PROJECT 3 - ANALYZE AB TESTING

I. Introduction

For this project, I would like to check the results of an A/B test run by an e-commerce website based on datasets including two file: ab_data.csv and countries.csv.

The objective of this A/B testing is to make an important decision: whether the company should keep the old page or change to new page in regards of better conversion rate.

For that objective, I would like to use Python with methods of statistics including probability, A/B Testing and Multiple regression

II. Part I – Probability

1. General information

First of all, pandas, numpy, random and matplotlib are imported to the Notebook

AB data is read also.

```
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)
```

```
df = pd.read_csv('ab_data.csv')
df.head()
```

	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

```
df.shape
```

```
(294478, 5)
```

There are 5 columns and 294478 rows in this dataset.

```
df['user_id'].nunique()
```

```
290584
```

```
df['converted'].mean()
```

```
0.11965919355605512
```

Regarding number of users, there are 290,584 unique users in this dataset.

The proportion of users converted generally is around 0.1197.

In the relationship between groups and landing page. There are 4 possibilities:

- Treatment and old page
- Treatment and new page
- Control and old page
- Control and new page

The number of times the new_page and treatment don't match is 3893. This is the total number of [Treatment and old page] and [Control and New page].

```
tm = df.query('group == "treatment"')
newpage = df.query('landing_page == "new_page"')
tm.query('landing_page == "old_page"')['timestamp'].count() + newpage.query('group == "control"')['timestamp'].count()
```

```
3893
```

f. Do any of the rows have missing values?

```
df.info()
```

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

Regarding the null value, according to the results above, there is no missing values in this dataset.

2. Detailed probabilities

Before going deep into details, I would like to make sure that there is no overlapping and duplicates of the rows where **treatment** does not match with **new_page** or **control** does not match with **old_page**.

```
df2 = df

df2.drop(df.query("(group == 'treatment' and landing_page == 'old_page') or (group == 'control' and landing_page == 'new_page')"
```

```
# 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]
```

```
0
```

```
df2['user_id'].nunique()
```

```
290584
```

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

```
df[df2.duplicated(['user_id'], keep=False)]['user_id'].unique()
```

```
array([773192])
```

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

```
df2[df2.duplicated(['user_id'], keep=False)]
```

	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

There is one user_id that is repeated and I would like to remove that duplicate as below

```
df2 = df2.drop_duplicates(['user_id'])
```

As the dataset is clean now, I would like to check the detailed probabilities

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

```
df2.query('group == "control"')['converted'].mean()
```

```
0.1203863045004612
```

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

```
df2.query('group == "treatment"')['converted'].mean()
```

```
0.11880806551510564
```

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

```
df2.query('landing_page == "new_page"')['user_id'].nunique() / df['user_id'].nunique()
```

```
0.5000619442226688
```

Consider your results from above parts, I can tell that there is not enough evidence to conclude because:

- Just only 50% of users receiving new page. Therefore, we are not sure what the rest 50% thinking about new page.
- The difference in conversion rate between control and treatment group is not significant enough to conclude.

Therefore, A/B testing is essential in next parts.

III. Part II - A/B Test

Under null hypothesis: conversion rate of new page is equal to or less than that of old page

Under alternative hypothesis: conversion rate of new page is better.

a. What is the **conversion rate** for under the null?

```
p_new = df2['converted'].mean()
print('Conversion rate for  pnew  under the null', p_new)
```

```
Conversion rate for  pnew  under the null 0.119597087245
```

b. What is the **conversion rate** for under the null?

p_{old}

```
p_old = df2['converted'].mean()
print('Conversion rate for  pold  under the null', p_old)
```

```
Conversion rate for  pold  under the null 0.119597087245
```

Under null hypothesis test, conversion rate of old page and new page is the same as 0.1196

c. What is , the number of individuals in the treatment group?

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

```
n_new :: 145310
```

d. What is , the number of individuals in the control group?

```
n_old = df2.query('group == "control"')['user_id'].count()
print('n_old ::', df2.query('group == "control"')['user_id'].count())
```

```
n_old :: 145274
```

There are 145,310 individuals in treatment group and 145,274 individuals in control group.

Transaction simulations are executed as below. The difference between conversion rate of new page and old page in simulations is -0.017

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

```
new_page_converted = np.random.binomial(n_new,p_new)
print('new_page_converted :: ',new_page_converted)
```

```
new_page_converted ::  17262
```

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

```
old_page_converted = np.random.binomial(n_old,p_old)
print('old_page_converted :: ',old_page_converted)
```

```
old_page_converted ::  17507
```

g. Find - for your simulated values from part (e) and (f).

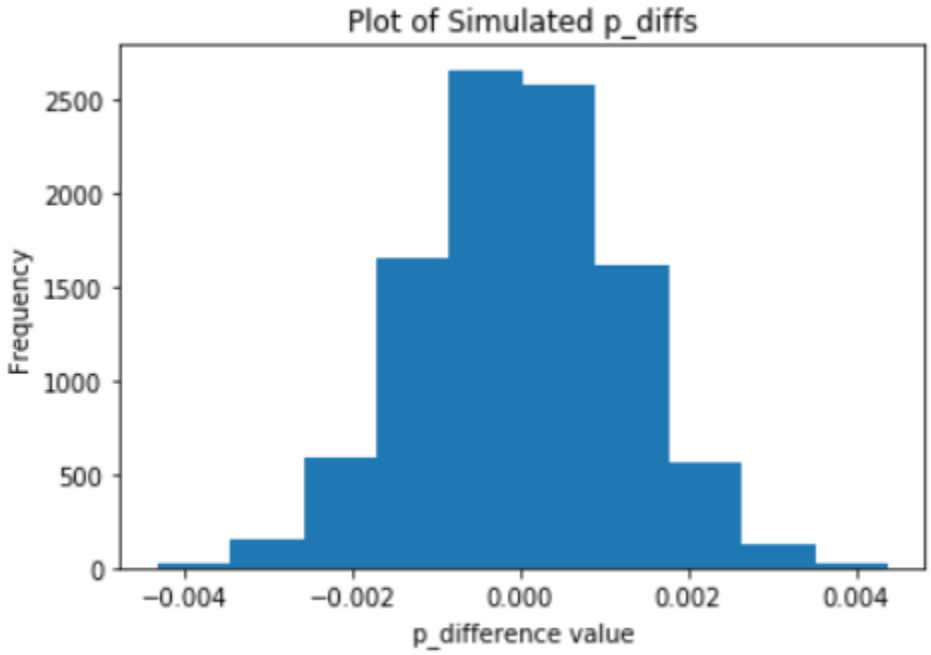
```
p_difference = (new_page_converted/n_new) - (old_page_converted/n_old)
print(p_difference)
```

```
-0.0017159064586
```

After that, I would like to 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**

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

Below is the histogram of simulated p_diffs




```
p_diff_original = df[df['landing_page'] == 'new_page']['converted'].mean() -  df[df['landing_page'] == 'old_page']['converted'].
print('orignial probability difference: ',p_diff_original)

p_diffs = np.array(p_diffs)
p_diff_proportion = (p_diff_original < p_diffs).mean()
print('proportion of p_diffs greater than orignial probability difference :: ',p_diff_proportion)
```

```
orignial probability difference:  -0.00157905659769
proportion of p_diffs greater than orignial probability difference ::  0.9045
```

The original probability difference is -0.00157905659769

The proportion of p_diffs greater than original probability difference is 0.9045. This is the p-value. **The p-value = 0.9045 > 0.05 means we fail to reject null.** Therefore, we can conclude that the new page doesnt have better conversion rate than the old page.

To make sure the conclusion is correct, we can double check the relationship between z_score and critical value.

```
import statsmodels.api as sm
import warnings
warnings.filterwarnings("ignore")

convert_old = sum(df2.query("landing_page == 'old_page'")['converted'])
convert_new = sum(df2.query("landing_page == 'new_page'")['converted'])
n_old = len(df2.query("landing_page == 'old_page'"))
n_new = len(df2.query("landing_page == 'new_page'"))
```

```
z_score, p_value = sm.stats.proportions_ztest([convert_old, convert_new], [n_old, n_new], alternative='smaller')
print('z_score :: ',z_score)
print('p_value :: ',p_value)

from scipy.stats import norm
# significant of z-score
print(norm.cdf(z_score))

# for our single-sides test, assumed at 95% confidence level, we calculate:
print('critical value ::', norm.ppf(1-(0.05)))
```

```
z_score ::  1.31092419842
p_value ::  0.905058312759
0.905058312759
critical value :: 1.64485362695
```

We can see that z_score = 1.311 < critical value = 1.644. Therefore, we fail to reject null. This means the new page doesn't make better conversion rate than the old page.

IV. Part III – Regression

Another method similar to hypothesis test is logistic regression.

1. Logistic regression with regard to group (control)

I would like to use **statsmodels** to fit the regression model to see if there is a significant difference in conversion rate.

A column for the intercept and a dummy variable column are created. I add an **intercept** column, as well as an **ab_page** column, which is 1 when an individual receives the **treatment** and 0 if **control**.

```
df2['intercept'] = 1
df2[['control', 'ab_page']]=pd.get_dummies(df2['group'])
df2.drop(labels=['control'], axis=1, inplace=True)
df2.head()
```

	user_id	timestamp	group	landing_page	converted	intercept	ab_page
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

Logistics regression are executed as below.

```
import statsmodels.api as sm
import scipy.stats as stats
logit = sm.Logit(df2['converted'],df2[['intercept' ,'ab_page']])
results = logit.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.

```
stats.chisqprob = lambda chisq, df: stats.chi2.sf(chisq, df)
results.summary()
```

Here’s the result after regression

Logit Regression Results

Dep. Variable:	converted	**No. Observations:**	290584
Model:	Logit	**Df Residuals:**	290582
Method:	MLE	**Df Model:**	1
Date:	Tue, 31 Mar 2020	**Pseudo R-squ.:**	8.077e-06
Time:	21:40:17	**Log-Likelihood:**	-1.0639e+05
converged:	True	**LL-Null:**	-1.0639e+05
		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

The **p-value of ab_page** is **0.19** which is larger than 0.05. Therefore, **we fail to reject null hypothesis**.
In addition, in this part, hypothesis is about H0 : p = p0 vs HA : p = p0
In previous part, H0: p_new - p_old =< 0 while HA: p_new - p_old > 0
That's why p-value is different.

2. **Logistic regression with regard to group (control) and countries data**

It is essential to consider other factors such as demographics including age, gender, location.... so that we can have broader view to analyze and to conclude hypothesis tests. There may be some disadvantages like whether these factors are reliable enough or may cause some conflicts in regression model.
In this part, I would like to take countries data into consideration that whether country had impacts on conversion rate or not.

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

# Convert country values into dummy varaibles
df_new[['CA','UK','US']]=pd.get_dummies(df_new['country'])
df_new.head()

# Doing regression again

mod = sm.Logit(df_new['converted'], df_new[['intercept', 'CA', 'UK','ab_page']])
results = mod.fit()
results.summary()
```

Dep. Variable:	converted	**No. Observations:**	290584
Model:	Logit	**Df Residuals:**	290580
Method:	MLE	**Df Model:**	3
Date:	Wed, 01 Apr 2020	**Pseudo R-squ.:**	2.323e-05
Time:	00:26:43	**Log-Likelihood:**	-1.0639e+05
converged:	True	**LL-Null:**	-1.0639e+05
		LLR p-value:	0.1760

	coef	std err	z	P>|z|	[0.025	0.975]
intercept	-1.9893	0.009	-223.763	0.000	-2.007	-1.972
CA	-0.0408	0.027	-1.516	0.130	-0.093	0.012
UK	0.0099	0.013	0.743	0.457	-0.016	0.036
ab_page	-0.0149	0.011	-1.307	0.191	-0.037	0.007

As you can see above, even with country values, the logistic regression still shows us that p-value of ab_page = 0.191 > 0.05. This means we fail to reject Null Hypothesis. This means that the new page doesn't make significantly better conversion rate than the old page. Therefore, it is suggested to keep to old page.

In addition, I would like to know if there are any significant effects on conversion in regard to the interaction between pages and countries. Then I would like to join data and create dummy variables as well.

```
# Join data
df_new['US_ind_ab_page'] = df_new['US']*df_new['ab_page']
df_new['CA_ind_ab_page'] = df_new['CA']*df_new['ab_page']
df_new.head()


# Regression
logit_h = sm.Logit(df_new['converted'], df_new[['intercept', 'ab_page', 'US', 'CA', 'US_ind_ab_page', 'CA_ind_ab_page']])
results = logit_h.fit()
results.summary()
```

Logit Regression Results

Dep. Variable:	converted	**No. Observations:**	290584
Model:	Logit	**Df Residuals:**	290578
Method:	MLE	**Df Model:**	5
Date:	Wed, 01 Apr 2020	**Pseudo R-squ.:**	3.482e-05
Time:	00:46:00	**Log-Likelihood:**	-1.0639e+05
converged:	True	**LL-Null:**	-1.0639e+05
		LLR p-value:	0.1920

	coef	std err	z	P>|z|	[0.025	0.975]
intercept	-1.9922	0.016	-123.457	0.000	-2.024	-1.961
ab_page	0.0108	0.023	0.475	0.635	-0.034	0.056
US	0.0057	0.019	0.306	0.760	-0.031	0.043
CA	-0.0118	0.040	-0.296	0.767	-0.090	0.066
US_ind_ab_page	-0.0314	0.027	-1.181	0.238	-0.084	0.021
CA_ind_ab_page	-0.0783	0.057	-1.378	0.168	-0.190	0.033

We can see that even within each country, the new page doesn't make better conversion rate because p-values of US_ind_ab_page and CA_ind_ab_page are still higher than 0.05. Therefore, we still fail to reject null hypothesis.

V. Conclusion

In conclusion, it is suggested the company should keep the old page instead of new page because regardless of trying different method statistics, we fail to reject null hypothesis.