Analyze_ab_test_results_notebook

August 31, 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

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 [1]: 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 [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]: df.shape[0]
```

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

c. The number of unique users in the dataset.

```
In [4]: df.user_id.nunique()
```

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

d. The proportion of users converted.

```
In [5]: df.query('converted==1').count()['converted']/df['converted'].count()
```

```
Out[5]: 0.11965919355605512
```

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

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

```
Out[6]: 3893
```

f. Do any of the rows have missing values?

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

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

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 [8]: df2=df.query('(landing_page=="new_page" and group=="treatment") or (landing_page=="old_p
```

```
In [10]: # 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[10]: 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 [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()]['user_id']
```

```
Out[12]: 2893    773192
         Name: user_id, dtype: int64
```

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

```
In [13]: df2[df2.user_id.duplicated()]
```

```
Out[13]:       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 [14]: df2.drop_duplicates(subset=["user_id"],inplace=True)
```

```
/opt/conda/lib/python3.6/site-packages/ipykernel_launcher.py:1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/indexing.html#
  """Entry point for launching an IPython kernel.
```

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]: df2.query('converted==1')['converted'].count()/df2.converted.count()
```

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

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

```
In [16]: df2.query('converted==1 and group=="control"')['converted'].count()/df2.query('group=="
```

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

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

```
In [17]: df2.query('converted==1 and group=="treatment"')['converted'].count()/df2.query('group=
```

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

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

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

```
Out[18]: 0.50006194422266881
```

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.

It is evident from above probability values there is less chance for new treatment page lead to more conversions than old control page

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 Hypothesis-> H_0 :coversion rate of old page greater than or equal new page (p_{old} >= p_{new})

Alternate Hypothesis-> H_1 :coversion rate of old page less than new page ($p_{old} < p_{new}$)

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

```
0.119597087245
```

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

```
In [20]: p_old=df2.converted.mean()
         print(p_old)
```

```
0.119597087245
```

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

```
In [21]: n_new=df2.query('group=="treatment"')['user_id'].count()
         print(n_new)
```

```
145310
```

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

```
In [22]: n_old=df2.query('group=="control"')['user_id'].count()
         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 [23]: new_page_converted = np.random.binomial(1,p_new,n_new)

         #new_page_converted = np.random.choice([1, 0], size=n_new, p=[p_new, (1-p_new)])
         new_page_converted.mean()
```

```
Out[23]: 0.11964076801321313
```

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 [24]: old_page_converted = np.random.binomial(1,p_old,n_old)

         #old_page_converted = np.random.choice([1, 0], size=n_old, p=[p_old, (1-p_old)])
         old_page_converted.mean()

Out[24]: 0.11757093492297314
```

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

```
In [25]: new_page_converted.mean()-old_page_converted.mean()

Out[25]: 0.0020698330902399892
```

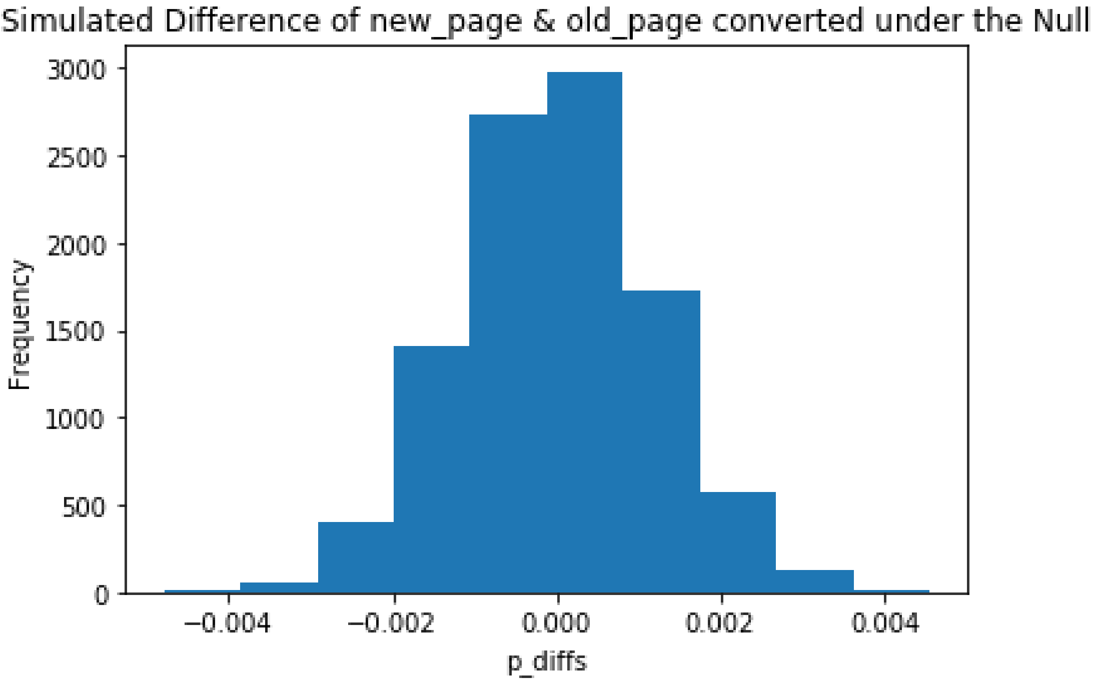
h. Create 10,000 p_{new} - p_{old} values using the same simulation process you used in parts (a) through (g) above. Store all 10,000 values in a NumPy array called **p_diffs**.

```
In [26]: p_diffs = []
         for _ in range(10000):
             new_page_converted = np.random.binomial(1,p_new,n_new)
             old_page_converted = np.random.binomial(1,p_old,n_old)
             p_diffs.append(new_page_converted.mean()-old_page_converted.mean())
```

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 [27]: # convert to numpy array
         p_diffs = np.array(p_diffs)

         # plot sampling distribution
         plt.hist(p_diffs)
         plt.xlabel('p_diffs')
         plt.ylabel('Frequency')
         plt.title('Simulated Difference of new_page & old_page converted under the Null');
```



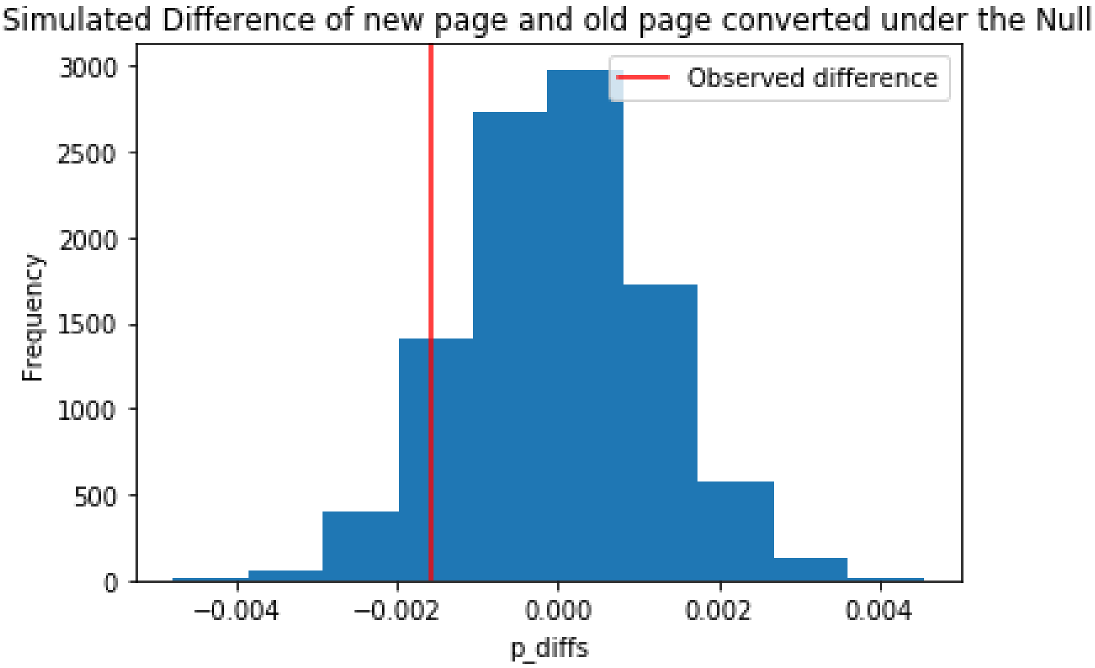
j. What proportion of the **p_diffs** are greater than the actual difference observed in **ab_data.csv**?

```
In [28]: df_control = df2.query('group == "control"')
         df_treatment = df2.query('group == "treatment"')

         # display observed difference
         obs_diff = df_treatment.converted.mean() - df_control.converted.mean()
         obs_diff

Out[28]: -0.0015782389853555567

In [29]: plt.hist(p_diffs)
         plt.axvline(x=obs_diff, color='r', label="Observed difference")
         plt.xlabel('p_diffs')
         plt.ylabel('Frequency')
         plt.title('Simulated Difference of new page and old page converted under the Null');
         plt.legend()
         plt.show()
```



```
In [30]: #calculate the proportion of p_diffs greater than the observe difference
         (p_diffs > obs_diff).mean()

Out[30]: 0.9076999999999995
```

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?

90.45% is the proportion of the p_diffs that are greater than the actual difference observed in ab_data.csv. In scientific studies this value is also called p-value. This value means that we cannot reject the null hypothesis and that we do not have proper evidence that the new page has a higher conversion rate than the old page.

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 [31]: import statsmodels.api as sm

         convert_old = df2.query('converted==1 and group=="control"')['user_id'].count()
         convert_new = df2.query('converted==1 and group=="treatment"')['user_id'].count()
         n_new = df2.query('group=="treatment"')['user_id'].count()
         n_old = df2.query('group=="control"')['user_id'].count()
```

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

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 [32]: z_score, p_value = sm.stats.proportions_ztest([convert_old, convert_new], [n_old, n_new

         z_score, p_value
```

```
Out[32]: (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.**?

Based on z-score and p-value from above we certain that we cannot reject null hypothesis. Null hypothesis being the converted rate of the old page is the same or greater than the converted rate of the new page.

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?

Since the response variable has two values either conversion or non-conversion, we choose logistic regression in this case

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 [33]: df2[['control','treatment']]= pd.get_dummies(df2['group'])
         df2 = df2.drop('control',axis = 1)
         df2.head()
```

```
/opt/conda/lib/python3.6/site-packages/pandas/core/frame.py:3140: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/indexing.html#
  self[k1] = value[k2]
```

```
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

            treatment
         0          0
         1          0
         2          1
         3          1
         4          0
```

```
In [36]: df2 = df2.rename(columns={'treatment': 'ab_page'})
         df2['intercept'] = 1
         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

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

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 [43]: lm = sm.Logit(df2['converted'],df2[['intercept','ab_page']])
         results = lm.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 [44]: results.summary2()
```

```
Out[44]: <class 'statsmodels.iolib.summary2.Summary'>
         """
                                  Results: Logit
         ==================================================================
         Model:              Logit            No. Iterations:   6.0000
         Dependent Variable: converted        Pseudo R-squared: 0.000
         Date:               2020-08-31 14:23 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**? **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**?

p-value associated with ab_page is 0.19. It is higher than 0.05. Thus, the coefficient is not significant.

The p-value is very different in part II and part III. In part II the p-value is 0.91. This might be because the tests of the regression model (PART III) assumes an intercept and because of differences in one or two-tailed testing.

Both p-value does not support alternate hypothesis

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?

Addition of new factors can help us in understanding conversion rates

Additions of factors may result in complexity

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 [50]: df3=pd.read_csv('countries.csv')

         df2=df2.join(df3.set_index('user_id'), on='user_id')
```

```
In [53]: df2.head()
         # Create the necessary dummy variables
         df2[['CA','UK', 'US']]= pd.get_dummies(df2['country'])
         lm = sm.Logit(df2['converted'],df2[['intercept','ab_page','CA','US']])
         results = lm.fit()
         results.summary2()
```

```
Optimization terminated successfully.
         Current function value: 0.366113
         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:               2020-08-31 14:37 AIC:              212781.1253
         No. Observations:   290584           BIC:              212823.4439
         Df Model:           3                Log-Likelihood:   -1.0639e+05
         Df Residuals:       290580           LL-Null:          -1.0639e+05
         Converged:          1.0000           Scale:            1.0000
         -------------------------------------------------------------------
                       Coef.   Std.Err.      z      P>|z|    [0.025   0.975]
         -------------------------------------------------------------------
         intercept    -1.9794   0.0127 -155.4145  0.0000  -2.0044  -1.9544
         ab_page      -0.0149   0.0114   -1.3069  0.1912  -0.0374   0.0075
         CA           -0.0506   0.0284   -1.7835  0.0745  -0.1063   0.0050
         US           -0.0099   0.0133   -0.7433  0.4573  -0.0359   0.0162
         ==================================================================

         """
```

From the p-value we can say there is no impact on conversion rate based on the countries. we can say country factor is not significant

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.

```
In [56]: df2['interaction_us_ab_page'] = df2.US *df2.ab_page
         df2['interaction_ca_ab_page'] = df2.CA *df2.ab_page
         lm = sm.Logit(df2['converted'],df2[['intercept','ab_page','US','interaction_us_ab_page'
         results = lm.fit()
         results.summary2()
```

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

```
Out[56]: <class 'statsmodels.iolib.summary2.Summary'>
         """
                                  Results: Logit
         ========================================================================
         Model:                 Logit              No. Iterations:    6.0000
         Dependent Variable:    converted          Pseudo R-squared:  0.000
         Date:                  2020-08-31 14:49   AIC:               212782.6602
         No. Observations:      290584             BIC:               212846.1381
         Df Model:              5                  Log-Likelihood:    -1.0639e+05
         Df Residuals:          290578             LL-Null:           -1.0639e+05
         Converged:             1.0000             Scale:             1.0000
         ------------------------------------------------------------------------
                                 Coef.  Std.Err.     z     P>|z|   [0.025  0.975]
         ------------------------------------------------------------------------
         intercept              -1.9922   0.0161 -123.4571 0.0000 -2.0238 -1.9606
         ab_page                 0.0108   0.0228    0.4749 0.6349 -0.0339  0.0555
         US                      0.0057   0.0188    0.3057 0.7598 -0.0311  0.0426
         interaction_us_ab_page -0.0314   0.0266   -1.1807 0.2377 -0.0835  0.0207
         CA                     -0.0118   0.0398   -0.2957 0.7674 -0.0899  0.0663
         interaction_ca_ab_page -0.0783   0.0568   -1.3783 0.1681 -0.1896  0.0330
         ========================================================================

         """
```

Base on the p-value for interaction factors as it is higher than 0.05, thus there is no significant effect on conversion rate due interaction between page and country

```
In [59]: df2.timestamp.min(),df2.timestamp.max()
```

```
Out[59]: ('2017-01-02 13:42:05.378582', '2017-01-24 13:41:54.460509')
```

1 CONCLUSION

Overall results from A/B testing and regression analysis shows that new page that was created has less or equal impact on conversion rate when compare to old page. Even there is no improvment on conversion rate based on countries and new page.

Since sample size is large and duration taken to get the samples is larger we can say it is better to create a another new page than countinuing testing this new page.