Analyze_ab_test_results_notebook

March 27, 2022

1 Analyze A/B Test Results

This project will assure you have mastered the subjects covered in the statistics lessons. We have organized the current notebook into the following sections:

- Section **??**
- Section **??**
- Section **??**
- Section **??**
- Section **??**
- Section **??**

Specific programming tasks are marked with a **ToDo** tag.
Introduction
A/B tests are very commonly performed by data analysts and data scientists. For this project, you will be working to understand the results of an A/B test run by an e-commerce website. Your goal is to work through this notebook to help the company understand if they should: - Implement the new webpage, - Keep the old webpage, or - Perhaps run the experiment longer to make their decision.

Each **ToDo** task below has an associated quiz present in the classroom. Though the classroom quizzes are **not necessary** to complete the project, they help ensure you are on the right track as you work through the project, and you can feel more confident in your final submission meeting the rubric specification.
Part I - Probability
To get started, let's import our libraries.

```
In [77]: import pandas as pd
         import numpy as np
         import random
         import matplotlib.pyplot as plt
         %matplotlib inline
         #We are setting the seed to assure you get the same answers on quizzes as we set up
         random.seed(42)
```

1.0.1 ToDo 1.1

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

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

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

Tip: Please save your work regularly.

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

```
In [78]: #reading in our csv file
         df=pd.read_csv('ab_data.csv')
         df.head()
```

Out[78]:

	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 [79]: #finding the number of rows in the dataset
         len(df)
```

```
Out[79]: 294478
```

c. The number of unique users in the dataset.

```
In [80]: #finding the number of unique rows in the dataset
         df['user_id'].nunique()
```

```
Out[80]: 290584
```

d. The proportion of users converted.

```
In [81]: #taking the sum of converted rows and dividing by the number of unique users to get teh
         df['converted'].sum()/df['user_id'].nunique()
```

```
Out[81]: 0.12126269856564711
```

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

```
In [82]: #taking a count of the returned queries that group is treatment but landing page is not
         df.query("group == 'treatment' and landing_page!='new_page'")['user_id'].count()
```

```
Out[82]: 1965
```

f. Do any of the rows have missing values?

```
In [83]: #taking a sum of the number of each row that has a null value
         df.isnull().sum()
```

```
Out[83]: user_id         0
         timestamp       0
         group           0
         landing_page    0
         converted       0
         dtype: int64
```

```
In [84]: #No there are no missing values
```

1.0.2 ToDo 1.2

In a particular row, the **group** and **landing_page** columns should have either of the following acceptable values:

user_id	timestamp	group	landing_page	converted
XXXX	XXXX	control	old_page	X
XXXX	XXXX	treatment	new_page	X

It means, the control group users should match with old_page; and treatment group users should matched with the new_page.

However, for the rows where treatment does not match with new_page or control does not match with old_page, we cannot be sure if such rows truly received the new or old wepage.

Use **Quiz 2** in the classroom to figure out how should we handle the rows where the group and landing_page columns don't match?

a. Now use the answer to the quiz to create a new dataset that meets the specifications from the quiz. Store your new dataframe in **df2**.

In [85]:
```
# Remove the inaccurate rows, and store the result in a new dataframe df2
#selecting just the control group with the old_page
df2first = df.query("group == 'control' and landing_page=='old_page'")
#selecting just the treatment group with the new_page
df2second =df.query("group == 'treatment' and landing_page=='new_page'")
#creating new array to hold them
df2 = [df2first, df2second]
#concat them together
df2 = pd.concat(df2)
df2
```

Out[85]:

	user_id	timestamp	group	landing_page	converted
0	851104	2017-01-21 22:11:48.556739	control	old_page	0
1	804228	2017-01-12 08:01:45.159739	control	old_page	0
4	864975	2017-01-21 01:52:26.210827	control	old_page	1
5	936923	2017-01-10 15:20:49.083499	control	old_page	0
7	719014	2017-01-17 01:48:29.539573	control	old_page	0
15	644214	2017-01-22 02:05:21.719434	control	old_page	1
16	847721	2017-01-17 14:01:00.090575	control	old_page	0
18	650559	2017-01-24 11:55:51.084801	control	old_page	0
19	935734	2017-01-17 20:33:37.428378	control	old_page	0
25	746742	2017-01-23 11:38:29.592148	control	old_page	0
28	913579	2017-01-24 09:11:39.164256	control	old_page	1
30	690284	2017-01-13 17:22:57.182769	control	old_page	0
34	710349	2017-01-11 22:24:44.226492	control	old_page	0
35	677533	2017-01-23 17:48:50.491821	control	old_page	0
36	831737	2017-01-11 21:18:20.911015	control	old_page	1
40	771087	2017-01-16 00:05:29.983919	control	old_page	0
42	896163	2017-01-22 09:10:20.753218	control	old_page	0
43	862225	2017-01-08 14:49:37.335432	control	old_page	1
44	939593	2017-01-05 09:15:31.984283	control	old_page	0
45	702260	2017-01-18 13:55:31.488221	control	old_page	0
50	670941	2017-01-05 08:16:41.306478	control	old_page	0
51	850231	2017-01-18 17:18:04.790584	control	old_page	1

```
55       685794  2017-01-20 14:54:58.150621    control     old_page          0
57       714733  2017-01-03 08:22:37.904146    control     old_page          0
58       710967  2017-01-10 02:19:22.842142    control     old_page          0
59       680201  2017-01-11 10:38:45.952887    control     old_page          0
60       790863  2017-01-19 11:02:39.220320    control     old_page          0
61       717595  2017-01-23 18:19:08.148166    control     old_page          0
62       779854  2017-01-11 21:28:30.735359    control     old_page          0
63       916307  2017-01-19 17:27:38.676600    control     old_page          0
...         ...                           ...         ...          ...       ...
294417   924332  2017-01-15 19:38:52.858024  treatment     new_page          0
294422   849625  2017-01-06 17:54:07.563311  treatment     new_page          0
294424   929723  2017-01-10 15:13:48.352399  treatment     new_page          0
294427   774769  2017-01-03 06:01:36.251836  treatment     new_page          0
294430   733871  2017-01-21 17:54:08.810964  treatment     new_page          1
294432   844588  2017-01-16 20:48:19.567178  treatment     new_page          0
294433   641244  2017-01-07 16:57:26.193171  treatment     new_page          0
294434   676072  2017-01-14 17:26:02.495442  treatment     new_page          0
294435   886374  2017-01-07 13:43:39.202634  treatment     new_page          0
294437   676732  2017-01-03 23:06:45.459467  treatment     new_page          0
294439   862218  2017-01-04 10:43:07.846494  treatment     new_page          0
294441   798826  2017-01-23 16:50:13.788528  treatment     new_page          0
294442   836721  2017-01-12 17:37:50.966955  treatment     new_page          0
294444   844901  2017-01-15 17:46:36.622726  treatment     new_page          0
294445   653124  2017-01-14 13:44:51.745491  treatment     new_page          0
294446   909437  2017-01-18 14:49:49.064452  treatment     new_page          0
294448   776137  2017-01-12 05:53:12.386730  treatment     new_page          0
294449   883344  2017-01-22 23:15:58.645325  treatment     new_page          0
294450   825594  2017-01-06 12:37:08.897784  treatment     new_page          0
294454   937338  2017-01-19 03:23:22.236666  treatment     new_page          0
294455   733101  2017-01-23 12:52:58.711914  treatment     new_page          0
294456   679096  2017-01-02 16:43:49.237940  treatment     new_page          0
294457   691699  2017-01-09 23:42:35.963486  treatment     new_page          0
294458   807595  2017-01-22 10:43:09.285426  treatment     new_page          0
294460   846225  2017-01-16 15:24:46.705903  treatment     new_page          0
294462   677163  2017-01-03 19:41:51.902148  treatment     new_page          0
294465   925675  2017-01-07 20:38:26.346410  treatment     new_page          0
294468   643562  2017-01-02 19:20:05.460595  treatment     new_page          0
294472   822004  2017-01-04 03:36:46.071379  treatment     new_page          0
294477   715931  2017-01-16 12:40:24.467417  treatment     new_page          0

[290585 rows x 5 columns]
```

In [86]:
```
# Double Check all of the incorrect rows were removed from df2 -
# Output of the statement below should be 0
df2[((df2['group'] == 'treatment') == (df2['landing_page'] == 'new_page')) == False].sh
```

Out[86]:
```
0
```

1.0.3 ToDo 1.3

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

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

```
In [87]: #using the shape to get the number of user_ids
         df2.shape[0]
```

```
Out[87]: 290585
```

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

```
In [88]: #finding the duplicated and printing the user id
         ddf2=df2[df2['user_id'].duplicated() == True]
         ddf2.user_id
```

```
Out[88]: 2893    773192
         Name: user_id, dtype: int64
```

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

```
In [89]: #displaying the row for that user id
         ddf2
```

```
Out[89]:       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 [90]: # Remove one of the rows with a duplicate user_id..

         #dropping the row by using the index
         df2= df2.drop(2893)
         # Check again if the row with a duplicate user_id is deleted or not
         ddf2=df2[df2['user_id'].duplicated() == True]
         ddf2.user_id
```

```
Out[90]: Series([], Name: user_id, dtype: int64)
```

1.0.4 ToDo 1.4

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

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

> **Tip**: The probability you'll compute represents the overall "converted" success rate in the population and you may call it $p_{population}$.

```
In [91]: #setting df2rows to the number of rows to use later and printing it out
         df2rows =df2.shape[0]
         df2rows
```

```
Out[91]: 290584

In [92]: #printing the values counts of converted 1 and not converted 0
         df2.converted.value_counts()

Out[92]: 0    255831
         1     34753
         Name: converted, dtype: int64

In [93]: #printing the number of all converted
         df2.converted.sum()

Out[93]: 34753

In [94]: #taking the number of converted and dividing by the number of rows
         df2.query("converted == 1").count()[0]/df2rows

Out[94]: 0.11959708724499628
```

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

```
In [95]: #taking the number in the control group that are converted and dividing by the total in
         cgroup =df2.query("group =='control' and converted == 1").count()[0]/df2.query("group =
         cgroup

Out[95]: 0.1203863045004612
```

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

```
In [96]: #taking the number in the treatment group that converted by the total number in the tre
         tgroup = df2.query("group =='treatment' and converted == 1").count()[0]/df2.query("grou
         tgroup

Out[96]: 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 [97]: # Calculate the actual difference (obs_diff) between the conversion rates for the two g
         obs_diff= df2.query("group =='treatment' and converted == 1").count()[0]/df2.query("gro
         obs_diff

Out[97]: -0.0015782389853555567
```

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

```
In [98]: #taking the number that recieved the new page divided by the total rows
         df2.query("landing_page == 'new_page'").count()[0]/df2rows
```

```
Out[98]: 0.50006194422266881
```

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

> **Your answer goes here.** Since the actual difference 0.001578 is so small I don't feel that the treatment group leads to a significantly greater number of conversions.

Part II - A/B Test

Since a timestamp is associated with each event, you could run a hypothesis test continuously as long as you observe the events.

However, then the hard questions would be: - Do you stop as soon as one page is considered significantly better than another or does it need to happen consistently for a certain amount of time?

- How long do you run to render a decision that neither page is better than another?

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

1.0.5 ToDo 2.1

For now, consider you need to make the decision just based on all the data provided.

> Recall that you just calculated that the "converted" probability (or rate) for the old page is *slightly* higher than that of the new page (ToDo 1.4.c).

If you want to assume that the old page is better unless the new page proves to be definitely better at a Type I error rate of 5%, what should be your null and alternative hypotheses (H_0 and H_1)?

You can state your hypothesis in terms of words or in terms of p_{old} and p_{new}, which are the "converted" probability (or rate) for the old and new pages respectively.

> **Put your answer here.**

0 = Pnew <= H1 = Pnew >

1.0.6 ToDo 2.2 - Null Hypothesis H_0 Testing

Under the null hypothesis H_0, assume that p_{new} and p_{old} are equal. Furthermore, assume that p_{new} and p_{old} both are equal to the **converted** success rate in the `df2` data regardless of the page. So, our assumption is:

$p_{new} = p_{old} = p_{population}$

In this section, you will:

- Simulate (bootstrap) sample data set for both groups, and compute the "converted" probability p for those samples.
- Use a sample size for each group equal to the ones in the `df2` data.
- Compute the difference in the "converted" probability for the two samples above.
- Perform the sampling distribution for the "difference in the converted probability" between the two simulated-samples over 10,000 iterations; and calculate an estimate.

Use the cells below to provide the necessary parts of this simulation. You can use **Quiz 5** in the classroom to make sure you are on the right track.

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

```
In [99]: df2.head()
```

Out[99]:

	user_id	timestamp	group	landing_page	converted
0	851104	2017-01-21 22:11:48.556739	control	old_page	0
1	804228	2017-01-12 08:01:45.159739	control	old_page	0
4	864975	2017-01-21 01:52:26.210827	control	old_page	1
5	936923	2017-01-10 15:20:49.083499	control	old_page	0
7	719014	2017-01-17 01:48:29.539573	control	old_page	0

```
In [100]: #setting df2size as the total number of rows
          df2size=df2.shape[0]

In [101]: #taking the number converted using the new page and dividing by the total size to get
          pnewconv=df2.query("converted == 1").count()[0]/df2rows
          pnewconv
```

Out[101]: 0.11959708724499628

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

```
In [102]: #taking the number of converted using the control group and dividing by total size to
          poldconv= df2.query("converted == 1").count()[0]/df2rows
          poldconv
```

Out[102]: 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 [103]: #setting nnew to the number in the treatment group
          nnew = df2.query('group =="treatment"')
          #setting nnew to the unique user ids in nnew
          nnew = nnew.user_id.nunique()
          nnew
```

Out[103]: 145310

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

```
In [104]: #setting nnew to the number in the control group
          nold = df2.query('group == "control"')
          #setting nold to the unique users ids in nold
          nold =nold.user_id.nunique()
          nold
```

Out[104]: 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 [105]: # Simulate a Sample for the treatment Group
          #using np.random.choice on nnew
          ##not right but need to use binomial
          new_page_converted=np.random.binomial(1, pnewconv, nold)
```

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 [106]: # Simulate a Sample for the control Group
          #using np.random.choice on the nold
          ##not right but need to use binomial
          old_page_converted=np.random.binomial(1, poldconv, nold)
```

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

```
In [107]: #the difference between the simulated groups
          new_page_converted - old_page_converted
```

```
Out[107]: array([0, 0, 0, ..., 0, 0, 1])
```

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 [108]: # Sampling distribution
          p_diffs = []
          for _ in range(10000):
              newconvsim = np.random.binomial(1, pnewconv, nnew)
              oldconvsim = np.random.binomial(1, poldconv, nold)
              p_diffs.append(newconvsim.mean() - oldconvsim.mean())
```

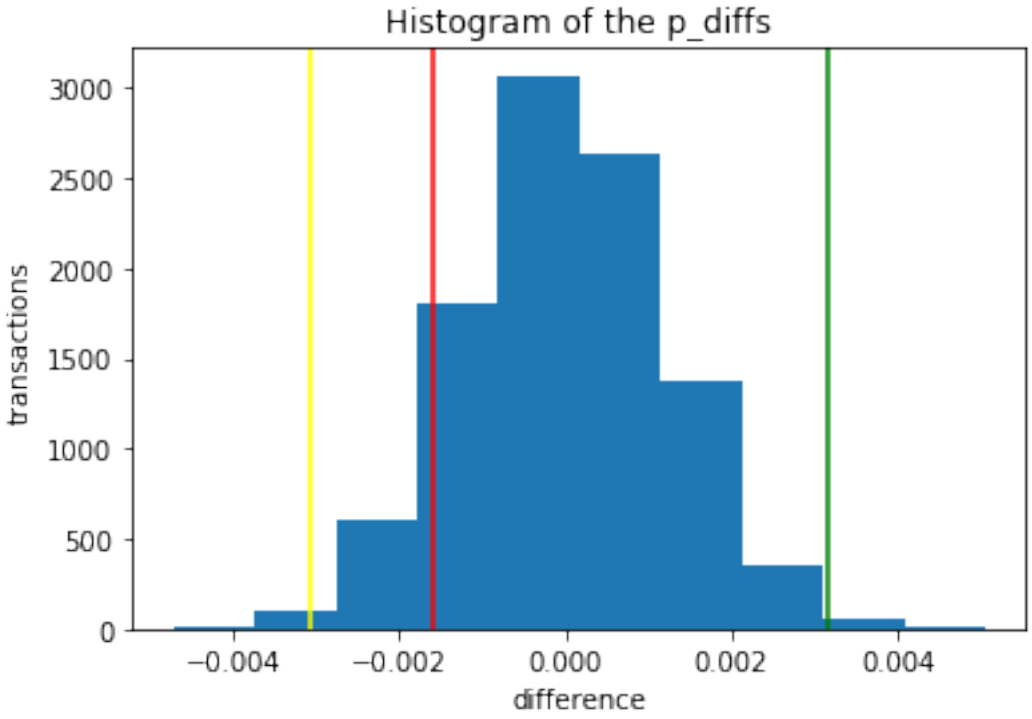
i. Histogram Plot a histogram of the **p_diffs**. Does this plot look like what you expected? Use the matching problem in the classroom to assure you fully understand what was computed here.

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

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

```
In [109]: p_diffs = np.array(p_diffs)
```

```
plt.hist(p_diffs);
plt.axvline(obs_diff, color = 'red');
plt.axvline(np.percentile(p_diffs, 0.5), color = 'yellow');
plt.axvline(np.percentile(p_diffs, 99.5), color = 'green');
plt.ylabel('transactions');
plt.xlabel('difference');
plt.title('Histogram of the p_diffs');
```



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

```
In [110]: #calculating the actual difference observed by taking the probability the treatment gr
          obsdiff = tgroup - cgroup
          obsdiff

Out[110]: -0.0015782389853555567

In [111]: #pvalue
          pvalue = (p_diffs < obsdiff).mean()
          #counting the number that are greater than the actual difference
          greaterthanobs = 0
          for i in p_diffs:
              if i>obsdiff:
                  greaterthanobs = greaterthanobs + 1
```

```
propgreatdiff = (greaterthanobs)/(len(p_diffs))
propgreatdiff
```

```
Out[111]: 0.9032
```

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)".

> **Put your answer here.** This is called the p value. Since the p-value of 0.9029 is larger than the Type I error rate of 0.05 we fail to reject the null hypothesis.

l. Using Built-in Methods for Hypothesis Testing We could also use a built-in to achieve similar results. Though using the built-in might be easier to code, the above portions are a walk-through of the ideas that are critical to correctly thinking about statistical significance.

Fill in the statements below to calculate the: - `convert_old`: number of conversions with the old_page - `convert_new`: number of conversions with the new_page - `n_old`: number of individuals who were shown the old_page - `n_new`: number of individuals who were shown the new_page

```
In [112]: df2.head(1)
```

```
Out[112]:    user_id                   timestamp    group landing_page  converted
          0   851104  2017-01-21 22:11:48.556739  control     old_page          0
```

```
In [113]: import statsmodels.api as sm

          # number of conversions with the old_page
          convert_old = df2.query("converted ==1 and landing_page == 'old_page'").shape[0]

          # number of conversions with the new_page
          convert_new =df2.query("converted ==1 and landing_page == 'new_page'").shape[0]

          # 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]

          #displaying these values
          convert_old,convert_new,n_old,n_new
```

```
Out[113]: (17489, 17264, 145274, 145310)
```

m. Now use `sm.stats.proportions_ztest()` to compute your test statistic and p-value. Here is a helpful link on using the built in.

The syntax is:

```
proportions_ztest(count_array, nobs_array, alternative='larger')
```

where, - `count_array` = represents the number of "converted" for each group - `nobs_array` = represents the total number of observations (rows) in each group - `alternative` = choose one of the values from `[two-sided, smaller, larger]` depending upon two-tailed, left-tailed, or right-tailed respectively. >**Hint**: It's a two-tailed if you defined H_1 as $(p_{new} = p_{old})$. It's a left-tailed if you defined H_1 as $(p_{new} < p_{old})$. It's a right-tailed if you defined H_1 as $(p_{new} > p_{old})$.

The built-in function above will return the z_score, p_value.

> **Tip**: You don't have to dive deeper into z-test for this exercise. **Try having an overview of what does z-score signify in general.**

```
In [114]: import statsmodels.api as sm
          # ToDo: Complete the sm.stats.proportions_ztest() method arguments
          #count = convert_old+convert_new

          #nobs = n_old+n_new

          z_score, p_value = sm.stats.proportions_ztest([convert_old, convert_new][::-1],[n_old,
          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.

> **Put your answer here.** That the standard deviation between our conversions is 1.31. The p-value is still about the 0.05 so we would still reject the null.

Part III - A regression approach

1.0.7 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.** Since this is a classification it should use 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 [115]: #create the intecept column
          df2['intercept']= 1
          #create the dummy variable column
          df2[['other_page','ab_page']] = pd.get_dummies(df2['group'])
          #dropping the other page
          df2 = df2.drop('other_page',axis=1)
          df2.head()
```

Out[115]:

	user_id	timestamp	group	landing_page	converted
0	851104	2017-01-21 22:11:48.556739	control	old_page	0
1	804228	2017-01-12 08:01:45.159739	control	old_page	0
4	864975	2017-01-21 01:52:26.210827	control	old_page	1
5	936923	2017-01-10 15:20:49.083499	control	old_page	0
7	719014	2017-01-17 01:48:29.539573	control	old_page	0

	intercept	ab_page
0	1	0
1	1	0
4	1	0
5	1	0
7	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 [116]: log_mod = sm.Logit(df2['converted'],df2[['intercept', 'ab_page']])
```

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

```
In [117]: results = log_mod.fit()
          results.summary2()
```

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

```
Out[117]: <class 'statsmodels.iolib.summary2.Summary'>
          """
                                Results: Logit
          =================================================================
          Model:              Logit            No. Iterations:   6.0000
          Dependent Variable: converted        Pseudo R-squared: 0.000
          Date:               2022-03-27 23:44 AIC:              212780.3502
          No. Observations:   290584           BIC:              212801.5095
          Df Model:           1                Log-Likelihood:   -1.0639e+05
          Df Residuals:       290582           LL-Null:          -1.0639e+05
          Converged:          1.0000           Scale:            1.0000
```

```
------------------------------------------------------------------
              Coef.    Std.Err.       z       P>|z|     [0.025   0.975]
------------------------------------------------------------------
intercept    -1.9888     0.0081  -246.6690   0.0000   -2.0046  -1.9730
ab_page      -0.0150     0.0114    -1.3109   0.1899   -0.0374   0.0074
==================================================================

"""
```

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).

> **Put your answer here.** The p-value is 0.905 for the A/B page test. This is for a one tailed test only testing for a positive effect of the new page. The regression section has a p-value of 0.1889. With the two tailed test under the regression approach section we are testing the page variable for the new and old page, not which direction it is going. This should account for the difference in p-values. While the ab_page was quite a bit higher than the 0.05 error rate. These recent results are slightly above the error rate of 0.05. This new slight difference shows that we should reject the null hypothesis. With such a large sample size I don't feel this is 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?

> **Put your answer here.** I don't think that the new landing page is much different than using the old one. It would be a good idea to add other factors into the regression model. Disadvantages are that the other factors may influence each other as well.

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

```
Out[118]:    user_id country
          0   834778      UK
          1   928468      US
          2   822059      UK
          3   711597      UK
          4   710616      UK
```

```
In [119]: # Join with the df2 dataframe
          df3 = df2.set_index('user_id').join(countries.set_index('user_id'))
          df3.head()
```

```
Out[119]:                          timestamp    group landing_page  converted  \
          user_id
          851104   2017-01-21 22:11:48.556739  control     old_page          0
          804228   2017-01-12 08:01:45.159739  control     old_page          0
          864975   2017-01-21 01:52:26.210827  control     old_page          1
          936923   2017-01-10 15:20:49.083499  control     old_page          0
          719014   2017-01-17 01:48:29.539573  control     old_page          0

                   intercept  ab_page country
          user_id
          851104           1        0      US
          804228           1        0      US
          864975           1        0      US
          936923           1        0      US
          719014           1        0      US
```

```
In [120]: # Create the necessary dummy variables
          df3[['US','UK','CA']] = pd.get_dummies(df3['country'])
          df3.head()
```

```
Out[120]:                          timestamp    group landing_page  converted  \
          user_id
          851104   2017-01-21 22:11:48.556739  control     old_page          0
          804228   2017-01-12 08:01:45.159739  control     old_page          0
          864975   2017-01-21 01:52:26.210827  control     old_page          1
          936923   2017-01-10 15:20:49.083499  control     old_page          0
          719014   2017-01-17 01:48:29.539573  control     old_page          0

                   intercept  ab_page country  US  UK  CA
          user_id
          851104           1        0      US   0   0   1
          804228           1        0      US   0   0   1
          864975           1        0      US   0   0   1
          936923           1        0      US   0   0   1
          719014           1        0      US   0   0   1
```

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 [121]:
```
# Fit your model, and summarize the results
log_mod = sm.Logit(df3['converted'],df3[['intercept','ab_page','US','CA',]])
results = log_mod.fit()
results.summary2()
```

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

Out[121]:
```
<class 'statsmodels.iolib.summary2.Summary'>
"""
                          Results: Logit
=================================================================
Model:              Logit            No. Iterations:   6.0000
Dependent Variable: converted        Pseudo R-squared: 0.000
Date:               2022-03-27 23:44 AIC:              212781.1253
No. Observations:   290584           BIC:              212823.4439
Df Model:           3                Log-Likelihood:   -1.0639e+05
Df Residuals:       290580           LL-Null:          -1.0639e+05
Converged:          1.0000           Scale:            1.0000
-----------------------------------------------------------------
               Coef.   Std.Err.      z      P>|z|    [0.025   0.975]
-----------------------------------------------------------------
intercept    -1.9794    0.0127  -155.4145  0.0000  -2.0044  -1.9544
ab_page      -0.0149    0.0114    -1.3069  0.1912  -0.0374   0.0075
US           -0.0506    0.0284    -1.7835  0.0745  -0.1063   0.0050
CA           -0.0099    0.0133    -0.7433  0.4573  -0.0359   0.0162
=================================================================

"""
```

In [122]:
```
#Creating dummy variables for an interaction between country and page on conversion
df3['UKab'] = df3['UK']*df3['ab_page']
df3['USab'] = df3['US']*df3['ab_page']
df3['CAab'] = df3['CA']*df3['ab_page']
df3.head()
```

Out[122]:

user_id	timestamp	group	landing_page	converted	intercept	ab_page	country	US	UK	CA	UKab	USab	CAab
851104	2017-01-21 22:11:48.556739	control	old_page	0	1	0	US	0	0	1	0	0	0
804228	2017-01-12 08:01:45.159739	control	old_page	0	1	0	US	0	0	1	0	0	0
864975	2017-01-21 01:52:26.210827	control	old_page	1	1	0	US	0	0	1	0	0	0
936923	2017-01-10 15:20:49.083499	control	old_page	0	1	0	US	0	0	1	0	0	0
719014	2017-01-17 01:48:29.539573	control	old_page	0	1	0	US	0	0	1	0	0	0

In [123]:
```
#Fitting model to new dummy variables
log_mod = sm.Logit(df3['converted'],df3[['intercept','ab_page','US','CA','USab','CAab'
results = log_mod.fit()
results.summary2()
```

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

Out[123]:

```
<class 'statsmodels.iolib.summary2.Summary'>
"""
                          Results: Logit
=================================================================
Model:              Logit            No. Iterations:   6.0000
Dependent Variable: converted        Pseudo R-squared: 0.000
Date:               2022-03-27 23:44 AIC:              212782.6602
No. Observations:   290584           BIC:              212846.1381
Df Model:           5                Log-Likelihood:   -1.0639e+05
Df Residuals:       290578           LL-Null:          -1.0639e+05
Converged:          1.0000           Scale:            1.0000
-----------------------------------------------------------------
               Coef.   Std.Err.      z      P>|z|    [0.025   0.975]
-----------------------------------------------------------------
intercept    -1.9922    0.0161  -123.4571  0.0000  -2.0238  -1.9606
ab_page       0.0108    0.0228     0.4749  0.6349  -0.0339   0.0555
US           -0.0118    0.0398    -0.2957  0.7674  -0.0899   0.0663
CA            0.0057    0.0188     0.3057  0.7598  -0.0311   0.0426
USab         -0.0783    0.0568    -1.3783  0.1681  -0.1896   0.0330
CAab         -0.0314    0.0266    -1.1807  0.2377  -0.0835   0.0207
=================================================================

"""
```

Put your conclusion answer here. It does seem that the p-value for CA is above the error rate of 0.05. I would say that this country may have an effect on conversion rate but probably not a large one with a p value of 0.4558. Even after using the conversion dummy variables the p value is only 0.1023. I don't think that with such a large amount of data that this is statistically significant. Due to this we should fail to reject the null hypothesis. It seems that moving to a new page may not provide much of a benefit.

Final Check!

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

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

Submission You may either submit your notebook through the "SUBMIT PROJECT" button at the bottom of this workspace, or you may work from your local machine and submit on the last page of this project lesson.

1. Before you submit your project, you need to create a .html or .pdf version of this notebook in the workspace here. To do that, run the code cell below. If it worked correctly, you should get a return code of 0, and you should see the generated .html file in the workspace directory (click on the orange Jupyter icon in the upper left).

2. Alternatively, you can download this report as .html via the **File** > **Download as** submenu, and then manually upload it into the workspace directory by clicking on the orange Jupyter icon in the upper left, then using the Upload button.

3. Once you've done this, you can submit your project by clicking on the "Submit Project" button in the lower right here. This will create and submit a zip file with this .ipynb doc and the .html or .pdf version you created. Congratulations!

```
In [124]: from subprocess import call
          call(['python', '-m', 'nbconvert', 'Analyze_ab_test_results_notebook.ipynb'])
```

```
Out[124]: 0
```

```
In [ ]:
```