```
#!/usr/bin/env python
# coding: utf-8

# ## 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](#intro)
# - [Part I - Probability](#probability)
# - [Part II - A/B Test](#ab_test)
# - [Part III - Regression](#regression)
#
#
# <a id='intro'></a>
# ### 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](https://review.udacity.com/#!/projects/37e27304-ad47-4eb0-a1ab-
8c12f60e43d0/rubric).
#
# <a id='probability'></a>
# #### Part I - Probability
#
# To get started, let's import our libraries.

# In[71]:


import pandas as pd
import numpy as np
import random
import matplotlib.pyplot as plt
get_ipython().run_line_magic('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:

# In[72]:


#read the dataset from a csv file, and see the first 5 rows of it
df = pd.read_csv('ab_data.csv')
df.head()


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

# In[73]:


df.shape[0]


# c. The number of unique users in the dataset.

# In[74]:


df.nunique()[0]


# d. The proportion of users converted.

# In[75]:


df.converted.mean()
```

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

# In[76]:


#the left hand side of the + operator is when the value of "treatment" in column
"group" in the dataset
#matches with any value in the "landing_page" column except the "new_page" value

##################################################################################
################################

#the right hand side of the + operator is when the value of "new_page" in column
"landing_page" in the dataset
#matches with any value in the "group" column except the "treatment" value

(df.query('group == "treatment"')).query('landing_page != "new_page"').shape[0] +
(df.query('group != "treatment"')).query('landing_page == "new_page"').shape[0]


# f. Do any of the rows have missing values?

# In[77]:


#No rows in the dataset have missing values
df.isnull().sum().sum()


# `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**.

# In[78]:


# Remove the inaccurate rows, and store the result in a new dataframe df2
# creating a new dataframe "df2" where "treatment" is aligned with "new_page"
# creating a new dataframe "df3" where "control" is aligned with "old_page"
# after that append "df3" to "df2"
# "df2" is now the desired dataframe
```

```
df2 = df[df['group'] == 'treatment']
df2 = df2[df2['landing_page'] == 'new_page']
df3 = df[df['group'] == 'control']
df3 = df3[df3['landing_page'] == 'old_page']
df2 = df2.append(df3, ignore_index=True)
df2.shape[0]


# In[79]:


# 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]


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

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

# In[80]:


# there is one duplicate row in the new dataframe "df2"
df2.nunique()[0]


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

# In[81]:


#row number 1404 0 based-indexing is the duplicated row in "df2" dataframe
df_=df2.duplicated(subset=['user_id'])
df_=df_[df_ == True]
df_.index[0]


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

# In[82]:


#printing all information regarding the row number 1404 0 based-indexing in the
"df2" dataframe
```

```
df2.iloc[df_.index[0]]


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

# In[83]:


#removing the only duplicated row in the "df2" dataframe
df2.drop(axis = 0,index = df_.index[0],inplace = True)


# `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?

# In[84]:


# probability of "converted" column, or people who converted
p_population = df2['converted']
p_population.sum()/ p_population.shape[0]


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

# In[119]:


p_converted_given_control = df2.query("converted == 1")
p_converted_given_control =
p_converted_given_control[p_converted_given_control['group'] == 'control']
p_control = df2[df2['group'] == 'control']
p_control = p_control.shape[0]
p_converted_given_control = p_converted_given_control.shape[0] / p_control
print("P(converted|control): ",p_converted_given_control)


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

# In[123]:
```

```
p_converted_given_treatment = df2.query("converted == 1")
p_converted_given_treatment =
p_converted_given_treatment[p_converted_given_treatment['group'] == 'treatment']
p_treatment = df2[df2['group'] == 'treatment']
p_treatment = p_treatment.shape[0]
p_converted_given_treatment = p_converted_given_treatment.shape[0] / p_treatment
print("P(converted|treatment): ",p_converted_given_treatment)


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

# In[87]:


#While the "treatment" group is consistent with the "new_page" landing page, so
the probability of both int "df2" dataframe are the same
#and same thing or result with "control" group and "old_page" landing page
print("percentage of 'treatment' group in dataframe 'df2' is ",df2[df2['group']
== 'treatment'].shape[0] / df2.shape[0])
print("percentage of 'control' group in dataframe 'df2' is ",df2[df2['group'] ==
'control'].shape[0] / df2.shape[0])


# 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.

# I think that there is **no sufficient evidence** to say that the new treatment
page leads to more conversions , as the population after cleaning it by
consisting users with in **treatment** group with receiving **new_page** and
users in **control** group with receiving **old_page** and drop the duplicates is
only 290584 users, they split into two half, one belongs to the **treatment**
group, and the other belongs to the **control** group, so the probablility for
each is **almost the same**, is it seems that the probablity of **treatment**
group is a very very little bit higher than the probability of the **control**
group as shown in the result of the **(d)** cell above.
#
# **P(treatment) = 0.5000619442226688**,
#
# **P(control) = 0.4999380557773312**
#
# and as the number of users is the same for both groups **(treatment, control)**
as well as ofcourse both landing pages **(new page, old page)** respectively and
```

```
probability of the **converted** users from the **treatment** group is
**P(converted|treatment):  0.11880806551510564**,
#
# probability of the **converted** users from the **control** group is
**P(converted|control):  0.1203863045004612**
#
# so it seems that more users in **control** group as well as receiving
**old_page** are converted more than users in **treatment** group who are
converted.
#
# so there is an evidence to say that the **new_page** **did not lead** to more
conversions.

# <a id='ab_test'></a>
# ### 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.

# set **null hypothesis** *Ho* to be: the **converted** probability for the old
page. **THE GREATEST**
#
#
# set **alter. hypothesis** *H1* to be: the **converted** probability for the new
page. **THE SMALLEST**
#
#
```

```
# `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. <br><br>
#
# Use a sample size for each page equal to the ones in **ab_data.csv**.  <br><br>
#
# Perform the sampling distribution for the difference in **converted** between
the two pages over 10,000 iterations of calculating an estimate from the
null.  <br><br>
#
# 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.<br><br>

# a. What is the **convert rate** for $p_{new}$ under the null?

# In[88]:


#using "df2" dataframe to calculate the probability of conversion
# given new page
#here we are looking at a null where there is no difference in conversion based
on the page,
#which means the conversions for each page are the same
Pnew = df2['converted'].mean()
Pnew


# b. What is the **convert rate** for $p_{old}$ under the null? <br><br>

# In[89]:


#using "df2" dataframe to calculate the probability of conversion
# given old page
#here we are looking at a null where there is no difference in conversion based
on the page,
#which means the conversions for each page are the same
Pold = df2['converted'].mean()
Pold


# c. What is $n_{new}$?
# In[90]:


#calculate the number of users landed on the new page
Nnew = df2[df2['landing_page'] == 'new_page'].shape[0]
Nnew


# d. What is $n_{old}$?

# In[91]:


#calculate the number of users landed on the old page
Nold = df2[df2['landing_page'] == 'old_page'].shape[0]
Nold


# 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**.

# In[92]:


new_page_converted = np.random.choice(a=[1,0],size=Nnew,p=[Pnew,1-Pnew])
new_page_converted.mean()


# 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**.

# In[93]:


old_page_converted = np.random.choice(a=[1,0],size=Nold,p=[Pold,1-Pold])
old_page_converted.mean()


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

# In[94]:


new_page_converted.mean() - old_page_converted.mean()
```

```
# 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**.

# In[95]:


p_diffs = []
for i in range(10000):
    p_diffs.append(np.random.choice(a=[1,0],size=Nnew,p=[Pnew,1-Pnew]).mean() -
np.random.choice(a=[1,0],size=Nold,p=[Pold,1-Pold]).mean())


# i. Plot a histogram of the **p_diffs**.  Does this plot look like what you
expected?  Use the matching problem in the classroom to assure you fully
understand what was computed here.

# In[96]:


#plotting the histogram of the 10000 sample mean values of new and old page
converted under the null
plt.hist(p_diffs,alpha=0.5,color='red');
plt.xlabel('sample mean differences of new and old page converted');
plt.ylabel('No. of occurrences');
plt.title('sampling distribution of the mean differences of new and old page
converted');


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

# In[126]:


p_diffs = np.array(p_diffs)
# Calculate the actual difference (obs_diff) between the conversion rates for the
two groups.
obs_diff = p_converted_given_treatment - p_converted_given_control
#calculate the proportion of p_diffs greater than the obs_diff
(p_diffs > obs_diff).mean()


# 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?
```

```python
# 90% is the proportion that **p_diffs** that are greater than the actual
difference observed in ab_data.csv, this proportion value in scientific studies
is called **p-value**, it is infers that the **new_page** is not needed as it
hasn't higher conversion probability than the old one.

# 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.

# In[129]:


import statsmodels.api as sm

convert_old = df2[df2['landing_page'] == 'old_page']
convert_old = convert_old[convert_old['converted'] == 1].shape[0]
convert_new = df2[df2['landing_page'] == 'new_page']
convert_new = convert_new[convert_new['converted'] == 1].shape[0]
n_old = df2[df2['landing_page'] == 'old_page'].shape[0]
n_new = df2[df2['landing_page'] == 'new_page'].shape[0]


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

# In[132]:


z_score, p_value = sm.stats.proportions_ztest([convert_old, convert_new], [n_old,
n_new],value=None, alternative='smaller', prop_var=False)

z_score, p_value


# 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.**?

# these values means that taking **null** in consideration.
#
```

```
# the **null** is the converted probability of the **old_page** is greater than
the converted probability of the **new_page**.
#
# the **p-value** is 90.5% which is higher than 5% significance level which means
that we cannot be confident with 95% confidence level that the convertion rate of
the **new_page** is higher than that of **old_page**.

# <a id='regression'></a>
# ### Part III - A regression approach
#
# `1.` In this final part, you will see that the result you acheived in the
previous A/B test can also be acheived by performing regression.<br><br>
#
# a. Since each row is either a conversion or no conversion, what type of
regression should you be performing in this case?

# The target varible (**ground truth**) is (**converted *vs* not-converted**)
which means it is a **binary-variable**, so it is a **classification model**, and
the regression type should be used in this case is the **Logistic regression**.

# b. The goal is to use **statsmodels** to fit the regression model you specified
in part **a.** to see if there is a significant difference in conversion based on
which page a customer receives.  However, you first need to create a column for
the intercept, and create a dummy variable column for which page each user
received.  Add an **intercept** column, as well as an **ab_page** column, which
is 1 when an individual receives the **treatment** and 0 if **control**.

# In[135]:


#get_dummies method create columns of the values on the specified column it takes
#as an argument returns the columns alphabetically ordered
df2[['control','ab_page']] = pd.get_dummies(df2['group'])


# In[139]:


#drop redundant column
df2.drop('control',axis=1,inplace=True)


# In[141]:


#adding needed column
```

```
df2['intercept'] = 1


# 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.

# In[144]:


#fitting the logistic regression model
model = sm.Logit(df2['converted'],df2[['intercept','ab_page']])
result = model.fit()


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

# In[145]:


result.summary()


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

# the **p-value** associated with **ab_page** is 0.190, it is different from that
of **part II** as it was **0.905** in **part II**.
#
# perhaps becuase the **Logistic regression model** assumes an **intercept**.

# 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?

# it is a very good idea to consider other factors into the model, as to make the
model be more accurate, reliable and generate accurate result.
#
# the disadvantage is perhaps the model underfit the data if the data was not
enough, and it will get more complex.
```

```
# g. Now along with testing if the conversion rate changes for different pages,
also add an effect based on which country a user lives. You will need to read in
the **countries.csv** dataset and merge together your datasets on the
approporiate rows.  [Here](https://pandas.pydata.org/pandas-
docs/stable/generated/pandas.DataFrame.join.html) 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.

# In[148]:


#joining the two data sets together like what is happening in SQL, using INNER
JOIN method
countries_df = pd.read_csv('countries.csv')
df_new = countries_df.set_index('user_id').join(df2.set_index('user_id'),
how='inner')


# In[154]:


df_new.columns


# In[150]:


### Create the necessary dummy variables

#get_dummies method create columns of the values on the specified column it takes
#as an argument returns the columns alphabetically ordered
df_new[['CA','UK','US']] = pd.get_dummies(df_new['country'])


# 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[153]:
```

```
### Fit Your Linear Model And Obtain the Results

#proportion conversion by landing_page and country columns to check
#the interaction between landing_page and country to see if there significant
effects on conversion
df_new.groupby(['ab_page','country'], as_index=False).mean()


# Looking at the data above, it may the influence of the **landing_page** work on
just only **CA** among the all three countries, or just only **US** and the same
as the last country in the **df_new** dataframe.

# In[155]:


df_new ['influ_ab_page_US'] = df_new['US'] * df_new['ab_page']
df_new ['influ_ab_page_CA'] = df_new['CA'] * df_new['ab_page']
df_new.head(10)


# In[156]:


model =
sm.Logit(df_new['converted'],df_new[['intercept','ab_page','US','influ_ab_page_US
','CA','influ_ab_page_CA']])
results = model.fit()
results.summary()


# # SUMMARY AND CONCLUSION ON MODEL
#
#
# *   The **p-value** is greater than the value **0.05** for both interactions.
# *   The influence of the **Landing_page** on **each** country is not different
compared to the influence of the **Landing_page** on all other countries.

# # CONCLUSION
# *   there is no sufficient enough evidence that the **new_page** increases the
convertion probability more than the **old_page** based on the results of the
three above tests **(probability, A/B test and Logistic Regression)** Not even
the **Countries** have a strong influence on the conversion probability.
```