Analyze_ab_test_results_notebook

January 12, 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 [2]: 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.
a. Read in the dataset from the `ab_data.csv` file and take a look at the top few rows here:

```
In [41]: df = pd.read_csv('ab_data.csv')
         df.head()
```

```
Out[41]:    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 [4]: df.shape[0]
```

```
Out[4]: 294478
```

c. The number of unique users in the dataset.

```
In [5]: df.user_id.nunique()
```

```
Out[5]: 290584
```

d. The proportion of users converted.

```
In [6]: df.converted.mean()
```

```
Out[6]: 0.11965919355605512
```

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

```
In [7]: df[(df['group'] == 'treatment') & (df['landing_page'] != 'new_page')].shape[0] +df[(df['
```

```
Out[7]: 3893
```

f. Do any of the rows have missing values?

```
In [8]: df.isnull().count()
```

```
Out[8]: user_id         294478
        timestamp       294478
        group           294478
        landing_page    294478
        converted       294478
        dtype: int64
```

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 [42]: # Remove the inaccurate rows, and store the result in a new dataframe df2
         # or ((df['group'] == 'control') & (df['landing_page'] == 'old_page'))
         df2 = df.loc[(df['group'] == 'treatment') & (df['landing_page'] == 'new_page')].append(
```

```
In [43]: # 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[43]: 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 [11]: df2.user_id.nunique()
```

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

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

```
In [12]: df2[(df2.user_id.duplicated())==True]
```

```
Out[12]:       user_id                   timestamp      group landing_page  converted
         2893   773192  2017-01-14 02:55:59.590927  treatment     new_page          0
```

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

```
In [32]: df2[(df2.user_id.duplicated())==True]
```

```
Out[32]:       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 [44]: # 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
         df2.drop_duplicates(subset=['user_id'], keep='first', inplace=True)
         # Check again if the row with a duplicate user_id is deleted or not
         df2[(df2.user_id.duplicated())==True]
```

```
Out[44]: Empty DataFrame
         Columns: [user_id, timestamp, group, landing_page, converted]
         Index: []
```

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?

```
In [14]: df2.converted.mean()
```

```
Out[14]: 0.11959708724499628
```

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

```
In [8]: control_cr = (df2.groupby('group')['converted']).mean()[0]
```

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

```
In [9]: experiment_cr = (df2.groupby('group')['converted']).mean()[1]
```

```
In [10]: # Calculate the actual difference (obs_diff) between the conversion rates for the two g
         obs_diff = experiment_cr - control_cr
         obs_diff
```

```
Out[10]: -0.0015782389853555567
```

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

```
In [18]: df2.query('landing_page=="new_page"').count()[0] / df2.shape[0]
```

```
Out[18]: 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.

> **Since the probability of receiving a new page is equal to the probability of receiving an old page, and the observed difference in conversion rate is negative, then statistically the old page leads to more conversion rate, however practically the difference is less than 1% which means that it is undefinitive that either pages lead to more 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.

> **0: - >= 0 1: - < 0**

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 [63]: p_new = df2.converted.mean()
         p_new
```

```
Out[63]: 0.11959708724499628
```

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

```
In [64]: p_old = df2.converted.mean()
         p_old
```

```
Out[64]: 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]: n_new = df2.query('group=="treatment"').count()[0]
         n_new

Out[21]: 145310
```

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

```
In [22]: n_old = df2.query('group=="control"').count()[0]
         n_old

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.

```
In [72]: # Simulate a Sample for the treatment Group
         new_page_df = df2.query('group=="treatment"')
         new_page_converted = np.random.choice(new_page_df.converted, n_new)
         p_sample_new = new_page_converted.mean()
         p_sample_new

Out[72]: 0.11832633679719221
```

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 [73]: # Simulate a Sample for the control Group
         old_page_df = df2.query('group=="control"')
         old_page_converted = np.random.choice(old_page_df.converted, n_old)
         p_sample_old = old_page_converted.mean()
         p_sample_old

Out[73]: 0.11991133995071382
```

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

```
In [74]: sample_diff = p_sample_new - p_sample_old
         sample_diff

Out[74]: -0.0015850031535216136
```

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`.

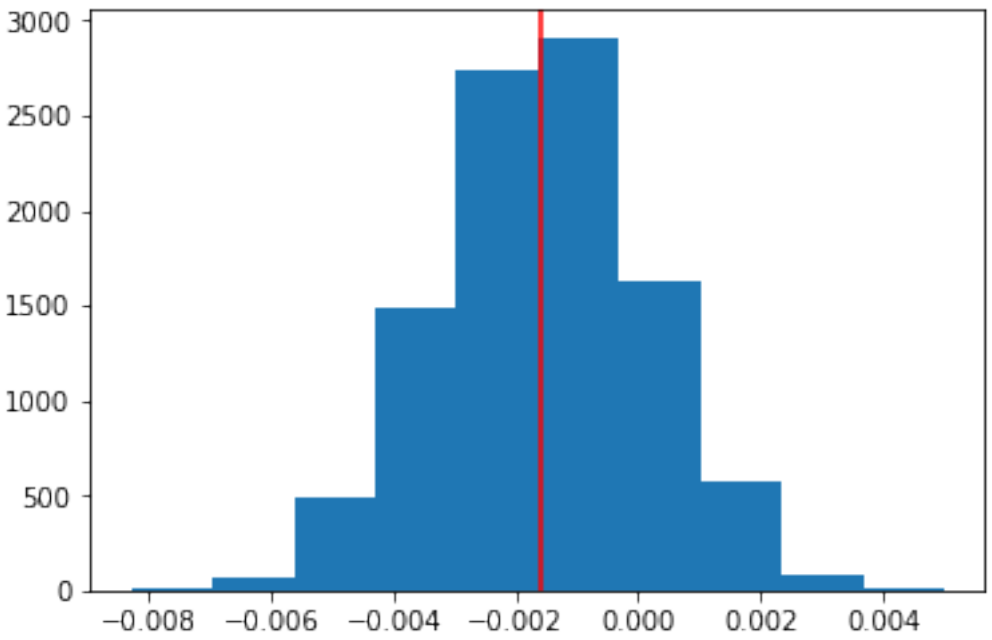
```
In [69]: # Sampling distribution
         p_diffs = []
         for _ in range(10000):
             b_sample = df2.sample(df2.shape[0], replace=True)
             new_page_df = b_sample.query('group=="treatment"')
             old_page_df = b_sample.query('group=="control"')
             new_page_converted = np.random.choice(new_page_df.converted, n_new)
             old_page_converted = np.random.choice(old_page_df.converted, n_old)
             p_sample_new = new_page_converted.mean()
             p_sample_old = old_page_converted.mean()
             p_diffs.append(p_sample_new - p_sample_old)
```

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

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

```
In [77]: p_diffs = np.array(p_diffs)
         plt.hist(p_diffs)
         plt.axvline(x=obs_diff,color='red')

Out[77]: <matplotlib.lines.Line2D at 0x7f450e433400>
```



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

```
In [78]: (p_diffs>obs_diff).mean()
```

```
Out[78]: 0.50590000000000002
```

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)".

> **- What we calculated in part j. above is the mean of the area under the sampling distribution where the difference in proportions of conversion rate is more than the observed difference of the population - The value is called P-Value - This value signifies that there isn't much difference between the old page and the new page - The P-value here (0.51) is large compared to Type I error rate (0.05) which signifies that we cannot reject the null hypothesis: which is the old page has higher or equal conversion rate compared to the new 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 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 [11]: import statsmodels.api as sm

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

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

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

         # number of individuals who received new_page
         n_new = df2.query('group=="treatment"').count()[0]
```

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.

```
In [28]: 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],a
         print(z_score, p_value)

-0.00328757967535 0.50131155217
```

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

- Z_{score} here means that the mean of conversion rate difference between the two groups is -0.3%. - So the Z_{score} is < Z_{α} (which is 1.645 for right-tailed test as in our case). - And for a right-tailed tests if Z_{score} is < Z_{α} then we don't reject the null hypothesis which is the old page is better or same as new page in terms of conversion rate. - From parts j. and k. we also found that we don't reject the null hypothesis. - Also here the p-value we calculated (0.51) is the same as the one we calculated in part j. which is also more than Type-I error (0.05) which also leads to that we don't reject the null hypothesis.

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?

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 [45]: import statsmodels.api as sm
         df2['intercept'] = 1
         df2[['ab_page0', 'ab_page']] = pd.get_dummies(df2['group'])
         df2.head()

Out[45]:    user_id                   timestamp      group landing_page  converted  \
         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
         6   679687  2017-01-19 03:26:46.940749  treatment     new_page          1
         8   817355  2017-01-04 17:58:08.979471  treatment     new_page          1
         9   839785  2017-01-15 18:11:06.610965  treatment     new_page          1
```

```
   intercept  ab_page0  ab_page
2          1         0        1
3          1         0        1
6          1         0        1
8          1         0        1
9          1         0        1
```

```
In [46]: df2 = df2.drop('ab_page0', axis=1)
         df2.head()
```

```
Out[46]:    user_id                   timestamp      group landing_page  converted  \
         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
         6   679687  2017-01-19 03:26:46.940749  treatment     new_page          1
         8   817355  2017-01-04 17:58:08.979471  treatment     new_page          1
         9   839785  2017-01-15 18:11:06.610965  treatment     new_page          1

            intercept  ab_page
         2          1        1
         3          1        1
         6          1        1
         8          1        1
         9          1        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 [9]: log_mod = sm.Logit(df2['converted'], df2[['intercept', 'ab_page']])
        results = log_mod.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 [11]: results.summary2()
```

```
Out[11]: <class 'statsmodels.iolib.summary2.Summary'>
         """
                                 Results: Logit
         =================================================================
         Model:              Logit            No. Iterations:   6.0000
         Dependent Variable: converted        Pseudo R-squared: 0.000
         Date:               2022-01-12 20:34 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**?

- In part II; the **p-value** tests H_0 & H_1 according to the **difference** in conversion rates between the old and new pages which means that the hypotheses are **two-sided**.
- In Logistic Regression Model The **p-value** associated with ab_page tests the H_0 & H_1 according to the coeffecient of **only one predictor** (ab_page or treatment page here) whether it is Zero or Non-Zero which means that the hypotheses are **one-sided**. The p-value here tests:
- 1- H_0 : The coeffecient of the predictor is equal to zero; which means that the predictor has no effect in predicting the response (high p-value or p-value > alpha)
- 2- H_1 : The coeffecient of the predictor is not equal to zero; which means that the predictor is statistically significant in predicting the response (low p-value or p-value < alpha)
- According to the above, the hypotheses H_0 & H_1 in part II is different to logistics regression model (two-sided and one-sided) **That's why the p-values are different**
- In our Logistic Regression Model; The p-value = 0.1899 which is > Type I error rate (0.05).
- This indicates that **We cannot reject the null hypothesis** which states that the predictor has no effect on the response. i.e; The new page (treatment page) has no effect in increasing the conversion rate.

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

> **Adding relevant variables to the regression model will improve predicting because it prevents bias in one predictor and help us find new relationships between added predictors and the response. However, if the added predictors already have relationship with the first predictor or among themselves, this could cause an issue (multicollinearity), and may lead to unreliable results from our model or cause direction shifting.**

g. Adding countries Now along with testing if the conversion rate changes for different pages, also add an effect based on which country a user lives in.

1. You will need to read in the **countries.csv** dataset and merge together your `df2` datasets on the appropriate rows. You call the resulting dataframe `df_merged`. Here are the docs for joining tables.

2. Does it appear that country had an impact on conversion? To answer this question, consider the three unique values, `['UK', 'US', 'CA']`, in the `country` column. Create dummy variables for these country columns.

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

```
In [47]: # Read the countries.csv
         countries_df = pd.read_csv('countries.csv')
         countries_df.head()
```

```
Out[47]:    user_id country
         0   834778      UK
         1   928468      US
         2   822059      UK
         3   711597      UK
         4   710616      UK
```

```
In [48]: # Join with the df2 dataframe
         df_merged = df2.merge(countries_df, on = 'user_id')
         df_merged.head()
```

```
Out[48]:    user_id                   timestamp      group landing_page  converted  \
         0   661590  2017-01-11 16:55:06.154213  treatment     new_page          0
         1   853541  2017-01-08 18:28:03.143765  treatment     new_page          0
         2   679687  2017-01-19 03:26:46.940749  treatment     new_page          1
         3   817355  2017-01-04 17:58:08.979471  treatment     new_page          1
         4   839785  2017-01-15 18:11:06.610965  treatment     new_page          1

            intercept  ab_page country
         0          1        1      US
         1          1        1      US
         2          1        1      CA
         3          1        1      UK
         4          1        1      CA
```

```
In [49]: # Create the necessary dummy variables
         df_dummies = pd.get_dummies(df_merged['country'])
```

```
In [50]: df_merged = df_merged.join(df_dummies)
         df_merged.head()
```

```
Out[50]:    user_id                   timestamp      group landing_page  converted  \
         0   661590  2017-01-11 16:55:06.154213  treatment     new_page          0
         1   853541  2017-01-08 18:28:03.143765  treatment     new_page          0
         2   679687  2017-01-19 03:26:46.940749  treatment     new_page          1
         3   817355  2017-01-04 17:58:08.979471  treatment     new_page          1
         4   839785  2017-01-15 18:11:06.610965  treatment     new_page          1

            intercept  ab_page country  CA  UK  US
```

```
0          1        1      US   0   0   1
1          1        1      US   0   0   1
2          1        1      CA   1   0   0
3          1        1      UK   0   1   0
4          1        1      CA   1   0   0
```

```
In [53]: log_mod = sm.Logit(df_merged['converted'], df_merged[['intercept', 'CA', 'UK']])
         results = log_mod.fit()
         results.summary2()

Optimization terminated successfully.
         Current function value: 0.366116
         Iterations 6


Out[53]: <class 'statsmodels.iolib.summary2.Summary'>
         """
                                  Results: Logit
         =================================================================
         Model:              Logit            No. Iterations:   6.0000
         Dependent Variable: converted        Pseudo R-squared: 0.000
         Date:               2022-01-12 21:34 AIC:              212780.8333
         No. Observations:   290584           BIC:              212812.5723
         Df Model:           2                Log-Likelihood:   -1.0639e+05
         Df Residuals:       290581           LL-Null:          -1.0639e+05
         Converged:          1.0000           Scale:            1.0000
         -----------------------------------------------------------------
                      Coef.   Std.Err.      z      P>|z|    [0.025   0.975]
         -----------------------------------------------------------------
         intercept    -1.9967    0.0068  -292.3145  0.0000  -2.0101  -1.9833
         CA           -0.0408    0.0269    -1.5178  0.1291  -0.0935   0.0119
         UK            0.0099    0.0133     0.7458  0.4558  -0.0161   0.0360
         =================================================================

         """
```

- **Here we found that the p-value for all countries is more than Type I error rate (0.05) which means that countries have no effect on predicting the conversion rate**

h. Fit your model and obtain the results Though you have now looked at the individual factors of country and page on conversion, we would now like to look at an interaction between page and country to see if are there significant effects on conversion. **Create the necessary additional columns, and fit the new model.**

Provide the summary results (statistical output), and your conclusions (written response) based on the results.

```
In [57]: # Fit your model, and summarize the results
         log_mod = sm.Logit(df_merged['converted'], df_merged[['intercept', 'ab_page', 'CA', 'UK
```

```
results = log_mod.fit()
results.summary2()
```

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

Out[57]:
```
<class 'statsmodels.iolib.summary2.Summary'>
"""
                          Results: Logit
=================================================================
Model:              Logit            No. Iterations:   6.0000
Dependent Variable: converted        Pseudo R-squared: 0.000
Date:               2022-01-12 22:25 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    -1.9893    0.0089  -223.7628  0.0000  -2.0067  -1.9718
ab_page      -0.0149    0.0114    -1.3069  0.1912  -0.0374   0.0075
CA           -0.0408    0.0269    -1.5161  0.1295  -0.0934   0.0119
UK            0.0099    0.0133     0.7433  0.4573  -0.0162   0.0359
=================================================================

"""
```

- From the Logistic Regression Model here, we found that the p-value for all the predictors is more than Type I error rate (0.05) which means that **we cannot reject the null hypothesis** for all the predictors (H_0 : The coeffecient of the predictor is equal to zero)
- Accordingly this means that none of the predictors here (page / countries) has signifcant effect on the prediction of the response (conversion rate) or it means that they have no effect on the predictor
- Practically, this means that implementing the new page design will not have a significant effect in increasing the conversion rate, so this will be wasting time and money. Either we keep using the old page or implement a new different design and test it again against increasing the conversion rate.

Final Check!

Congratulations! You have reached the end of the A/B Test Results project! You should be very proud of all you have accomplished! ## Submission You may either submit your notebook through the "SUBMIT PROJECT" button at the bottom of this workspace, or you may work from your local machine and submit on the last page of this project lesson.

1. Before you submit your project, you need to create a .html or .pdf version of this notebook in the workspace here. To do that, run the code cell below. If it worked correctly, you should

get a return code of 0, and you should see the generated .html file in the workspace directory (click on the orange Jupyter icon in the upper left).

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

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

```
In [60]: from subprocess import call
         call(['python', '-m', 'nbconvert', 'Analyze_ab_test_results_notebook.ipynb'])
```

```
Out[60]: 0
```

```
In [ ]:
```