A/B Test (python).

In [54]:
```
import pandas as pd
import numpy as np
import random
import matplotlib.pyplot as plt
%matplotlib inline
random.seed(42)
```

In [55]:
```
df=pd.read_csv('ab_data.csv')
df.head(2)
```

Out[55]:

	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

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

In [56]:
```
df.shape[0]
```

Out[56]:
```
294478
```

c. The number of unique users in the dataset.

In [57]:
```
df.user_id.nunique()
```

Out[57]:
```
290584
```

d. The proportion of users converted.

In [58]:
```
(df[df['converted']==1].converted.count()/df.user_id.nunique())*100
```

Out[58]:
```
12.126269856564711
```

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

In [59]:
```
print("treatment don't match."+": "+str(df.query('group=="treatment" and landing_page !="n
print("control don't match."+": "+str(df.query('group=="control" and landing_page !="old_p
```

```
treatment don't match.: 1965
control don't match.: 1928
```

In [60]:
```
print("The number of times the new_page and treatment don't match."+": "+str(1965+1928))
```

```
The number of times the new_page and treatment don't match.: 3893
```

f. Do any of the rows have missing values?

In [61]:
```
df.isnull().any()
```

Out[61]:
```
user_id         False
timestamp       False
```

```
group           False
landing_page    False
converted       False
dtype: bool
```

2. For the rows where **treatment** does not match with **new_page** or **control** does not match with **old_page**, we cannot be sure if this row truly received the new or old page. Use **Quiz 2** in the classroom to figure out 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 [62]:
```
df2=df.copy()
i=df2.query('group=="treatment" and landing_page !="new_page"').index
i2=df2.query('group=="control" and landing_page !="old_page"').index
df2=df2.drop(i)
df2=df2.drop(i2)
```

In [63]:
```
# Double Check all of the correct rows were removed - this should be 0
df2[((df2['group'] == 'treatment') == (df2['landing_page'] == 'new_page')) == False].shape
```

Out[63]:
```
0
```

a. How many unique **user_id**s are in **df2**?

In [64]:
```
df2.user_id.nunique()
```

Out[64]:
```
290584
```

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

In [65]:
```
df2[df2['user_id'].duplicated()]
```

Out[65]:

	user_id	timestamp	group	landing_page	converted
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 [66]:
```
df2[df2['user_id'].duplicated()]
```

Out[66]:

	user_id	timestamp	group	landing_page	converted
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 [67]:
```
df2=df2.drop_duplicates(subset=['user_id'],keep='first', inplace=False)
df2[df2['user_id'].duplicated()]
```

Out[67]:

user_id	timestamp	group	landing_page	converted

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

In [68]:

```
df2.converted.sum()/df2.shape[0]
```

Out[68]:
```
0.11959708724499628
```

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

In [69]:
```
df2[df2['group']=='control'].converted.sum()/df2[df2['group']=='control'].converted.count
```

Out[69]:
```
0.1203863045004612
```

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

In [70]:
```
df2[df2['group']=='treatment'].converted.sum()/df2[df2['group']=='treatment'].converted.co
```

Out[70]:
```
0.11880806551510564
```

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

In [71]:
```
df2[df2['landing_page']=='new_page'].landing_page.count()/df2.shape[0]
```

Out[71]:
```
0.50006194422266881
```

NOTE

From those results there's no evidence for that the new treatment page leads to more conversions,but by those results i cannot determine that the old page is better as nearly 12% convertion rate in each group , that lead us to ask more essential questions like (depending on the people themselves ,may change aversion and novelty effect effect in that and also this depend on their age or others) ,(12% convertion rate for treatment group ,lead me to ask why they converted , they may be didnot like somethings in the new page or it'snot Comfortable on the eyes).

the propability values show us there is nearly the same converted rate for both group.

In my personal opinion i can take the decision by given more time to test that and more different people and i can put the age in my consideration and the nationality for the crowd and more .

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.

H0:p_{old} >= p_{new}

H1:p_{new} > p_{old}

H0: p_{new} - p_{old} <=0

H1 : 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.

H0:p_{old} = p_{new}

H1:p_{new} != p_{old}

H0: p_{new} - p_{old} =0

H1 : p_{new} - p_{old} !=0

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 **conversion rate** for p_{new} under the null?

In [72]:
```
# under the null we assume that p_new=p_old
p_new=df2.converted.mean()
```

```
p_new
```

Out[72]:
```
0.11959708724499628
```

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

In [73]:
```
# under the null we assume that p_new=p_old
p_old=df2.converted.mean()
p_old
```

Out[73]:
```
0.11959708724499628
```

c. What is n_{new}, the number of individuals in the treatment group?

In [74]:
```
n_new=df2.query('group =="treatment"').user_id.nunique()
n_new
```

Out[74]:
```
145310
```

d. What is n_{old}, the number of individuals in the control group?

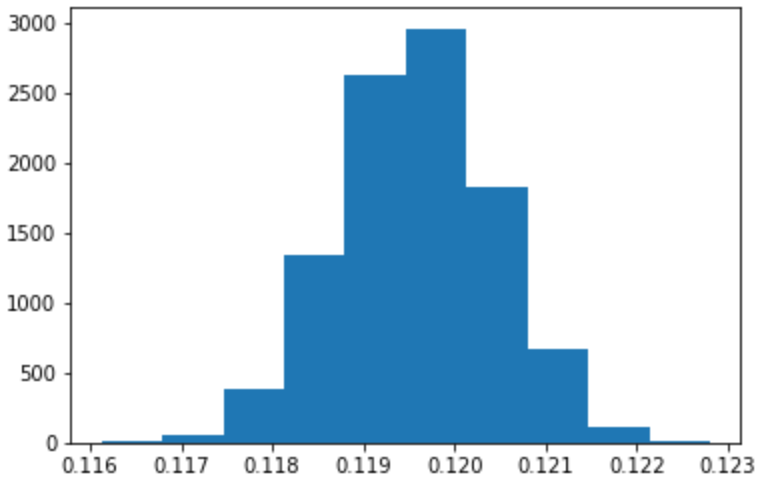
In [75]:
```
n_old=df2.query('group =="control"').user_id.nunique()
n_old
```

Out[75]:
```
145274
```

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

In [76]:
```
# they have plotting shape according to central limit theory.
new_converted_simulation = np.random.binomial(n_new, p_new, 10000)/n_new
plt.hist(new_converted_simulation)
```

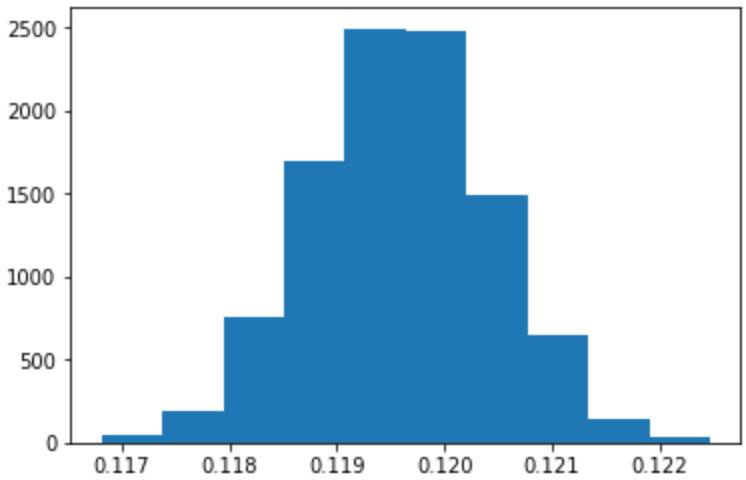
Out[76]:
```
(array([    5.,    56.,   383.,  1337.,  2632.,  2958.,  1825.,   670.,
         116.,    18.]),
 array([ 0.11612415,  0.11679237,  0.1174606 ,  0.11812883,  0.11879705,
         0.11946528,  0.12013351,  0.12080173,  0.12146996,  0.12213819,
         0.12280641]),
 <a list of 10 Patch objects>)
```



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

In [77]:
```
old_converted_simulation = np.random.binomial(n_old, p_old, 10000)/n_old
plt.hist(old_converted_simulation)
```

Out[77]:
```
(array([   47.,   194.,   762.,  1692.,  2496.,  2485.,  1492.,   654.,
         148.,    30.]),
 array([ 0.11681375,  0.11737888,  0.11794402,  0.11850916,  0.1190743 ,
         0.11963944,  0.12020458,  0.12076972,  0.12133486,  0.1219    ,
         0.12246513]),
 <a list of 10 Patch objects>)
```



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

In [78]:
```
diff= new_converted_simulation - old_converted_simulation
diff.mean()
```

Out[78]:
```
-2.5551972157581913e-06
```

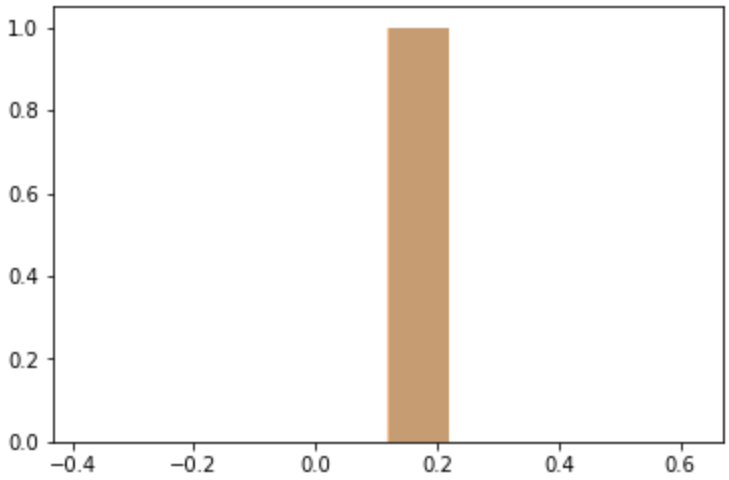
h. Create 10,000 p_{new} - p_{old} values using the same simulation process you used in parts (a) through (g) above. Store all 10,000 values in a NumPy array called **p_diffs**.

In [96]:
```
sample=df2.sample(df2.shape[0],replace=True)
bootstamp=sample.sample(df2.shape[0],replace=True)
```

In [97]:
```
#bootsamp=sample.sample(df2.shape[0],replace=True)
new_page=bootstamp.query('group =="treatment"')
old_page=bootstamp.query('group =="control"')
p_diffs = []
new_converted= np.random.binomial(new_page.user_id.nunique(),new_page.converted.mean(), 1
old_converted= np.random.binomial(old_page.user_id.nunique(),old_page.converted.mean(), 1
p_diffs = new_converted - old_converted
```

In [99]:
```
plt.hist(np.array(new_page.converted.mean()),alpha=0.5)
plt.hist(np.array(old_page.converted.mean()),alpha=0.5)
```

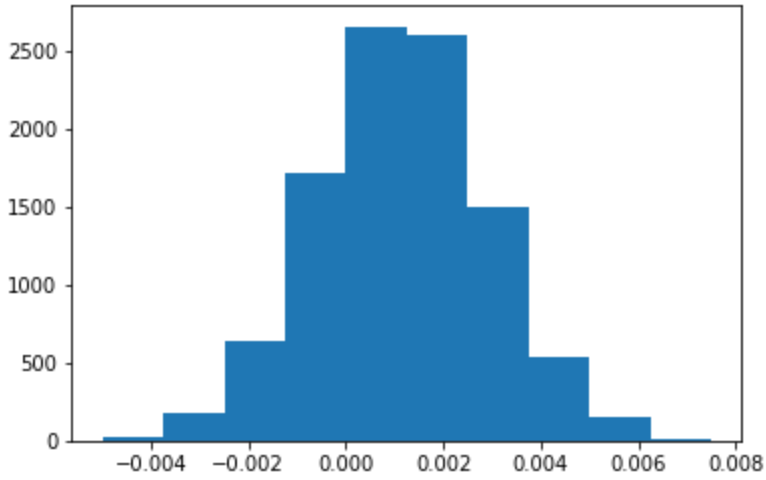
Out[99]:
```
(array([ 0.,  0.,  0.,  0.,  0.,  1.,  0.,  0.,  0.,  0.]),
 array([-0.38115337, -0.28115337, -0.18115337, -0.08115337,  0.01884663,
         0.11884663,  0.21884663,  0.31884663,  0.41884663,  0.51884663,
         0.61884663]),
 <a list of 10 Patch objects>)
```



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 [100…
```
p_diffs=np.array(p_diffs)
plt.hist(p_diffs)
```

Out[100…
```
(array([   25.,   176.,   639.,  1717.,  2651.,  2598.,  1491.,   541.,
          146.,    16.]),
 array([ -4.99818231e-03,  -3.74871097e-03,  -2.49923963e-03,
         -1.24976829e-03,  -2.96946845e-07,   1.24917439e-03,
          2.49864573e-03,   3.74811707e-03,   4.99758841e-03,
          6.24705975e-03,   7.49653109e-03]),
 <a list of 10 Patch objects>)
```



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

In [101…
```
obs_diff= df2[df2['group'] == 'treatment']['converted'].mean() - df2[df2['group'] == 'cont
obs_diff
```

Out[101…
```
-0.0015782389853555567
```

In [102…
```
#p_value
(p_diffs>obs_diff).mean()
```

Out[102…
```
0.90489999999999996
```

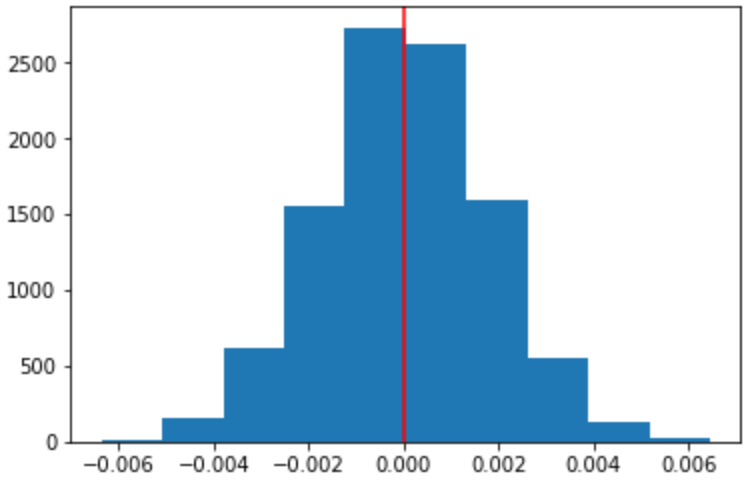
In [104…
```
null_values=np.random.normal(0,p_diffs.std(),10000)
#p_value for the first hypothesis H1: p_new>p_old
```

```
(null_values>obs_diff).mean()
```

Out[104…
```
0.8175
```

In [105…
```
plt.hist(null_values)
plt.axvline(diff.mean(),color='r')
```

Out[105…
```
<matplotlib.lines.Line2D at 0x7fe453319320>
```



In [106…
```
# for other hypothesis H0: p_new=p_old
bootsamp=sample.sample(100000,replace=True)
sample_mean=bootsamp.converted.mean()
null_mean=df2.converted.mean()
(null_values<sample_mean).mean()+(null_values>null_mean+(null_mean - sample_mean)).mean()
```

Out[106…
```
1.0
```

k. Please explain using the vocabulary you've learned in this course 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.

I computed the p_value for the hypothesis which means is the probability of finding the observed, or more extreme, results when the null hypothesis (H0) of a study question is true.

the more p_value the less statistical difference between the two pages . for this case p_value > 0.05 which lead me to fail in rejection the null hypothsis.[pold>=pnew]

for null hypothesis H0:p_new=p_old , the p_value is 1 which lead us to the statistic is in null values which lead us again to fail to reject null hypothesis.

z-score test.

In [107…
```
import statsmodels.api as sm
convert_old = df2.query('group=="control"').converted.sum()
convert_new = df2.query('group=="treatment"').converted.sum()
n_new = df2.query('group=="treatment"').user_id.nunique()
n_old = df2.query('group=="control"').user_id.nunique()
```

```
/opt/conda/lib/python3.6/site-packages/statsmodels/compat/pandas.py:56: FutureWarning: The pandas.core.datetools module is deprecated and will be removed in a future version. Please use the pandas.tseries module instead.
  from pandas.core import datetools
```

In [108…

```
from statsmodels.stats.proportion import proportions_ztest
stat,pval = proportions_ztest([convert_new, convert_old], [n_new, n_old], alternative = '
print('{0:0.3f}'.format(pval))
```

```
0.905
```

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.

from this value i can say the null hypthsis as p_value> 0.05 . and they also agree with the findings in j and k parts (also not the same value but the same meaning as p_value in both > 0.05 which lead us to fail to reject null hypothesis)

A bigger p-value than the error ratio means that there is stronger evidence in favor of the null hypothesis.

Part III - A regression approach

1. In this final part, you will see that the result you achieved in the A/B test in Part II above 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.

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 in df2 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 [109…

```
df2['intercept']=1
df2['ab_page'] = pd.get_dummies(df2['landing_page'])['new_page']
df2['landing_page']=pd.get_dummies(df['landing_page'])
df2.head()
```

Out[109…

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

In [110…

```
print(df2['landing_page'].mean())
print(df2['converted'].mean())
print(df2.groupby(['landing_page','ab_page'])['converted'].mean())
```

```
0.500061944223
0.119597087245
```

```
landing_page  ab_page
0             0          0.120386
1             1          0.118808
Name: converted, dtype: float64
```

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

In [111…
```
import statsmodels.api as sm
logit_mod = sm.Logit(df2['converted'],df2[['intercept', 'ab_page']])
results = logit_mod.fit()
#results.summary()
```

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

In [112…
```
from scipy import stats
stats.chisqprob = lambda chisq, df2: stats.chi2.sf(chisq, df2)
results.summary()
```

Out[112…

Logit Regression Results

Dep. Variable:	converted	**No. Observations:**	290584
Model:	Logit	**Df Residuals:**	290582
Method:	MLE	**Df Model:**	1
Date:	Sat, 20 Feb 2021	**Pseudo R-squ.:**	8.077e-06
Time:	13:47:20	**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

In [113…
```
# this is multiplicable change in the odd
np.exp(-0.0150)
```

Out[113…
```
0.98511193960306265
```

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

from multiplicable change for each one unit increase in ab_page,converted is 0.98 times as likely holding all else constant

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

p-value is 0.190 but the z-test has the same value of the p_value from part two is 0.9 as The greater the effect size, the greater the difference in p_diffs.

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 **Part II**?

Put your answer here.

[H0] p_new - p_old =0

[H1] p_new - p_old !=0

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.

for sure it is a good idea to add more additional terms into the regresion model like the age of the users , using time and time spending and more, to get more truthful result which we can use to determine if we will launch the new page or not.also may be multicollinearity as well as Simpson-paradox effect.

about the disadvantages i think there would be if we added additional terms as the results may be confused to understand or get the wrong description.

g. Now along with testing if the conversion rate changes for different pages, also add an effect based on which country a user lives in. 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.

Answer:

from those results The conversion ratios for people who are in countries [US 11.9%, UK 12.05% ,CA 11.53%],so there is an impact of counrty as the people in UK are more converted than in US but we can say that if they are equal numbers of from those countries in the test .

In [122…

```
data=pd.read_csv('countries.csv')
df_2=df2.join(data.set_index('user_id'),on='user_id')
df_2.head(1)
```

Out[122…

	user_id	timestamp	group	landing_page	converted	intercept	ab_page	country
0	851104	2017-01-21 22:11:48.556739	control	0	0	1	0	US

In [128…

```
print("US"+" "+str(df_2.query('country == "US"').converted.sum()/df_2.query('country == "U
print("UK"+" "+str(df_2.query('country == "UK"').converted.sum()/df_2.query('country == "U
print("CA"+" "+str(df_2.query('country == "CA"').converted.sum()/df_2.query('country == "C
```

```
US 0.119546800642
UK 0.12059448569
CA 0.115318297814
```

In [129…

```
df_3=df_2.copy()
_dummies = pd.get_dummies(df_3['country'])
df_3 = df_3.join(_dummies)
df_3.head(1)
```

Out[129…

	user_id	timestamp	group	landing_page	converted	intercept	ab_page	country	CA	UK	US
0	851104	2017-01-21 22:11:48.556739	control	0	0	1	0	US	0	0	1

In [130…

```
lm3 = sm.Logit(df_3['converted'], df_3[['intercept', 'ab_page', 'UK' , 'US']])
results = lm3.fit()
```

```
results.summary()
```

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

Out[130…

Logit Regression Results

Dep. Variable:	converted	**No. Observations:**	290584
Model:	Logit	**Df Residuals:**	290580
Method:	MLE	**Df Model:**	3
Date:	Sat, 20 Feb 2021	**Pseudo R-squ.:**	2.323e-05
Time:	14:31:00	**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	-2.0300	0.027	-76.249	0.000	-2.082	-1.978
ab_page	-0.0149	0.011	-1.307	0.191	-0.037	0.007
UK	0.0506	0.028	1.784	0.074	-0.005	0.106
US	0.0408	0.027	1.516	0.130	-0.012	0.093

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.

In [131…

```
df_3['ab_UK'] = df_3['ab_page'] * df_3['UK']
df_3['ab_US'] = df_3['ab_page'] * df_3['US']
lm3 = sm.Logit(df_3['converted'], df_3[['intercept', 'ab_page', 'UK' , 'US', 'ab_UK', 'ab_
results = lm3.fit()
results.summary()
```

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

Out[131…

Logit Regression Results

Dep. Variable:	converted	**No. Observations:**	290584
Model:	Logit	**Df Residuals:**	290578
Method:	MLE	**Df Model:**	5
Date:	Sat, 20 Feb 2021	**Pseudo R-squ.:**	3.482e-05
Time:	14:31:39	**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	-2.0040	0.036	-55.008	0.000	-2.075	-1.933
ab_page	-0.0674	0.052	-1.297	0.195	-0.169	0.034

UK	0.0118	0.040	0.296	0.767	-0.066	0.090
US	0.0175	0.038	0.465	0.642	-0.056	0.091
ab_UK	0.0783	0.057	1.378	0.168	-0.033	0.190
ab_US	0.0469	0.054	0.872	0.383	-0.059	0.152

summary:

from the results i can see all the p_values for them is greater than error rate 0.05 which lead us there is an effect of the country on conversion rate .

the ordering in effecting in conversion rate UK>US, as p_value for them UK greater than US.

In [173…
```
# this is multiplicable change in the odd
np.exp(-0.0674),np.exp(-0.0118),np.exp(-0.0175),np.exp(-0.0783),np.exp(-0.0468)
```

Out[173…
```
(0.93482119806188357,
 0.98826934696725466,
 0.9826522356650732,
 0.92468697883481343,
 0.95427823415290935)
```

In [174…
```
df2.timestamp.min(),df2.timestamp.max()
```

Out[174…
```
('2017-01-02 13:42:05.378582', '2017-01-24 13:41:54.460509')
```

conclusion:

in my opinion old page and new page have nearlly the same conversion ratio which to lead to accept the null hypothesis, for that the decision about which one is better should talk more time in , the testing time should be expanded .

from the cell above this test only take 22 day which isnot enough time to can get decision based on .