a_b_test_anaylysis

March 20, 2020

0.1 Analyze A/B Test Results

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.

Part I - Probability

To get started, let's import our libraries.

[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:

[2]:
```
df = pd.read_csv('ab_data.csv')
```

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

[3]:
```
df.shape
```

[3]:
```
(294478, 5)
```

c. The number of unique users in the dataset.

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

[4]:
```
290584
```

d. The proportion of users converted.

[5]:
```
overall_converted_prop = df.converted.sum() / df.shape[0]
overall_converted_prop
```

[5]:
```
0.11965919355605512
```

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

[6]:
```
new_page_control = df[((df['landing_page'] == 'new_page') & (df['group'] !=␣
↪'treatment')) |
                      ((df['landing_page'] != 'new_page') & (df['group'] ==␣
↪'treatment'))]
new_page_control.shape[0]
```

[6]:
```
3893
```

f. Do any of the rows have missing values?

[7]:
```
# If the statement below returns true, that means there is no missing values
new_page_control.notnull().shape == new_page_control.shape
```

[7]:
```
True
```

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**.

[8]:
```
df2 = df[((((df.group == 'treatment') & (df.landing_page == 'new_page')) | ((df.
↪group == 'control') & (df.landing_page == 'old_page'))))].copy()
```

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

[9]:
```
0
```

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

a. How many unique **user_ids** are in **df2**?

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

[10]:
```
290584
```

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

[11]:
```
# Get the index of the user who has duplicated information from the numpy array
# and filter the user id column using that index
idx = list(df2.user_id.duplicated().values).index(True)
df2.user_id.to_frame().iloc[idx]
```

[11]:
```
user_id    773192
Name: 2893, dtype: int64
```

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

[12]:
```
repeated_users = df[df.user_id == 773192]
repeated_users
```

[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**.

[13]:
```
# Drop the second record
df2.drop(repeated_users.index[1], inplace=True)
```

[14]:
```
# Test
df2.user_id.duplicated().sum()
```

[14]:
```
0
```

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?

[15]:
```
df.converted.mean()
```

[15]:
```
0.11965919355605512
```

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

[16]:
```
control = df2.group == 'control'
df2[control].converted.mean()
```

[16]:
```
0.1203863045004612
```

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

[17]:
```
treatment = df2.group == 'treatment'
df2[treatment].converted.mean()
```

[17]:
```
0.11880806551510564
```

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

[18]:
```
new_page = df2.landing_page == 'new_page'
df2[new_page].shape[0] / df2.shape[0]
```

[18]:
```
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.

`From observations above, we cannot yet safely decide that one page is better than the other. Th`

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

Alternative hypothesis: $p_{old} < p_{new}$

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

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

[19]:
```
0.11959708724499628
```

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

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

[20]:
```
0.11959708724499628
```

```
As we assumed that p_new and p_old are equal to the actual observed difference in the dataset,
```

c. What is n_{new}?

[21]:
```
df2.landing_page.value_counts()
```

[21]:
```
new_page    145310
old_page    145274
Name: landing_page, dtype: int64
```

[22]:
```
n_new = 145310
```

d. What is n_{old}?

[23]:
```
n_old = 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**.

[24]:
```
new_page_converted = np.random.binomial(n_new, p_new)
new_page_converted
```

[24]:
```
17426
```

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**.

[25]:
```
old_page_converted = np.random.binomial(n_old, p_old)
old_page_converted
```

[25]:
```
17672
```

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

[26]:
```
p_diff_old = new_page_converted / n_new - old_page_converted / n_old
p_diff_old
```

[26]:
```
-0.0017230696837877701
```

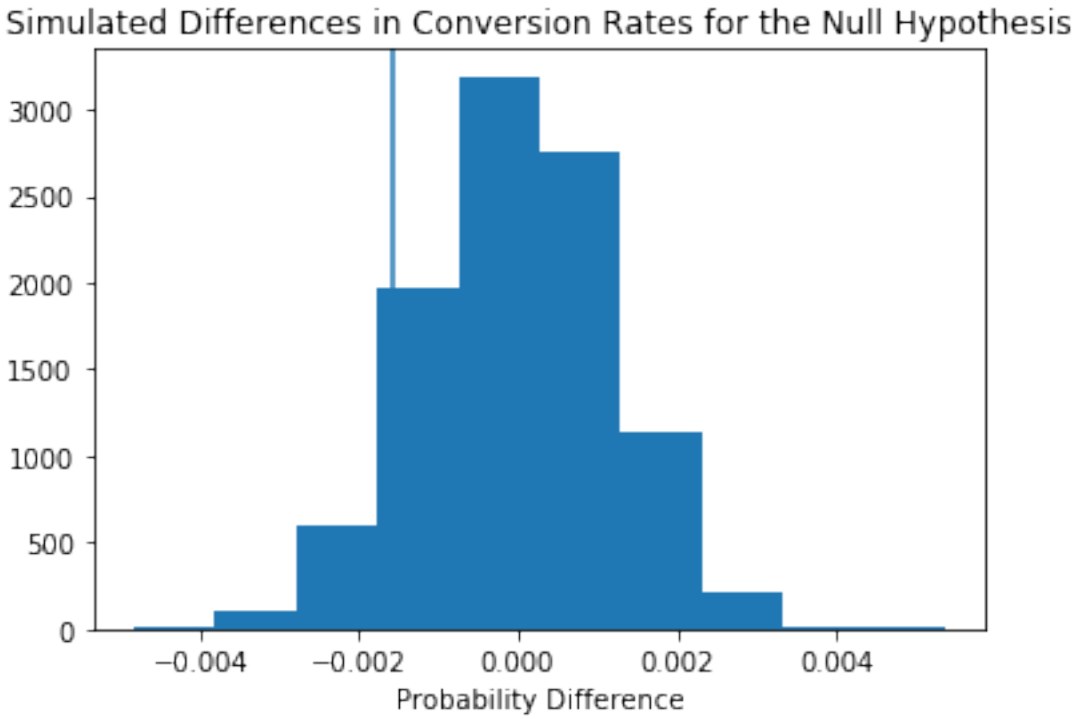
h. Simulate 10,000 p_{new} - p_{old} values using this same process similarly to the one you calculated in parts **a. through g.** above. Store all 10,000 values in a numpy array called **p_diffs**.

[27]:
```
p_diffs = []
```

[28]:
```
for _ in range(10000):
    new_page_converted = np.random.choice([1, 0], size=n_new, p=[p_new,␣
 ↪(1-p_new)])
    old_page_converted = np.random.choice([1, 0], size=n_old, p=[p_old,␣
 ↪(1-p_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.

[29]:
```
obs_difference = df2[df2.group == 'treatment'].converted.mean() - df2[df2.group␣
 ↪== 'control'].converted.mean()
plt.hist(p_diffs)
plt.xlabel('Probability Difference')
plt.title('Simulated Differences in Conversion Rates for the Null Hypothesis')
plt.axvline(obs_difference);
```



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

[30]:
```
(np.array(p_diffs) > obs_difference).mean()
```

[30]:
```
0.9065
```

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?

`In the above cell, we calculated the p_value. It is equal to 0.9091 which means that we fa`

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.

[31]:
```
import statsmodels.api as sm
from statsmodels.stats.proportion import proportions_ztest

convert_old = df2[df2.group == 'control'].converted.sum()
convert_new = df2[df2.group == 'treatment'].converted.sum()
n_old = df2[df2.group == 'control'].shape[0]
n_new = df2[df2.group == 'treatment'].shape[0]
```

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

[32]:
```
test_stat, p_value = proportions_ztest([convert_new, convert_old], [n_new,␣
↪n_old], alternative='larger')
test_stat, p_value
```

[32]:
```
(-1.3109241984234394, 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.**?

`Above we calculated the z score and p value. As I stated above, p values represent the probabi`

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?

`As there are only two outcomes, the correct approach would be a 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 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**.

[33]:
```
df2['intercept'] = 1
```

[34]:
```
df2['ab_page'] = 0
df2.loc[(df2['group'] == 'treatment'), 'ab_page'] = 1
df2.sample(5)
```

[34]:
```
        user_id                   timestamp      group landing_page  \
245296   888668  2017-01-17 21:23:48.760002    control     old_page
263972   789110  2017-01-23 15:13:56.155431  treatment     new_page
288334   710339  2017-01-20 18:42:50.894080  treatment     new_page
19860    832043  2017-01-11 13:53:09.495519    control     old_page
122420   694747  2017-01-10 22:21:19.717100    control     old_page

        converted  intercept  ab_page
245296          0          1        0
263972          1          1        1
288334          0          1        1
19860           0          1        0
122420          0          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.

[35]:
```
logit_mod = sm.Logit(df2['converted'], df2[['intercept', 'ab_page']])
results = logit_mod.fit()
```

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

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

[36]:
```
results.summary()
```

[36]:
```
<class 'statsmodels.iolib.summary.Summary'>
"""
                           Logit Regression Results
==============================================================================
Dep. Variable:              converted   No. Observations:               290584
Model:                          Logit   Df Residuals:                   290582
Method:                           MLE   Df Model:                            1
Date:                Fri, 20 Mar 2020   Pseudo R-squ.:               8.077e-06
Time:                        23:22:29   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**?

[37]:
```
ab_page_p_value = results.pvalues[1]
ab_page_p_value
```

[37]:
```
0.1898862112362789
```

```
The p value associated with ab_page is, as seen above, 0.18988. I was not sure why the p values
```

[38]:
```
1 - (ab_page_p_value / 2)
```

[38]:
```
0.9050568943818605
```

```
It is the same as proportions_ztest p_value and the ones in part j and k.
```

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?

```
Almost every web page have users from across the globe. Whether users convert or not also depe
```

g. 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 approporiate 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.

[39]:
```
countries_df = pd.read_csv('./countries.csv')
df_new = countries_df.set_index('user_id').join(df2.set_index('user_id'),␣
 ↪how='inner')
```

[40]:
```
### Create the necessary dummy variables
df_new[['CA', 'UK', 'US']] = pd.get_dummies(df_new['country'])
df_new.sample(5)
```

[40]:
```
         country                   timestamp      group landing_page  \
user_id
687327       UK  2017-01-17 05:57:30.680263  treatment     new_page
```

```
697715       US  2017-01-19 03:54:42.909603  treatment     new_page
641541       UK  2017-01-06 15:34:28.508313    control     old_page
913280       UK  2017-01-06 19:33:13.007218  treatment     new_page
717681       US  2017-01-16 12:10:59.142907    control     old_page

         converted  intercept  ab_page  CA  UK  US
user_id
687327           1          1        1   0   1   0
697715           0          1        1   0   0   1
641541           0          1        0   0   1   0
913280           0          1        1   0   1   0
717681           0          1        0   0   0   1
```

[41]:
```
country_model = sm.Logit(df_new['converted'], df_new[['intercept', 'US', 'UK']])
results = country_model.fit()
results.summary()
```

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

[41]:
```
<class 'statsmodels.iolib.summary.Summary'>
"""
                           Logit Regression Results
==============================================================================
Dep. Variable:              converted   No. Observations:               290584
Model:                          Logit   Df Residuals:                   290581
Method:                           MLE   Df Model:                            2
Date:                Fri, 20 Mar 2020   Pseudo R-squ.:               1.521e-05
Time:                        23:22:33   Log-Likelihood:            -1.0639e+05
converged:                       True   LL-Null:                   -1.0639e+05
Covariance Type:            nonrobust   LLR p-value:                    0.1984
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
intercept     -2.0375      0.026    -78.364      0.000      -2.088      -1.987
US             0.0408      0.027      1.518      0.129      -0.012       0.093
UK             0.0507      0.028      1.786      0.074      -0.005       0.106
==============================================================================
"""
```

As the p values associated with countries are higher than the Type I error rate, they don't hav

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.

[42]:
```
# Create dummy variables to form interaction terms
df_new["ca_page"] = df_new["ab_page"] * df_new["CA"]
df_new["uk_page"] = df_new["ab_page"] * df_new["UK"]
```

[43]:
```
### Fit Your Linear Model And Obtain the Results
all_in_model = sm.Logit(df_new['converted'], df_new[['intercept', 'CA', 'UK',␣
 ↪'ab_page', 'ca_page']])
results = all_in_model.fit()
results.summary()
```

```
Optimization terminated successfully.
         Current function value: 0.366111
         Iterations 6
```

[43]:
```
<class 'statsmodels.iolib.summary.Summary'>
"""
                           Logit Regression Results
==============================================================================
Dep. Variable:              converted   No. Observations:               290584
Model:                          Logit   Df Residuals:                   290579
Method:                           MLE   Df Model:                            4
Date:                Fri, 20 Mar 2020   Pseudo R-squ.:               2.827e-05
Time:                        23:22:37   Log-Likelihood:            -1.0639e+05
converged:                       True   LL-Null:                   -1.0639e+05
Covariance Type:            nonrobust   LLR p-value:                    0.1981
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
intercept     -1.9906      0.009   -221.524      0.000      -2.008      -1.973
CA            -0.0134      0.038     -0.357      0.721      -0.087       0.060
UK             0.0099      0.013      0.744      0.457      -0.016       0.036
ab_page       -0.0123      0.012     -1.047      0.295      -0.035       0.011
ca_page       -0.0552      0.053     -1.035      0.301      -0.160       0.049
==============================================================================
"""
```

Conclusions

Statistical tests from Parts II and III provided enough evidence to correctly base our decisions on. In Part II we performed tests to see which of our hyptheses is true in two ways. First way, which is a traditional hypothesis test, gave us a p value of 0.906. Therefore, we failed to reject the null hypothesis, which was old page was equal to or better than the new page.

In the second way, we performed a proportions_ztest. We found the z score and p value, -1.31 and 0.905 respectively. This also suggested that we should not move away from the null hypothesis.

In Part III, we fit a logistic regression model to see whether new page or old page had an influence on conversion. The p value associated with pages was 0.1899 which again, indicates that we fail to reject the null hypothesis we set up in the beginning of Part II. However, p value greater than the

Type I error threshold also signifies that the difference is not statistically significant. Furthermore, change in country does not affect the conversion rate significantly.

Overall, based on the tests above, the old page is the same or sometimes, better than the new page.

Note: Even though I correctly fit a regression model with interaction terms I am not completely sure as to why we would want to create new columns by muplitplying the country and page columns. Besides, as we did not cover interaction terms in the classroom, I was not able to interpret the results of that last model.