13]:
```
0
```

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 [14]:
```
df2_user=df2.user_id.unique()
print(df2_user.shape)
```

```
(290584,)
```

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

In [15]:
```
df2[df2.duplicated(['user_id'])]
```

Out[15]:

	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 [16]:
```
df2[df2.duplicated(['user_id'],keep=False)]
```

Out[16]:

	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 [17]:
```
# 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_duplicates(subset = ['user_id'],keep = 'last')
df2
```

Out[17]:

	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
...	...	...	...	...	...
294473	751197	2017-01-03 22:28:38.630509	control	old_page	0
294474	945152	2017-01-12 00:51:57.078372	control	old_page	0
294475	734608	2017-01-22 11:45:03.439544	control	old_page	0
294476	697314	2017-01-15 01:20:28.957438	control	old_page	0
294477	715931	2017-01-16 12:40:24.467417	treatment	new_page	0

290584 rows × 5 columns

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 $p_{population}$.

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

In [18]:
```
df2['converted'].mean ()
```

Out[18]:
```
0.11959708724499628
```

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

In [19]:
```
df2.groupby(['group']).mean()
```

Out[19]:

group	user_id	converted
control	788164.072594	0.120386
treatment	787845.719290	0.118808

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

In [20]:
```
df2.groupby(['group']).mean()
```

Out[20]:

group	user_id	converted
control	788164.072594	0.120386
treatment	787845.719290	0.118808

Calculate the actual difference (`obs_diff`) between the conversion rates for the two groups. You will need that later.

In [21]:
```
# Calculate the actual difference (obs_diff) between the conversion rates for the two groups.
obs_diff=0.120386-0.118808
print(obs_diff)
```

```
0.0015780000000000001
```

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

In [22]:
```
len(df2.query("landing_page == 'new_page'"))

new_page_probability=145311/290585
print(new_page_probability)
```

```
0.5000636646764286
```

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

between a and d there is alittle diffrence as the probability of an individual converting regardless of the page they receive=0.119 and the probability that an individual received the new page =0.5

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.

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.
Notice that because of the time stamp associated with each event, you could technically run a hypothesis test continuously as each observation was observed.

However, then the hard question is do you stop as soon as one page is considered significantly better than another or does it need to happen consistently for a certain amount of time? How long do you run to render a decision that neither page is better than another?

These questions are the difficult parts associated with A/B tests in general.

1. For now, consider you need to make the decision just based on all the data provided. If you want to assume that the old page is better unless the new page proves to be definitely better at a Type I error rate of 5%, what should your null and alternative hypotheses be?

   and are the converted rates for the old and new pages respectively.

2. Assume under the null hypothesis, and both have "true" success rates equal to the converted success rate regardless of page - that is and are equal. Furthermore, assume they are equal to the converted rate in ab_data.csv regardless of the page.

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 [23]:
```
P_new = df2.converted.mean()
print(P_new)
```

```
0.11959708724499628
```

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

In [24]:
```
p_old = df2['converted'].mean()
print(p_old)
```

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

```
145310
```

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

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

```
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 [27]:
```
# Simulate a Sample for the treatment Group
new_page_converted = np.random.binomial(1,P_new,n_new)
new_page_converted.mean()
```

Out[27]:
```
0.11843644621842957
```

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 [28]:
```
# Simulate a Sample for the control Group
old_page_converted = np.random.binomial(1,p_old,n_old)

old_page_converted.mean()
```

Out[28]:
```
0.11900959566061374
```

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

In [29]:
```
new_page_converted.mean()-old_page_converted.mean()
```

Out[29]:
```
-0.0005731494421841732
```

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 [30]:
```
# Sampling distribution
p_diffs = []
for _ in range(10000):
    new_page_converted = np.random.binomial(1,P_new,n_new)  # bootstrapping
    old_page_converted = np.random.binomial(1,p_old,n_old) # bootstrapping
    p_diffs.append(new_page_converted.mean() - old_page_converted.mean())
```

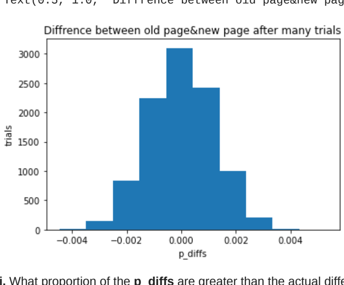
i. Histogram
Plot a histogram of the **p_diffs**. Does this plot look like what you expected? Use the matching problem in the classroom to assure you fully understand what was computed here.

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

In [62]:
```
p_diffs = np.array(p_diffs)

# plot sampling distribution
plt.hist(p_diffs)
plt.xlabel('p_diffs')
plt.ylabel('trials')
plt.title('Diffrence between old page&new page after many trials')
```

Out[62]:
```
Text(0.5, 1.0, 'Diffrence between old page&new page after many trials')
```



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

In [32]:
```
actual_diff = df2.converted[df2.group == 'treatment'].mean() - df2.converted[df2.group == 'control'].mean()
(actual_diff < p_diffs).mean()
```

Out[32]:
```
0.9065
```

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)".

we observed difference in means of converted old page and converted new page. we had random choices of these mean converted values for the observed difference, . The actual difference was calculated from the dataset ab_data.csv. This value that we calculated, difference in means, is the p-value. Our p-value is more than critical value of 0.05 so we fail to reject the null hypothesis, new page couldn't attract user more 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 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 [33]:
```
import statsmodels.api as sm

# number of conversions with the old_page
convert_old = df2.query('group == "control" & converted == 1')['converted'].count()

# number of conversions with the new_page
convert_new =df2.query('group == "treatment" & converted == 1')['converted'].count()

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

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

convert_old,convert_new,n_old,n_new
```

Out[33]:
```
(17489, 17264, 145274, 145310)
```

m. Now use `sm.stats.proportions_ztes t()` 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.

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_{α}.
> For a left-tailed test, reject null if Z_{score} < Z_{α}.

Reference:

- Example 9.1.2 on this page/09%3A_Two-Sample_Problems/9.01%3A_Comparison_of_Two_Population_Means-_Large_Independent_Samples), courtesy www.stats.libretexts.org

---

In [34]:
```
import statsmodels.api as sm
# ToDo: Complete the sm.stats.proportions_ztest() method arguments
z_score, p_value = sm.stats.proportions_ztest([convert_old,convert_new],[n_old,n_new],alternative='smaller')
print(z_score, p_value)
```

```
1.3109241984234394 0.9050583127590245
```

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.**?

> **Put your answer here.** Z-score indicates how much a given value differs from the standard deviation. The Z-score=1.31 which is less than critical value at 95% and p_value=0.905 which indicate that there is no improvement in convert for new landing page

Part III - A regression approach

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?

> **Put your answer here.**This is 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 [35]:
```
df2['intercept'] = 1
df2[['control', 'ab_page']] = pd.get_dummies(df2['group'])
df2.drop(['control'], axis=1, inplace=True)
df2.head()
```

Out[35]:

	user_id	timestamp	group	landing_page	converted	intercept	ab_page
0	851104	2017-01-21 22:11:48.556739	control	old_page	0	1	0
1	804228	2017-01-12 08:01:45.159739	control	old_page	0	1	0
2	661590	2017-01-11 16:55:06.154213	treatment	new_page	0	1	1
3	853541	2017-01-08 18:28:03.143765	treatment	new_page	0	1	1
4	864975	2017-01-21 01:52:26.210827	control	old_page	1	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 [58]:
```
import statsmodels.api as sm

logit_mod = sm.Logit(df2['converted'], df2[['intercept', 'ab_page']])
```

In [56]:
```
np.exp(0.0150)
```

Out[56]:
```
1.015113064615719
```

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

In [63]:
```
results = logit_mod.fit()
results.summary()
```

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

Out[63]:

Logit Regression Results

Dep. Variable:	converted	**No. Observations:**	290584
Model:	Logit	**Df Residuals:**	290582
Method:	MLE	**Df Model:**	1
Date:	Thu, 02 Jun 2022	**Pseudo R-squ.:**	8.077e-06
Time:	07:16:30	**Log-Likelihood:**	-1.0639e+05
converged:	True	**LL-Null:**	-1.0639e+05
Covariance Type:	nonrobust	**LLR p-value:**	0.1899

	coef	std err	z	P>|z|	[0.025	0.975]
intercept	-1.9888	0.008	-246.669	0.000	-2.005	-1.973
ab_page	-0.0150	0.011	-1.311	0.190	-0.037	0.007

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

- 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).

> **Put your answer here.** p_value=-1.3109

The difference is, in part II, we performed a one-sided test, in part III where in the logistic regression part its two sided test

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?

> **Put your answer here.** There can be many other factors that can be taken into consideration to add into our regression model.

time of day that the user used the page, might influence when people sign up online Type of user use the page accoeding to it they are old or young ,young user prefer animations and picture and simple information and shape to be more attractive ,old user interested in details

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.

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

In [38]:
```
df_new = pd.read_csv('countries.csv')
df_new.head()
```

Out[38]:

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

In [39]:
```
# Join with the df2 dataframe
df_new = df_new.set_index('user_id').join(df2.set_index('user_id'), how='inner')
```

In [40]:
```
# Create the necessary dummy variables
df_new[['CA', 'UK', 'US']] = pd.get_dummies(df_new['country'])
df_new.head()
```

Out[40]:

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

- 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 [59]:
```
# Fit your model, and summarize the results


df_new['CA_ab_page'] = df_new['CA'] * df_new['ab_page']
df_new['UK_ab_page'] = df_new['UK'] * df_new['ab_page']
df_new['US_ab_page'] = df_new['US'] * df_new['ab_page']
df_new.head()
```

Out[59]:

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

In [51]:
```
log_mod = sm.Logit(df_new['converted'], df_new[['intercept', 'US_ab_page', 'UK_ab_page', 'CA_ab_page']])
results = log_mod.fit()
results.summary2()
```

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

Out[51]:

Model:	Logit	Pseudo R-squared:	0.000
Dependent Variable:	converted	AIC:	212778.9383
Date:	2022-06-02 05:30	BIC:	212821.2568
No. Observations:	290584	Log-Likelihood:	-1.0639e+05
Df Model:	3	LL-Null:	-1.0639e+05
Df Residuals:	290580	LLR p-value:	0.067853
Converged:	1.0000	Scale:	1.0000
No. Iterations:	6.0000		

	Coef.	Std.Err.	z	P>|z|	[0.025	0.975]
intercept	-1.9888	0.0081	-246.6690	0.0000	-2.0046	-1.9730
US_ab_page	-0.0183	0.0126	-1.4486	0.1475	-0.0430	0.0064
UK_ab_page	0.0074	0.0180	0.4098	0.6819	-0.0279	0.0427
CA_ab_page	-0.0827	0.0380	-2.1763	0.0295	-0.1571	-0.0082

In [52]:
```
np.exp(results.params)
```

Out[52]:
```
intercept     0.136863
US_ab_page    0.981901
UK_ab_page    1.007417
CA_ab_page    0.920649
dtype: float64
```

Summary: By looking at the odds ratio of the interaction between country and ab_page, converting users in each country(0.98, 1.007, 0.92). We thus fail to reject the null hypothesis; the new page does not convert more than the old page.

conclussion

Based on the statistical tests we used, the Z-test, logistic regression model, and actual difference observed, the results have shown that the new and old page have equal chance of converting users. We fail to reject the null hypothesis. I recommend to keep the old page to save money and time

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).

1. 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.

1. 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 [102…
```
from subprocess import call
call(['python', '-m', 'nbconvert', 'Analyze_ab_test_results_notebook.ipynb'])
```

Out[102…
```
1
```

In [ ]: