Analyze_ab_test_results_notebook

March 20, 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 [1]: import pandas as pd
        import numpy as np
        import random
        import matplotlib.pyplot as plt
        %matplotlib inline
        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 [2]: df = pd.read_csv('ab_data.csv')
        df.head()
```

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

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

```
In [3]: len(df)
```

```
Out[3]: 294478
```

c. The number of unique users in the dataset.

```
In [4]: len(df['user_id'].unique())
```

```
Out[4]: 290584
```

d. The proportion of users converted.

```
In [5]: sum(df['converted'])/len(df['user_id'].unique())
```

```
Out[5]: 0.12126269856564711
```

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

```
In [6]: len(df.query("group == 'treatment' and landing_page != 'new_page'"))
```

```
Out[6]: 1965
```

f. Do any of the rows have missing values?

```
In [7]: df.isna().sum()
```

```
Out[7]: user_id         0
        timestamp       0
        group           0
        landing_page    0
        converted       0
        dtype: int64
```

1.0.2 It is found that there is no missing values

1.0.3 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]: df2 = df[((df['group'] == 'control') & (df['landing_page'] == 'old_page')) | ((df['group
```

```
In [9]: df2[((df2['group'] == 'treatment') == (df2['landing_page'] == 'new_page')) == False].sha
```

```
Out[9]: 0
```

1.0.4 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]: len(df2)
```

```
Out[10]: 290585
```

```
In [11]: len(df2['user_id'].unique())
```

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

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

```
In [12]: df2.groupby(['user_id']).count().sort_values('converted', ascending = False).head(1)
```

```
Out[12]:          timestamp  group  landing_page  converted
         user_id
         773192           2      2             2          2
```

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

```
In [13]: df2[df2['user_id'] == 773192]
```

```
Out[13]:       user_id                   timestamp      group landing_page  converted
         1899   773192  2017-01-09 05:37:58.781806  treatment     new_page          0
         2893   773192  2017-01-14 02:55:59.590927  treatment     new_page          0
```

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

```
In [14]: df2 = df2.drop(2893)
         len(df2)
```

```
Out[14]: 290584
```

1.0.5 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 [15]: p_pop = df2['converted'].sum()/len(df2)
         p_pop
```

```
Out[15]: 0.11959708724499628
```

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

```
In [16]: conv_control = df2[df2['group']=='control']['converted'].sum()/len(df2[df2['group']=='c
         conv_control
```

```
Out[16]: 0.1203863045004612
```

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

```
In [17]: conv_treatment = df2[df2['group']=='treatment']['converted'].sum()/len(df2[df2['group']
         conv_treatment
```

```
Out[17]: 0.11880806551510564
```

```
In [18]: obs_diff = conv_treatment - conv_control
         obs_diff
```

```
Out[18]: -0.0015782389853555567
```

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

```
In [19]: len(df2[df2['group']=='treatment'])/len(df2)
```

```
Out[19]: 0.5000619442226688
```

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

1.0.6 Answer

#### The actual difference is showing a negative value, slightly disfavoring the new page. However, a solid conclusion should only be made after performing the A/B test.

## 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.

1.0.7 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.

1.0.8 Answer

The null hypothesis is that the conversion rate of the old page is higher or equal to that of the new page

$H_0 : p_{old} >= p_{new}$

The alternative hypothesis is that the conversion rate of the old page is lower than that of the new page

$H_1 : p_{old} < p_{new}$

1.0.9 ToDo 2.2 - Null Hypothesis H_0 Testing

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

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

In this section, you will:

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

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

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

```
In [20]: p_new = p_pop
         p_new
```

```
Out[20]: 0.11959708724499628
```

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

```
In [21]: p_old = p_pop
         p_pop
```

```
Out[21]: 0.11959708724499628
```

```
In [22]: p_new - p_old
```

```
Out[22]: 0.0
```

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

```
In [23]: n_new = len(df2[df2['group'] == 'treatment'])
         n_new
```

```
Out[23]: 145310
```

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

```
In [24]: n_old = len(df2[df2['group'] == 'control'])
         n_old
```

```
Out[24]: 145274
```

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

```
In [25]: new_page_converted = np.random.choice([0,1],n_new,p = (p_new,(1-p_new)))
         new_page_converted
```

```
Out[25]: array([1, 1, 1, ..., 1, 1, 1])
```

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 [26]: old_page_converted = np.random.choice([0,1],n_old,p = (p_old,(1-p_old)))
         old_page_converted
```

```
Out[26]: array([1, 1, 1, ..., 1, 1, 1])
```

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

```
In [27]: p2_new = new_page_converted.mean()
         p2_new
```

```
Out[27]: 0.87909985548138458
```

```
In [28]: p2_old = old_page_converted.mean()
         p2_old
```

```
Out[28]: 0.88044660434764654
```

```
In [29]: simulated_diff = p2_new - p2_old
         simulated_diff
```

```
Out[29]: -0.0013467488662619598
```

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

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

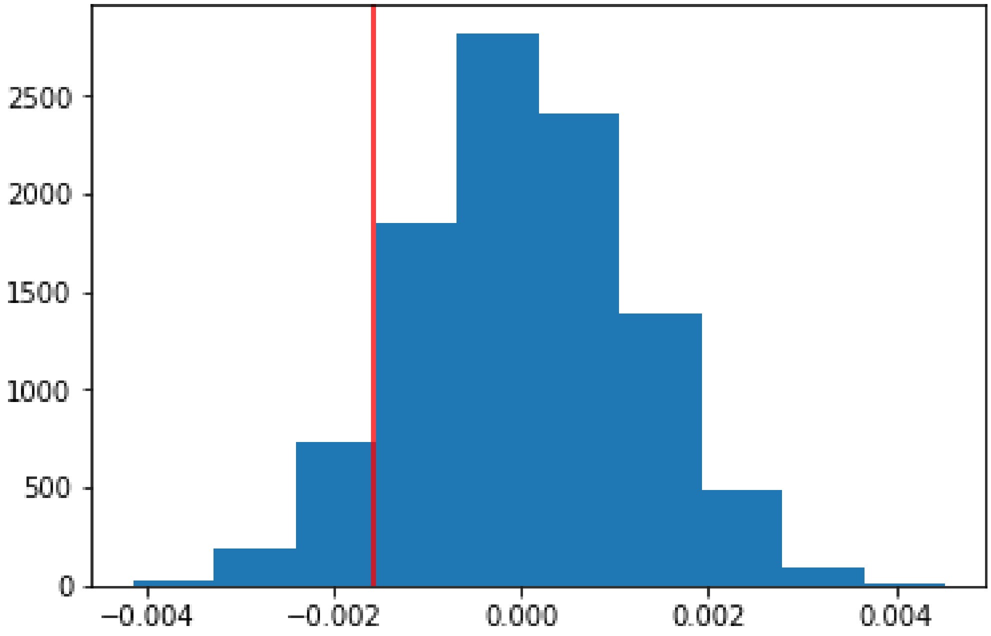
```
In [30]: p_diffs = []
         new_converted_simulation = np.random.binomial(n_new, p_new, 10000)/n_new
         old_converted_simulation = np.random.binomial(n_old, p_old, 10000)/n_old
         p_diffs = new_converted_simulation - old_converted_simulation
```

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 [31]: plt.hist(p_diffs)
         plt.axvline(obs_diff, color = 'r')
```

```
Out[31]: <matplotlib.lines.Line2D at 0x7fbdd33385f8>
```



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

```
In [32]: (p_diffs > obs_diff).mean()
```

```
Out[32]: 0.9093999999999999
```

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)".

1.0.10 Answer:

The value computed is the p-value.

It represents the the probability of obtaining test results, under assumption that null hypothesis is correct.

From the value found, it can be interpretted that more than 90% of the test results support the null hypothesis.

Besides, it is much above the threshold of the acceptable Type I error rate, which is only 5%

In other words, the results fail to reject the null hypothesis.

This concludes that the new page fails to outperform the old page in terms of conversion rate.

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 [33]: df2.head()
```

```
Out[33]:    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
```

```
In [34]: import statsmodels.api as sm

         convert_old = df2[df2['landing_page']== 'old_page']['converted'].sum()

         convert_new = df2[df2['landing_page']== 'new_page']['converted'].sum()

         n_old = len(df2[df2['landing_page']== 'old_page'])

         n_new = len(df2[df2['landing_page']== 'new_page'])

         convert_old, convert_new, n_old, n_new
```

```
/opt/conda/lib/python3.6/site-packages/statsmodels/compat/pandas.py:56: FutureWarning: The panda
  from pandas.core import datetools
```

```
Out[34]: (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.

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

```
In [35]: import statsmodels.api as sm
         z_score = sm.stats.proportions_ztest([convert_new, convert_old],[n_new, n_old], alterna
         z_score
```

```
Out[35]: (-1.3109241984234394, 0.90505831275902449)
```

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.**?

1.0.11 Answer

While the p-value focuses on the test observation and whether they are the same or exterme when null values applies, the z-test focuses on how far our observations from the null hypothesis and whether it should be rejected. However the conclusion is similar. The p-value from the two-sample z-test is equal to that calculated from the A/B test. The p-value of 0.905 means that 90.5% of the observations where similar to the values of the null hypothesis; thus failed to reject it. the z-score of -1.31 means that the standard deviation of our observations fails to exceed the critical value -1.645 and stays within the borders of the standard deviation of the null hypothesis, thus the null hypothesis is accepted.

### Part III - A regression approach

1.0.12 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?

As we attempt to predict categorial data, logistic regression is used to predicts probability between 0 and 1; however, if other quantitative data is included as well, multiple lineaer regression will be needed

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 [36]: df2['intercept'] = 1
         df2['ab_page'] = pd.get_dummies(df2['group'])['treatment']
         df2.head()

Out[36]:    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

            intercept  ab_page
         0          1        0
         1          1        0
         2          1        1
         3          1        1
         4          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 [37]: model = sm.Logit(df2['converted'],df2[['intercept','ab_page']])
         result = model.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 [38]: result.summary2()

Out[38]: <class 'statsmodels.iolib.summary2.Summary'>
         """
                                   Results: Logit
         ==================================================================
         Model:              Logit            No. Iterations:   6.0000
         Dependent Variable: converted        Pseudo R-squared: 0.000
         Date:               2022-02-19 08:10 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**?

1.0.13 Answer

The p-value associated with ab_page = 0.1899. It differs from the p-value of the prevoius A/B test because different hypothesis is being tested.

In the A/B test:
$H_0 : p_{old} >= p_{new}$
$H_1 : p_{old} < p_{new}$
In the regression approach:
$H_0 : p_{old} = p_{new}$
$H_1 : p_{old} \quad p_{new}$
if we tried to exam the ($H_1 : p_{old} \quad p_{new}$)using the A/B test method, it would end up with the same p-value as shown below:

```
In [39]: null_mean = 0
         (p_diffs < obs_diff).mean() + (p_diffs > null_mean+(null_mean-obs_diff)).mean()

Out[39]: 0.1958
```

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?

1.0.14 Answer

Considering other factors is definitely useful to create a more predictive model for our target dependent variable. Introducing more relevant independent variables would improve the R-squared fo our model. However, the downside of this approach is the multicollinearity where the independent variable should be correlated with the dependent variable, not with one another. To avoid this issue, we should check the presence of flipped correlation coefficient or check that the VIFs of our variables are below 10.

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 [40]: df3 = pd.read_csv('countries.csv')
         df3.head()
```

```
Out[40]:    user_id country
         0   834778      UK
         1   928468      US
         2   822059      UK
         3   711597      UK
         4   710616      UK
```

```
In [41]: df_merged = df2.join(df3.set_index('user_id'), on='user_id')
         df_merged.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

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

```
In [42]: df_merged[['CA', 'UK', 'US']] = pd.get_dummies(df_merged['country'])
         df_merged.head()
```

```
Out[42]:    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

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

```
In [43]: model = sm.Logit(df_merged['converted'], df_merged[['intercept','CA','US']])
         result = model.fit()
         result.summary2()
```

```
Optimization terminated successfully.
         Current function value: 0.366116
         Iterations 6
```

```
Out[43]: <class 'statsmodels.iolib.summary2.Summary'>
         """
                                  Results: Logit
         ==================================================================
         Model:              Logit            No. Iterations:   6.0000
         Dependent Variable: converted        Pseudo R-squared: 0.000
         Date:               2022-02-19 08:10 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    -1.9868    0.0114  -174.1736  0.0000  -2.0092  -1.9645
         CA           -0.0507    0.0284    -1.7863  0.0740  -0.1064   0.0049
         US           -0.0099    0.0133    -0.7458  0.4558  -0.0360   0.0161
         ==================================================================

         """
```

1.0.15 Comment:

Out of the three countries, only UK had a significant p-value. This outcome may predict that the UK users are showing more conversion rate potential; but using the country as the only factor may be useless as we can't tell which page causes more conversion rate. We may need to merge between the two inependent variables(country and page) to get a meaningful result. To do so, adding an interaction between these two independent variables.

```
In [44]: df_merged['CA_converted'] = df_merged['ab_page']*df_merged['CA']
         df_merged['UK_converted'] = df_merged['ab_page']*df_merged['UK']
         df_merged['US_converted'] = df_merged['ab_page']*df_merged['US']
         df_merged.head()
```

```
Out[44]:    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
```

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

   US_converted
0             0
1             0
2             1
3             1
4             0
```

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

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

```
In [45]: model = sm.Logit(df_merged['converted'], df_merged[['intercept','ab_page','CA','UK','CA
         result = model.fit()
         result.summary2()
```

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

```
Out[45]: <class 'statsmodels.iolib.summary2.Summary'>
         """
                                  Results: Logit
         ==================================================================
         Model:              Logit            No. Iterations:   6.0000
         Dependent Variable: converted        Pseudo R-squared: 0.000
         Date:               2022-02-19 08:10 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        -1.9865   0.0096 -206.3440 0.0000 -2.0053 -1.9676
         ab_page          -0.0206   0.0137   -1.5052 0.1323 -0.0473  0.0062
         CA               -0.0175   0.0377   -0.4652 0.6418 -0.0914  0.0563
         UK               -0.0057   0.0188   -0.3057 0.7598 -0.0426  0.0311
```

```
CA_converted     -0.0469    0.0538    -0.8718 0.3833 -0.1523  0.0585
UK_converted      0.0314    0.0266     1.1807 0.2377 -0.0207  0.0835
==================================================================

"""
```

1.0.16 Conclusion:

From the above regression, when checking the effect of the new page in each country, the corresponding p-vales are high enough to fail in rejecting the null hypothesis. This confirms the conclusions established by the A/B tes, the regression method and the two-sample z-test regardless of the user location.

However; before deciding to unlaunch the new page, it is important to note that the data is collected through a period of 22 days, which is relatively not long enough to avoid the novelty effect or change aversion bias.

The recommended course of action would be to collect more data regarding the conversion rate of both pages through a longer timeframe to double check the result. Another approach to avoid unwanted bias is to make sure that the new page is not viewed by current users who are familiar with the old page (This can be done through ip address filter for example).

```
In [46]: df_merged['timestamp']= pd.to_datetime(df_merged['timestamp'])
         df_merged['timestamp'].max() - df_merged['timestamp'].min()

Out[46]: Timedelta('21 days 23:59:49.081927')
```

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

Out[47]: 0
```