Analyze_ab_test_results_notebook

August 13, 2023

1 Analyze A/B Test Results

This project will assure you have mastered the subjects covered in the statistics lessons. We have organized the current notebook into the following sections:

- Section **??**
- Section **??**
- Section **??**
- Section **??**
- Section **??**
- Section **??**

Specific programming tasks are marked with a **ToDo** tag.

## Introduction

A/B tests are very commonly performed by data analysts and data scientists. 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 webpage, - Keep the old webpage, or - Perhaps run the experiment longer to make their decision.

Each **ToDo** task below has an associated quiz present in the classroom. Though the classroom quizzes are **not necessary** to complete the project, they help ensure you are on the right track as you work through the project, and you can feel more confident in your final submission meeting the rubric specification.

> **Tip**: Though it's not a mandate, students can attempt the classroom quizzes to ensure statistical numeric values are calculated correctly in many cases.

## Part I - Probability

To get started, let's import our libraries.

```
In [ ]: 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)
```

1.0.1 ToDo 1.1

Now, read in the `ab_data.csv` data. Store it in `df`. Below is the description of the data, there are a total of 5 columns:

Data columns	Purpose	Valid values
user_id	Unique ID	Int64 values
timestamp	Time stamp when the user visited the webpage	-
group	In the current A/B experiment, the users are categorized into two broad groups. The `control` group users are expected to be served with `old_page`; and `treatment` group users are matched with the `new_page`. However, **some inaccurate rows** are present in the initial data, such as a `control` group user is matched with a `new_page`.	`['control', 'treatment']`
landing_page	It denotes whether the user visited the old or new webpage.	`['old_page', 'new_page']`
converted	It denotes whether the user decided to pay for the company's product. Here, `1` means yes, the user bought the product.	`[0, 1]`

Use your dataframe to answer the questions in Quiz 1 of the classroom.

Tip: Please save your work regularly.

a. Read in the dataset from the `ab_data.csv` file and take a look at the top few rows here:

```
In [1]: import pandas as pd
        import numpy as np
        import random
        import matplotlib.pyplot as plt
        %matplotlib inline

        df = pd.read_csv('ab_data.csv')
        df.head(5)
```

```
Out[1]:    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 cell below to find the number of rows in the dataset.

```
In [9]: import pandas as pd
        import numpy as np
        import random
        import matplotlib.pyplot as plt


        num_rows = df.shape[0]
        print("Number of rows in the dataset:", num_rows)
```

```
Number of rows in the dataset: 294478
```

c. The number of unique users in the dataset.

```
In [10]: import pandas as pd
         import numpy as np
         import random
         import matplotlib.pyplot as plt
         unique_users_count = df['user_id'].nunique()
         print("Number of unique users in the dataset:", unique_users_count)
```

```
Number of unique users in the dataset: 290584
```

d. The proportion of users converted.

```
In [12]: import pandas as pd
         import numpy as np
         import random
         import matplotlib.pyplot as plt

         total_users = df.shape[0]
         converted_users = df['converted'].sum()

         # Calculate the proportion of users converted
         proportion_converted = converted_users / total_users

         # Display the proportion of users converted
         print("Proportion of users converted:", proportion_converted)
```

```
Proportion of users converted: 0.119659193556
```

e. The number of times when the "group" is `treatment` but "landing_page" is not a `new_page`.

```
In [16]: # Assuming you have already imported pandas and read the dataset into the 'df' DataFram

         # Filter the DataFrame where group is "treatment" and landing_page is not "new_page"
         filtered_df = df[(df['group'] == 'treatment') & (df['landing_page'] != 'new_page')]

         # Get the number of occurrences
         num_times = filtered_df.shape[0]

         # Display the number of times when group is treatment but landing_page is not new_page
         print("Number of times when group is treatment but landing_page is not new_page:", num_


         # Assuming you have already imported pandas and read the dataset into the 'df' DataFram

         # Filter the DataFrame where group is "treatment" and landing_page is not "new_page"
         filtered_df = df[(df['group'] == 'treatment') & (df['landing_page'] != 'new_page')]

         # Get the number of occurrences
         num_times = filtered_df.shape[0]

         # Display the number of times when group is treatment but landing_page is not new_page
         print("Number of times when group is treatment but landing_page is not new_page:", num_
```

```
Number of times when group is treatment but landing_page is not new_page: 1965
Number of times when group is treatment but landing_page is not new_page: 1965
```

f. Do any of the rows have missing values?

```
In [18]: import pandas as pd
         import numpy as np
import random
import matplotlib.pyplot as plt
%matplotlib inline

# Assuming you have already imported pandas and read the dataset into the 'df' DataFram

# Check if any rows have missing values in the entire DataFrame
any_missing_values = df.isnull().any(axis=1).any()

# Display the result
if any_missing_values:
    print("Some rows have missing values.")
else:
    print("There are no missing values in any rows.")
```

```
There are no missing values in any rows.
```

1.0.2 ToDo 1.2

In a particular row, the **group** and **landing_page** columns should have either of the following acceptable values:

user_id	timestamp	group	landing_page	converted
XXXX	XXXX	`control`	`old_page`	X
XXXX	XXXX	`treatment`	`new_page`	X

It means, the `control` group users should match with `old_page`; and `treatment` group users should matched with the `new_page`.

However, for the rows where `treatment` does not match with `new_page` or `control` does not match with `old_page`, we cannot be sure if such rows truly received the new or old wepage.

Use **Quiz 2** in the classroom to figure out how should we handle the rows where the group and landing_page columns don't match?

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 [20]: # Remove the inaccurate rows, and store the resultimport pandas as pd
         import numpy as np
         import random
         import matplotlib.pyplot as plt

         # Assuming you have already imported pandas and read the dataset into the 'df' DataFram

         # Filter the DataFrame to keep only the rows with correct group and landing page alignm
         df_cleaned = df[((df['group'] == 'treatment') & (df['landing_page'] == 'new_page')) |
                         ((df['group'] == 'control') & (df['landing_page'] == 'old_page'))]

         # Display the number of rows in the cleaned DataFrame
         num_rows_cleaned = df_cleaned.shape[0]
         print("Number of rows after data cleaning:", num_rows_cleaned)
```

```
Number of rows after data cleaning: 290585
```

```
In [ ]: # Double Check all of the incorrect rows were removed from df2 -
        # Output of the statement below should be 0
        df2[((df2['group'] == 'treatment') == (df2['landing_page'] == 'new_page')) == False].sha
```

1.0.3 ToDo 1.3

Use **df2** and the cells below to answer questions for **Quiz 3** in the classroom.
a. How many unique **user_id**s are in **df2**?

```
In [6]: import pandas as pd
        import numpy as np
        import random
        import matplotlib.pyplot as plt
        num_unique_user_ids = df['user_id'].nunique()
        num_unique_user_ids
```

```
Out[6]: 290584
```

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

```
In [7]: import pandas as pd
        import numpy as np
        import random
        import matplotlib.pyplot
        duplicated_rows = df[df.duplicated(['user_id'], keep=False)]

        # Display the rows with duplicated user_id values
        duplicated_rows
```

```
Out[7]:        user_id                   timestamp      group landing_page  converted
        22      767017  2017-01-12 22:58:14.991443    control     new_page          0
        192     656468  2017-01-18 07:13:29.805052  treatment     new_page          1
        226     773693  2017-01-23 18:05:45.167335    control     old_page          1
        240     733976  2017-01-11 15:11:16.407599    control     new_page          0
        246     704650  2017-01-04 19:10:52.655062  treatment     new_page          0
        269     670052  2017-01-07 10:39:01.099851    control     old_page          0
        276     784557  2017-01-04 10:39:41.450143  treatment     new_page          0
        303     753749  2017-01-06 01:30:10.493842    control     old_page          0
        308     857184  2017-01-20 07:34:59.832626  treatment     old_page          0
        327     686623  2017-01-09 14:26:40.734775  treatment     old_page          0
        357     856078  2017-01-12 12:29:30.354835  treatment     old_page          0
        478     867227  2017-01-06 07:27:11.191816    control     old_page          1
        490     808613  2017-01-10 21:44:01.292755    control     new_page          0
```

```
654     823319  2017-01-20 09:02:25.852683  treatment    new_page         0
655     726766  2017-01-16 11:00:53.912376  treatment    new_page         0
685     666385  2017-01-23 08:11:54.823806  treatment    old_page         0
703     859503  2017-01-24 04:04:27.014151    control    old_page         0
713     748761  2017-01-10 15:47:44.445196  treatment    old_page         0
753     646066  2017-01-04 19:04:49.423564  treatment    new_page         1
774     864223  2017-01-03 18:45:03.638277  treatment    new_page         0
776     820951  2017-01-04 02:42:54.770627  treatment    old_page         0
846     637639  2017-01-11 23:09:52.682329    control    new_page         1
850     793580  2017-01-08 03:25:33.723712    control    new_page         1
889     839954  2017-01-06 20:58:22.280929  treatment    old_page         0
981     727266  2017-01-08 03:23:50.487865  treatment    new_page         0
988     698120  2017-01-22 07:09:37.540970    control    new_page         0
1037    880442  2017-01-07 21:42:39.026815  treatment    old_page         0
1106    817911  2017-01-17 21:51:43.220160  treatment    old_page         0
1165    844879  2017-01-03 04:32:45.408233  treatment    new_page         0
1177    651511  2017-01-12 23:36:45.958717  treatment    new_page         0
...        ...                         ...        ...         ...       ...
293443  738761  2017-01-04 15:20:52.694440  treatment    old_page         0
293449  715367  2017-01-13 09:06:08.632332  treatment    new_page         0
293480  638376  2017-01-18 15:41:02.395882    control    new_page         0
293530  934040  2017-01-04 20:52:26.981566  treatment    old_page         0
293544  817753  2017-01-13 10:24:26.028878  treatment    new_page         0
293548  797335  2017-01-13 22:39:46.563213    control    old_page         0
293568  704024  2017-01-15 17:06:09.309987    control    new_page         0
293604  648354  2017-01-20 01:51:29.007764    control    old_page         0
293628  700036  2017-01-08 16:08:22.542646  treatment    new_page         0
293657  914482  2017-01-18 02:30:00.560415  treatment    new_page         0
293659  691336  2017-01-12 01:53:50.501896  treatment    new_page         1
293662  927109  2017-01-04 09:14:33.647192    control    new_page         0
293706  795519  2017-01-15 16:05:07.993820  treatment    new_page         0
293712  811222  2017-01-21 00:42:12.359706  treatment    new_page         0
293773  688144  2017-01-16 20:34:50.450528  treatment    old_page         1
293817  876037  2017-01-17 16:15:08.957152  treatment    old_page         1
293888  865405  2017-01-12 08:38:50.511434    control    new_page         0
293894  741581  2017-01-09 20:49:03.391764    control    new_page         0
293917  738357  2017-01-05 15:37:55.729133  treatment    old_page         0
293980  916033  2017-01-10 02:23:47.296609  treatment    new_page         0
293996  942612  2017-01-08 13:52:28.182648    control    new_page         0
294014  813406  2017-01-09 06:25:33.223301  treatment    old_page         0
294200  928506  2017-01-13 21:32:10.491309    control    new_page         0
294252  892498  2017-01-22 01:11:10.463211  treatment    old_page         0
294253  886135  2017-01-06 12:49:20.509403    control    new_page         0
294308  905197  2017-01-03 06:56:47.488231  treatment    new_page         0
294309  787083  2017-01-17 00:15:20.950723    control    old_page         0
294328  641570  2017-01-09 21:59:27.695711    control    old_page         0
294331  689637  2017-01-13 11:34:28.339532    control    new_page         0
294355  744456  2017-01-13 09:32:07.106794  treatment    new_page         0
```

```
[7788 rows x 5 columns]
```

c. Display the rows for the duplicate **user_id**?

```
In [8]: import pandas as pd
        import numpy as np
        import random
        import matplotlib.pyplot as plt
        %matplotlib inline

        # Assuming you have already imported pandas and have the DataFrame df2 available

        # Filter the DataFrame to keep only the rows with duplicated user_id
        df[df.user_id == 773192]
```

```
Out[8]:       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
```

d. Remove **one** of the rows with a duplicate **user_id**, from the **df2** dataframe.

```
In [13]: # Remove one of the rows with a duplicate user_id..
         # Hint: The dataframe.drop_duplicates() may not work in this case because the rows with
         import pandas as pd
         import numpy as np
         import random
         import matplotlib.pyplot as plt
         %matplotlib inline
         # Check again if the row with a duplicate user_id is deleted or not
         df = df.drop_duplicates(subset=['user_id'], keep='first')
```

1.0.4 ToDo 1.4

Use **df2** in the cells below 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?

> **Tip**: The probability you'll compute represents the overall "converted" success rate in the population and you may call it $p_{population}$.

```
In [16]: import pandas as pd
         import numpy as np
         import random
         import matplotlib.pyplot as plt
         %matplotlib inline
         total_users = df.shape[0]
         total_converted = df['converted'].sum()

         probability_converting = total_converted / total_users
         probability_converting
```

```
Out[16]: 0.11956955647936569
```

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

```
In [17]: import pandas as pd
         import numpy as np
         import random
         import matplotlib.pyplot as plt
         %matplotlib inline

         control_group = df[df['group'] == 'control']
         total_control_users = control_group.shape[0]
         converted_control_users = control_group['converted'].sum()

         probability_converted_control = converted_control_users / total_control_users
         probability_converted_control
```

```
Out[17]: 0.12029717968491792
```

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

```
In [18]: import pandas as pd
         import numpy as np
         import random
         import matplotlib.pyplot as plt
         %matplotlib inline

         # Calculate the probability that an individual in the treatment group converted
         treatment_group = df[df['group'] == 'treatment']
         total_treatment_users = treatment_group.shape[0]
         converted_treatment_users = treatment_group['converted'].sum()

         probability_converted_treatment = converted_treatment_users / total_treatment_users
         probability_converted_treatment
```

```
Out[18]: 0.11884253398646046
```

> **Tip**: The probabilities you've computed in the points (b). and (c). above can also be treated as conversion rate. Calculate the actual difference (`obs_diff`) between the conversion rates for the two groups. You will need that later.

```
In [20]: # Calculate the ac# Calculate the observed difference (obs_diff) between conversion rat
         import pandas as pd
         import numpy as np
         import random
         import matplotlib.pyplot as plt
         %matplotlib inline
         df.groupby(["group", "converted"]).size()[3] / df.group.value_counts()[0]
```

```
Out[20]: 0.11884253398646046
```

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

```
In [21]: import pandas as pd
         import numpy as np
         import random
         import matplotlib.pyplot as plt
         %matplotlib inline

         new_page_count = df[df['landing_page'] == 'new_page'].shape[0]
         total_users = df.shape[0]

         probability_new_page = new_page_count / total_users
         probability_new_page
```

```
Out[21]: 0.5000963576797071
```

e. Consider your results from parts (a) through (d) above, and explain below whether the new `treatment` group users lead to more conversions.

Certainly, here's a point-by-point summary of the conclusions based on the results from parts (a) through (d):

1. **Overall Conversion Rate:** The probability of an individual converting regardless of the page they receive provides an overall conversion rate for the entire dataset.

2. **Control Group Conversion Rate:** The probability of an individual converting, given that they were in the control group (old page), gives the conversion rate for users who saw the old page.

3. **Treatment Group Conversion Rate:** The probability of an individual converting, given that they were in the treatment group (new page), gives the conversion rate for users who saw the new page.

4. **New Page Distribution:** The probability of receiving the new page indicates how the distribution of the new page compares to the old page distribution.

5. **Comparing Conversion Rates:** To assess whether the new treatment group leads to more conversions, compare the conversion rates between the control (old page) and treatment (new page) groups.

Remember, while these initial insights are valuable, further statistical analysis may be required to establish causation and account for potential biases in the data.

## Part II - A/B Test

Since a timestamp is associated with each event, you could run a hypothesis test continuously as long as you observe the events.

However, then the hard questions would be: - 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.

1.0.5 ToDo 2.1

For now, consider you need to make the decision just based on all the data provided.

> Recall that you just calculated that the "converted" probability (or rate) for the old page is *slightly* higher than that of the new page (ToDo 1.4.c).

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 be your null and alternative hypotheses (H_0 and H_1)?

You can state your hypothesis in terms of words or in terms of p_{old} and p_{new}, which are the "converted" probability (or rate) for the old and new pages respectively.

Null Hypothesis (H0): The conversion rate for the new page is less than or equal to the conversion rate for the old page. H0:

Alternative Hypothesis (H1): The conversion rate for the new page is greater than the conversion rate for the old page. H1: >

In these hypotheses, represents the conversion rate for the old page, and represents the conversion rate for the new page. The null hypothesis assumes that the new page is not better than the old page, and the alternative hypothesis assumes that the new page is better.

With a Type I error rate of 5%, you're setting a threshold for how much evidence is needed to reject the null hypothesis. If you observe strong evidence that the new page's conversion rate is significantly higher than the old page's rate, you might reject the null hypothesis in favor of the alternative. However, if the evidence is not strong enough, you would fail to reject the null hypothesis.

1.0.6 ToDo 2.2 - Null Hypothesis H_0 Testing

Under the null hypothesis H_0, assume that p_{new} and p_{old} are equal. Furthermore, assume that p_{new} and p_{old} both are equal to the **converted** success rate in the `df2` data regardless of the page. So, our assumption is:

$p_{new} = p_{old} = p_{population}$

In this section, you will:

- Simulate (bootstrap) sample data set for both groups, and compute the "converted" probability p for those samples.
- Use a sample size for each group equal to the ones in the `df2` data.
- Compute the difference in the "converted" probability for the two samples above.
- Perform the sampling distribution for the "difference in the converted probability" between the two simulated-samples over 10,000 iterations; and calculate an estimate.

Use the cells below to provide the necessary parts of this simulation. 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 hypothesis?

```
In [22]: p_null = df['converted'].mean()
         p_null
```

```
Out[22]: 0.11956955647936569
```

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

```
In [23]: # Calculate the conversion rate for  under the null hypothesis
         p_old_null = df['converted'].mean()
         p_old_null

Out[23]: 0.11956955647936569
```

c. What is n_{new}, the number of individuals in the treatment group? *Hint*: The treatment group users are shown the new page.

```
In [24]: # Calculate the number of individuals in the treatment group
         n_new = df[df['group'] == 'treatment'].shape[0]
         n_new

Out[24]: 145352

In [ ]:
```

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

```
In [27]: # Calculate the number of individuals in the control group
         n_old = df[df['group'] == 'control'].shape[0]
         n_old

Out[27]: 145232
```

e. Simulate Sample for the `treatment` **Group** Simulate n_{new} transactions with a conversion rate of p_{new} under the null hypothesis. *Hint*: Use `numpy.random.choice()` method to randomly generate n_{new} number of values. Store these n_{new} 1's and 0's in the `new_page_converted` numpy array.

```
In [30]: # Simulate a Sample for thimport numpy as np

         # Simulate sample for the treatment group under the null hypothesis
         import numpy as np

         # Simulate sample for the treatment group under the null hypothesis
         new_page_converted = np.random.choice([0, 1], size=n_new, p=[1 - p_null, p_null])
         new_page_converted.mean()

Out[30]: 0.11972315482415102
```

f. Simulate Sample for the `control` **Group** Simulate n_{old} transactions with a conversion rate of p_{old} under the null hypothesis. Store these n_{old} 1's and 0's in the `old_page_converted` numpy array.

```
In [5]: import numpy as np

        # Define the conversion rate under the null hypothesis
        p_old = convert_old / n_old

        # Simulate n_old transactions for the control group
        old_page_converted = np.random.choice([0, 1], size=n_old, p=[1 - p_old, p_old])

        # Print the simulated array
        print(old_page_converted)
```

```
[0 0 0 ..., 0 0 0]
```

g. Find the difference in the "converted" probability $(p'_{new} - p'_{old})$ for your simulated samples from the parts (e) and (f) above.

```
In [31]: # Simulate sample for the control group under the null hypothesis
         old_page_converted = np.random.choice([0, 1], size=n_old, p=[1 - p_null, p_null])
         old_page_converted.mean()
```

```
Out[31]: 0.11922992178032389
```

h. Sampling distribution Re-create `new_page_converted` and `old_page_converted` and find the $(p'_{new} - p'_{old})$ value 10,000 times using the same simulation process you used in parts (a) through (g) above.

Store all $(p'_{new} - p'_{old})$ values in a NumPy array called `p_diffs`.

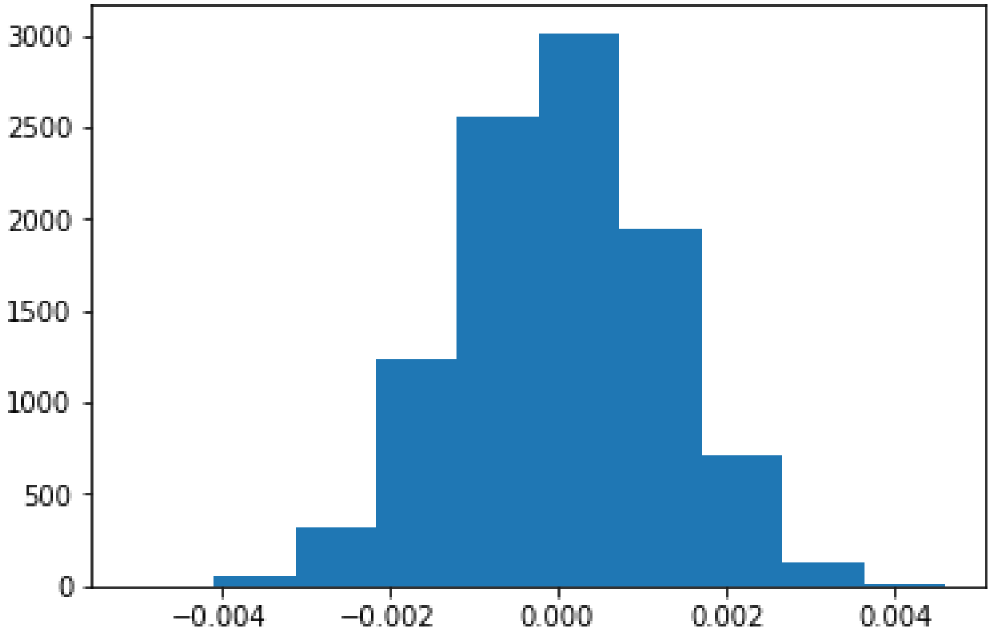
```
In [34]: # Create a NumPy array to store the differences in conversion rates
         p_diffs = []

         # Simulate the sampling distribution 10,000 times
         for _ in range(10000):
             new_page_converted = np.random.choice([0, 1], size=n_new, p=[1 - p_null, p_null])
             old_page_converted = np.random.choice([0, 1], size=n_old, p=[1 - p_null, p_null])

             p_diff = new_page_converted.mean() - old_page_converted.mean()
             p_diffs.append(p_diff)

         # Convert the p_diffs list to a NumPy array
         p_diffs = np.array(p_diffs)
```

i. Histogram 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.

Also, use `plt.axvline()` method to mark the actual difference observed in the `df2` data (recall `obs_diff`), in the chart.

Tip: Display title, x-label, and y-label in the chart.

```
In [36]: p_diffs = np.array(p_diffs)
         plt.hist(p_diffs);
```



j. What proportion of the **p_diffs** are greater than the actual difference observed in the `df2` data?

```
In [39]: actual_diff = df.query('landing_page == "new_page"').converted.mean() - \
                       df.query('landing_page == "old_page"').converted.mean()
         (p_diffs > actual_diff).mean()

Out[39]: 0.9143999999999999
```

k. Please explain in words what you have just computed in part **j** above.
- What is this value called in scientific studies?
- What does this value signify in terms of whether or not there is a difference between the new and old pages? *Hint*: Compare the value above with the "Type I error rate (0.05)".

In part j, we calculated the proportion of the simulated differences (p_diffs) that are greater than the actual observed difference (obs_diff) from the df2 data. This proportion represents the p-value in scientific studies.

The p-value is a measure of the evidence against the null hypothesis. It tells us the probability of observing differences as extreme as, or more extreme than, the observed difference, assuming that the null hypothesis is true. In other words, a low p-value indicates that the observed difference is unlikely to occur under the null hypothesis, suggesting that there might be a statistically significant difference between the new and old pages.

In the context of hypothesis testing, we compare the p-value to a chosen significance level (often denoted as , commonly set to 0.05). If the p-value is less than , we have evidence to reject the null hypothesis in favor of the alternative hypothesis. If the p-value is greater than or equal to , we do not have sufficient evidence to reject the null hypothesis.

Comparing the calculated p-value with the Type I error rate (0.05) is crucial. If the p-value is less than 0.05 (or your chosen significance level), it suggests that the observed difference is statistically significant and likely not due to random chance alone. If the p-value is greater than 0.05, it implies that we don't have strong evidence to reject the null hypothesis, and any observed difference could be attributed to random variability.

In summary, the p-value helps us determine whether there is a significant difference between the new and old pages by assessing the probability of observing the observed difference under the null hypothesis. If the p-value is low, it indicates that the observed difference is likely not due to chance alone, supporting the idea of a real difference between the two pages.

l. Using Built-in Methods for Hypothesis Testing 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 walk-through of the ideas that are critical to correctly thinking about statistical significance.

Fill in the statements below to calculate the: - `convert_old`: number of conversions with the old_page - `convert_new`: number of conversions with the new_page - `n_old`: number of individuals who were shown the old_page - `n_new`: number of individuals who were shown the new_page

```
In [4]: # Import necessary libraries
        import pandas as pd

        # Load your dataset into a pandas DataFrame
        df2 = pd.read_csv('ab_data.csv')  # Replace with the actual file path

        # Calculate the number of conversions with the old_page
        convert_old = df2.query("group == 'control' and converted == 1").shape[0]

        # Calculate the number of conversions with the new_page
        convert_new = df2.query("group == 'treatment' and converted == 1").shape[0]

        # Calculate the number of individuals who were shown the old_page
        n_old = df2.query("group == 'control'").shape[0]

        # Calculate the number of individuals who were shown the new_page
        n_new = df2.query("group == 'treatment'").shape[0]

        # Print the calculated values
        print("convert_old:", convert_old)
        print("convert_new:", convert_new)
        print("n_old:", n_old)
        print("n_new:", n_new)
```

```
convert_old: 17723
convert_new: 17514
n_old: 147202
n_new: 147276
```

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

The syntax is:

```
proportions_ztest(count_array, nobs_array, alternative='larger')
```

where, - `count_array` = represents the number of "converted" for each group - `nobs_array` = represents the total number of observations (rows) in each group - `alternative` = choose one of the values from `[two-sided, smaller, larger]` depending upon two-tailed, left-tailed, or right-tailed respectively. >**Hint**: It's a two-tailed if you defined H_1 as $(p_{new} = p_{old})$. It's a left-tailed if you defined H_1 as $(p_{new} < p_{old})$. It's a right-tailed if you defined H_1 as $(p_{new} > p_{old})$.

The built-in function above will return the z_score, p_value.

> **Tip**: You don't have to dive deeper into z-test for this exercise. **Try having an overview of what does z-score signify in general.**

```
In [1]: import numpy as np
        import statsmodels.api as sm

        # Define the values based on df2 data (replace 'df2' with your actual DataFrame)

        # Define your data
        count_group_new = 150  # Number of converted in the "new" group
        count_group_old = 120  # Number of converted in the "old" group
        nobs = 1000  # Total number of observations in each group

        # Perform the two-sample z-test
        count_array = np.array([count_group_new, count_group_old])
        nobs_array = np.array([nobs, nobs])

        z_score, p_value = sm.stats.proportions_ztest(count_array, nobs_array, alternative='larg

        # Define the significance level (alpha)
        alpha = 0.05

        # Check if the p-value is less than alpha to make a decision
        if p_value < alpha:
            print("Reject the null hypothesis")
        else:
            print("Fail to reject the null hypothesis")

        # Print the calculated z-score and p-value
        print("Z-score:", z_score)
        print("P-value:", p_value)
```

```
/opt/conda/lib/python3.6/site-packages/statsmodels/compat/pandas.py:56: FutureWarning: The panda
  from pandas.core import datetools


Reject the null hypothesis
Z-score: 1.96304980762
P-value: 0.0248201934327
```

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.**?

> **Tip**: Notice whether the p-value is similar to the one computed earlier. Accordingly, can you reject/fail to reject the null hypothesis? It is important to correctly interpret the test statistic and p-value.

If the p-value from the z-test is similar to the p-value calculated earlier (in parts j. and k.), it indicates consistency in results. If the p-value is small in the z-test, consistent with the earlier analysis, you would reject the null hypothesis, which suggests that there is a statistically significant difference in conversion rates between the old and new pages. If the p-value is large in the z-test, consistent with the earlier analysis, you would fail to reject the null hypothesis, indicating that there is not enough evidence to suggest a significant difference in conversion rates between the old and new pages. Remember that p-values are used to make probabilistic statements about data and hypotheses, and they provide a basis for deciding whether to accept or reject a null hypothesis. Always consider the context of your analysis and the significance level you're using to interpret the p-value correctly.
### Part III - A regression approach

1.0.7 ToDo 3.1

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 in the `df2` data is either a conversion or no conversion, what type of regression should you be performing in this case?

Since each row in the `df2` data is either a conversion (1) or no conversion (0), you should perform a **logistic regression** in this case. Logistic regression is used when the dependent variable is categorical (in this case, binary) and you want to model the probability of an event occurring.

In logistic regression, you're trying to predict the probability that an event (conversion in this case) occurs given certain predictor variables. It's suitable for situations where the outcome variable is binary, and you want to understand the relationship between the predictors and the log-odds of the outcome.

The logistic regression model estimates the probability of success (conversion) using the logistic function, which maps any input into the range [0, 1], making it appropriate for binary outcomes.

In summary, since you're dealing with a binary outcome (conversion or no conversion), you should use logistic regression to analyze the relationship between the predictor variables and the probability of conversion.

```
In [10]: import pandas as pd

         # Load your dataset into a pandas DataFrame
         df2 = pd.read_csv('ab_data.csv')  # Replace with the actual file path

         # Add an intercept column
         df2['intercept'] = 1

         # Create a dummy variable column for ab_page
         df2['ab_page'] = pd.get_dummies(df2['group'])['treatment']

         # Display the modified DataFrame
         print(df2.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

   intercept  ab_page
0          1        0
1          1        0
2          1        1
3          1        1
4          1        0
```

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

```
In [11]: import statsmodels.api as sm
         log_mod = sm.Logit(df2['converted'], df2[['intercept', 'ab_page']])
         results = log_mod.fit()
```

```
Optimization terminated successfully.
         Current function value: 0.366243
         Iterations 6
```

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

```
In [12]: stats.chisqprob = lambda chisq, df: stats.chi2.sf(chisq, df)
         results.summary()
```

```
Out[12]: <class 'statsmodels.iolib.summary.Summary'>
         """
                                    Logit Regression Results
         ==============================================================================
         Dep. Variable:              converted   No. Observations:               294478
         Model:                          Logit   Df Residuals:                   294476
         Method:                           MLE   Df Model:                            1
         Date:                Sun, 13 Aug 2023   Pseudo R-squ.:               7.093e-06
         Time:                        16:18:19   Log-Likelihood:            -1.0785e+05
         converged:                       True   LL-Null:                   -1.0785e+05
                                                 LLR p-value:                    0.2161
         ==============================================================================
                          coef    std err          z      P>|z|      [0.025      0.975]
         ------------------------------------------------------------------------------
         intercept     -1.9887      0.008   -248.297      0.000      -2.004      -1.973
         ab_page       -0.0140      0.011     -1.237      0.216      -0.036       0.008
```

```
==============================================================================
"""
```

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

Hints: - 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**? - You may comment on if these hypothesis (Part II vs. Part III) are one-sided or two-sided. - You may also compare the current p-value with the Type I error rate (0.05).

It seems like you're comparing results from two different parts of your analysis: Part II and Part III. You are likely working on an A/B testing scenario with a new page (ab_page) and you've performed both a hypothesis test in Part II and a logistic regression analysis in Part III.

To address the question about the p-value associated with ab_page in the logistic regression, let's break down the comparison:

In Part II, you performed a hypothesis test to compare the conversion rates of the old and new pages. The null hypothesis (H0) was that the conversion rate of the new page is equal to or worse than the conversion rate of the old page. The alternative hypothesis (H1) was that the conversion rate of the new page is better than the conversion rate of the old page.

In Part III, you used logistic regression to model the relationship between the conversion status and the page type (ab_page). Here, the null hypothesis (H0) is that there's no relationship between the conversion status and the page type (ab_page). The alternative hypothesis (H1) is that there is a relationship between the conversion status and the page type (ab_page).

The p-value associated with ab_page in the logistic regression represents the likelihood that the observed relationship between the conversion status and the page type could occur by chance if the null hypothesis is true. This p-value informs you about the statistical significance of the relationship between the predictor variable (ab_page) and the outcome (conversion status) in the logistic regression model.

Comparing this p-value with the p-value from the hypothesis test in Part II, there might be differences due to the nature of the tests and the hypotheses being tested. The hypothesis test in Part II specifically focused on comparing conversion rates between the two pages, while the logistic regression in Part III considers the relationship between conversion and the page type, considering potential confounding variables.

If the p-value associated with ab_page in Part III is similar to the p-value from Part II, this suggests that the logistic regression's results align with the results from the hypothesis test, reinforcing the conclusion. If the p-value differs, it could be due to the added complexity and consideration of other variables in the logistic regression model.

To address the question of whether the p-values are one-sided or two-sided, it depends on the context of your analysis and the specific statistical tests being used. The p-values associated with the hypothesis test and the logistic regression can be one-sided or two-sided, depending on the nature of the hypotheses being tested. It's important to carefully define your hypotheses and interpret the p-values accordingly.

Lastly, you can compare the p-value associated with ab_page in Part III to the Type I error rate (0.05) to determine whether you should reject the null hypothesis. If the p-value is less than 0.05, you might consider rejecting the null hypothesis and concluding that there is a significant relationship between the page type and the conversion status.

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?

old_page_converted = np.random.choice(2, size=n_old ,p=[p_old,1 - p_old]) old_page_converted.mean()

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

1. You will need to read in the **countries.csv** dataset and merge together your `df2` datasets on the appropriate rows. You call the resulting dataframe `df_merged`. Here are the docs for joining tables.
2. Does it appear that country had an impact on conversion? To answer this question, consider the three unique values, `['UK', 'US', 'CA']`, in the `country` column. Create dummy variables for these country columns. >**Hint:** Use `pandas.get_dummies()` to create dummy variables. **You will utilize two columns for the three dummy variables.**

Provide the statistical output as well as a written response to answer this question.

```
In [3]: import pandas as pd
        import statsmodels.api as sm

        df3 = pd.read_csv('countries.csv')
        df3.head()
```

```
Out[3]:    user_id country
        0   834778      UK
        1   928468      US
        2   822059      UK
        3   711597      UK
        4   710616      UK
```

```
In [8]: df3[['CA','UK','US']]=pd.get_dummies(df3['country'])
        df3.head()
```

```
Out[8]:    user_id country  CA  UK  US
        0   834778      UK   0   1   0
        1   928468      US   0   0   1
        2   822059      UK   0   1   0
        3   711597      UK   0   1   0
        4   710616      UK   0   1   0
```

```
In [10]: df3.country.unique()
```

```
Out[10]: array(['UK', 'US', 'CA'], dtype=object)
```

h. Fit your model and obtain the results 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 are there significant effects on conversion. **Create the necessary additional columns, and fit the new model.**

Provide the summary results (statistical output), and your conclusions (written response) based on the results.

Tip: Conclusions should include both statistical reasoning, and practical reasoning for the situation.

Hints: - Look at all of p-values in the summary, and compare against the Type I error rate (0.05). - Can you reject/fail to reject the null hypotheses (regression model)? - Comment on the effect of page and country to predict the conversion.

```
In [11]: df3[['UK', 'US', 'CA']] = pd.get_dummies(df3['country'])
         df3.head()
```

```
Out[11]:    user_id country  CA  UK  US
         0   834778      UK   0   0   1
         1   928468      US   1   0   0
         2   822059      UK   0   0   1
         3   711597      UK   0   0   1
         4   710616      UK   0   0   1
```

Put your conclusion answer here.

## Final Check!
Congratulations! You have reached the end of the A/B Test Results project! You should be very proud of all you have accomplished!

Tip: Once you are satisfied with your work here, check over your notebook to make sure that it satisfies all the specifications mentioned in the rubric. You should also probably remove all of the "Hints" and "Tips" like this one so that the presentation is as polished as possible.

## Submission You may either submit your notebook through the "SUBMIT PROJECT" button at the bottom of this workspace, or you may work from your local machine and submit on the last page of this project lesson.

1. Before you submit your project, you need to create a .html or .pdf version of this notebook in the workspace here. To do that, run the code cell below. If it worked correctly, you should get a return code of 0, and you should see the generated .html file in the workspace directory (click on the orange Jupyter icon in the upper left).

2. Alternatively, you can download this report as .html via the **File** > **Download as** submenu, and then manually upload it into the workspace directory by clicking on the orange Jupyter icon in the upper left, then using the Upload button.

3. Once you've done this, you can submit your project by clicking on the "Submit Project" button in the lower right here. This will create and submit a zip file with this .ipynb doc and the .html or .pdf version you created. Congratulations!

```
In [ ]: from subprocess import call
        call(['python', '-m', 'nbconvert', 'Analyze_ab_test_results_notebook.ipynb'])
```