Project 3; Analyze AB Test Results

August 14, 2020

1 Analyze A/B Test Results

1.1 Table of Contents

Introduction A/B tests are commonly performed to understand the results of A/B tests. For this project a company is considering making changes to its e-commerce website. The goal is 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

[1]:
```
#import required libraries
import pandas as pd
import numpy as np
import random
import matplotlib.pyplot as plt

#plot visualisation in notebook
%matplotlib inline

#set the seed to assure replicability
random.seed(42)
```

[2]:
```
#read in the data
df = pd.read_csv(r'C:\Users\noama\ab_data.csv')

#display top 5 rows of dataset
df.head()
```

[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
```

[3]:
```
#count number of rows in the dataset
df.shape[0]
```

[3]:
```
294478
```

[4]:
```
#count number of unique users in the dataset
df.user_id.nunique()
```

[4]:
```
290584
```

[5]:
```
#calculate proportion of users converted
df.converted.mean()
```

[5]:
```
0.11965919355605512
```

[6]:
```
#The number of times the new_page and treatment don't line up
non_align = df[(df['group'] == "treatment") & (df['landing_page'] ==␣
↪"old_page") | (df['group'] == "control") & (df['landing_page'] ==␣
↪"new_page")]
non_align.shape[0]
```

[6]:
```
3893
```

[7]:
```
#find and count rows with missing values
df.isna().sum()
```

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

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.

[8]:
```
#drop egregious rows
IndexNames = df[(df['group'] == "treatment") & (df['landing_page'] ==␣
↪"old_page") | (df['group'] == "control") & (df['landing_page'] ==␣
↪"new_page")].index
df2 = df.drop(IndexNames)
```

[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

[10]:
```
#recaculate count number of unique user_ids in new dataset with offending rows␣
↪removed
df2['user_id'].nunique()
```

[10]: 290584

[11]:
```
#search for erronously repeated unique user_id
df2['user_id'].value_counts()
```

[11]:
```
773192    2
630732    1
811737    1
797392    1
795345    1
         ..
650647    1
648598    1
654741    1
652692    1
630836    1
Name: user_id, Length: 290584, dtype: int64
```

[12]:
```
#isolate row information for the repeat user_id
df2.query('user_id == "773192"')
```

[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

[13]:
```
#Remove one of the rows with a duplicate user_id
df2 = df2.drop_duplicates(['user_id'], keep='first')
```

Question: What is the probability of an individual converting regardless of the page they receive?

[14]:
```
#calculate mean conversion rate
df2['converted'].mean()
```

[14]: 0.11959708724499628

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

[15]:
```
#calculate conversion rate among control group
df2.query('group == "control"').converted.mean()
```

[15]: 0.1203863045004612

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

[16]:
```
#calculate conversion rate among treatment group
df2.query('group == "treatment"').converted.mean()
```

[16]: 0.11880806551510564

Question: What is the observed difference in conversion rate between the two landing pages?

[17]:
```
obs_diff = df2.query('group == "treatment"').converted.mean() - df2.
↪query('group == "control"').converted.mean()
obs_diff
```

[17]: -0.0015782389853555567

Question: What is the probability that an individual received the new page?

[18]:
```
#calculate probability of exposure to new page
df2.query('landing_page == "new_page"').shape[0] / df2.shape[0]
```

[18]: 0.5000619442226688

The baseline conversion rate can be considered 0.12, given this is the rate of conversion irrespective of landing page. The observed difference in conversion rate between the treatment group and control group is less than a 0.1%, suggesting the new page does not in fact lead to more conversions. Indeed, the reverse may even be true since the conversion is is higher among the control group than it is for the treatment group.

Part II - A/B Test

Challenges with A/B tests in general include answers to the following questions: > How long to run the expereriment for? > Does one stop as soon as an effect is detected? Or does the experiment need to be run for a certain amount of time? > If not, how long does one run the expeiremnt to render a decision that neither control or treatment differs in outcome?

`1.` For now, we will consider a decision needs to be made based on all the data provided. If you were to assume that the old page is better unless the new page definitievly proves to be at a Type I error rate of 5%, the null and alternative hypotheses would be stated as such:

$$H_0 : p_{new} - P_{old} <= 0$$

$$H_1 : p_{new} - P_{old} > 0$$

`2.` Under a different scenario, 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. In this context, the hypothesis would be stated as such:

$$H_0 : p_{new} - P_{old} = 0$$

$$H_1 : p_{new} - P_{old} \neq 0$$

Conditions:

-For this experiment the sample size for each page is equal to the ones in **ab_data.csv**.

Question: What is the **convert rate** for p_{new} under the null?

[19]:
```
#calculate conversion rate for new page under null hypothesis
p_new = df2['converted'].mean()
p_new
```

[19]: 0.11959708724499628

Question: What is the **convert rate** for p_{old} under the null?

[20]:
```
#calculate conversion rate for old page under null hypothesis
p_old = df2['converted'].mean()
p_old
```

[20]: 0.11959708724499628

Question: What is n_{new}?

[21]:
```
#calculate number of visits to new page
n_new = df2.query('landing_page == "new_page"')
n_new.shape[0]
```

[21]: 145310

Question: What is n_{old}?

[22]:
```
#calculate number of visits to old page
n_old = df2.query('landing_page == "old_page"')
n_old.shape[0]
```

[22]: 145274

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**.

[23]:
```
#initiate empty list
new_page_converted = []

#simulate 145310 transactions under null
for _ in range(n_new.shape[0]):
    b_samp = df2.sample(1, replace = True)
    # append the info
    new_page_converted.append(b_samp.iloc[0,4])
```

```
#convert list to numpy array
new_page_converted = np.array(new_page_converted)
new_page_converted
```

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]:
```
#initiate empty list
old_page_converted = []

#simulate 145274 transactions under null
for _ in range(n_old.shape[0]):
    b_samp = df2.sample(1, replace = True)
    # append the info
    old_page_converted.append(b_samp.iloc[0, 4])

#convert list to numpy array
old_page_converted = np.array(old_page_converted)
old_page_converted
```

Question: Find p_{new} - p_{old} for simulated values.

[27]:
```
#calculate mean difference between treatment groups
sim_diff = (new_page_converted.mean() - old_page_converted.mean())
sim_diff
```

[27]:
```
-0.001117196712849794
```

Simulate 10,000 p_{new} - p_{old} values using this same process performed earlier. Store all 10,000 values in a numpy array called **p_diffs**.

[28]:
```
#simulate 10000 new page conversion rates
new_converted_sim = np.random.binomial(n_new.shape[0], p_new, 10000)/n_new.
 ↪shape[0]

#simulate 10000 old page conversion rates
old_converted_sim = np.random.binomial(n_old.shape[0], p_old, 10000)/n_old.
 ↪shape[0]

#calculate difference of 10000 conversion rates between different landing pages
p_diffs = new_converted_sim - old_converted_sim
p_diffs
```

[28]:
```
array([-1.47496536e-03, -2.28257397e-05,  3.49436520e-03, …,
       -8.47213602e-05,  3.00704248e-04, -7.17720924e-04])
```

Question: What proportion of the **p_diffs** are greater than the actual difference observed in

ab_data.csv?

[29]:
```
#calculate p-value associated with the array of differences
(p_diffs > obs_diff).mean()
```

[29]: 0.9089

Question: 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?

p-value. This is the probability of oberserving a result as extreme, if not more than the one actually observed. A p-value of 0.9089 suggests a degree of difference in conversion rates between the two different landing pages. However, the role of chance (randomness) in creating this difference cannot be ruled out (as the value is above alpha – the threshold above which statistcal signifiance is achieved).

We could also use a built-in to achieve similar results. Though using the built-in might be easier to code, the above is a walkthrough of the ideas critical to statistical significance. The code below uses built-in functions to calculate the number of conversions for each page, as well as the number of individuals who received each page.

`n_old` and `n_new` refer the the number of rows associated with the old page and new pages, respectively.

[30]:
```
#import stats module
from statsmodels.stats.proportion import proportions_ztest

#extract conversion rates for the landing pages
convert_old = df2.query('landing_page == "old_page" & converted == "1"').
↪shape[0]
convert_new = df2.query('landing_page == "new_page" & converted == "1"').
↪shape[0]

#extract numer of individuals exposed to each page
n_old = df2.query('landing_page == "old_page"').shape[0]
n_new = df2.query('landing_page == "new_page"').shape[0]
```

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

[31]:
```
#calculate statistic, as well as associated p-value
stat, pval = proportions_ztest([convert_new, convert_old], [n_new, n_old],␣
↪alternative = 'larger')
print('{0:0.3f}'.format(stat), '{0:0.3f}'.format(pval))
```

```
-1.311 0.905
```

Question: What do the z-score and p-value computed in the previous question mean for the conversion rates of the old and new pages? Do they agree with earlier findings?

The z-score and p-value computed is consistent with earlier findings that the difference in conversion rates between the two landings pages is not large enough to conclude that either one is better.

Part III - A regression approach

1. In this final part, a regression approach will be adopted to achieve similar results to that acheived in the previous A/B test.

Question: Since each row is either a conversion or no conversion, what type of regression should you be performing in this case?

Logistic Regression.

The goal is to use **statsmodels** to fit the regression model specified in part **a.** to see if there is a significant difference in conversion based on which page a customer receives. However, a column must be created to account for the intercept, and a dummy variable column for which page each user received. For the **ab_page** column, a 1 represents when an individual receives the **treatment** and 0 if **control**.

[32]:
```
#create intercept column
df2['intercept']  = 1
```

[33]:
```
#create dummy variables using the categorical variable treatment group
df2[['control', 'ab_page']] = pd.get_dummies(df2['group'])
```

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

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

# instantiate model
logit_mod = sm.Logit(df2['converted'], df2[['intercept', 'ab_page']])

#apply model to data
results = logit_mod.fit()

#display results of model
results.summary()
```

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

[34]:
```
<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:                  Thu, 16 Jul 2020   Pseudo R-squ.:               8.077e-06
Time:                          13:57:52   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
==============================================================================
"""
```

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

The p-value associated with ab_page is 0.190. It dovetails from the one tailed test performed in Part II since a two-tailed test is now being performed.

Question: 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?

R-squared is the proportion of variation explained by the model. Capturing more variables should, in theory, add to the proportion of variation explained by the model. Adding more variables however also increases the likelihood of multicollinearity, or correlation among independent variables. This may cause pose challenges to interpretation.

Question: Does the country in which users live appear to have an impact on conversion rates?

[35]:
```
#load countries data
countries_df = pd.read_csv('./countries.csv')

#append country information onto clean dataframe
df_new = countries_df.set_index('user_id').join(df2.set_index('user_id'),␣
 ↪how='inner')
df_new.head(1)
```

[35]:
```
        country                   timestamp    group landing_page  converted  \
user_id
834778       UK  2017-01-14 23:08:43.304998  control     old_page          0

         intercept  control  ab_page
user_id
834778           1        1        0
```

[36]:
```
#count unique values in countries column
df_new['country'].value_counts()
```

[36]:
```
US    203619
UK     72466
CA     14499
Name: country, dtype: int64
```

[37]:
```
# Create the dummy variables from countries column
df_new[['CA', 'UK', 'US']] = pd.get_dummies(df_new['country'])
df_new.head()
```

[37]:
```
        country                   timestamp      group landing_page  \
user_id
834778       UK  2017-01-14 23:08:43.304998    control     old_page
928468       US  2017-01-23 14:44:16.387854  treatment     new_page
822059       UK  2017-01-16 14:04:14.719771  treatment     new_page
711597       UK  2017-01-22 03:14:24.763511    control     old_page
710616       UK  2017-01-16 13:14:44.000513  treatment     new_page

         converted  intercept  control  ab_page  CA  UK  US
user_id
834778           0          1        1        0   0   1   0
928468           0          1        0        1   0   0   1
822059           1          1        0        1   0   1   0
711597           0          1        1        0   0   1   0
710616           0          1        0        1   0   1   0
```

[38]:
```
# instantiate model
logit_mod2 = sm.Logit(df_new['converted'], df_new[['intercept', 'UK', 'US']])

#apply model to data
results2 = logit_mod2.fit()

#display results of model
results2.summary()
```

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

[38]:
```
<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:                Thu, 16 Jul 2020   Pseudo R-squ.:               1.521e-05
Time:                        13:57:56   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
UK             0.0507      0.028      1.786      0.074      -0.005       0.106
US             0.0408      0.027      1.518      0.129      -0.012       0.093
==============================================================================
"""
```

The country in which a user resides does not appear to have an effect on conversion rates. This is evident by the p-values associated with each of the variables in the model above.

Though we have considered country and page conversion rates individually, we now need to consider what is known as interaction effects to see if there are significant effects on conversion rates when considered collectively.

Question: What is the interaction between the country in which a user resides and the page to which they were exposed?

[39]:
```
#create interaction term between country (UK) and landing page (new page)
df_new['UK_ab_page'] = df_new['UK'] * df_new['ab_page']
df_new.head()
```

[39]:
```
        country                   timestamp      group landing_page  \
user_id
834778       UK  2017-01-14 23:08:43.304998    control     old_page
928468       US  2017-01-23 14:44:16.387854  treatment     new_page
822059       UK  2017-01-16 14:04:14.719771  treatment     new_page
711597       UK  2017-01-22 03:14:24.763511    control     old_page
710616       UK  2017-01-16 13:14:44.000513  treatment     new_page

         converted  intercept  control  ab_page  CA  UK  US  UK_ab_page
user_id
834778           0          1        1        0   0   1   0           0
928468           0          1        0        1   0   0   1           0
822059           1          1        0        1   0   1   0           1
711597           0          1        1        0   0   1   0           0
710616           0          1        0        1   0   1   0           1
```

[40]:
```
# instantiate model
logit_mod3 = sm.Logit(df_new['converted'], df_new[['intercept', 'ab_page',␣
↪'UK', 'UK_ab_page']])

#apply model to data
results3 = logit_mod3.fit()
```

```
#display results of model
results3.summary()
```

```
Optimization terminated successfully.
         Current function value: 0.366114
         Iterations 6
```

[40]:

```
<class 'statsmodels.iolib.summary.Summary'>
"""
                           Logit Regression Results
==============================================================================
Dep. Variable:              converted   No. Observations:               290584
Model:                          Logit   Df Residuals:                   290580
Method:                           MLE   Df Model:                            3
Date:                Thu, 16 Jul 2020   Pseudo R-squ.:               2.036e-05
Time:                        13:57:59   Log-Likelihood:            -1.0639e+05
converged:                       True   LL-Null:                   -1.0639e+05
Covariance Type:            nonrobust   LLR p-value:                    0.2278
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
intercept     -1.9876      0.009   -213.551      0.000      -2.006      -1.969
ab_page       -0.0236      0.013     -1.788      0.074      -0.050       0.002
UK            -0.0046      0.019     -0.247      0.805      -0.041       0.032
UK_ab_page     0.0345      0.026      1.307      0.191      -0.017       0.086
==============================================================================
"""
```

[41]:

```
#create interaction term between country (US) and landing page (new page)
df_new['US_ab_page'] = df_new['US'] * df_new['ab_page']
df_new.head()
```

[41]:

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

         converted  intercept  control  ab_page  CA  UK  US  UK_ab_page  \
user_id
834778           0          1        1        0   0   1   0           0
928468           0          1        0        1   0   0   1           0
822059           1          1        0        1   0   1   0           1
711597           0          1        1        0   0   1   0           0
```

```
710616            0           1         0         1    0    1    0            1

         US_ab_page
user_id
834778            0
928468            1
822059            0
711597            0
710616            0
```

[42]:

```
# instantiate model
logit_mod4 = sm.Logit(df_new['converted'], df_new[['intercept', 'ab_page',␣
 ↪'US', 'US_ab_page']])

#apply model to data
results4 = logit_mod4.fit()

#display results of model
results4.summary()
```

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

[42]:

```
<class 'statsmodels.iolib.summary.Summary'>
"""
                           Logit Regression Results
==============================================================================
Dep. Variable:              converted   No. Observations:               290584
Model:                          Logit   Df Residuals:                   290580
Method:                           MLE   Df Model:                            3
Date:                Thu, 16 Jul 2020   Pseudo R-squ.:               1.077e-05
Time:                        13:58:03   Log-Likelihood:            -1.0639e+05
converged:                       True   LL-Null:                   -1.0639e+05
Covariance Type:            nonrobust   LLR p-value:                    0.5143
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
intercept     -1.9942      0.015   -135.158      0.000      -2.023      -1.965
ab_page       -0.0019      0.021     -0.093      0.926      -0.043       0.039
US             0.0077      0.018      0.436      0.663      -0.027       0.042
US_ab_page    -0.0186      0.025     -0.746      0.456      -0.068       0.030
==============================================================================
"""
```

Consistent with earlier findings that neither country nor landing page played a significacnt role in conversion rates indivually, the two variables considered together do not appear to have a statistically significant impact on conversion rates. This is true

for both UK and US residents.

Conclusions

The goal for this project was to help a company understand the consequences of switching the landing page of its e-commerce web page. Utilising conversion rates among groups exposed to both the old landing page and the proposed new landing pageas a metric for "success", the results were analysed from multiple perspectives using probability, simulation and regression.

Results from a pure propabilitic approach suggest the old landing page has a slightly higher conversion rate among users than the new landing page. Simulation of the theoretical difference in conversion rates between the two landing pages suggests the observed difference may even be due to chance (randomness). This was consistent with the findings of the third and final apporach adopted, regression.