Homework 7: Testing Hypotheses

Reading:

- Testing Hypotheses
- A/B Testing

For all problems that you must write our explanations and sentences for, you **must** provide your answer in the designated space. Moreover, throughout this homework and all future ones, please be sure to not re-assign variables throughout the notebook! For example, if you use `max_temperature` in your answer to one question, do not reassign it later on.

```
# Don't change this cell; just run it.

import numpy as np
from datascience import *

# These lines do some fancy plotting magic.
import matplotlib
%matplotlib inline
import matplotlib.pyplot as plt
plt.style.use('fivethirtyeight')
import warnings
warnings.simplefilter('ignore', FutureWarning)
```

1. Spam Calls

Part 1: 781 Fun

Yanay gets a lot of spam calls. An area code is defined to be a three digit number from 200-999 inclusive. In reality, many of these area codes are not in use, but for this question we'll simplify things and assume they all are. **Throughout these questions, you should assume that Yanay's area code is 781.**

Question 1. Assuming each area code is just as likely as any other, what's the probability that two back-to-back spam calls both have an area code of 781?

```
prob_781 = 1/799*1/799
prob_781
```

```
    1.5664135864448834e-06
```

Question 2. Rohan already knows that Yanay's area code is 781. Rohan randomly guesses the last 7 digits (0-9 inclusive) of his phone number. What's the probability that Rohan correctly guesses Yanay's number, assuming he's equally likely to choose any digit?

Note: A phone number contains an area code and 7 additional digits, i.e. xxx-xxx-xxxx

```
prob_yanay_num = pow(1/10,7)
prob_yanay_num
```

```
1.0000000000000004e-07
```

Yanay suspects that there's a higher chance that the spammers are using his area code (781) to trick him into thinking it's someone from his area calling him. Ashley thinks that this is not the case, and that spammers are just choosing area codes of the spam calls at random from all possible area codes (*Remember, for this question we're assuming the possible area codes are 200-999, inclusive*). Yanay wants to test his claim using the 50 spam calls he received in the past month.

Here's a dataset of the area codes of the 50 spam calls he received in the past month.

```
from google.colab import drive
drive.mount('/content/drive')
```

```
Mounted at /content/drive
```

```
# Just run this cell
spam = Table().read_table('/content/drive/MyDrive/spam.csv')
spam
```

Area Code
891
924
516
512
328
613
214
781
591
950

... (40 rows omitted)

Question 3. Define the null hypothesis and alternative hypothesis for this investigation.

Hint: Don't forget that your null hypothesis should fully describe a probability model that we can use for simulation later.

Null Hypothesis (H0): The probability of receiving a spam call with the area code 781 is not higher than the probability of receiving a spam call with any other area code from 200 to 999.

Alternative Hypothesis (H1): The probability of receiving a spam call with the area code 781 is higher than the probability of receiving a spam call with any other area code from 200 to 999.

Question 4. Which of the following test statistics would be a reasonable choice to help differentiate between the two hypotheses?

Hint: For a refresher on choosing test statistics, check out the textbook section on Test Statistics.

1. The proportion of area codes that are 781 in 50 random spam calls
2. The total variation distance (TVD) between probability distribution of randomly chosen area codes, and the observed distribution of area codes. (*Remember the possible area codes are 200-999 inclusive*)
3. The probability of getting an area code of 781 out of all the possible area codes.
4. The proportion of area codes that are 781 in 50 random spam calls divided by 2
5. The number of times you see the area code 781 in 50 random spam calls

Assign `reasonable_test_statistics` to an array of numbers corresponding to these test statistics.

```
reasonable_test_statistics = [1,5]
```

For the rest of this question, suppose you decide to use the number of times you see the area code 781 in 50 spam calls as your test statistic.

Question 5. Write a function called `simulate` that generates exactly one simulated value of your test statistic under the null hypothesis. It should take no arguments and simulate 50 area codes under the assumption that the result of each area is sampled from the range 200-999 inclusive with equal probability. Your function should return the number of times you saw the 781 area code in those 50 random spam calls.

```
import random

def simulate():
    simulated_area_codes = [random.randint(200, 999) for _ in range(50)]
    count_781 = simulated_area_codes.count(781)

    return count_781

# Test the function

# Call your function to make sure it works
simulate()
```

```
    0
```

Question 6. Generate 20,000 simulated values of the number of times you see the area code 781 in 50 random spam calls. Assign `test_statistics_under_null` to an array that stores the result of each of these trials.

Hint: Use the function you defined in Question 5.

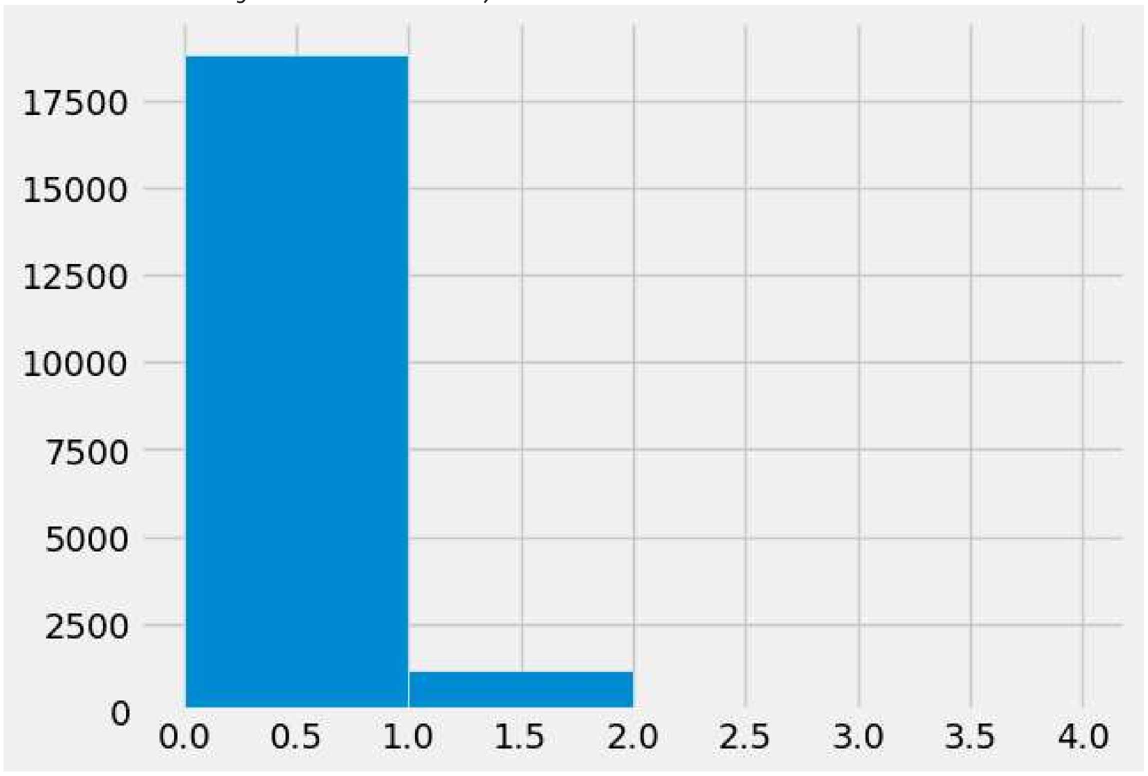
```
test_statistics_under_null = [simulate() for _ in range(20000)]
repetitions = sum(test_statistics_under_null)

...
repetitions
```

```
    1228
```

Question 7. Using the results from Question 6, generate a histogram of the empirical distribution of the number of times you saw the area code 781 in your simulation. **NOTE: Use the provided bins when making the histogram**

```
bins = np.arange(0,5,1) # Use these provided bins
plt.hist(test_statistics_under_null, bins=bins)
```

```
    (array([ 18808.,   1156.,     36.,      0.]),
     array([ 0.,  1.,  2.,  3.,  4.]),
     <BarContainer object of 4 artists>)
```



Question 8. Compute an empirical P-value for this test.

```
# First calculate the observed value of the test statistic from the `spam` table.
observed_val = 50

count_greater_equal_observed = sum(1 for value in test_statistics_under_null if value >= observed_val)

p_value = count_greater_equal_observed / len(test_statistics_under_null)

p_value
```

```
0.0
```

Question 9. Suppose you use a P-value cutoff of 1%. What do you conclude from the hypothesis test? Why?

We would conclude that we should reject the null hypothesis because our values do not reach the p value cutoff

Part 2: Multiple Spammers

Instead of checking if the area code is equal to his own, Yanay decides to check if the area code matches the area code of one of the 8 places he's been to recently, and wants to test if it's more likely to receive a spam call with an area code from any of those 8 places. These are the area codes of the places he's been to recently: 781, 617, 509, 510, 212, 858, 339, 626.

Question 10. Define the null hypothesis and alternative hypothesis for this investigation.

Reminder: Don't forget that your null hypothesis should fully describe a probability model that we can use for simulation later.

NUll: The probability of receiving a spam call with an area code from any of the 8 places Yanay has recently visited is not higher than the probability of receiving a spam call with an area code from any other location outside of those 8 places. ALT: The probability of receiving a spam call with an area code from any of the 8 places Yanay has recently visited is higher than the probability of receiving a spam call with an area code from any other location outside of those 8 places.

Suppose you decide to use the number of times you see any of the area codes of the places Yanay has been to in 50 spam calls as your test statistic.

Question 11. Write a function called `simulate_visited_area_codes` that generates exactly one simulated value of your test statistic under the null hypothesis. It should take no arguments and simulate 50 area codes under the assumption that the result of each area is sampled from the range 200-999 inclusive with equal probability. Your function should return the number of times you saw any of the area codes of the places Yanay has been to in those 50 spam calls.

Hint: You may find the textbook section on the `sample_proportions` function to be useful.

```
model_proportions = [781, 617, 509, 510, 212, 858, 339, 626]

def simulate_visited_area_codes():
    # Simulate 50 random area codes
    simulated_area_codes = [random.randint(200, 999) for _ in range(50)]

    # Count the number of times any of the visited area codes appear
    count_visited_area_codes = sum(1 for code in simulated_area_codes if code in model_proportions)

    return count_visited_area_codes

simulate_visited_area_codes()
```

```
     3
```

Question 12. Generate 20,000 simulated values of the number of times you see any of the area codes of the places Yanay has been to in 50 random spam calls. Assign `test_statistics_under_null` to an array that stores the result of each of these trials.

Hint: Use the function you defined in Question 11.

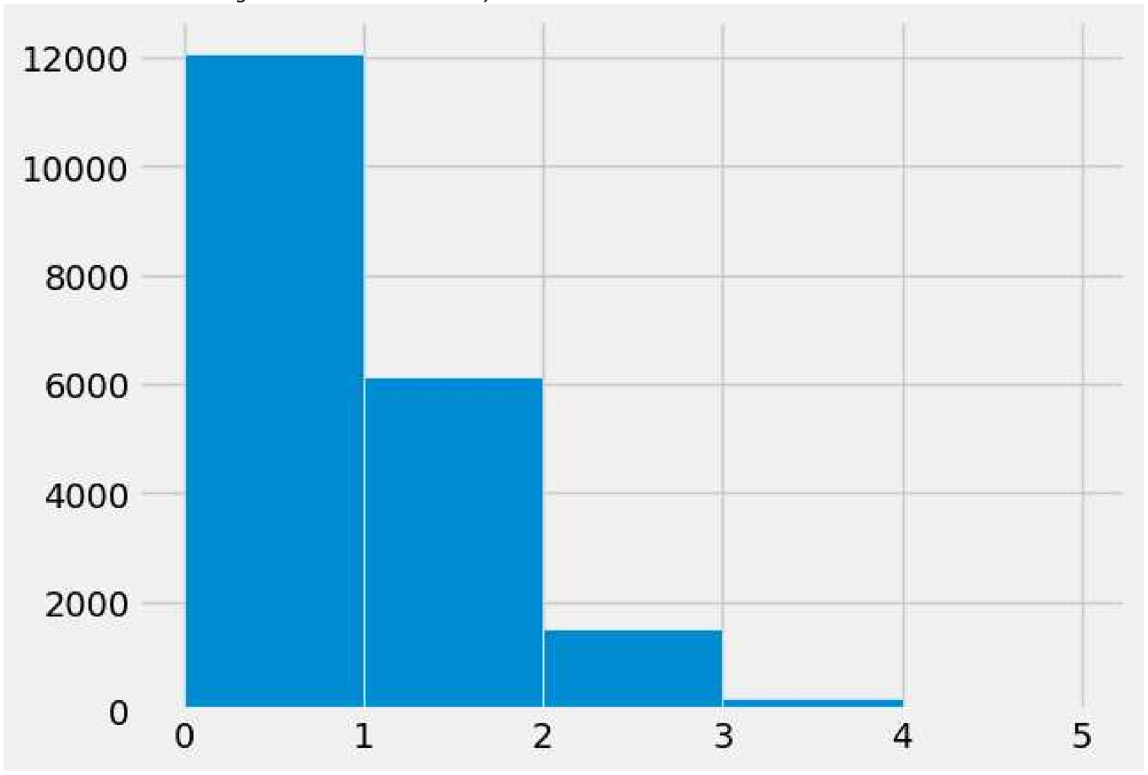
```
visited_test_statistics_under_null = test_statistics_under_null = [simulate_visited_area_codes() for _ in

repetitions = ...
...
visited_test_statistics_under_null
```

```
      0,
      0,
      0,
      1,
      1,
      0,
      0,
      0,
      0,
      2,
      0,
      0,
      0,
      1,
      2,
      0,
      1,
      1,
      0,
      1,
      0,
      0,
      1,
      0,
      1,
      ...]
```

Question 13. Using the results from Question 12, generate a histogram of the empirical distribution of the number of times you saw any of the area codes of the places Yanay has been to in your simulation. **NOTE: Use the provided bins when making the histogram**

```
bins_visited = np.arange(0,6,1) # Use these provided bins
plt.hist(test_statistics_under_null, bins=bins_visited)
```

```
(array([ 12075.,   6142.,   1520.,    235.,     28.]),
 array([ 0.,  1.,  2.,  3.,  4.,  5.]),
 <BarContainer object of 5 artists>)
```



Question 14. Compute an empirical P-value for this test.

```
visited_area_codes = make_array(781, 617, 509, 510, 212, 858, 339, 626)
# First calculate the observed value of the test statistic from the `spam` table.
visited_observed_value = sum(1 for code in test_statistics_under_null if code in visited_area_codes)
p_value = p_value = count_greater_equal_observed / len(test_statistics_under_null)
p_value
```

```
0.0
```

Question 15. Suppose you use a P-value cutoff of 0.05% (**Note: that's 0.05%, not our usual cutoff of 5%**). What do you conclude from the hypothesis test? Why?

I also missed this cutoff... I'm pretty sure I'm doing this right hahaha- I guess we again reject the null in favorite of the alternative due to us not hitting the cutoff

Question 16. Is `p_value`:

- (a) the probability that the spam calls favored the visited area codes,
- (b) the probability that they didn't favor, or
- (c) neither

If you chose (c), explain what it is instead.

b

Question 17. Is 0.05% (the P-value cutoff):

- (a) the probability that the spam calls favored the visited area codes,
- (b) the probability that they didn't favor, or
- (c) neither

If you chose (c), explain what it is instead.

I'm a bit confused by this question but if our P value was at the cutoff that means that the simulated experiments showed our original experiment to support the original hypothesis due to the simulations lining up with that hypothesis.

Question 18. Suppose you run this test for 4000 different people after observing each person's last 50 spam calls. When you reject the null hypothesis for a person, you accuse the spam callers of favoring the area codes that person has visited. If the spam callers were not actually favoring area codes that people have visited, can we compute how many times we will incorrectly accuse the spam callers of favoring area codes that people have visited? If so, what is the number? Explain your answer. Assume a 0.05% P-value cutoff.

multiply the number of tests 4000 by the false positive rate .05%, so we get 2

Part 3: Practice with A/B Tests

Yanay collects information about this month's spam calls. The table `with_labels` is a sampled table, where the `Area Code Visited` column contains either `"Yes"` or `"No"` which represents whether or not Yanay has visited the location of the area code. The `Picked Up` column is `1` if Yanay picked up and `0` if he did not pick up.

```
# Just run this cell
with_labels = Table().read_table("/content/drive/MyDrive/spam_picked_up.csv")
with_labels
```

Area Code Visited	Picked Up
No	0
No	1
No	1
Yes	0
No	0
No	0
Yes	0
No	1
No	1
No	1

... (40 rows omitted)

Yanay is going to perform an A/B Test to see whether or not he is more likely to pick up a call from an area code he has visited. Specifically, his null hypothesis is that there is no difference in the distribution of calls he picked up between visited and not visited area codes, with any difference due to chance. His alternative hypothesis is that there is a difference between the two categories, specifically that he thinks that he is more likely to pick up if he has visited the area code. We are going to perform a permutation test to test this. Our test statistic will be the difference in proportion of calls picked up between the area codes Yanay visited and the area codes he did not visit.

Question 19. Complete the `difference_in_proportion` function to have it calculate this test statistic, and use it to find the observed value. The function takes in a sampled table which can be any table that has the same columns as `with_labels`. We'll call `difference_in_proportion` with the sampled table `with_labels` in order to find the observed difference in proportion.

```
def difference_in_proportion(sample):
    # Take a look at the code for `proportion_visited` and use that as a
    # hint of what `proportions` should be assigned to
    proportions = sample.group('Area Code Visited', np.mean)
    proportion_visited = proportions.where("Area Code Visited", "Yes").column("Picked Up mean").item(0)
    proportion_not_visited = proportions.where("Area Code Visited", "No").column("Picked Up mean").item(
    return proportion_visited - proportion_not_visited

observed_diff_proportion = difference_in_proportion(with_labels)
observed_diff_proportion
```

```
    0.21904761904761905
```

Question 20. To perform a permutation test we shuffle the labels, because our null hypothesis is that the labels don't matter because the distribution of calls he picked up between visited and not visited area codes come from same underlying distribution. The labels in this case is the `"Area Code Visited"` column containing `"Yes"` and `"No"`.

Write a function to shuffle the table and return a test statistic using the function you defined in question 19.

Hint: To shuffle labels, we sample without replacement and then replace the appropriate column with the new shuffled column.

```
def shuffle_labels(table):
    shuffled_labels = table.sample(with_replacement=False).column("Area Code Visited")
    shuffled_table = table.drop("Area Code Visited").with_column("Area Code Visited", shuffled_labels)
    return shuffled_table

def simulate_one_stat():
    shuffled = shuffle_labels(with_labels)
    return difference_in_proportion(shuffled)

one_simulated_test_stat = simulate_one_stat()
one_simulated_test_stat
```

```
     0.028571428571428858
```

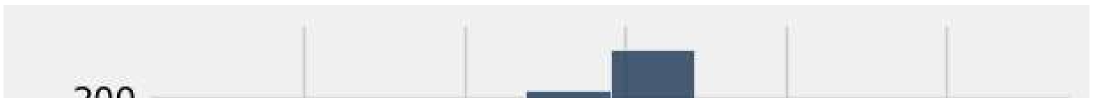
Question 21. Generate 1,000 simulated test statistic values. Assign `test_stats` to an array that stores the result of each of these trials.

Hint: Use the function you defined in Question 20.

We also provided code that'll generate a histogram for you after generating a 1000 simulated test statistic values.

```
def simulate_test_stats(num_simulations):
    test_stats = []
    for _ in range(num_simulations):
        test_stat = simulate_one_stat()
        test_stats.append(test_stat)
    return test_stats
trials = 1000
test_stats = simulate_test_stats(trials)
```

```
# here's code to generate a histogram of values and the red dot is the observed value
Table().with_column("Simulated Proportion Difference", test_stats).hist("Simulated Proportion Difference")
plt.plot(observed_diff_proportion, 0, 'ro', markersize=15);
```


200