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:

- Introduction
- Part I - Probability
- Part II - A/B Test
- Part III - Regression
- Final Check
- Submission

Specific programming tasks are marked with a **ToDo** tag.

Introduction¶

A/B tests are very commonly performed by data analysts and data scientists. For this project, you will be working to understand the results of an A/B test run by an e-commerce website. Your goal is to work through this notebook to help the company understand if they should:

- Implement the new webpage,
- Keep the old webpage, or
- Perhaps run the experiment longer to make their decision.

Each **ToDo** task below has an associated quiz present in the classroom. Though the classroom quizzes are **not necessary** to complete the project, they help ensure you are on the right track as you work through the project, and you can feel more confident in your final submission meeting the rubric specification.

> **Tip**: Though it's not a mandate, students can attempt the classroom quizzes to ensure statistical numeric values are calculated correctly in many cases.

Part I - Probability¶

To get started, let's import our libraries.

In [1]:

```
import pandas as pd
import numpy as np
import random
import 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.core.datetools module is deprecated and will be removed in a future version.
Please use the pandas.tseries module instead.
  from pandas.core import datetools
```

ToDo 1.1¶

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

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

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

Tip: Please save your work regularly.

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

In [2]:

```
df=pd.read_csv('ab_data.csv')
df.head()
```

Out[2]:

	user_id	timestamp	group	landing_page	converted
0	851104	2017-01-21 22:11:48.556739	control	old_page	0
1	804228	2017-01-12 08:01:45.159739	control	old_page	0
2	661590	2017-01-11	treatment	new_page	0

	user_id	timestamp	group	landing_page	converted
		16:55:06.154213			
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]:

```
len(df)
```

Out[3]:

```
294478
```

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
```

d. The proportion of users converted.

In [5]:

```
len(df.query("converted==1"))/len(df)
```

Out[5]:

```
0.11965919355605512
```

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

In [6]:

```
x=len(df.query("group=='treatment'and landing_page=='old_page' "))
y=len(df.query("group=='control'and landing_page=='new_page' "))
x+y
```

Out[6]:

3893

f. Do any of the rows have missing values?

In [7]:

```
len(df.isnull())
```

Out[7]:

294478

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
x=df.query("group=='control'and landing_page=='new_page'").index
y=df.query("group=='treatment'and landing_page=='old_page'").index
df2=df.drop(y)
df2=df2.drop(x)
```

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
```

ToDo 1.3¶

Use **df2** and the cells below to answer questions for **Quiz 3** in the classroom.

a. How many unique **user_id**s are in **df2**?

In [10]:

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

Out[10]:

```
290584
```

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

In [11]:

```
z=df2[df2.duplicated('user_id')]
z.index
```

Out[11]:

```
Int64Index([2893], dtype='int64')
```

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

In [12]:

```
z
```

Out[12]:

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

d. Remove **one** of the rows with a duplicate **user_id**, from the **df2** dataframe.

In [13]:

```
# Remove one of the rows with a duplicate user_id..
# Hint: The dataframe.drop_duplicates() may not work in this case because the rows with
duplicate user_id are not entirely identical.
df2=df2.drop(z.index)
# Check again if the row with a duplicate user_id is deleted or not
```

ToDo 1.4¶

Use **df2** in the cells below to answer the quiz questions related to **Quiz 4** in the classroom.

a. What is the probability of an individual converting regardless of the page they receive?

> **Tip**: The probability you'll compute represents the overall "converted" success rate in the

> population and you may call it $p_{population}$.

In [14]:

```
len(df2.query("converted==1"))/len(df2)
```

Out[14]:

```
0.11959708724499628
```

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

In [15]:

```
x=len(df2.query("group=='control'and converted==1"))/len(df2.query("group=='control'"))
x
```

Out[15]:

```
0.1203863045004612
```

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

In [16]:

```
y=len(df2.query("group=='treatment'and
converted==1"))/len(df2.query("group=='treatment'"))
y
```

Out[16]:

```
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 [17]:

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

Out[17]:

```
-0.0015782389853555567
```

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

In [18]:

```
len(df2.query("landing_page=='new_page'"))/len(df2)
```

Out[18]:

```
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.

> **probabilities prove that 'old_page' has more submition ratio than 'new_page'but there is factor we are ignoring that dataset we are workin in contains both old users and new users so the test would be un fair becouse old users will convert any way becouse they know the benifits of getting course from this site we should compare the time they spend to submit in old_page to the new_page with the same amount of information and description of course as possiable . to see how realy new_page is improving user experience or not**

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.

> **H0: p_new - p_old <=0 H1: 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 [19]:

```
len(df2.query(" converted ==1"))/len(df2)
```

Out[19]:

```
0.11959708724499628
```

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

In [20]:

```
len(df2.query(" converted ==1"))/len(df2)
```

Out[20]:

```
0.11959708724499628
```

c. What is n_{new}, the number of individuals in the treatment group?

Hint: The treatment group users are shown the new page.

In [21]:

```
len(df2.query("group=='treatment'"))
```

Out[21]:

```
145310
```

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

In [22]:

```
len(df2.query("group=='control'"))
```

Out[22]:

```
145274
```

e. Simulate Sample for the `treatment` Group
Simulate n_{new} transactions with a conversion rate of p_{new} under the null hypothesis.

Hint: Use `numpy.random.choice()` method to randomly generate n_{new} number of values.
Store these n_{new} 1's and 0's in the `new_page_converted` numpy array.

In [23]:

```
# Simulate a Sample for the treatment Group
x=df2.query("group=='treatment'")
new_page_converted=np.random.choice(x.converted,145310,replace=True)
new_page_converted.shape[0]
```

Out[23]:

```
145310
```

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

In [24]:

```
# Simulate a Sample for the control Group
y=df2.query("group=='control'")
old_page_converted=np.random.choice(y.converted,145274,replace=True)
old_page_converted.shape[0]
```

Out[24]:

```
145274
```

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

In [25]:

```
newp=new_page_converted.sum()/len(new_page_converted)
oldp=old_page_converted.sum()/len(new_page_converted)
newp-oldp
```

Out[25]:

```
-0.002188424747092424
```

h. Sampling distribution

Re-create `new_page_converted` and `old_page_converted` and find the $(p{'}_{new}$ - $p{'}_{old})$ value 10,000 times using the same simulation process you used in parts (a) through (g) above.

Store all $(p{'}_{new}$ - $p{'}_{old})$ values in a NumPy array called `p_diffs`.

In [26]:

```
# Sampling distribution
p_diffs = []
for _ in range(10000):
    sample_n=np.random.choice(new_page_converted,145310,replace=True)
    sample_o=np.random.choice(old_page_converted,145274,replace=True)
    p_n=sample_n.sum()/len(sample_n)
    p_o=sample_o.sum()/len(sample_o)
    p_diffs.append(p_n-p_o)
```

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.

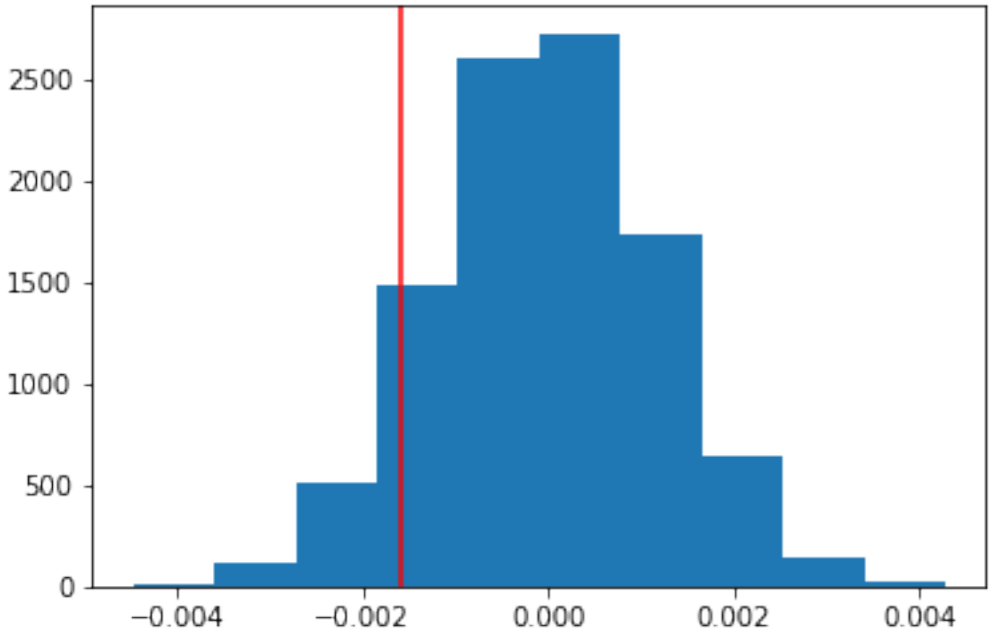
> **Tip**: Display title, x-label, and y-label in the chart.

In [34]:

```
plt.hist(p_diffs);
plt.axvline(obs_diff,color='red')
```

Out[34]:

```
<matplotlib.lines.Line2D at 0x7f13c4745908>
```



In [ ]:

In [37]:

```
p_diffs=np.array(p_diffs)
(p_diffs>obs_diff).mean()
```

Out[37]:

```
0.90459999999999996
```

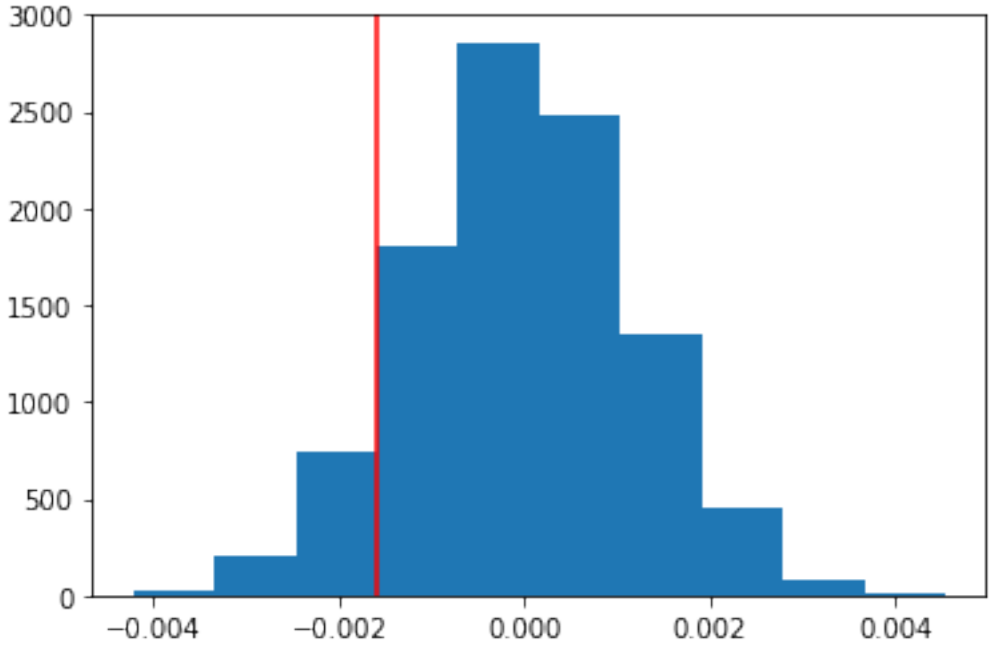
j. What proportion of the **p_diffs** are greater than the actual difference observed in the df2 data?

In [38]:

```
null_vals=np.random.normal(0,p_diffs.std(),10000)
plt.hist(null_vals);
plt.axvline(obs_diff,color='red')
```

Out[38]:

```
<matplotlib.lines.Line2D at 0x7f13c4236128>
```



In [40]:

```
null_vals=np.array(null_vals)
null_vals=np.array(null_vals)
(null_vals>obs_diff).mean()
```

Out[40]:

```
0.90110000000000001
```

k. Please explain in words what you have just computed in part **j** above.

- What is this value called in scientific studies? p_value
- What does this value signify in terms of whether or not there is a difference between the new and old pages? *show us extreme values in that greater than ops_mean in favor fo h1*: Compare the value above with the "Type I error rate (0.05)" it seems that we reject the null hypothesis.

> **p_value show us extreme values in that greater than ops_mean in favor fo h1: Compare the value above with the "Type I error rate (0.05)" it seems that we fail to reject null.**

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(" landing_page =='old_page' "))

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

c_r=[convert_old,convert_new]

n_r=[n_old,n_new]
```

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.

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

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/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]:

```
# ToDo: Complete the sm.stats.proportions_ztest() method arguments
z_score, p_value = sm.stats.proportions_ztest(c_r,n_r,alternative='smaller')
print(z_score, p_value)
```

```
1.31092419842 0.905058312759
```

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

> **Tip**: Notice whether the p-value is similar to the one computed earlier. Accordingly, can you reject/fail to reject the null hypothesis? It is important to correctly interpret the test statistic and p-value.
>
> **z_score test statistic p_value is extreme values in favor of H1 the have almost the same p_value but test statistic fall in rejection region so this test say we should reject the null**

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?

> **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 [54]:

```
df2['intercept']=1
df2[['non','ab_page']]=pd.get_dummies(df['group'])
df2[['ncon','con']]=pd.get_dummies(df['converted'])
x=df2
x=x.drop('non',axis=1)
x=x.drop('ncon',axis=1)
```

```
x.head()
```

Out[54]:

	user_id	timestamp	group	landing_page	converted	intercept	ab_page	con
0	851104	2017-01-21 22:11:48.556739	control	old_page	0	1	0	0

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

c. Use **statsmodels** to instantiate your regression model on the two columns you created in part (b). above, then fit the model to predict whether or not an individual converts.

In [55]:

```
lm=sm.Logit(x['con'], x[['intercept','ab_page']])
result=lm.fit()
result.summary2()

Optimization terminated successfully.
         Current function value: 448.424145
         Iterations 6
```

Out[55]:

Model:	Logit	No. Iterations:	6.0000
Dependent Variable:	con	Pseudo R-squared:	-0.000
Date:	2022-12-17 22:42	AIC:	260609767.5214
No. Observations:	290584	BIC:	260609788.6807
Df Model:	1	Log-Likelihood:	-1.3030e+08
Df Residuals:	290582	LL-Null:	-1.3030e+08
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.

In [ ]:

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).

> **0.1899, this is p_value realted ap_page meen that it isnot staticly signifcant in predecting convertion with treatment group becouse it tests the response value in either direction of regression line (two-tailed test) that response value should be above or under the regression line in a range if it located farther than this range so it should be an extreme value part 2 we test p_value in one direction only**

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?

> **i think data frame we working on has poor information we cant just compare students from different countries only based on the number of students who converted in each country, i think we can add some factors to data frame . like education level, does english in their country is mother language or not, did they take a course before on adacity or not, time they spent until converting, we can compare students from the same country and they have the same factors value and divide them into two groups one for new page and the other for old page, and with the result we get we can fit it into regression model to see our prediction fit our data base or not .**

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 [56]:

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

x=x.join(dc.set_index('user_id'), on='user_id')
```

```
x[['ca','uk','us']]=pd.get_dummies(x['country'])
# Create the necessary dummy variables
x.head()
```

Out[56]:

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

In [58]:

```
lm=sm.Logit(x['con'],x[['intercept','ab_page','us','uk']])
result=lm.fit()
result.summary2()

Optimization terminated successfully.
         Current function value: 448.436079
         Iterations 6
```

Out[58]:

Model:	Logit	No. Iterations:	6.0000
Dependent Variable:	con	Pseudo R-squared:	-0.000
Date:	2022-12-17 22:46	AIC:	260616706.9997
No. Observations:	290584	BIC:	260616749.3183
Df Model:	3	Log-Likelihood:	-1.3031e+08
Df Residuals:	290580	LL-Null:	-1.3030e+08
Converged:	1.0000	Scale:	1.0000

	Coef.	Std.Err.	z	P>|z|	[0.025	0.975]
intercept	-2.0300	0.0266	-76.2488	0.0000	-2.0822	-1.9778

ab_page	-0.0149	0.0114	-1.3069	0.1912	-0.0374	0.0075
us	0.0408	0.0269	1.5161	0.1295	-0.0119	0.0934
uk	0.0506	0.0284	1.7835	0.0745	-0.0050	0.1063

In [45]:

```
# they have high p_value the arenot  staticly signifcant ,it dosnt has impact on
convertion

#  they have different confdient value we should add higher order  terms
```

In [ ]:

In [39]:

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 [42]:

```
x['us_ap']=x['ab_page']*x['us']
x['uk_ap']=x['ab_page']*x['uk']
lm=sm.Logit(x['con'],x[['intercept','ab_page','us','uk','uk_ap','us_ap']])
result2=lm.fit()
result2.summary2()

# Fit your model, and summarize the results

Optimization terminated successfully.
         Current function value: 448.445345
```

```
         Iterations 6
```

Out[42]:

Model:	Logit	No. Iterations:	6.0000
Dependent Variable:	con	Pseudo R-squared:	-0.000
Date:	2022-12-17 17:35	AIC:	260622096.3876
No. Observations:	290584	BIC:	260622159.8655
Df Model:	5	Log-Likelihood:	-1.3031e+08
Df Residuals:	290578	LL-Null:	-1.3030e+08
Converged:	1.0000	Scale:	1.0000

	Coef.	Std.Err.	z	P>|z|	[0.025	0.975]
intercept	-2.0040	0.0364	-55.0077	0.0000	-2.0754	-1.9326
ab_page	-0.0674	0.0520	-1.2967	0.1947	-0.1694	0.0345
us	0.0175	0.0377	0.4652	0.6418	-0.0563	0.0914
uk	0.0118	0.0398	0.2957	0.7674	-0.0663	0.0899
uk_ap	0.0783	0.0568	1.3783	0.1681	-0.0330	0.1896
us_ap	0.0469	0.0538	0.8718	0.3833	-0.0585	0.1523

In [43]:

```
Optimization terminated successfully.
         Current function value: 448.436079
         Iterations 6
```

Out[43]:

Model:	Logit	No. Iterations:	6.0000
Dependent Variable:	con	Pseudo R-squared:	-0.000
Date:	2022-12-17 17:58	AIC:	260616706.9997
No. Observations:	290584	BIC:	260616749.3183
Df Model:	3	Log-Likelihood:	-1.3031e+08
Df Residuals:	290580	LL-Null:	-1.3030e+08
Converged:	1.0000	Scale:	1.0000

	Coef.	Std.Err.	z	P>|z|	[0.025	0.975]
intercept	-2.0300	0.0266	-76.2488	0.0000	-2.0822	-1.9778
ab_page	-0.0149	0.0114	-1.3069	0.1912	-0.0374	0.0075
us	0.0408	0.0269	1.5161	0.1295	-0.0119	0.0934
uk	0.0506	0.0284	1.7835	0.0745	-0.0050	0.1063

> **with p_value in the table say that country is not staticly signifcant to predict the conversion rate according to treatment group we fail to reject null according to these colums , we should get more accurate data so that we can predict behavior of students, there is no different in R-squared value in both summaries,us and uk have different confidence value maybe they need higher order terms,but not with ab_page , uk_ap and us_ap they have high p_vale compare to alpha 0.05 they are not staticly signifcant, the effect of page and country to predict the conversion not effective conclusion: (we should stick with the old page).**

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).

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 [86]:

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

Out[86]:

```
0
```

In [ ]: