

▼ 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.

```
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:

```
#read dataset as df
df=pd.read_csv(r"/content/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	

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

```
#number of rows
df.shape[0]
```

294478

c. The number of unique users in the dataset.

```
# unique users in dataset
len(df['user_id'].unique())
```

290584

d. The proportion of users converted.

```
#proportion of users converted
convert_overall = df.converted.mean()
convert_overall
```

0.11965919355605512

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

```
df.query('(group == "treatment" and landing_page != "new_page") or (group != "treatment" and
```

3893

f. Do any of the rows have missing values?

```
df.isnull().values.any()
```

```
False
```

Unsupported Cell Type. Double-Click to inspect/edit the content.

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**.

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

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

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

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

```
290584
```

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

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

```
1899    773192
2893    773192
Name: user_id, dtype: int64
```

▼ result

user_id 773192 has repeated

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

```
df2[df2['user_id'] == 773192]
```

	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	

▼ result

Different time stamp

d. Remove **one** of the rows with a duplicate **user_id**, but keep your dataframe as **df2**.

```
df2 = df2.drop(df2[(df2.user_id == 773192) & (df2['timestamp'] == '2017-01-09 05:37:58.781806')])
df2[df2['user_id'] == 773192]
```

	user_id	timestamp	group	landing_page	converted	
2893	773192	2017-01-14 02:55:59.590927	treatment	new_page	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?

```
converted_users2 = float(df2.query('converted == 1')['user_id'].nunique())
p2 = converted_users2/float(df2.shape[0])
print("The probability of an individual converting regardless of the page they receive is {0:".format(p2))
```

The probability of an individual converting regardless of the page they receive is 11.9

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

```
converted_controlusers2 = float(df2.query('converted == 1 and group == "control"')['user_id'].nunique())
control_users2 = float(df2.query('group == "control"')['user_id'].nunique())
cp2 = converted_controlusers2 / control_users2
print(" Given that an individual was in the control group, the probability they converted is {0:".format(cp2))
```

Given that an individual was in the control group, the probability they converted is 1



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

```
converted_controlusers2 = float(df2.query('converted == 1 and group == "treatment"')['user_id'].nunique())
treat_users2 = float(df2.query('group == "treatment"')['user_id'].nunique())
tp2 = converted_controlusers2 / treat_users2
print(" Given that an individual was in the treatment group, the probability they converted is 11.11%")
```

Given that an individual was in the treatment group, the probability they converted is 11.11%



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

```
new_page_users2 = float(df2.query('landing_page == "new_page"')['user_id'].nunique())
Newpage_p2 = new_page_users2 / float(df2.shape[0])
print("The probability that an individual received the new page is {0:.2%}".format(Newpage_p2))
```

The probability that an individual received the new page is 50.01%

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.

```
new_c2 = float(df2.query('converted == 1 and landing_page == "new_page"')['user_id'].nunique())
new_users2 = float(df2.query('landing_page == "new_page"')['user_id'].nunique())
print(" Given that an individual was in new landing page, the probability they converted is {0:.2%}".format(new_c2/new_users2))
```

Given that an individual was in new landing page, the probability they converted is 11.11%



▼ 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.

Null hypothesis is $H_0: p_{new} - p_{old} \leq 0$

Alternative hypothesis is $H_1: p_{new} - p_{old} > 0$

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?

```
# Compute converted success rate, which equals to the converted success rate regardless of page
p_new = round(float(df2.query('converted == 1')['user_id'].nunique())/float(df2['user_id'].nunique()), 4)
p_new
```

0.1196

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

```
# Compute old converted success rate, which equals to the converted success rate regardless of page
p_old = round(float(df2.query('converted == 1')['user_id'].nunique())/float(df2['user_id'].nunique()), 4)
p_old
```

```
p_old
```

```
0.1196
```

c. What is n_{new} ?

```
#Compute the number of unique users who has new page using df2 dataframe
N_new = df2.query('landing_page == "new_page"')['user_id'].nunique()
N_new
```

```
145310
```

d. What is n_{old} ?

```
#Compute the number of unique users who has old page using df2 dataframe
N_old = df2.query('landing_page == "old_page"')['user_id'].nunique()
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**.

```
#Simulate n_new transactions with a convert rate of p_new under the null
new_page_converted = np.random.choice([0,1],N_new, p=(p_new,1-p_new))

new_page_converted
```

```
array([1, 1, 1, ..., 1, 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**.

```
#Simulate n_old transactions with a convert rate of p_old under the null
old_page_converted = np.random.choice([0,1],N_old, p=(p_old,1-p_old))

old_page_converted
```

```
array([1, 1, 1, ..., 1, 1, 1])
```

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

```
#Compute the probability of new page converted rate
new_page_converted.mean()
```

```
0.881081825063657
```

```
#Compute the probability of old page converted rate
old_page_converted.mean()
```

```
0.8803502347288572
```

```
#Find pnew - pold for your simulated values from part (e) and (f).
new_page_converted.mean() - old_page_converted.mean()
```

```
0.0007315903347997477
```

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**.

```
#Import timeit package
import timeit
start = timeit.default_timer()
```

```
# Sampling distribution
p_diffs = []
new_page_converted= np.random.binomial(N_new, p_new, 10000)
old_page_converted = np.random.binomial(N_old, p_old, 10000)
p_diffs = new_page_converted/N_new - old_page_converted/N_old
p_diffs = np.array(p_diffs)
```

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.

```
p_diffs = np.array(p_diffs)
```

```
plt.hist(p_diffs)
```



```
(array([ 28., 113., 530., 1490., 2531., 2665., 1687., 745., 188.,
        23.]),
 array([-0.0043451, -0.00349715, -0.00264919, -0.00180124, -0.00095328,
        -0.00010533, 0.00074263, 0.00159058, 0.00243854, 0.00328649,
        0.00413445]),
 <a list of 10 Patch objects>)
```



```
# Create number of users with all new_page users and all old_page users
convert_new = df2.query('converted == 1 and landing_page == "new_page"')['user_id'].count()
convert_old = df2.query('converted == 1 and landing_page == "old_page"')['user_id'].count()

# Compute actual converted rate
actual_cvt_new = float(convert_new)/ float(N_new)
actual_cvt_old = float(convert_old)/ float(N_old)
```

```
# Compute observed difference in converted rate
obs_diff = actual_cvt_new - actual_cvt_old

# Display observed difference in converted rate
obs_diff
```

```
-0.0015782389853555567
```

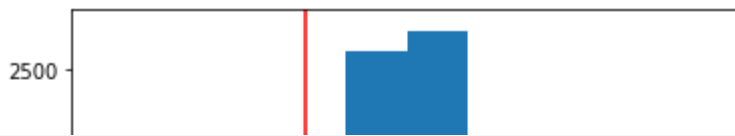
j. What proportion of the **p_diffs** are greater than the actual difference observed in **ab_data.csv**?

```
# create distribution under the null hypothesis
null_vals = np.random.normal(0, p_diffs.std(), p_diffs.size)
```

```
#Plot Null distribution
plt.hist(null_vals)
#Plot vertical line for observed statistic

plt.axvline(x=obs_diff,color='red')
```

<matplotlib.lines.Line2D at 0x7f81e2b490d0>



```
#Compute proportion of the p_diffs are greater than the actual difference observed in  
(null_vals > obs_diff).mean()
```

0.906



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. Type I error rate of 5%, and $P_{old} > \alpha$, we fail to reject the null. Therefore, the data show, with a type I error rate of 0.05, that the old page has higher probability of convert rate than new 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.

```
import statsmodels.api as sm  
  
convert_old = df2.query('landing_page == "old_page" & converted == 1')['user_id'].count()  
  
convert_new = df2.query('landing_page == "new_page" & converted == 1')['user_id'].count()  
  
n_old = df2.query('group == "control"')['user_id'].count()  
  
n_new = df2.query('group == "treatment"')['user_id'].count()
```

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

```
z_score, p_value = sm.stats.proportions_ztest(np.array([convert_new, convert_old]), np.array([n_new, n_old]))
```

```
z_score, p_value
```

```
(-1.3109241984234394, 0.9050583127590245)
```

```
from scipy.stats import norm
```

```
norm.cdf(z_score)
```

```
0.09494168724097551
```

```
norm.ppf(1-(0.05/2))
```

```
1.959963984540054
```

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. Since the z-score of -1.3109241984234394 does not exceed the critical value of 1.959963984540054, we fail to reject the null hypothesis that old page users has a better or equal converted rate than old page users. Therefore, the converted rate for new page and old page have no difference. This result is the same as parts J. and K. result.

▼ Part III - A regression approach

1. In this final part, you will see that the result you achieved in the previous A/B test 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?

Put your answer here. Logistic Regression, due to the fact that response variable is categorical variable. Logistic regression is multiple regression but with an outcome variable that is a categorical variable and predictor variables that are continuous

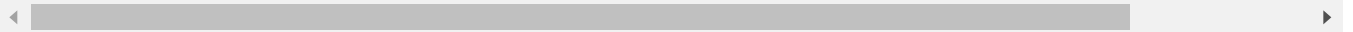
b. The goal is to use **statsmodels** to fit the regression model you specified in part **a.** to see if there is a significant difference in conversion based on which page a customer receives. However, you first need to create a column for the intercept, and create a dummy variable column for which 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**.

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

```
#create a dummy variable column for which page each user received  
df2= df2.join(pd.get_dummies(df2['landing_page']))
```

```
df2['ab_page'] = pd.get_dummies(df['group']) ['treatment']  
df2.head()
```

	user_id	timestamp	group	landing_page	converted	intercept	new_page	old_
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.

```
#Create Logit regression model for converted variable and ab_page, and us control as baseline  
lo = sm.Logit(df2['converted'], df2[['intercept','ab_page']])
```

```
result = lo.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.

```
print (result.summary())
```

Logit Regression Results

```

=====
Dep. Variable:          converted    No. Observations:          290584
Model:                  Logit        Df Residuals:              290582
Method:                  MLE         Df Model:                  1
Date:                   Sun, 16 Oct 2022    Pseudo R-squ.:            8.077e-06
Time:                   02:05:42          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**?

Put your answer here. The p-value associated with ab_page is 0.190. The null in c-e part is that there is no difference between the treatment and control group. Alternative hypotheses is that there is difference between between the treatment and control group


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?

Put your answer here. Other factor can be the time(timestamp variable). We can check if the converted rate depends on certain time of the day or certain day when user browse the website. For timestamp variable, we can further convert time as categorical variable which includes "Morning, afternoon, and evening", or "weekday and weekend". Disadvantage for adding additional terms into regression model is that it will make interpretate the model more complex and also, if new terms are dependable variable with the exisiting explanatory term, we need to add higher order term to help predict the result better.

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 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.

```
countries = pd.read_csv(r"/content/countries.csv")
countries.head()
```



	user_id	country
0	834778	UK
1	928468	US
2	822059	UK
3	711597	UK
4	710616	UK

```
#Join ab dataset with country dataset
df3 = df2.merge(countries, on = 'user_id', how='left')
df3.head()
```

	user_id	timestamp	group	landing_page	converted	intercept	new_page	old_
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	



```
countries['country'].unique()
```

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

```
### Create the necessary dummy variables
df3[['CA', 'UK', 'US']] = pd.get_dummies(df3['country'])
```

```
df3 = df3.drop(df3['CA'])

#Create intercept variable
df3['intercept'] = 1

#Create Logit regression model for converted and country, and us CA and old page as baseline
logit3 = sm.Logit(df3['converted'], df3[['intercept','new_page','UK','US']])
result = logit3.fit()
result.summary()
```

Optimization terminated successfully.

Current function value: 0.366115

Iterations 6

Logit Regression Results

Dep. Variable:	converted	No. Observations:	290582
Model:	Logit	Df Residuals:	290578
Method:	MLE	Df Model:	3
Date:	Sun, 16 Oct 2022	Pseudo R-squ.:	2.325e-05
Time:	02:06:11	Log-Likelihood:	-1.0639e+05
converged:	True	LL-Null:	-1.0639e+05
Covariance Type:	nonrobust	LLR p-value:	0.1757

	coef	std err	z	P> z	[0.025	0.975]
intercept	-2.0300	0.027	-76.248	0.000	-2.082	-1.978
new_page	-0.0150	0.011	-1.308	0.191	-0.037	0.007
UK	0.0506	0.028	1.784	0.075	-0.005	0.106
US	0.0408	0.027	1.516	0.129	-0.012	0.093

```
1/np.exp(-0.0150),np.exp(0.0506),np.exp(0.0408)
```

```
(1.015113064615719, 1.0519020483004984, 1.0416437559600236)
```

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.

```
### Fit Your Linear Model And Obtain the Results
#Create a new intereacton variable between new page and country US and UK
df3['UK_new_page'] = df3['new_page']* df3['UK']
df3['US_new_page'] = df3['new_page']* df3['US']
```

```
#Create logistic regression for the intereaction variable between new page and country using
logit4 = sm.Logit(df3['converted'], df3[['intercept','new_page','UK_new_page','US_new_page'],
result4 = logit4.fit()
result4.summary()
```

Optimization terminated successfully.
Current function value: 0.366110
Iterations 6

Logit Regression Results

Dep. Variable:	converted	No. Observations:	290582
Model:	Logit	Df Residuals:	290576
Method:	MLE	Df Model:	5
Date:	Sun, 16 Oct 2022	Pseudo R-squ.:	3.484e-05
Time:	02:06:26	Log-Likelihood:	-1.0639e+05
converged:	True	LL-Null:	-1.0639e+05
Covariance Type:	nonrobust	LLR p-value:	0.1917

	coef	std err	z	P> z	[0.025	0.975]
intercept	-2.0040	0.036	-55.008	0.000	-2.075	-1.933
new_page	-0.0674	0.052	-1.297	0.195	-0.169	0.034
UK_new_page	0.0783	0.057	1.378	0.168	-0.033	0.190
US_new_page	0.0469	0.054	0.871	0.384	-0.059	0.152
UK	0.0118	0.040	0.296	0.767	-0.066	0.090
US	0.0176	0.038	0.466	0.641	-0.056	0.091

```
#exponentiated the CV to inteprete the result  
np.exp(result4.params)
```

```
intercept      0.134794  
new_page       0.934776  
UK_new_page    1.081428  
US_new_page    1.047978  
UK             1.011854  
US             1.017705  
dtype: float64
```

▼ Interpreting Result:

From the above Logit Regression Results, we can see the coefficient of intereaction variable "UK_new_page" and "US_new_page" are different from the coefficient of new_page itself.

Also,only intercept's p-value is less than 0.05, which is statistically significant enough for converted rate. Other variable in the summary are not statistically significant. Additionally, Z-score for all X variables are not large enough to be significant for predicting converted rate.

Therefore, the country a user lives is not significant on the converted rate considering the page the user land in.

For every unit for new_page decreases, convert will be 7.0% more likely to happen, holding all other variable constant.

Convert is 1.08 times more likely to happen for UK and new page users than CA and new page users, holding all other variable constant.

Convert is 1.04 times more likely to happen for US and new page users than CA and new page users, holding all other variable constant.

Convert is 1.18 % more likely to happen for the users in UK than CA, holding all other variable constant.

Convert is 1.76 % more likely to happen for the users in US than CA, holding all other variable constant.

```
#Import sklearn model to split, test and score data,and fit data model
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import confusion_matrix, precision_score, recall_score, accuracy_score
from sklearn.model_selection import train_test_split
```

```
#Define X and Y variable
x = df3[['new_page','UK_new_page','US_new_page','UK','US']]
y = df3['converted']

#Split data into train and test data
X_train, X_test, y_train, y_test = train_test_split(x, y, test_size=0.1, random_state=0)
```

```
lm = LogisticRegression()
```

```
lm.fit(X_train,y_train) # fit the train data
```

```
LogisticRegression()
```

```
print(lm.score(X_test,y_test))
```

```
0.8819298668226712
```

```
#change datatype of timestamp to datetime for better analysis
df3['timestamp']=pd.to_datetime(df3['timestamp'])
```

```
#calculate duration of our experiment
exp_duration=df3['timestamp'].max()-df3['timestamp'].min()
print(exp_duration)
```

```
21 days 23:59:49.081927
```

Conclusions

None of the variables have significant p-values. Therefore, we will fail to reject the null and conclude that there is not sufficient evidence to suggest that there is an interaction between country and page received that will predict whether a user converts or not.

for statistical significance: Based on A/B test and logistic regression the new page has no effect on conversion rate and from logistic regression the country has no effect on conversion rate too so the website should keep the old page or run the experiment for more time

for practical significance: the experiment should run for a longer time than almost 22 days so we can judge the new page

[Colab paid products](#) - [Cancel contracts here](#)

✓ 0s completed at 5:06 AM

