Analyze_ab_test_results_notebook

October 14, 2020

0.1 Analyze A/B Test Results

You may either submit your notebook through the workspace here, or you may work from your local machine and submit through the next page. Either way assure that your code passes the project RUBRIC. **Please save regularly.**

This project will assure you have mastered the subjects covered in the statistics lessons. The hope is to have this project be as comprehensive of these topics as possible. Good luck!

0.2 Table of Contents

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

Introduction

A/B tests are very commonly performed by data analysts and data scientists. It is important that you get some practice working with the difficulties of these

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 page, keep the old page, or perhaps run the experiment longer to make their decision.

As you work through this notebook, follow along in the classroom and answer the corresponding quiz questions associated with each question. The labels for each classroom concept are provided for each question. This will assure you are on the right track as you work through the project, and you can feel more confident in your final submission meeting the criteria. As a final check, assure you meet all the criteria on the RUBRIC.

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. Now, read in the ab_data.csv data. Store it in df. **Use your dataframe to answer the questions in Quiz 1 of the classroom.**

a. Read in the dataset and take a look at the top few rows here:

```
In [3]: df = pd.read_csv('ab_data.csv')
        df.head

Out[3]: <bound method NDFrame.head of          user_id                   timestamp      group lan
        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
        5        936923  2017-01-10 15:20:49.083499    control     old_page          0
        6        679687  2017-01-19 03:26:46.940749  treatment     new_page          1
        7        719014  2017-01-17 01:48:29.539573    control     old_page          0
        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
        10       929503  2017-01-18 05:37:11.527370  treatment     new_page          0
        11       834487  2017-01-21 22:37:47.774891  treatment     new_page          0
        12       803683  2017-01-09 06:05:16.222706  treatment     new_page          0
        13       944475  2017-01-22 01:31:09.573836  treatment     new_page          0
        14       718956  2017-01-22 11:45:11.327945  treatment     new_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
        17       888545  2017-01-08 06:37:26.332945  treatment     new_page          1
        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
        20       740805  2017-01-12 18:59:45.453277  treatment     new_page          0
        21       759875  2017-01-09 16:11:58.806110  treatment     new_page          0
        22       767017  2017-01-12 22:58:14.991443    control     new_page          0
        23       793849  2017-01-23 22:36:10.742811  treatment     new_page          0
        24       905617  2017-01-20 14:12:19.345499  treatment     new_page          0
        25       746742  2017-01-23 11:38:29.592148    control     old_page          0
        26       892356  2017-01-05 09:35:14.904865  treatment     new_page          1
        27       773302  2017-01-12 08:29:49.810594  treatment     new_page          0
        28       913579  2017-01-24 09:11:39.164256    control     old_page          1
        29       736159  2017-01-06 01:50:21.318242  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
        294451   875688  2017-01-14 07:19:49.042869    control     old_page          0
        294452   927527  2017-01-12 10:52:11.084740    control     old_page          0
        294453   789177  2017-01-17 18:17:56.215378    control     old_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
294459   924816  2017-01-20 10:59:03.481635    control     old_page          0
294460   846225  2017-01-16 15:24:46.705903  treatment     new_page          0
294461   740310  2017-01-10 17:22:19.762612    control     old_page          0
294462   677163  2017-01-03 19:41:51.902148  treatment     new_page          0
294463   832080  2017-01-19 13:18:27.352570    control     old_page          0
294464   834362  2017-01-17 01:51:56.106436    control     old_page          0
294465   925675  2017-01-07 20:38:26.346410  treatment     new_page          0
294466   923948  2017-01-09 16:33:41.104573    control     old_page          0
294467   857744  2017-01-05 08:00:56.024226    control     old_page          0
294468   643562  2017-01-02 19:20:05.460595  treatment     new_page          0
294469   755438  2017-01-18 17:35:06.149568    control     old_page          0
294470   908354  2017-01-11 02:42:21.195145    control     old_page          0
294471   718310  2017-01-21 22:44:20.378320    control     old_page          0
294472   822004  2017-01-04 03:36:46.071379  treatment     new_page          0
294473   751197  2017-01-03 22:28:38.630509    control     old_page          0
294474   945152  2017-01-12 00:51:57.078372    control     old_page          0
294475   734608  2017-01-22 11:45:03.439544    control     old_page          0
294476   697314  2017-01-15 01:20:28.957438    control     old_page          0
294477   715931  2017-01-16 12:40:24.467417  treatment     new_page          0

[294478 rows x 5 columns]>
```

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

```
In [5]: df.shape[0]

Out[5]: 294478
```

c. The number of unique users in the dataset.

```
In [7]: df.nunique()

Out[7]: user_id         290584
        timestamp       294478
        group                2
        landing_page         2
        converted            2
        dtype: int64
```

d. The proportion of users converted.

```
In [8]: size = df[df['converted'] == 1].shape[0]
        print(size)

35237
```

```
In [10]: proportion = (size / df.shape[0])*100
         print(proportion)
```

```
11.96591935560551
```

e. The number of times the `new_page` and `treatment` don't match.

```
In [12]: df.query('group == "treatment" and landing_page != "new_page"').shape[0] + \
         df.query('group == "control" and landing_page != "old_page"').shape[0]
```

```
Out[12]: 3893
```

f. Do any of the rows have missing values?

```
In [13]: df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 294478 entries, 0 to 294477
Data columns (total 5 columns):
user_id         294478 non-null int64
timestamp       294478 non-null object
group           294478 non-null object
landing_page    294478 non-null object
converted       294478 non-null int64
dtypes: int64(2), object(3)
memory usage: 11.2+ MB
```

2. For the rows where **treatment** does not match with **new_page** or **control** does not match with **old_page**, we cannot be sure if this row truly received the new or old page. Use **Quiz 2** in the classroom to figure out how we should handle these rows.

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 [18]: df2 = df.query("group == 'control' and landing_page == 'old_page'")
         df2 = df2.append(df.query("group == 'treatment' and landing_page == 'new_page'"))
```

```
In [19]: # Double Check all of the correct rows were removed - this should be 0
         df2[((df2['group'] == 'treatment') == (df2['landing_page'] == 'new_page')) == False].sh
```

```
Out[19]: 0
```

3. Use **df2** and the cells below to answer questions for **Quiz3** in the classroom.

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

```
In [20]:  df2.user_id.nunique()
```

```
Out[20]: 290584
```

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

```
In [21]: df2[df2['user_id'].duplicated()]['user_id']
```

```
Out[21]: 2893    773192
         Name: user_id, dtype: int64
```

c. What is the row information for the repeat **user_id**?

```
In [22]: df2[df2['user_id'].duplicated()]
```

```
Out[22]:       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**, but keep your dataframe as **df2**.

```
In [23]: df2 = df2.drop(2893)
```

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 [24]: df.converted.mean()
```

```
Out[24]: 0.11965919355605512
```

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

```
In [26]: control_prop = df2.query("group  == 'control'")['converted'].mean()
         control_prop
```

```
Out[26]: 0.1203863045004612
```

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

```
In [27]: treatment_prop = df2.query("group == 'treatment'")['converted'].mean()
         treatment_prop
```

```
Out[27]: 0.11880806551510564
```

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

```
In [28]: df2.query('landing_page == "new_page"').shape[0] /df2.shape[0]
```

```
Out[28]: 0.5000619442226688
```

e. Consider your results from parts (a) through (d) above, and explain below whether you think there is sufficient evidence to conclude that the new treatment page leads to more conversions.

No, there is no sufficient evidence to say that the new treatment page leads to more conversions.

Because the probability that users in group of treatment will convert is actually slightly less than the probability that user in the group of control will convert.

Part II - A/B Test

Notice that because of the time stamp associated with each event, you could technically run a hypothesis test continuously as each observation was observed.

However, then the hard question is 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.` For now, consider you need to make the decision just based on all the data provided. 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 your null and alternative hypotheses be? You can state your hypothesis in terms of words or in terms of p_{old} and p_{new}, which are the converted rates for the old and new pages.

Null hypotheses : * $H_0 : p_{old} >= p_{new}$ * $H_1 : p_{old} < p_{new}$

in other words : * $H_0 : p_{new} <= p_{old}$ * $H_1 : p_{new} > p_{old}$

`2.` Assume under the null hypothesis, p_{new} and p_{old} both have "true" success rates equal to the **converted** success rate regardless of page - that is p_{new} and p_{old} are equal. Furthermore, assume they are equal to the **converted** rate in **ab_data.csv** regardless of the page.

Use a sample size for each page equal to the ones in **ab_data.csv**.

Perform the sampling distribution for the difference in **converted** between the two pages over 10,000 iterations of calculating an estimate from the null.

Use the cells below to provide the necessary parts of this simulation. If this doesn't make complete sense right now, don't worry - you are going to work through the problems below to complete this problem. 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?

```
In [31]: p_new = df2['converted'].mean()
         print(p_new)
```

```
0.119597087245
```

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

```
In [33]: p_null
```

```
Out[33]: 0.11959708724499628
```

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

```
In [34]: n_new = df2.query("landing_page == 'new_page'").shape[0]
         print(n_new)
```

```
145310
```

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

```
In [35]: n_old = df2.query("landing_page == 'old_page'").shape[0]
         print(n_old)
```

```
145274
```

e. Simulate n_{new} transactions with a conversion rate of p_{new} under the null. Store these n_{new} 1's and 0's in **new_page_converted**.

```
In [ ]: new_page_converted = np.random.binomial(1, p_null, n_new)
```

f. Simulate n_{old} transactions with a conversion rate of p_{old} under the null. Store these n_{old} 1's and 0's in **old_page_converted**.

```
In [37]: old_page_converted = np.random.binomial(1, p_null, n_old)
```

g. Find p_{new} - p_{old} for your simulated values from part (e) and (f).

```
In [47]: old_page_converted.mean() - new_page_converted.mean()
```

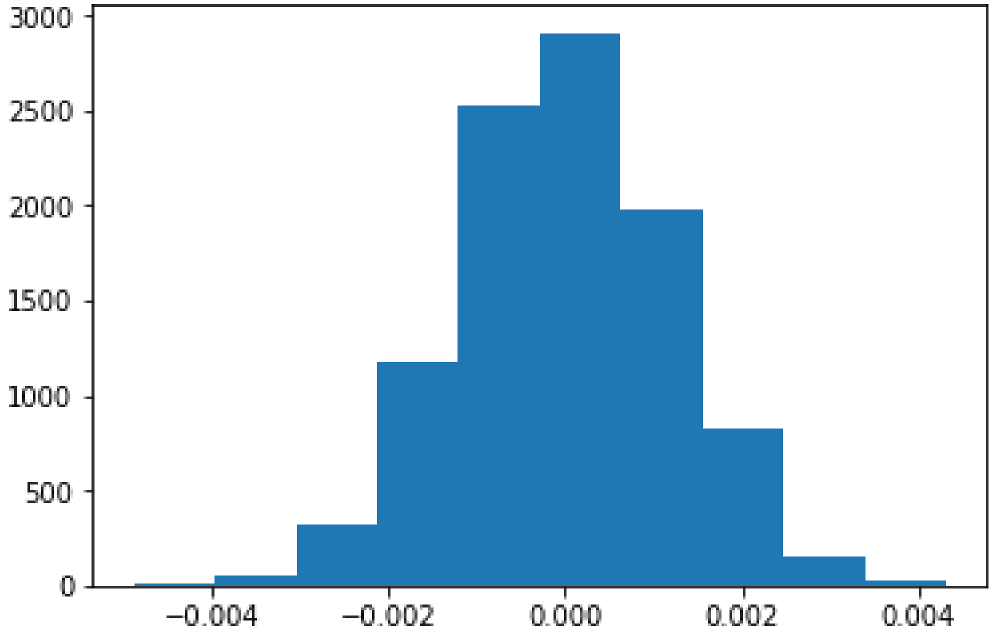
```
Out[47]: -0.00040369307873495963
```

h. Create 10,000 p_{new} - p_{old} values using the same simulation process you used in parts (a) through (g) above. Store all 10,000 values in a NumPy array called **p_diffs**.

```
In [52]: p_diffs = []
         new_converted_simulation = np.random.binomial(n_new, p_null, 10000)/n_new
         old_converted_simulation = np.random.binomial(n_old, p_null, 10000)/n_old
         p_diffs = new_converted_simulation - old_converted_simulation
```

i. 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.

```
In [53]: plt.hist(p_diffs);
```



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

```
In [55]: act_diff = df[df['group'] == 'treatment']['converted'].mean() - df[df['group'] == 'cont
         print(act_diff)

-0.00147959979408


In [56]: p_diffs - np.array(p_diffs)
         p_diffs

Out[56]: array([  7.13626516e-04,  -6.55786080e-04,  -7.76860379e-05, ...,
                  7.13740776e-04,  -5.80171122e-04,   1.31940740e-03])

In [57]: (act_diff < p_diffs).mean()

Out[57]: 0.89219999999999999
```

k. Please explain using the vocabulary you've learned in this course what you just computed in part **j.** What is this value called in scientific studies? What does this value mean in terms of whether or not there is a difference between the new and old pages?

I've computed the p-value in j , which is the probability that we will observe this statistic, given the null hypothesis is true.

Since the p-value is large here, So we fail to reject the null hypothesis.

l. We could also use a built-in to achieve similar results. Though using the built-in might be easier to code, the above portions are a walkthrough of the ideas that are critical to correctly thinking about statistical significance. Fill in the below to calculate the number of conversions for each page, as well as the number of individuals who received each page. Let `n_old` and `n_new` refer the the number of rows associated with the old page and new pages, respectively.

```
In [58]: import statsmodels.api as sm

         convert_old = df2.query('group == "control"')['converted'].sum()
         convert_new = df2.query('group == "treatment"')['converted'].sum()
         n_old =  df2.query('landing_page == "old_page"').shape[0]
         n_new =  df2.query('landing_page == "new_page"').shape[0]
         print(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
```

```
17489 17264 145274 145310
```

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

```
In [97]: z_score, p_value = sm.stats.proportions_ztest(np.array([convert_new, convert_old])
                                                        ,np.array([n_new, n_old]), alternative =
         z_score, p_value
```

```
Out[97]: (-1.3109241984234394, 0.905058312759024449)
```

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.**?

The z-score here is -1.31 inside our critical value of 1.959 and the p-value is still large, So it is likely that our statistic is from the null

This means the z-score and p-value agree with the findings in parts j and k that we cannot reject the null hypothesis.

Part III - A regression approach

`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 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** to fit the regression model you specified in part **a.** to see if there is a significant difference in conversion based on which page a customer receives. However, you first need to create in df2 a column for the intercept, and create a dummy variable column for which page each user received. Add an **intercept** column, as well as an **ab_page** column, which is 1 when an individual receives the **treatment** and 0 if **control**.

In [189]:
```
df2['intercept']=1
df2['ab_page']=0
ab_page_index = df2[df2['group']=='treatment'].index
df2.loc[ab_page_index, "ab_page"] = 1

df2.head()
```

Out[189]:
```
                          timestamp    group landing_page  converted  ab_page  \
user_id
851104   2017-01-21 22:11:48.556739  control     old_page          0        0
804228   2017-01-12 08:01:45.159739  control     old_page          0        0
864975   2017-01-21 01:52:26.210827  control     old_page          1        0
936923   2017-01-10 15:20:49.083499  control     old_page          0        0
719014   2017-01-17 01:48:29.539573  control     old_page          0        0

         intercept country  us  uk  ca  US  UK  CA
user_id
851104           1      US   0   0   1   1   0   0
804228           1      US   0   0   1   1   0   0
864975           1      US   0   0   1   1   0   0
936923           1      US   0   0   1   1   0   0
719014           1      US   0   0   1   1   0   0
```

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

In [190]:
```
lm = sm.OLS(df2['converted'], df2[['intercept', 'ab_page']])
results=lm.fit()
results.summary()
```

Out[190]:
```
<class 'statsmodels.iolib.summary.Summary'>
"""
                            OLS Regression Results
==============================================================================
Dep. Variable:              converted   R-squared:                       0.000
Model:                            OLS   Adj. R-squared:                  0.000
Method:                 Least Squares   F-statistic:                     1.719
Date:                Tue, 13 Oct 2020   Prob (F-statistic):              0.190
Time:                        18:43:51   Log-Likelihood:                -85267.
No. Observations:              290584   AIC:                         1.705e+05
Df Residuals:                  290582   BIC:                         1.706e+05
Df Model:                           1
```

```
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
intercept      0.1204      0.001    141.407      0.000       0.119       0.122
ab_page       -0.0016      0.001     -1.311      0.190      -0.004       0.001
==============================================================================
Omnibus:                   125553.456   Durbin-Watson:                   2.000
Prob(Omnibus):                  0.000   Jarque-Bera (JB):           414313.355
Skew:                           2.345   Prob(JB):                         0.00
Kurtosis:                       6.497   Cond. No.                         2.62
==============================================================================

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is correctly speci
"""
```

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

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

Our hypothesis here is:
H_0 : p_{new} - p_{old} = 0
H_1 : p_{new} - p_{old} != 0

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?

We should consider other factors into the regression model as they might influence the conversions too. For instance student segments [new v/s returning candidates] might create change aversion or even, the opposite as a predisposition to conversion. Seasonality like new terms or New years might mean more interest in new skills/ resolutions. Timestamps are inlcuded but without regionality, they do not indicate if seasonality was a factor or not. [as different countries follow different term and weather patterns. Factors like device on which tests were taken or course which was looked at, prior academic background, age, might alter experience and ultimately, conversions. These are limitations which should be at least kept in mind while making the final decision. The disadvantages to adding additional terms into the regression model is that even with additional factors we can never account for all influencing factors or accomodate them. Plus, small pilots and pivots sometimes work better in practice than long-drawn research without execution.

g. Now along with testing if the conversion rate changes for different pages, also add an effect based on which country a user lives in. You will need to read in the **countries.csv** dataset and merge together your datasets on the appropriate rows. Here are the docs for joining tables.

Does it appear that country had an impact on conversion? Don't forget to create dummy variables for these country columns - **Hint: You will need two columns for the three dummy variables.** Provide the statistical output as well as a written response to answer this question.

```
In [173]: df_countries = pd.read_csv('countries.csv')
          df_countries.head()

Out[173]:    user_id country
          0   834778      UK
          1   928468      US
          2   822059      UK
          3   711597      UK
          4   710616      UK

In [175]: df2[['CA', 'UK', 'US']] = pd.get_dummies(df2['country'])
          df2

Out[175]:                         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
          644214   2017-01-22 02:05:21.719434    control     old_page          1
          847721   2017-01-17 14:01:00.090575    control     old_page          0
          650559   2017-01-24 11:55:51.084801    control     old_page          0
          935734   2017-01-17 20:33:37.428378    control     old_page          0
          746742   2017-01-23 11:38:29.592148    control     old_page          0
          913579   2017-01-24 09:11:39.164256    control     old_page          1
          690284   2017-01-13 17:22:57.182769    control     old_page          0
          710349   2017-01-11 22:24:44.226492    control     old_page          0
          677533   2017-01-23 17:48:50.491821    control     old_page          0
          831737   2017-01-11 21:18:20.911015    control     old_page          1
          771087   2017-01-16 00:05:29.983919    control     old_page          0
          896163   2017-01-22 09:10:20.753218    control     old_page          0
          862225   2017-01-08 14:49:37.335432    control     old_page          1
          939593   2017-01-05 09:15:31.984283    control     old_page          0
          702260   2017-01-18 13:55:31.488221    control     old_page          0
          670941   2017-01-05 08:16:41.306478    control     old_page          0
          850231   2017-01-18 17:18:04.790584    control     old_page          1
          685794   2017-01-20 14:54:58.150621    control     old_page          0
          714733   2017-01-03 08:22:37.904146    control     old_page          0
          710967   2017-01-10 02:19:22.842142    control     old_page          0
          680201   2017-01-11 10:38:45.952887    control     old_page          0
          790863   2017-01-19 11:02:39.220320    control     old_page          0
          717595   2017-01-23 18:19:08.148166    control     old_page          0
          779854   2017-01-11 21:28:30.735359    control     old_page          0
          916307   2017-01-19 17:27:38.676600    control     old_page          0
```

```
...                             ...         ...           ...          ...
924332   2017-01-15 19:38:52.858024  treatment     new_page            0
849625   2017-01-06 17:54:07.563311  treatment     new_page            0
929723   2017-01-10 15:13:48.352399  treatment     new_page            0
774769   2017-01-03 06:01:36.251836  treatment     new_page            0
733871   2017-01-21 17:54:08.810964  treatment     new_page            1
844588   2017-01-16 20:48:19.567178  treatment     new_page            0
641244   2017-01-07 16:57:26.193171  treatment     new_page            0
676072   2017-01-14 17:26:02.495442  treatment     new_page            0
886374   2017-01-07 13:43:39.202634  treatment     new_page            0
676732   2017-01-03 23:06:45.459467  treatment     new_page            0
862218   2017-01-04 10:43:07.846494  treatment     new_page            0
798826   2017-01-23 16:50:13.788528  treatment     new_page            0
836721   2017-01-12 17:37:50.966955  treatment     new_page            0
844901   2017-01-15 17:46:36.622726  treatment     new_page            0
653124   2017-01-14 13:44:51.745491  treatment     new_page            0
909437   2017-01-18 14:49:49.064452  treatment     new_page            0
776137   2017-01-12 05:53:12.386730  treatment     new_page            0
883344   2017-01-22 23:15:58.645325  treatment     new_page            0
825594   2017-01-06 12:37:08.897784  treatment     new_page            0
937338   2017-01-19 03:23:22.236666  treatment     new_page            0
733101   2017-01-23 12:52:58.711914  treatment     new_page            0
679096   2017-01-02 16:43:49.237940  treatment     new_page            0
691699   2017-01-09 23:42:35.963486  treatment     new_page            0
807595   2017-01-22 10:43:09.285426  treatment     new_page            0
846225   2017-01-16 15:24:46.705903  treatment     new_page            0
677163   2017-01-03 19:41:51.902148  treatment     new_page            0
925675   2017-01-07 20:38:26.346410  treatment     new_page            0
643562   2017-01-02 19:20:05.460595  treatment     new_page            0
822004   2017-01-04 03:36:46.071379  treatment     new_page            0
715931   2017-01-16 12:40:24.467417  treatment     new_page            0

         ab_page  intercept country  us  uk  ca  US  UK  CA
user_id
851104         0          1      US   0   0   1   1   0   0
804228         0          1      US   0   0   1   1   0   0
864975         0          1      US   0   0   1   1   0   0
936923         0          1      US   0   0   1   1   0   0
719014         0          1      US   0   0   1   1   0   0
644214         0          1      US   0   0   1   1   0   0
847721         0          1      US   0   0   1   1   0   0
650559         0          1      CA   1   0   0   0   0   1
935734         0          1      US   0   0   1   1   0   0
746742         0          1      US   0   0   1   1   0   0
913579         0          1      US   0   0   1   1   0   0
690284         0          1      US   0   0   1   1   0   0
710349         0          1      UK   0   1   0   0   1   0
677533         0          1      US   0   0   1   1   0   0
```

831737	0	1	UK	0	1	0	0	1	0
771087	0	1	US	0	0	1	1	0	0
896163	0	1	UK	0	1	0	0	1	0
862225	0	1	US	0	0	1	1	0	0
939593	0	1	US	0	0	1	1	0	0
702260	0	1	US	0	0	1	1	0	0
670941	0	1	US	0	0	1	1	0	0
850231	0	1	US	0	0	1	1	0	0
685794	0	1	US	0	0	1	1	0	0
714733	0	1	US	0	0	1	1	0	0
710967	0	1	US	0	0	1	1	0	0
680201	0	1	US	0	0	1	1	0	0
790863	0	1	US	0	0	1	1	0	0
717595	0	1	US	0	0	1	1	0	0
779854	0	1	US	0	0	1	1	0	0
916307	0	1	UK	0	1	0	0	1	0
...	...	...	...	..	..	..	..	..	..
924332	1	1	US	0	0	1	1	0	0
849625	1	1	US	0	0	1	1	0	0
929723	1	1	CA	1	0	0	0	0	1
774769	1	1	US	0	0	1	1	0	0
733871	1	1	US	0	0	1	1	0	0
844588	1	1	US	0	0	1	1	0	0
641244	1	1	US	0	0	1	1	0	0
676072	1	1	US	0	0	1	1	0	0
886374	1	1	US	0	0	1	1	0	0
676732	1	1	UK	0	1	0	0	1	0
862218	1	1	US	0	0	1	1	0	0
798826	1	1	US	0	0	1	1	0	0
836721	1	1	US	0	0	1	1	0	0
844901	1	1	US	0	0	1	1	0	0
653124	1	1	CA	1	0	0	0	0	1
909437	1	1	UK	0	1	0	0	1	0
776137	1	1	US	0	0	1	1	0	0
883344	1	1	CA	1	0	0	0	0	1
825594	1	1	UK	0	1	0	0	1	0
937338	1	1	UK	0	1	0	0	1	0
733101	1	1	US	0	0	1	1	0	0
679096	1	1	US	0	0	1	1	0	0
691699	1	1	US	0	0	1	1	0	0
807595	1	1	US	0	0	1	1	0	0
846225	1	1	US	0	0	1	1	0	0
677163	1	1	US	0	0	1	1	0	0
925675	1	1	US	0	0	1	1	0	0
643562	1	1	CA	1	0	0	0	0	1
822004	1	1	CA	1	0	0	0	0	1
715931	1	1	UK	0	1	0	0	1	0

```
[290584 rows x 13 columns]
```

In [176]:
```
df_countries.country.unique()
```

Out[176]:
```
array(['UK', 'US', 'CA'], dtype=object)
```

In [178]:
```
country_dummies = pd.get_dummies(df_countries['country'])
df_new = df_countries.join(country_dummies)
```

In [179]:
```
df_new.head()
```

Out[179]:
```
   user_id country  CA  UK  US
0   834778      UK   0   1   0
1   928468      US   0   0   1
2   822059      UK   0   1   0
3   711597      UK   0   1   0
4   710616      UK   0   1   0
```

In [182]:
```
lm = sm.OLS(df2['converted'], df2[['intercept', 'UK', 'US']])
results = lm.fit()
results.summary()
```

Out[182]:
```
<class 'statsmodels.iolib.summary.Summary'>
"""
                            OLS Regression Results
==============================================================================
Dep. Variable:              converted   R-squared:                       0.000
Model:                            OLS   Adj. R-squared:                  0.000
Method:                 Least Squares   F-statistic:                     1.605
Date:                Tue, 13 Oct 2020   Prob (F-statistic):              0.201
Time:                        18:38:07   Log-Likelihood:                -85267.
No. Observations:              290584   AIC:                         1.705e+05
Df Residuals:                  290581   BIC:                         1.706e+05
Df Model:                           2
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
intercept      0.1153      0.003     42.792      0.000       0.110       0.121
UK             0.0053      0.003      1.787      0.074      -0.001       0.011
US             0.0042      0.003      1.516      0.130      -0.001       0.010
==============================================================================
Omnibus:                   125552.384   Durbin-Watson:                   2.000
Prob(Omnibus):                  0.000   Jarque-Bera (JB):           414306.036
Skew:                           2.345   Prob(JB):                         0.00
Kurtosis:                       6.497   Cond. No.                         9.94
==============================================================================

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is correctly speci
"""
```

h. 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 there significant effects on conversion. Create the necessary additional columns, and fit the new model.

Provide the summary results, and your conclusions based on the results.
Finishing Up

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 report to make sure that it is satisfies all the areas of the rubric (found on the project submission page at the end of the lesson). You should also probably remove all of the "Tips" like this one so that the presentation is as polished as possible.

0.3 Directions to Submit

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).

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.

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