hw07

March 29, 2025

[1]:
```
# Initialize Otter
import otter
grader = otter.Notebook("hw07.ipynb")
```

1 Homework 7: Testing Hypotheses

Please complete this notebook by filling in the cells provided. Before you begin, execute the previous cell to load the provided tests.

Helpful Resource:

- Python Reference: Cheat sheet of helpful array & table methods used in Data 8!

Recommended Readings:

- Sampling Methods Guide
- Testing Hypotheses
- A/B Testing

Please complete this notebook by filling in the cells provided. **Before you begin, execute the cell below to setup the notebook by importing some helpful libraries.** Each time you start your server, you will need to execute this cell again.

For all problems that you must write 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. Otherwise, you will fail tests that you thought you were passing previously!

Deadline:

This assignment is **due Wednesday, 10/16 at 5:00pm PT**. Submissions after this time will be accepted for 24 hours and will incur a 20% penalty. Any submissions later than this 24 hour period will not be accepted unless an extension has been granted as per the policies page. Turn it in by Tuesday, 10/15 at 5:00pm PT for 5 extra credit points.

Note: This homework has hidden tests on it. That means even though tests may say 100% passed, it doesn't mean your final grade will be 100%. We will be running more tests for correctness once everyone turns in the homework.

Directly sharing answers is not okay, but discussing problems with the course staff or with other students is encouraged. Refer to the policies page to learn more about how to learn cooperatively.

You should start early so that you have time to get help if you're stuck. Office hours are held Monday through Friday in Warren Hall 101B. The office hours schedule appears here.

The point breakdown for this assignment is given in the table below: | Category | Points | | — | — | | Autograder (Coding questions) | 80 | | Written (Q1.2, Q1.3, Q1.7, Q2.1, Q2.5) | 20 | | **Total** | 100 |

[2]:
```
# Run this cell to set up the notebook, but please don't change it.

# These lines import the Numpy and Datascience modules.
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)
```

1.1 1. Vaccinations Across The Nation

A vaccination clinic has two types of vaccines against a disease. Each person who comes in to be vaccinated gets either Vaccine 1 or Vaccine 2. One week, everyone who came in on Monday, Wednesday, and Friday was given Vaccine 1. Everyone who came in on Tuesday and Thursday was given Vaccine 2. The clinic is closed on weekends.

Doctor DeNero at the clinic said, "Oh wow, the distribution of vaccines is like tossing a coin that lands heads with probability $\frac{3}{5}$. If the coin lands on heads, you get Vaccine 1 and if the coin lands on tails, you get Vaccine 2."

But Doctor Sahai said, "No, it's not. We're not doing anything like tossing a (biased) coin."

That week, the clinic gave Vaccine 1 to 211 people and Vaccine 2 to 107 people. Conduct a test of hypotheses to see which doctor's position is better supported by the data.

Question 1.1. Given the information above, what was the sample size for the data, and what was the percentage of people who got **Vaccine 1? (4 points)**

Note: Your percent should be a number between 0 and 100, not a proportion between 0 and 1.

[3]:
```
sample_size = 211 + 107
percent_V1 = (211 / sample_size) * 100

print(f"Sample Size: {sample_size}")
print(f"Vaccine 1 Percent: {percent_V1}")
```

```
Sample Size: 318
Vaccine 1 Percent: 66.35220125786164
```

[4]:
```
grader.check("q1_1")
```

[4]:
```
q1_1 results: All test cases passed!
```

Question 1.2. State the null hypothesis. It should reflect the position of either Dr. DeNero or Dr. Sahai. **(4 points)**

Note: Check out 11.3 for a refresher on hypotheses.

The possibility of a person getting vaccine 1 or baccine 2 is like tossing a biased coin

---

Question 1.3. State the alternative hypothesis. It should reflect the position of the doctor you did not choose to represent in Question 1.2. **(4 points)**

Note: Check out 11.3 for a refresher on hypotheses.

The possibility of a person getting vaccine 1 or baccine 2 is nothing like tossing a biased coin

---

Question 1.4. One of the test statistics below is appropriate for testing these hypotheses. Assign the variable `valid_test_stat` to the number corresponding to the correct test statistic. **(4 points)**

Hint: Recall that large values of the test statistic should favor the alternative hypothesis.

1. percent of heads - 60
2. |percent of heads - 60|
3. percent of heads - 50
4. |percent of heads - 50|

[11]:
```
valid_test_stat = 2
valid_test_stat
```

[11]:
```
2
```

[12]:
```
grader.check("q1_4")
```

[12]:
```
q1_4 results: All test cases passed!
```

Question 1.5. Using your answer from Questions 1.1 and 1.4, find the observed value of the test statistic and assign it to the variable `observed_statistic`. Recall that the observed statistic is the test statistic value that was observed in the real life data. **(4 points)**

[13]:
```
observed_statistic = abs(percent_V1 - 60)
observed_statistic
```

[13]:
```
6.352201257861637
```

[14]:
```
grader.check("q1_5")
```

[14]:
```
q1_5 results: All test cases passed!
```

Question 1.6. In order to perform this hypothesis test, you must simulate the test statistic. From the four options below, pick the assumption that is needed for this simulation. Assign `assumption_needed` to an integer corresponding to the assumption. **(4 points)**

1. The statistic must be simulated under the null hypothesis.
2. The statistic must be simulated under the alternative hypothesis.
3. The statistic must be simulated under both hypotheses.
4. No assumptions are needed. We can just simulate the statistic.

[9]:
```
assumption_needed = 1
assumption_needed
```

[9]:
```
1
```

[10]:
```
grader.check("q1_6")
```

[10]:
```
q1_6 results: All test cases passed!
```

Question 1.7. Simulate 10,000 values of the test statistic under the assumption you picked in Question 1.6. **(4 points)**

As usual, start by defining a function that simulates one value of the statistic. Your function should use `sample_proportions`. (You may find a variable defined in Question 1.1 useful here!) Then, write a `for` loop to simulate multiple values and collect them in the array `simulated_statistics`.

Use as many lines of code as you need. We have included the code that visualizes the distribution of the simulated values. The red dot represents the observed statistic you found in Question 1.5.

[66]:
```
def one_simulated_statistic():
    return abs(sample_proportions(sample_size, make_array(0.6, 0.4)).
 ↪item(0)*100 - 60)
```

[67]:
```
# Run the this cell a few times to see how the simulated statistic changes
one_simulated_statistic()
```

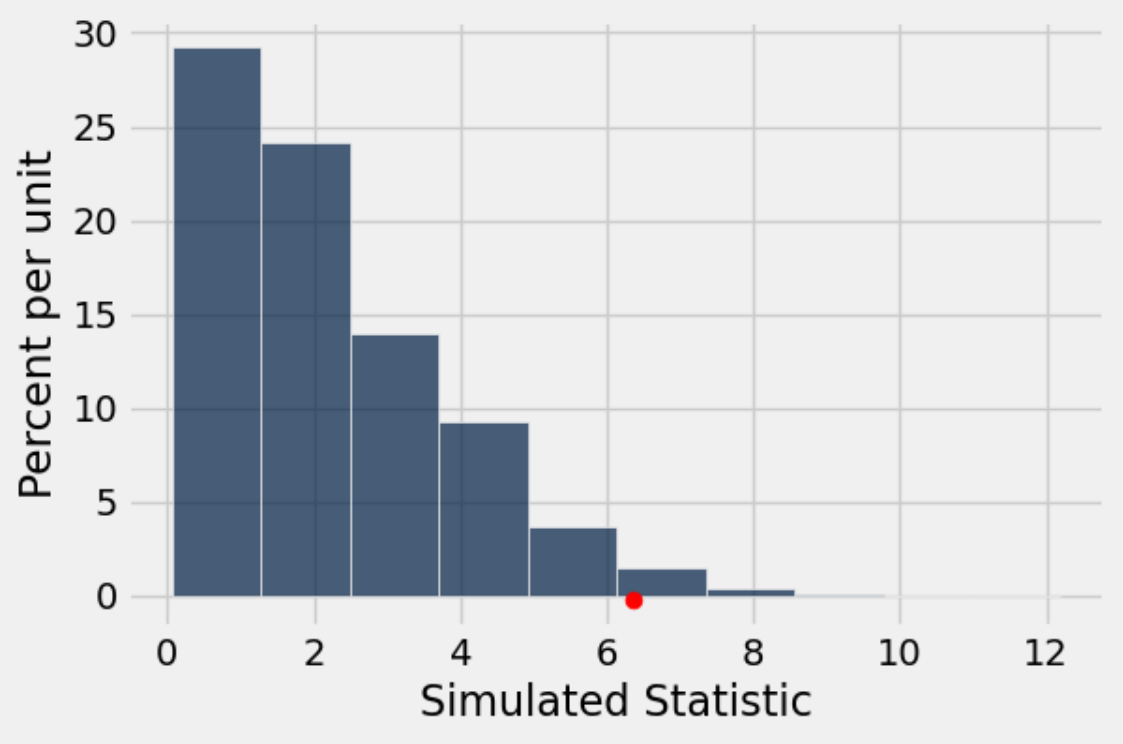
[67]:
```
3.2075471698113205
```

[68]:
```
num_simulations = 10000

simulated_statistics = make_array()
for _ in range(num_simulations):
```

```
    simulated_statistics = np.append(simulated_statistics,␣
 ↪one_simulated_statistic())
```

[69]:
```
# Run this cell to produce a histogram of the simulated statistics

Table().with_columns('Simulated Statistic', simulated_statistics).hist()
plt.scatter(observed_statistic, -0.002, color='red', s=40)
```

[69]:
```
<matplotlib.collections.PathCollection at 0x197099c1b20>
```



Question 1.8. Using `simulated_statistics`, `observed_statistic`, and `num_simulations`, find the empirical p-value based on the simulation. **(4 points)**

Hint: Reading 11.3.6 might be helpful for this question.

[70]:
```
p_value = np.count_nonzero(simulated_statistics >= observed_statistic) /␣
 ↪num_simulations
p_value
```

[70]:
```
0.0204
```

[71]:
```
grader.check("q1_8")
```

[71]:
```
q1_8 results: All test cases passed!
```

Question 1.9. Assign `correct_doctor` to the number corresponding to the correct statement below. Use the 5% cutoff for the p-value. **(4 points)**

1. The data support Dr. DeNero's position more than they support Dr. Sahai's.
2. The data support Dr. Sahai's position more than they support Dr. DeNero's.

As a reminder, here are the two claims made by Dr. DeNero and Dr. Sahai: > **Doctor DeNero:** "Oh wow, it's just like tossing a coin that lands heads with chance $\frac{3}{5}$. Heads you get Vaccine 1 and Tails you get Vaccine 2."

Doctor Sahai: "No, it's not. We're not doing anything like tossing a coin."

[75]:
```
correct_doctor = 2 if p_value <= 0.05 else 1
correct_doctor
```

[75]:
```
2
```

[76]:
```
grader.check("q1_9")
```

[76]:
```
q1_9 results: All test cases passed!
```

1.2 2. Using TVD as a Test Statistic

Before beginning this section, please read this section of the textbook on TVD!

Total variation distance (TVD) is a special type of test statistic that we use when we want to compare two distributions of *categorical data*. It is often used when we observe that a set of observed proportions/probabilities is different than what we expect under the null model.

Consider a six-sided die that we roll 6,000 times. If the die is fair, we would expect that each face comes up $\frac{1}{6}$ of the time. By random chance, a fair die won't always result in equal proportions (that is, we won't get exactly 1,000 of each face). However, if we suspect that the die might be unfair based on the data, we can conduct a hypothesis test using TVD to compare the expected $[\frac{1}{6}, \frac{1}{6}, \frac{1}{6}, \frac{1}{6}, \frac{1}{6}, \frac{1}{6}]$ distribution to what is actually observed.

In this part of the homework, we'll look at how we can use TVD to determine the effect that different factors have on happiness.

We will be working with data from the Gallup World Poll that is presented in the World Happiness Report, a survey of the state of global happiness. The survey ranked 137 countries by overall happiness and estimated the influence that economic production, social support, life expectancy, freedom, absence of corruption, and generosity had on population happiness. The study has been repeated for several years, but we'll be looking at data from the 2023 survey.

Run the cell below to load in the `happiness_scores` table.

[77]:
```
happiness_scores = Table.read_table("happiness_scores.csv").drop(12, 13, 14).
↪take(np.arange(137))
happiness_scores.show(5)
```

```
<IPython.core.display.HTML object>
```

Participants in the study were asked to evaluate their life satisfaction from a scale of 0 (worst possible life) to 10 (best possible life). The responses for each country were averaged to create the `Happiness Score`.

The columns `Economy (Log GDP per Capita)`, `Family`, `Health (Life Expectancy)`, `Freedom`, `Generosity`, and `Trust (Government Corruption)` estimate the extent to which each factor influences happiness, both for better or for worse. The happiness score is the sum of these factors; the larger a factor is, the more it contributes to overall happiness. [In other words, if you add up all the factors (in addition to a "Difference from Dystopia" value we excluded in the dataset), you get the happiness score.]

Let's look at the different factors that affect happiness in the United States. Run the cell below to view the row in `us_happiness` that contains data for the United States.

[78]:
```
us_happiness = happiness_scores.where("Country", "United States")
us_happiness
```

[78]:
```
Country       | Happiness Rank | Happiness Score | Standard error of happiness
score | Lower Confidence Interval | Upper Confidence Interval | Economy (Log GDP
per capita) | Family | Health (life expectancy) | Freedom | Generosity | Trust
(Government Corruption)
United States | 15             | 6.894           | 0.047
| 6.986                      | 6.802                       | 1.98
| 1.46    | 0.39                       | 0.557    | 0.21       | 0.172
```

To compare the different factors, we'll look at the proportion of the happiness score that is attributed to each variable. You can find these proportions in the table `us_happiness_factors` after running the cell below.

Note: The factors shown in `us_happiness` don't add up exactly to the happiness score, so we adjusted the proportions to only account for the data we have access to. The proportions were found by dividing each Happiness Factor value by the sum of all Happiness Factor values in `us_happiness`.

[79]:
```
us_happiness_factors = Table().read_table("us_happiness_factors.csv")
us_happiness_factors
```

[79]:
```
Happiness Factor              | Proportion of Happiness Score
Economy (GDP per Capita)      | 0.41521
Family                        | 0.306036
Health (Life Expectancy)      | 0.0818086
Freedom                       | 0.116865
Trust (Government Corruption) | 0.0361179
Generosity                    | 0.0439626
```

Question 2.1. Suppose we want to test whether or not each factor contributes the same amount to the overall Happiness Score. Define the null hypothesis, alternative hypothesis, and test statistic in the cell below. **(4 points)**

Note: Please format your answer as follows: - Null Hypothesis: … - Alternative Hypothesis: … - Test Statistic: …

- Null Hypothesis: All factors contribute to the overall happiness score equally
- Alternative Hypothesis: Some factors contribute to the overall happiness score more or less than others
- Test Statistic: sum(abs(actual_score - expected_score)) / 2

Question 2.2. Write a function `calculate_tvd` that takes in the observed distribution (`obs_dist`) and expected distribution under the null hypothesis (`null_dist`) and calculates the total variation distance. Use this function to set `observed_tvd` to be equal to the observed test statistic. **(4 points)**

[89]:
```
null_distribution = make_array(1/6, 1/6, 1/6, 1/6, 1/6, 1/6)

def calculate_tvd(obs_dist, null_dist):
    return np.sum(np.abs(obs_dist - null_dist)) / 2.0

observed_tvd = calculate_tvd(us_happiness_factors.column('Proportion of␣
 ↪Happiness Score'), null_distribution)
observed_tvd
```

[89]:
```
0.38791256366666665
```

[90]:
```
grader.check("q2_2")
```

[90]:
```
q2_2 results: All test cases passed!
```

Question 2.3. Create an array called `simulated_tvds` that contains 10,000 simulated values under the null hypothesis. Assume that the original sample consisted of 1,000 individuals. **(4 points)**

Hint: The `sample_proportions` function may be helpful to you. Refer to the Python Reference Sheet to read up on it!

[93]:
```
simulated_tvds = make_array()

for _ in range(10000):
    simulated_statistics = sample_proportions(1000, null_distribution)
    simulated_tvds = np.append(simulated_tvds,␣
 ↪calculate_tvd(simulated_statistics, null_distribution))

simulated_tvds
```

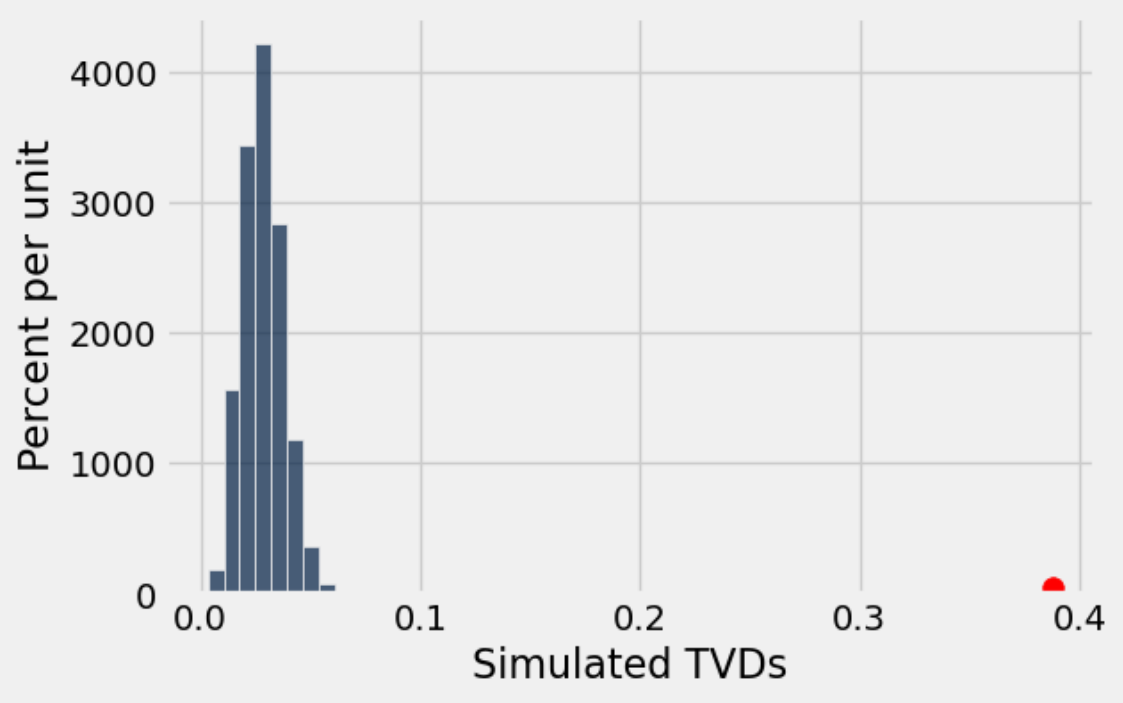
[93]:
```
array([ 0.01966667,  0.03133333,  0.026     , …,  0.02966667,
        0.031     ,  0.04366667])
```

[94]:
```
grader.check("q2_3")
```

[94]:
```
q2_3 results: All test cases passed!
```

Run the cell below to plot a histogram of your simulated test statistics, as well as a red dot representing the observed value of the test statistic.

[ ]:
```
Table().with_column("Simulated TVDs", simulated_tvds).hist()
plt.scatter(observed_tvd, 0.5, color='red', s=70, zorder=2)
plt.show()
```



Question 2.4. Use your simulated statistics to calculate the p-value of your test. Make sure that this number is consistent with what you observed in the histogram above. **(4 points)**

[98]:
```
p_value_tvd = np.count_nonzero(simulated_tvds >= observed_tvd) /␣
↪len(simulated_tvds)
p_value_tvd
```

[98]:
```
0.0
```

[99]:
```
grader.check("q2_4")
```

[99]:
```
q2_4 results: All test cases passed!
```

Question 2.5. What can you conclude about how each factor contributes to the overall happiness score in the US? Explain your answer using the results of your hypothesis test. Assume a p-value cutoff of 5%. **(4 points)**

Because the p_value is less than 0.05, so the null hypothesis is rejected, so that every factor does not contribute equally to the overall happiness score

1.3 3. Who is Older?

Data scientists have drawn a simple random sample of size 500 from a large population of adults. Each member of the population happened to identify as either "male" or "female". (Though many people identify outside of the gender binary, in this particular population of interest, each member happened to identify as either male or female.) Data was collected on several attributes of the sampled people, including age. The table `sampled_ages` contains one row for each person in the sample, with columns containing the individual's gender identity.

[100]:
```
sampled_ages = Table.read_table('age.csv')
sampled_ages.show(5)
```

```
<IPython.core.display.HTML object>
```

Question 3.1. How many females were there in our sample? Please use the provided skeleton code. **(4 points)**

Hint: Keep in mind that `.group` sorts categories in alphabetical order!

[107]:
```
num_females = sampled_ages.group("Gender").where("Gender", "female").
  ↪column('count').item(0)
num_females
```

[107]:
```
260
```

[108]:
```
grader.check("q3_1")
```

[108]:
```
q3_1 results: All test cases passed!
```

Question 3.2. Complete the cell below so that `avg_male_vs_female` evaluates to `True` if the sampled males are older than the sampled females on average, and `False` otherwise. Use Python code to achieve this. **(4 points)**

[124]:
```
group_mean_tbl = sampled_ages.group("Gender", np.mean)
group_means = group_mean_tbl.column(1)      # array of mean ages
avg_male_vs_female = group_means.item(1) > group_means.item(0)
avg_male_vs_female
```

[124]:
```
True
```

[115]:
```
grader.check("q3_2")
```

[115]: q3_2 results: All test cases passed!

Question 3.3. The data scientists want to use the data to test whether males are older than females. One of the following statements is their null hypothesis and another is their alternative hypothesis. Assign `null_statement_number` and `alternative_statement_number` to the numbers of the correct statements in the code cell below. **(4 points)**

1. In the sample, the males and females have the same distribution of ages; the sample averages of the two groups are different due to chance.
2. In the population, the males and females have the same distribution of ages; the sample averages of the two groups are different due to chance.
3. The age distributions of males and females in the population are different due to chance.
4. The males in the sample are older than the females, on average.
5. The males in the population are older than the females, on average.
6. The average ages of the males and females in the population are different.

[126]:
```
null_statement_number = 5
alternative_statement_number = 2
```

[127]:
```
grader.check("q3_3")
```

[127]: q3_3 results: All test cases passed!

Question 3.4. The data scientists have decided to use a permutation test. Assign `permutation_test_reason` to the number corresponding to the reason they made this choice. **(4 points)**

1. Since a person's age shouldn't be related to their gender, it doesn't matter who is labeled "male" and who is labeled "female", so you can use permutations.
2. Under the null hypothesis, permuting the labels in the `sampled_ages` table is equivalent to drawing a new random sample with the same number of males and females as in the original sample.
3. Under the null hypothesis, permuting the rows of `sampled_ages` table is equivalent to drawing a new random sample with the same number of males and females as in the original sample.

Note: Check out 12.1 for a refresher on random permutations and permutation tests.

[128]:
```
permutation_test_reason = 1
permutation_test_reason
```

[128]: 1

[129]:
```
grader.check("q3_4")
```

[129]: q3_4 results: All test cases passed!

Question 3.5. To test their hypotheses, the data scientists have followed our textbook's advice and chosen a test statistic where the following statement is true: Large values of the test statistic favor the alternative hypothesis.

The data scientists' test statistic is one of the two options below. Which one is it? Assign the appropriate number to the variable `correct_test_stat`. **(4 points)**

1. "male age average - female age average" in a sample created by randomly shuffling the male/female labels
2. "|male age average - female age average|" in a sample created by randomly shuffling the male/female labels

[138]:
```
correct_test_stat = 2
correct_test_stat
```

[138]:
```
2
```

[139]:
```
grader.check("q3_5")
```

[139]:
```
q3_5 results: All test cases passed!
```

Question 3.6. Complete the cell below so that `observed_statistic_ab` evaluates to the observed value of the data scientists' test statistic. Use as many lines of code as you need, and remember that you can use any quantity, table, or array that you created earlier. **(4 points)**

[140]:
```
observed_statistic_ab = abs(group_means.item(1) - group_means.item(0))
observed_statistic_ab
```

[140]:
```
1.314102564102562
```

[141]:
```
grader.check("q3_6")
```

[141]:
```
q3_6 results: All test cases passed!
```

Question 3.7. Assign `shuffled_labels` to an array of shuffled male/female labels. The rest of the code puts the array in a table along with the data in `sampled_ages`. **(4 points)**

[136]:
```
shuffled_labels = sampled_ages.sample(with_replacement =False).column(0)
original_with_shuffled_labels = sampled_ages.with_columns('Shuffled Label',␣
 ↪shuffled_labels)
original_with_shuffled_labels
```

[136]:
```
Gender | Age  | Shuffled Label
male   | 23   | female
male   | 29   | male
male   | 29   | male
female | 49   | female
```

```
female | 33   | male
male   | 31   | male
male   | 60   | male
male   | 38   | female
female | 60   | female
female | 27   | female
… (490 rows omitted)
```

[137]:
```
grader.check("q3_7")
```

[137]:
```
q3_7 results: All test cases passed!
```

Question 3.8. The comparison below uses the array `shuffled_labels` from Question 3.7 and the count `num_females` from Question 3.1.

For this comparison, assign the correct letter as a string (e.g. `correct_q8 = 'A'`) from one of the following options to the variable `correct_q8`. **Pretend this is a midterm problem and do not solve it using a code cell. (4 points)**

```
comp = np.count_nonzero(shuffled_labels == 'female') == num_females
```

A. `comp` is set to `True`. B. `comp` is set to `False`. C. `comp` is set to `True` or `False`, depending on how the shuffle came out.

[144]:
```
correct_q8 = 'A'
correct_q8
```

[144]:
```
'A'
```

[145]:
```
grader.check("q3_8")
```

[145]:
```
q3_8 results: All test cases passed!
```

Question 3.9. Define a function `simulate_one_statistic` that takes no arguments and returns one simulated value of the test statistic. We've given you a skeleton, but feel free to approach this question in a way that makes sense to you. Use as many lines of code as you need. Refer to the code you have previously written in this problem, as you might be able to re-use some of it. **(4 points)**

[146]:
```
def simulate_one_statistic():
    "Returns one value of our simulated test statistic"
    shuffled_labels =np.random.permutation(sampled_ages.column('Gender'))
    shuffled_tbl =Table().with_columns('Shuffled Gender',␣
 ↪shuffled_labels,'Age', sampled_ages.column('Age'))
    group_means =shuffled_tbl.group('Shuffled Gender', np.mean).column(1)
    return abs(group_means.item(1) - group_means.item(0))
```

[147]:
```
grader.check("q3_9")
```

[147]:
```
q3_9 results: All test cases passed!
```

After you have defined your function, run the following cell a few times to see how the statistic varies.

[148]:
```
simulate_one_statistic()
```

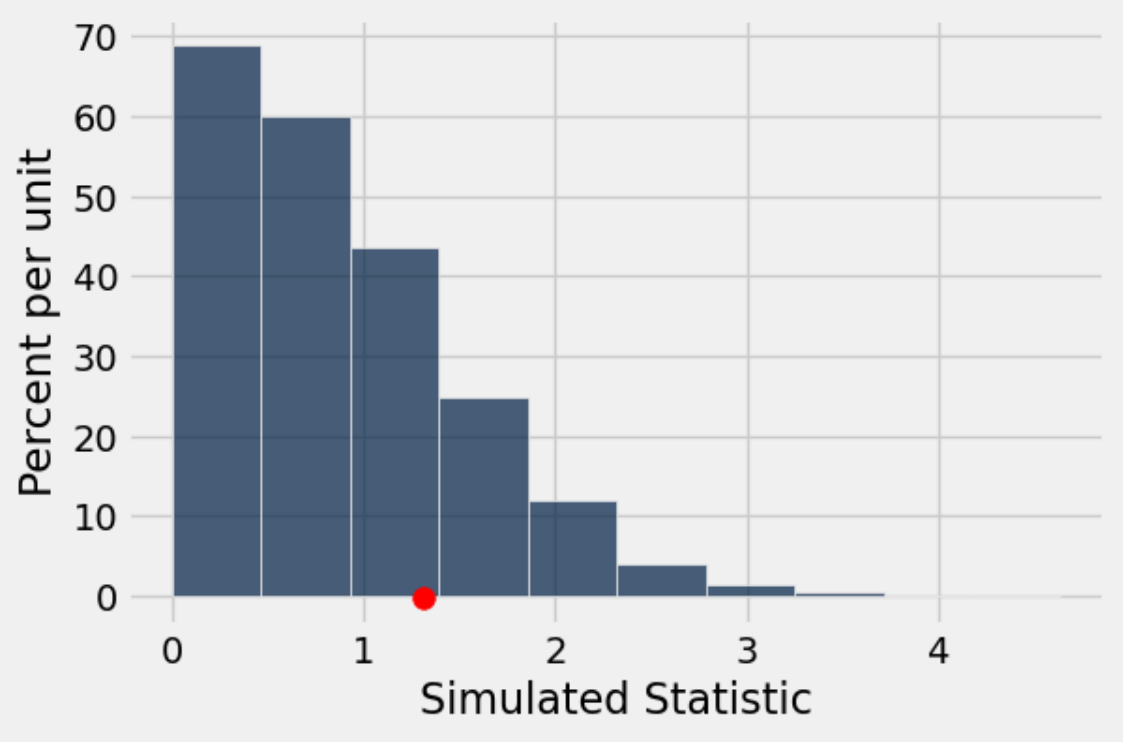
[148]:
```
0.3285256410256423
```

Question 3.10. Complete the cell to simulate 5,000 values of the statistic. We have included the code that draws the empirical distribution of the statistic and shows the value of `observed_statistic_ab` from Question 3.6. Feel free to use as many lines of code as you need. **(4 points)**

Note: This cell will take around a minute to run.

[149]:
```
simulated_statistics_ab = make_array()

for _ in range(5000):
    simulated_statistics_ab = np.append(simulated_statistics_ab,␣
 ↪simulate_one_statistic())

# Do not change these lines
Table().with_columns('Simulated Statistic', simulated_statistics_ab).hist()
plt.scatter(observed_statistic_ab, -0.002, color='red', s=70)
```

[149]:
```
<matplotlib.collections.PathCollection at 0x1970baa9cd0>
```



Question 3.11. Use the simulation to find an empirical approximation to the p-value. Assign `p_val` to the appropriate p-value from this simulation. Then, assign `conclusion` to either `null_hyp` or `alt_hyp`. **(4 points)**

Note: Assume that we use the 5% cutoff for the p-value.

[150]:
```
# These are variables provided for you to use.
null_hyp = 'The data are consistent with the null hypothesis.'
alt_hyp = 'The data support the alternative more than the null.'

p_val = np.count_nonzero(simulated_statistics_ab >= observed_statistic_ab) /␣
 ↪len(simulated_statistics_ab)
conclusion = null_hyp

p_val, conclusion # Do not change this line
```

[150]:
```
(0.229, 'The data are consistent with the null hypothesis.')
```

[151]:
```
grader.check("q3_11")
```

[151]:
```
q3_11 results: All test cases passed!
```

You're done with Homework 6!

Important submission steps: 1. Run the tests and verify that they all pass. 2. Choose **Save Notebook** from the **File** menu, then **run the final cell**. 3. Click the link to download the zip file. 4. Go to Gradescope and submit the zip file to the corresponding assignment. The name of this assignment is "HW 06 Autograder".

It is your responsibility to make sure your work is saved before running the last cell.

1.4 Pets of Data 8

Gus is enjoying the weather we're having lately. Congrats on surviving the heat and finishing homework 7!

1.5 Written Work Submission

Below, you will see two cells. Running the first cell will automatically generate a PDF of all questions that need to be manually graded, and running the second cell will automatically generate a zip with your autograded answers. You are responsible for submitting both the coding portion (the zip) and the written portion (the PDF) to their respective Gradescope portals. **Please save before exporting!**

> **Important: You must correctly assign the pages of your PDF after you submit to the correct gradescope assignment. If your pages are not correctly assigned and/or not in the correct PDF format by the deadline, we reserve the right to award no points for your written work.**

If there are issues with automatically generating the PDF in the first cell, you can try downloading the notebook as a PDF by clicking on `File -> Save and Export Notebook As... -> Webpdf`. If that doesn't work either, you can manually take screenshots of your answers to the manually graded questions and submit one single PDF of your screenshots. Either way, **you are responsible for ensuring your submision follows our requirements, we will NOT be granting regrade requests for submissions that don't follow instructions.**

You must submit the PDF generated via one of these methods, we will not accept screenshots or Word documents.

1.6 Submission

Make sure you have run all cells in your notebook in order before running the cell below, so that all images/graphs appear in the output. The cell below will generate a zip file for you to submit. **Please save before exporting!**

```
[ ]: # Save your notebook first, then run this cell to export your submission.
grader.export(pdf=False, run_tests=True)
```