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 (https://review.udacity.com/#!/rubrics/1214/view) 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 statsmodels.api as sm
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)
```

```
/opt/conda/lib/python3.6/site-packages/statsmodels/compat/pandas.py:56: FutureWarning: The pandas.co
re.datetools module is deprecated and will be removed in a future version. Please use the pandas.tse
ries module instead.
  from pandas.core import datetools
```

ToDo 1.1

Now, read in the `ab_data.csv` data. Store it in `df` .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)
```

- There are 294478 observations in this dataset.

c. The number of unique users in the dataset.

In [4]:

```
df.nunique()
```

Out[4]:

```
user_id         290584
timestamp       294478
group                2
landing_page         2
converted            2
dtype: int64
```

- There are 290584 unique users.

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]:

```
not_aligned1 = len(df.query('group == "treatment" & landing_page != "new_page"'))
not_aligned2 = len(df.query('group != "treatment" & landing_page == "new_page"'))

not_aligned1 + not_aligned2
```

Out[6]:

```
3893
```

- 3893 is the number of rows where treatment is not aligned with new_page or control is not aligned with old_page.

f. Do any of the rows have missing values?

In [7]:

```
df.isnull().count()
```

Out[7]:

```
user_id         294478
timestamp       294478
group           294478
landing_page    294478
converted       294478
dtype: int64
```

- No missing values.

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
df2 = df.query('(group == "control" and landing_page == "old_page") or (group == "treatment" and landing_page ==
"new_page")')
```

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].shape[0]
```

Out[9]:

```
0
```

In [10]:

```
df2.shape
```

Out[10]:

```
(290585, 5)
```

- The new dataset has 290585 observations after dropping the rows where treatment is not aligned with new_page or control is not aligned with old_page.

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 [11]:

```
df2.nunique().user_id
```

Out[11]:

```
290584
```

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

In [12]:

```
duplicate_user = df2[df2.duplicated('user_id', keep=False) == True]
duplicate_user
```

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

- user_id 773192 is duplicated.

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

In [13]:

```
duplicate_user
```

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..
# Hint: The dataframe.drop_duplicates() may not work in this case because the rows with duplicate user_id are not
entirely identical.

df2 = df2.drop(1899)

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

Out[14]:

	user_id	timestamp	group	landing_page	converted
2893	773192	2017-01-14 02:55:59.590927	treatment	new_page	0

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 [15]:

```
p_converted = df2['converted'].mean()
p_converted
```

Out[15]:

```
0.11959708724499628
```

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

In [16]:

```
control = df2.query("group == 'control'")['converted'].mean()
control
```

Out[16]:

```
0.1203863045004612
```

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

In [17]:

```
treatment = df2.query("group == 'treatment'")['converted'].mean()
treatment
```

Out[17]:

```
0.11880806551510564
```

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

In [18]:

```
# Calculate the actual difference (obs_diff) between the conversion rates for the two groups.
obs_diff = treatment - control
obs_diff
```

Out[18]:

```
-0.0015782389853555567
```

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

In [19]:

```
new_page = len(df2.query("landing_page == 'new_page'"))/df2.shape[0]
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.

> **The difference between the probability of conversion if an individual was in the treatment group (11.88%), and the probability of conversion if an individual was in the control group (12.038%) is very small (0.158%) to be considered significant and it doesn't .**

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.

Null Hypotheses

$$H_0: p_{new} - p_{old} \leq 0$$

Alternative Hypotheses

$$H_1: p_{new} - p_{old} > 0$$

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 [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 = p_new
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("group == 'treatment'")['user_id'].count()
n_new
```

Out[22]:

```
145310
```

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

In [23]:

```
n_old = df2.query("group == 'control'")['user_id'].count()
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

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

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

In [25]:

```
# Simulate a Sample for the control Group

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

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

In [26]:

```
new_page_converted.mean() - old_page_converted.mean()
```

Out[26]:

```
1.1520355659627723e-05
```

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 i in range (10000):
    new_page_converted = np.random.choice([1,0], size = n_new, replace = True, p = (p_new, 1-p_new))
    old_page_converted = np.random.choice([1,0], size = n_old, replace = True, p = (p_old, 1-p_old))
    converted_diff = new_page_converted.mean() - old_page_converted.mean()
    p_diffs.append(converted_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.

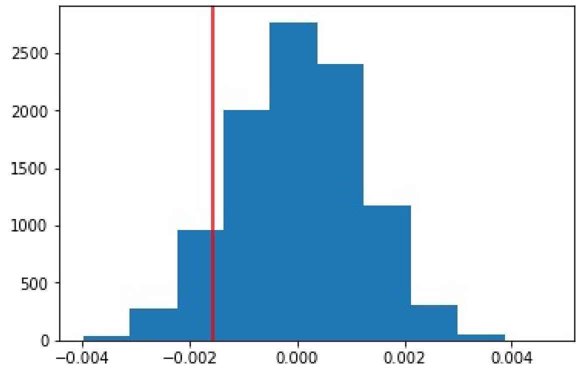
> **Tip**: Display title, x-label, and y-label in the chart.

In [28]:

```
# convert to numpy array
p_diffs = np.array(p_diffs)

# plot sampling distribution
plt.hist(p_diffs);

# plot line for observed statistic
plt.axvline(x=obs_diff, color='red');
```



The simulated Sampling distribution follows a normal distribution (bell shaped) with mean falling at zero. The actual difference (obs_diff) between the conversion rates for the two groups falls slightly bellow the distribution mean. This suggests statistical significance.

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

In [29]:

```
p_value = (p_diffs > obs_diff).mean()
p_value
```

Out[29]:

```
0.9061000000000002
```

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)".

The value calculated in `k.` is called P-value

The p-value (0.906) is greater than Type I error rate (0.05), therefore we don't have enough evidence to reject the null hypotheses. This means we don't have enough statistically significant evidence to support launching the new web page, thus the e-commerce company will have to keep the old web 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 [30]:

```
import statsmodels.api as sm

# number of conversions with the old_page
convert_old = len(df2.query("group == 'control' and converted == 1"))

# number of conversions with the new_page
convert_new = len(df2.query("group == 'treatment' and converted == 1"))

# number of individuals who were shown the old_page
n_old = len(df2.query("group == 'control'"))

# number of individuals who received new_page
n_new = len(df2.query("group == 'treatment'"))
```

m. Now use `sm.stats.proportions_ztest()` to compute your test statistic and p-value. Here (https://www.statsmodels.org/stable/generated/statsmodels.stats.proportion.proportions_ztest.html) 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}.

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

In other words, we determine whether or not the Z_{score} lies in the "rejection region" in the distribution. A "rejection region" is an interval where the null hypothesis is rejected iff the Z_{score} lies in that region.

Reference:

- Example 9.1.2 on this page (https://stats.libretexts.org/Bookshelves/Introductory_Statistics/Book%3A_Introductory_Statistics_(Shafer_and_Zhang)/09%3A_Two-Sample_Problems/9.01%3A_Comparison_of_Two_Population_Means-_Large_Independent_Samples), courtesy www.stats.libretexts.org

---

> **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_old], alternative = 'larger')
print(z_score, p_value)
```

```
-1.31092419842 0.905058312759
```

n. What do the z-score and p-value you computed in the previous question mean for the conversion rates of the old and new pages? Do they agree with the findings in parts **j.** and **k.**?

> **In this z-test, a z-score of (-1.3109) is less than the z-critical value (1.6) and since the hypotheses is right-tailed, we can conclude that we fail to reject the null hypotheses.**
>
> **Also, the p-value (0.9050) from z-test is similar to the one computed earlier and larger than 0.05. This means the conversion rates for new web page don't exceed that of old web 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?

> **Since we are looking to predict 1 of 2 possible outcomes. That is if there is a conversion or not depending on web page type either old or new. Logistic regression will be used**

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]:

```
# intercept
df2['intercept'] = 1
```

In [33]:

```
# dummy variable column
df2[['ab', 'ab_page']] = pd.get_dummies(df2['group'])
```

In [34]:

```
# drop 'ab' column
df2.drop(['ab'], axis=1, inplace=True)
df2.head()
```

Out[34]:

	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 [35]:

```
from scipy import stats
stats.chisqprob = lambda chisq, df: stats.chi2.sf(chisq, df)
```

In [36]:

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

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

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

In [37]:

```
results.summary()
```

Out[37]:

Logit Regression Results

Dep. Variable:	converted	**No. Observations:**	290584
Model:	Logit	**Df Residuals:**	290582
Method:	MLE	**Df Model:**	1
Date:	Sun, 02 Jan 2022	**Pseudo R-squ.:**	8.077e-06
Time:	23:51:59	**Log-Likelihood:**	-1.0639e+05
converged:	True	**LL-Null:**	-1.0639e+05
		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

In [38]:

```
np.exp(results.params)
```

Out[38]:

```
intercept    0.136863
ab_page      0.985123
dtype: float64
```

In [39]:

```
1/_
```

Out[39]:

```
intercept    7.306593
ab_page      1.015102
dtype: float64
```

- Holding all other variables constant, the new page is 1.015 times less likely to convert a user.

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

Hints:

- What are the null and alternative hypotheses associated with your regression model, and how do they compare to the null and alternative hypotheses in **Part II**?
- You may comment on if these hypothesis (Part II vs. Part III) are one-sided or two-sided.
- You may also compare the current p-value with the Type I error rate (0.05).

> **Since this is a logistic regression model, the tests concerns whether the parameter = 0 (null hypotheses) or the paramete != 0 (alternative hypotheses) (one tailed). This is different from the null and alternative hypotheses in Part II (two tailed) where we test whether the difference in conversion rates (new-old) is smaller than and equal to zero or not. Part II and part III are two different approaches to A/B testing with different hypotheses.**
>
> **The p-value associated with ab_page (new page) is (0.190). if p-value < 0.05 (Type1 error), it means that the relationship between an explanatory variable and the response variable is statisticallt significant. This means that p-value of new page is not statistically significant while the p-value of old page of value zero Statistically significant**

f. Now, you are considering other things that might influence whether or not an individual converts. Discuss why it is a good idea to consider other factors to add into your regression model. Are there any disadvantages to adding additional terms into your regression model?

> **It would be a good idea to think about other variables to include in the regression model. It can assist us in assessing and comparing one or more predictor variables to the response value. However there are disadvantages to adding more variables into the regression model. Problems like multicollinearity, correlated errors, in addition that linear relationship simply may not exist. Although there are solutions like variance inflation factors and higher order terms, one must be careful when adding one more predictor to a regression model.**

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 (https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.join.html)](https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.join.html) 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 [40]:

```
# Read the countries.csv
country = pd.read_csv('countries.csv')
```

In [41]:

```
# Join with the df2 dataframe
df_merged = df2.join(country.set_index('user_id'), on = 'user_id')
df_merged.head()
```

Out[41]:

	user_id	timestamp	group	landing_page	converted	intercept	ab_page	country
0	851104	2017-01-21 22:11:48.556739	control	old_page	0	1	0	US
1	804228	2017-01-12 08:01:45.159739	control	old_page	0	1	0	US
2	661590	2017-01-11 16:55:06.154213	treatment	new_page	0	1	1	US
3	853541	2017-01-08 18:28:03.143765	treatment	new_page	0	1	1	US
4	864975	2017-01-21 01:52:26.210827	control	old_page	1	1	0	US

In [42]:

```
# Create the necessary dummy variables
df_merged['country'].unique()
```

Out[42]:

```
array(['US', 'CA', 'UK'], dtype=object)
```

In [43]:

```
df_merged[['CA', 'UK', 'US']] = pd.get_dummies(df_merged['country'])
df_merged.head()
```

Out[43]:

	user_id	timestamp	group	landing_page	converted	intercept	ab_page	country	CA	UK	US
0	851104	2017-01-21 22:11:48.556739	control	old_page	0	1	0	US	0	0	1
1	804228	2017-01-12 08:01:45.159739	control	old_page	0	1	0	US	0	0	1
2	661590	2017-01-11 16:55:06.154213	treatment	new_page	0	1	1	US	0	0	1
3	853541	2017-01-08 18:28:03.143765	treatment	new_page	0	1	1	US	0	0	1
4	864975	2017-01-21 01:52:26.210827	control	old_page	1	1	0	US	0	0	1

In [54]:

```
# Fit your model, and summarize the results
# use US as baseline
lm = sm.Logit(df_merged['converted'], df_merged[['intercept','ab_page','CA','UK']])
results = lm.fit()
results.summary()
```

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

Out[54]:

Logit Regression Results

Dep. Variable:	converted	**No. Observations:**	290584
Model:	Logit	**Df Residuals:**	290580
Method:	MLE	**Df Model:**	3
Date:	Sun, 02 Jan 2022	**Pseudo R-squ.:**	2.323e-05
Time:	23:56:17	**Log-Likelihood:**	-1.0639e+05
converged:	True	**LL-Null:**	-1.0639e+05
		LLR p-value:	0.1760

	coef	std err	z	P>|z|	[0.025	0.975]
intercept	-1.9893	0.009	-223.763	0.000	-2.007	-1.972
ab_page	-0.0149	0.011	-1.307	0.191	-0.037	0.007
CA	-0.0408	0.027	-1.516	0.130	-0.093	0.012
UK	0.0099	0.013	0.743	0.457	-0.016	0.036

In [55]:

```
np.exp(results.params)
```

Out[55]:

```
intercept    0.136795
ab_page      0.985168
CA           0.960062
UK           1.009932
dtype: float64
```

In [56]:

```
1/_
```

Out[56]:

```
intercept    7.310207
ab_page      1.015056
CA           1.041599
UK           0.990165
dtype: float64
```

- Canadian users are 1.04 times less likely to convert, holding all other variables constant.
- UK users are 0.99 times more likely to convert, holding all other variables constant.

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 [57]:

```
#  interaction between page and country
df_merged['CA_new'] = df_merged['CA'] * df_merged['ab_page']
df_merged['UK_new'] = df_merged['UK'] * df_merged['ab_page']
```

In [58]:

```
lm_1 = sm.Logit(df_merged['converted'],df_merged[['intercept','ab_page','CA','UK','CA_new','UK_new']])
results = lm_1.fit()
results.summary()
```

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

Out[58]:

Logit Regression Results

Dep. Variable:	converted	**No. Observations:**	290584
Model:	Logit	**Df Residuals:**	290578
Method:	MLE	**Df Model:**	5
Date:	Mon, 03 Jan 2022	**Pseudo R-squ.:**	3.482e-05
Time:	00:00:20	**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
ab_page	-0.0206	0.014	-1.505	0.132	-0.047	0.006
CA	-0.0175	0.038	-0.465	0.642	-0.091	0.056
UK	-0.0057	0.019	-0.306	0.760	-0.043	0.031
CA_new	-0.0469	0.054	-0.872	0.383	-0.152	0.059
UK_new	0.0314	0.027	1.181	0.238	-0.021	0.084

In [59]:

```
np.exp(results.params)
```

Out[59]:

```
intercept    0.137178
ab_page      0.979646
CA           0.982625
UK           0.994272
CA_new       0.954198
UK_new       1.031896
dtype: float64
```

In [60]:

```
1/_
```

Out[60]:

```
intercept    7.289813
ab_page      1.020776
CA           1.017682
UK           1.005761
CA_new       1.048001
UK_new       0.969090
dtype: float64
```

> **Based of p-values of countries which is greater than (0.05), it implies that changes in the convert decision are unrelated to the user's country.**

> **Conclusion:**
> **With a p-value of 0.9, we do not have sufficient evidence to reject the null hypothesis in Part II. It concludes that there is no evidence that a new page affects a user's conversion rate. Furthermore, the z-test as well as the regression model in Part III returned the same results.**
> **After including the country variable to determine whether or not the countries had an effect on conversion. And it was found that changes in the conversion rate are unrelated to the user's country.**
> **Therefore, we do not have sufficient evidence to reject the null hypothesis. As a result, there is no reason to switch to the new page if the old one is performing adequately.**

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 [61]:

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

Out[61]:

```
0
```

In [ ]:

```
```