Analyze A/B Test Results

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!

Table of Contents

- Introduction
- Part I - Probability
- Part II - A/B Test
- Part III - Regression

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
%autosave 180
%matplotlib inline
#We are setting the seed to assure you get the same answers on quizzes as w
random.seed(42)
```

```
Autosaving every 180 seconds
```

`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")
```

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

In [3]:
```
df.head()
```

Out[3]:

	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

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.converted.mean()
```

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

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

In [6]:
```
# number of times the new_page and treatment don't line up
df[((df['group'] == 'treatment') == (df['landing_page'] == 'new_page'))==Fa
```

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

f. Do any of the rows have missing values?

In [7]:
```
df.isnull().any()
```

Out[7]:
```
user_id         False
timestamp       False
group           False
landing_page    False
converted       False
dtype: bool
```

`2.` For the rows where **treatment** is not aligned with **new_page** or **control** is not aligned with **old_page**, we cannot be sure if this row truly received the new or old page. Use **Quiz 2** in the classroom to provide 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[((df['group'] == 'treatment') == (df['landing_page'] == 'new_page'))
```

```
df2.shape[0]
```

Out[8]:
```
290585
```

In [9]:
```
# Double Check all of the correct rows were removed - this should be 0
df2[((df2['group'] == 'treatment') == (df2['landing_page'] == 'new_page'))
```

Out[9]:
```
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 [10]:
```
df2.user_id.nunique()
```

Out[10]:
```
290584
```

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

In [11]:
```
df2[df2.user_id.duplicated(keep=False)]
```

Out[11]:

	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

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

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

Out[12]:

	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**, but keep your dataframe as **df2**.

In [13]:
```
df2=df2.drop_duplicates(subset="user_id", keep = 'first')
df2[df2.user_id.duplicated(keep=False)]
```

Out[13]:

user_id	timestamp	group	landing_page	converted

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

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

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

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

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

In [15]:
```
df2[df2['group'] == "control"]['converted'].mean()
```

Out[15]:
```
0.1203863045004612
```

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

In [16]:
```
df2[df2['group'] == "treatment"]['converted'].mean()
```

Out[16]:
```
0.11880806551510564
```

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

In [17]:
```
df2[df2['landing_page'] == "new_page"].count()[0]/df2.shape[0]
```

Out[17]:
```
0.5000619442226688
```

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

Your answer goes here.

No, there is not enought to suggest there is sufficient evidence that the treatment page lead to more conversions. This is because there was only a slight difference in probability of conversion between the treatment and control groups.

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.

Put your answer here.

$$H_0 : p_{old} - p_{new} \geq 0$$

$$H_1 : p_{old} - p_{new} < 0$$

`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 **convert rate** for p_{new} under the null?

In [18]:
```
p_new = df2.converted.mean()
p_new
```

Out[18]:
```
0.11959708724499628
```

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

In [19]:
```
p_old = df2.converted.mean()
p_old
```

Out[19]:
```
0.11959708724499628
```

c. What is n_{new}?

In [20]:
```
n_new = df2[df2['group'] == "treatment"]['converted'].count()
n_new
```

Out[20]:
```
145310
```

d. What is n_{old}?

In [21]:
```
n_old = df2[df2['group'] == "control"]['converted'].count()
n_old
```

Out[21]:
```
145274
```

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

In [22]:
```
new_page_converted = np.random.choice([0, 1], p=[1-p_new, p_new], size=[1,
new_page_converted
```

```
Out[22]: array([[0, 0, 0, ..., 0, 0, 1]])
```

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

```
In [23]:
old_page_converted = np.random.choice([0, 1], p=[1-p_old, p_old], size=[1, 
old_page_converted
```

```
Out[23]: array([[0, 1, 0, ..., 0, 0, 0]])
```

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

```
In [24]:
obs_diff = new_page_converted.mean() - old_page_converted.mean()
obs_diff
```

```
Out[24]: 0.000720641373470457
```

```
In [25]:
df2.query('group == "treatment"')['converted'].mean()-df2.query('group == "
```

```
Out[25]: -0.0015782389853555567
```

h. Simulate 10,000 p_{new} - p_{old} values using this same process similarly to the one you calculated in parts **a. through g.** above. Store all 10,000 values in a numpy array called **p_diffs**.

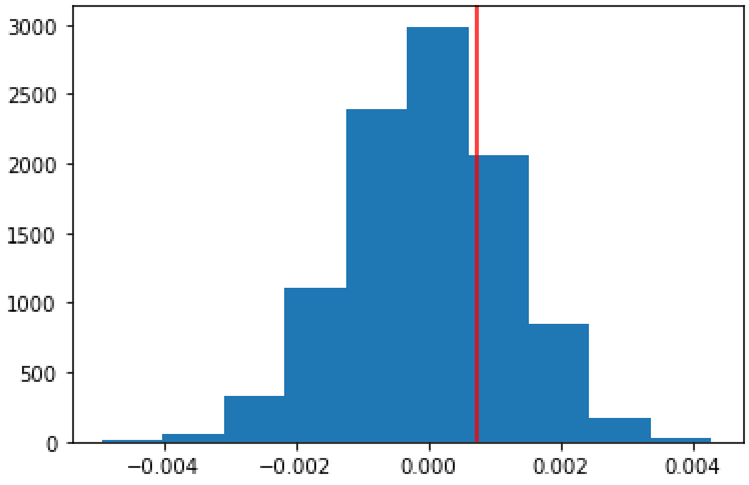
```
In [26]:
new_converted_simulation = np.random.binomial(n_new, p_new, 10000)/n_new
old_converted_simulation = np.random.binomial(n_old, p_old, 10000)/n_old
p_diffs = new_converted_simulation - old_converted_simulation

## Alternative but less efficient code-- use Numpy when possible
# new_page_converted_means, old_page_converted_means, p_diffs = [], [], []
# for _ in range(10000):
#     bootsamp = df2.sample(200, replace = True)
#     new_page_converted_mean = bootsamp[bootsamp['group'] == "treatment"][
#     old_page_converted_mean = bootsamp[bootsamp['group'] == "control"]['c
#     # append the info
#     new_page_converted_means.append(new_page_converted_mean)
#     old_page_converted_means.append(old_page_converted_mean)
#     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]:
#plt.hist(new_page_converted_means, alpha = 0.5);
#plt.hist(old_page_converted_means, alpha = 0.5);
```

```
In [28]:
plt.hist(p_diffs)
plt.axvline(x=obs_diff, color = 'red');
```

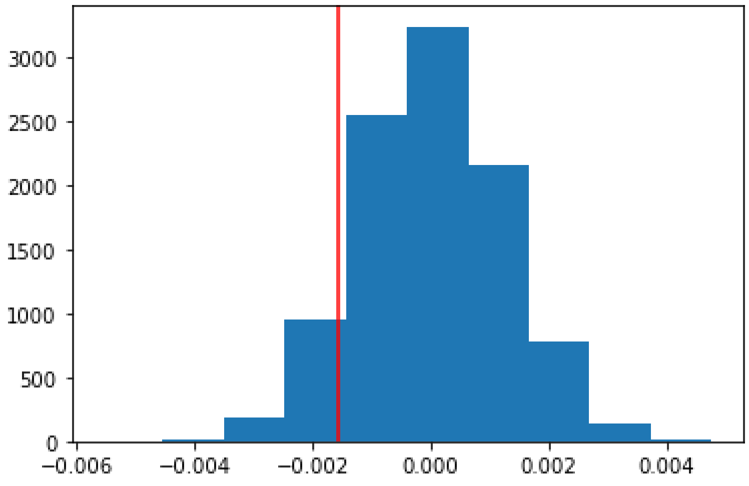


j. What proportion of the **p_diffs** are greater than the actual difference observed in **ab_data.csv**?

In [29]:
```
# Calculate the actual difference observed in ab_data.csv
act_diff=df2.query('group == "treatment"')['converted'].mean()-df2.query('g
act_diff
```

Out[29]:
```
-0.0015782389853555567
```

In [30]:
```
# Simulate a sampling distribution for the null value
p_diffs = np.array(p_diffs)
null_value = np.random.normal(0, np.std(p_diffs), 10000)
plt.hist(null_value)
plt.axvline(act_diff, color="red");
```



In [31]:
```
(null_value>act_diff).mean()
```

Out[31]:
```
0.9068
```

k. In words, explain 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?

Put your answer here. The value of 0.9037 represents the so-called p_value which describes the probability of finding the observed results given that null hypothesis is true. In this case, a p-value of 0.9037 is very high-- much higher than the .05 error rate, so it is likely that null value came from the sampling distribution. Therefore, we fail to reject the null hypothesis, that is we cannot conclude that the new page is more effective in coverting customers 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 [32]:
```
import statsmodels.api as sm

# sum of the converted users
convert_old = sum((df2.group == 'control') & (df2.converted == 1))
convert_new = sum((df2.group == 'treatment') & (df2.converted == 1))

#number of individuals who received each page
n_old = sum(df2.group == 'control')
n_new = sum(df2.group == 'treatment')
```

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 [33]:
```
z_score, p_value = sm.stats.proportions_ztest([convert_new, convert_old], [
```

In [34]:
```
z_score
```

Out[34]:
```
-1.3109241984234394
```

In [35]:
```
p_value
```

Out[35]:
```
0.9050583127590245
```

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.**?

Put your answer here. The z-score is a value on the standard distribution that can be used to calculate the p-score. The p-value, as stated above, simply represents the probabability of accepting the null hypothesis. The p-value of 0.9050 derived from the z-score is nearly identical to the p-value (0.9037) derived from simulation and bootstrapping is exercises j. and k.

Part III - A regression approach

`1.` In this final part, you will see that the result you acheived in the previous A/B test can also be acheived 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?

Put your answer here. Logistical 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 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 [36]:
```
#from statsmodels.formula.api import ols
df2['intercept'] = 1
df2[['ab_page', 'old_page']] = pd.get_dummies(df2['landing_page'])
df2.head()
```

Out[36]:

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

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

In [37]:
```
import statsmodels.api as sm;

logit = sm.Logit(df2['converted'],df2[['intercept' ,'ab_page']])
```

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

In [38]:
```
results = logit.fit()
results.summary()
```

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

Out[38]:

```
                          Logit Regression Results
   Dep. Variable:         converted   No. Observations:      290584
           Model:             Logit       Df Residuals:      290582
          Method:               MLE           Df Model:           1
            Date:  Mon, 09 Aug 2021     Pseudo R-squ.:    8.077e-06
            Time:          12:56:23     Log-Likelihood:  -1.0639e+05
       converged:              True            LL-Null:  -1.0639e+05
 Covariance Type:         nonrobust        LLR p-value:      0.1899

              coef   std err          z   P>|z|   [0.025   0.975]
 intercept  -1.9888    0.008   -246.669   0.000   -2.005   -1.973
   ab_page  -0.0150    0.011     -1.311   0.190   -0.037    0.007
```

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 the **Part II**?

Answer: The p-value for ab_page is 0.190 likely because this a two-tailed test. The p-value from the A/B test above was a one-tailed test.

f. Now along with testing if the conversion rate changes for different pages, also add an effect based on which country a user lives. 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 [39]:

```
countries_df = pd.read_csv('./countries.csv')
df_new = countries_df.set_index('user_id').join(df2.set_index('user_id'), h
```

In [40]:

```
df_new.head()
```

Out[40]:

user_id	country	timestamp	group	landing_page	converted	intercept	ab_page
834778	UK	2017-01-14 23:08:43.304998	control	old_page	0	1	0
928468	US	2017-01-23 14:44:16.387854	treatment	new_page	0	1	1
822059	UK	2017-01-16 14:04:14.719771	treatment	new_page	1	1	1
711597	UK	2017-01-22 03:14:24.763511	control	old_page	0	1	0
710616	UK	2017-01-16 13:14:44.000513	treatment	new_page	0	1	1

In [41]:

```
df_new.country.unique()
```

Out[41]:

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

In [42]:

```
### Create the necessary dummy variables
df_new[['CA','UK','US']] = pd.get_dummies(df_new['country'])
df_new.drop('CA',axis=1, inplace=True)
df_new.head()
```

Out[42]:

user_id	country	timestamp	group	landing_page	converted	intercept	ab_page
834778	UK	2017-01-14 23:08:43.304998	control	old_page	0	1	0
928468	US	2017-01-23 14:44:16.387854	treatment	new_page	0	1	1
822059	UK	2017-01-16 14:04:14.719771	treatment	new_page	1	1	1
711597	UK	2017-01-22 03:14:24.763511	control	old_page	0	1	0
710616	UK	2017-01-16 13:14:44.000513	treatment	new_page	0	1	1

In [44]:

```
logit2 = sm.Logit(df_new['converted'],df_new[['intercept' ,'ab_page', 'UK',
results2 = logit2.fit()
results2.summary2()
```

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

Out[44]:

Model:	Logit	Pseudo R-squared:	0.000
Dependent Variable:	converted	AIC:	212781.1253
Date:	2021-08-09 13:03	BIC:	212823.4439
No. Observations:	290584	Log-Likelihood:	-1.0639e+05
Df Model:	3	LL-Null:	-1.0639e+05
Df Residuals:	290580	LLR p-value:	0.17599
Converged:	1.0000	Scale:	1.0000
No. Iterations:	6.0000		

	Coef.	Std.Err.	z	P>|z|	[0.025	0.975]
intercept	-2.0300	0.0266	-76.2488	0.0000	-2.0822	-1.9778
ab_page	-0.0149	0.0114	-1.3069	0.1912	-0.0374	0.0075
UK	0.0506	0.0284	1.7835	0.0745	-0.0050	0.1063
US	0.0408	0.0269	1.5161	0.1295	-0.0119	0.0934

g. 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 [46]:

```
### Fit Your Linear Model And Obtain the Results
df_new['UK_ab_page'] = df_new['UK'] * df_new['ab_page']
df_new['US_ab_page'] = df_new['US'] * df_new['ab_page']
df_new.head()
```

Out[46]:

	country	timestamp	group	landing_page	converted	intercept	ab_page
user_id							
834778	UK	2017-01-14 23:08:43.304998	control	old_page	0	1	0
928468	US	2017-01-23 14:44:16.387854	treatment	new_page	0	1	1
822059	UK	2017-01-16 14:04:14.719771	treatment	new_page	1	1	1
711597	UK	2017-01-22 03:14:24.763511	control	old_page	0	1	0
710616	UK	2017-01-16 13:14:44.000513	treatment	new_page	0	1	1

In [47]:

```
logit3 = sm.Logit(df_new['converted'],df_new[['intercept' ,'ab_page', 'UK',
results3 = logit3.fit()
results3.summary2()
```

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

```
Out[47]:             Model:            Logit   Pseudo R-squared:        0.000
       Dependent Variable:        converted                AIC:  212782.6602
                     Date: 2021-08-09 13:14                BIC:  212846.1381
         No. Observations:           290584     Log-Likelihood:  -1.0639e+05
                 Df Model:                5            LL-Null:  -1.0639e+05
             Df Residuals:           290578        LLR p-value:      0.19199
                Converged:           1.0000              Scale:       1.0000
           No. Iterations:           6.0000

                   Coef.  Std.Err.         z   P>|z|   [0.025   0.975]
       intercept -2.0040    0.0364  -55.0077  0.0000  -2.0754  -1.9326
         ab_page -0.0674    0.0520   -1.2967  0.1947  -0.1694   0.0345
              UK  0.0118    0.0398    0.2957  0.7674  -0.0663   0.0899
              US  0.0175    0.0377    0.4652  0.6418  -0.0563   0.0914
      UK_ab_page  0.0783    0.0568    1.3783  0.1681  -0.0330   0.1896
      US_ab_page  0.0469    0.0538    0.8718  0.3833  -0.0585   0.1523
```

Summary and conlusion: The coefficent for "UK_ab_page" is larger than that of "US_ab_page" which suggests that UK-based customers visiting the page were more likely to convert than than US-based customers. In addition, the p-scores for "UK_ab_page" and "US_ab_page" are quite small (below error rate of 0.5), which suggests that we should reject the null hypothesis that the old page results in more conversations than the new page. The addition of the country data as variable is vital as the p-values change when this data is integrating. The simulation A/B test did not take the country data into consideration, and resulted in a high p-value, suggesting that we cannot reject the null hypothesis. However, the more robust regression model that included the country data showed low p-values so we should reject the null hypothesis and replace the old page with the new page to increase customer conversion.

In other words, based on the data available, we do not have sufficient evidence to suggest that the new page results in more conversions than the old page.

```
In [ ]:
```