lab09_master

November 22, 2019

1 Lab 9: Statistic Inference Review

Welcome to Lab 9! Over the past several weeks you've learned the fundamentals of statistic inference, from hypothesis testing to confidence intervals. This lab will review the main topics in inference. As we move on in the course you should be comfortable defining hypotheses, picking a test statistic, interpreting p-values, and building confidence intervals.

[1]:
```
# 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)

# These lines load the tests.
from client.api.notebook import Notebook
ok = Notebook('lab09.ok')
_ = ok.submit()
```

2 Overview

A hypothesis test is motivated by the data we observe or by a belief we have about the world. Many times after collecting data, we notice a trend in our data that is different from what we expect. We want to determine if that deviation can be explained by something other than random chance: is the world actually different from what we expect, or was our data just "unlucky"? In order to make an objective conclusion about the "truth", we conduct a hypothesis test.

When conducting a statistical test, we usually† follow this basic structure: 1. Define a null and alternative hypothesis. 2. Choose a p-value cutoff (usually .05 or .01). 2. Choose a test statistic and calculate the observed test statistic (from your real data). 3. Create a distribution of test statistics under the null hypothesis by simulating data. 4. Calculate a p-value using your simulated test statistics. 5. Using the p-value cutoff from step 2, determine if your data are more consistent with your null or alternative hypothesis: * If the p-value is above the cutoff, the data are more consistent with the null hypothesis. * If the p-value is below the cutoff, the data are more consistent with the alternative hypothesis.

Depending on the type of test you want to do, each step will be a little different. You should be familiar with the different types of tests we've covered and their corresponding hypotheses and test statistics.

†For some null hypotheses, we can't always follow this process. In some of those cases, we can use confidence intervals to conduct hypothesis tests. See the "Hypothesis tests using confidence intervals" section below for more information on how those work.

2.0.1 Hypothesis tests using confidence intervals

Sometimes, we might not be able to simulate data under the null hypothesis. For example, if your null hypothesis is of the form "The mean of some population is 40 inches" and your alternative is of the form "The mean of the population is not 40 inches", and you observe a sample with mean 37 inches, it's not clear how you'd use the null hypothesis to sample. In a case like this, you can use a confidence interval to conduct the hypothesis test.

The basic structure is similar to the one described at the start of the lab, but with some modifications: 1. Define a null and alternative hypothesis (they must be of the form "The mean is X" and "The mean is not X"). 2. Choose a p-value cutoff, and call it q. 3. Construct a (100-q)% interval using bootstrap sampling (for example, if your p-value cutoff q is .01, or 1%, then construct a 99% confidence interval). 4. Using the confidence interval, determine if your data are more consistent with your null or alternative hypothesis: * If the null hypothesis mean X is in your confidence interval, the data are more consistent with the null hypothesis. * If the null hypothesis mean X is *not* in your confidence interval, the data are more consistent with the alternative hypothesis.

Answer the following questions to test your knowledge of the different variations of hypothesis tests.

2.1 The Data

Throughout this lab, we'll 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 155 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 - we'll be looking at data from the 2016 survey.

Run the cell below to load the dataset

```
[2]: happiness_scores = Table.read_table("happiness_scores.csv")
     happiness_scores
```

```
[2]: Country     | Region                        | Happiness Rank | Happiness Score |
     Lower Confidence Interval | Upper Confidence Interval | Economy (GDP per Capita)
```

```
| Family  | Health (Life Expectancy) | Freedom | Trust (Government Corruption) |
Generosity
Denmark     | Western Europe            | 1              | 7.526           |
7.46                   | 7.592                         | 1.44178
| 1.16374 | 0.79504                  | 0.57941 | 0.44453                       |
0.36171
Switzerland | Western Europe            | 2              | 7.509           |
7.428                  | 7.59                          | 1.52733
| 1.14524 | 0.86303                  | 0.58557 | 0.41203                       |
0.28083
Iceland     | Western Europe            | 3              | 7.501           |
7.333                  | 7.669                         | 1.42666
| 1.18326 | 0.86733                  | 0.56624 | 0.14975                       |
0.47678
Norway      | Western Europe            | 4              | 7.498           |
7.421                  | 7.575                         | 1.57744
| 1.1269  | 0.79579                  | 0.59609 | 0.35776                       |
0.37895
Finland     | Western Europe            | 5              | 7.413           |
7.351                  | 7.475                         | 1.40598
| 1.13464 | 0.81091                  | 0.57104 | 0.41004                       |
0.25492
Canada      | North America             | 6              | 7.404           |
7.335                  | 7.473                         | 1.44015
| 1.0961  | 0.8276                   | 0.5737  | 0.31329                       |
0.44834
Netherlands | Western Europe            | 7              | 7.339           |
7.284                  | 7.394                         | 1.46468
| 1.02912 | 0.81231                  | 0.55211 | 0.29927                       |
0.47416
New Zealand | Australia and New Zealand | 8              | 7.334           |
7.264                  | 7.404                         | 1.36066
| 1.17278 | 0.83096                  | 0.58147 | 0.41904                       |
0.49401
Australia   | Australia and New Zealand | 9              | 7.313           |
7.241                  | 7.385                         | 1.44443
| 1.10476 | 0.8512                   | 0.56837 | 0.32331                       |
0.47407
Sweden      | Western Europe            | 10             | 7.291           |
7.227                  | 7.355                         | 1.45181
| 1.08764 | 0.83121                  | 0.58218 | 0.40867                       |
0.38254
… (147 rows omitted)
```

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

The columns `Economy (GDP per Capita)`, `Family`, `Health (Life Expectancy)`, `Freedom`, `Trust (Government Corruption)`, and `Generosity` estimate the extent to which each factor influences happiness, both for better or for worse. The higher the value, the more influential that factor was in calculating the country's happiness score. 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.

3 1. Using TVD as a Test Statistic

Total variation distance (TVD) is a special type of test statistic that we use when we want to compare two distributions. 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 6000 times. If the die is fair, we would expect that each face comes up 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 1000 of each face). However, if we suspect that based on the data, the die might be unfair, we can conduct a hypothesis test using TVD to compare the expected [1/6, 1/6, 1/6, 1/6, 1/6, 1/6] distribution to what is actually observed.

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

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.

[3]:
```
us_happiness = happiness_scores.where("Country", "United States")
us_happiness
```

[3]:
```
Country       | Region        | Happiness Rank | Happiness Score | Lower
Confidence Interval | Upper Confidence Interval | Economy (GDP per Capita) |
Family  | Health (Life Expectancy) | Freedom | Trust (Government Corruption) |
Generosity
United States | North America | 13             | 7.104           | 7.02
| 7.188                     | 1.50796                     | 1.04782 | 0.779
| 0.48163 | 0.14868                           | 0.41077
```

Remember that the columns `Economy (GDP per Capita)`, `Family`, `Health (Life Expectancy)`, `Freedom`, `Trust (Government Corruption)`, and `Generosity` describe the contribution that each factor has on the country's happiness - the sum of these values is approximately equal to the country's Happiness Score.

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.

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

```
[4]: Happiness Factor                | Proportion of Happiness Score
     Economy (GDP per Capita)        | 0.344609
     Family                          | 0.239455
     Health (Life Expectancy)        | 0.178022
     Freedom                         | 0.110065
     Trust (Government Corruption)   | 0.0339773
     Generosity                      | 0.0938718
```

Question 1.1. 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. Check your work with a neighbor, LA, or uGSI.

```
BEGIN QUESTION
name: q1_1
```

SOLUTION:

Null Hypothesis: Each factor contributes an equal amount to the happiness score. Any deviation is due to random chance.

Alternative Hypothesis: Some factors contribute more to the happiness score than other factors.

Test Statistic: the total variation distance (TVD) between the observed score proportions and the expected score proportions under the null.

Question 1.2. 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.

```
BEGIN QUESTION
name: q1_2
```

```
[5]: null_distribution = np.ones(6) * (1/6)

     def calculate_tvd(obs_dist, null_dist):
         # BEGIN SOLUTION
         return sum(abs(obs_dist - null_dist))/2
         # END SOLUTION

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

```
[5]: 0.26208562431156396
```

Question 1.3. Create an array called `simulated_tvds` that contains 10000 simulated values under the null hypothesis. Assume that the original sample consisted of 1000 individuals.

```
BEGIN QUESTION
```

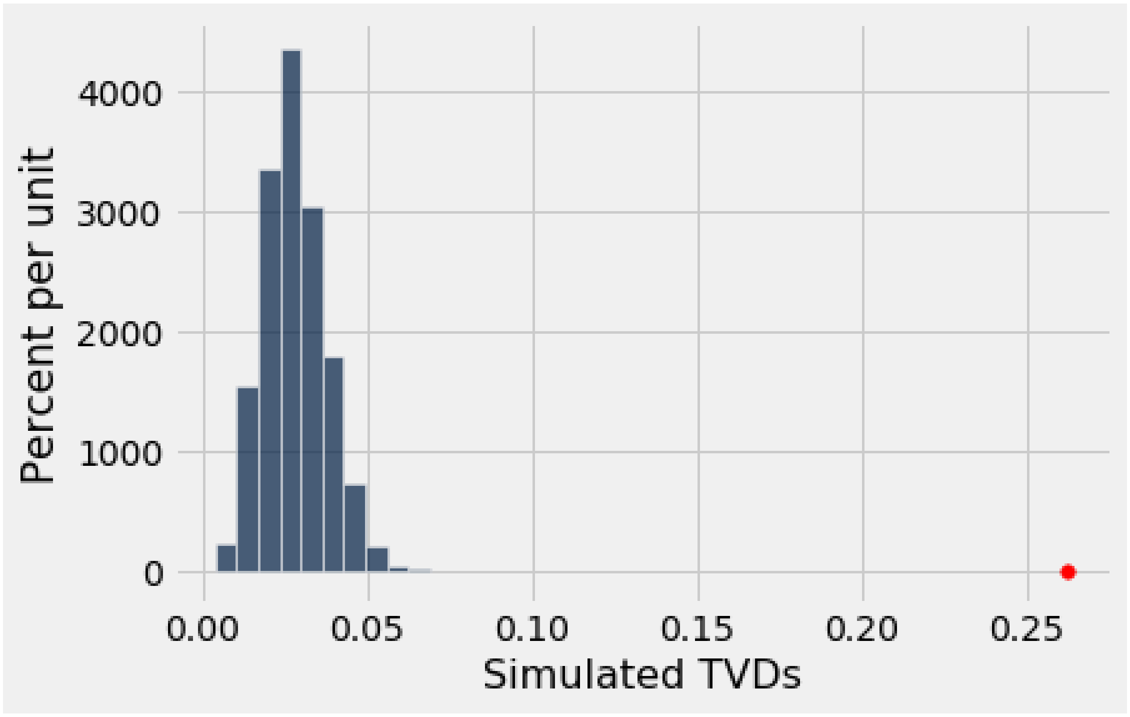
name: q1_3

```
[8]: simulated_tvds = make_array() # SOLUTION

     # BEGIN SOLUTION
     for i in np.arange(10000):
         simulated_proportions = sample_proportions(1000, null_distribution)
         one_tvd = calculate_tvd(simulated_proportions, null_distribution)
         simulated_tvds = np.append(simulated_tvds, one_tvd)
     # END SOLUTION
```

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

```
[12]: Table().with_column("Simulated TVDs", simulated_tvds).hist()
      plt.scatter(observed_tvd , 0, color='red', s=30, zorder=2);
      plt.show();
```



Question 1.4. Use your simluated statistics to calculate the p-value of your test. Make sure that this number is consistent with what you observed in the histogram above.

BEGIN QUESTION
name: q1_4

```
[13]: p_value_1 = np.count_nonzero(simulated_tvds >= observed_tvd) # SOLUTION
      p_value_1
```

```
[13]: 0
```

Question 1.5. 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 0.05

```
BEGIN QUESTION
name: q1_5
```

SOLUTION: The p-value is less than our cutoff, so the data is more consistent with the alternative hypothesis. The factors do not equally contribute to the overall happiness score of a country.

4 2. A/B Testing

A/B testing allows us to determine if two numerical samples come from the same underlying distribution. We perform an A/B test if we think that knowing the "label" of an individual tells us something about its value for a numerical variable.

In the birth weights example given in class, we suspected that knowing the smoker status of a mother gave us insight into whether their baby's birth weight would be higher or lower. If smoker status has no effect on birth weights, then the birth weights should have the same distributions for both smoker and non-smoker mothers. If mothers who are smokers have babies with lower birth weights, we would observe two different distributions for smoker and non-smoker mothers.

We can compare two distributions by finding the difference in their means. If the difference in means is small, the data suggest that the distributions are similar. If the difference in means is large, the data suggest that the distributions are different.

In this section of the lab, we'll use an A/B test to determine if happiness scores differ by geographic region.

What do the happiest countries in the world have in common? Run the cell below to view the countries with the top 5 highest happiness scores.

```
[15]: happiness_scores.take(np.arange(5))
```

```
[15]: Country     | Region         | Happiness Rank | Happiness Score | Lower
      Confidence Interval | Upper Confidence Interval | Economy (GDP per Capita) |
      Family  | Health (Life Expectancy) | Freedom | Trust (Government Corruption) |
      Generosity
      Denmark     | Western Europe | 1              | 7.526           | 7.46
      | 7.592                     | 1.44178                    | 1.16374 | 0.79504
      | 0.57941 | 0.44453                       | 0.36171
      Switzerland | Western Europe | 2              | 7.509           | 7.428
      | 7.59                      | 1.52733                    | 1.14524 | 0.86303
      | 0.58557 | 0.41203                       | 0.28083
      Iceland     | Western Europe | 3              | 7.501           | 7.333
      | 7.669                     | 1.42666                    | 1.18326 | 0.86733
      | 0.56624 | 0.14975                       | 0.47678
```

```
Norway      | Western Europe | 4                | 7.498             | 7.421
| 7.575                      | 1.57744                           | 1.1269  | 0.79579
| 0.59609 | 0.35776                          | 0.37895
Finland     | Western Europe | 5                | 7.413             | 7.351
| 7.475                      | 1.40598                           | 1.13464 | 0.81091
| 0.57104 | 0.41004                          | 0.25492
```

All five of these countries are in Western Europe! It looks like there might be an association between region and happiness score. Run the cell below to view the average happiness score for each region, sorted in descending order.

[16]:
```
happiness_scores.select("Region", "Happiness Score").group("Region", np.mean).
↪sort(1, descending = True)
```

[16]:
```
Region                          | Happiness Score mean
Australia and New Zealand       | 7.3235
North America                   | 7.254
Western Europe                  | 6.68567
Latin America and Caribbean     | 6.10175
Eastern Asia                    | 5.62417
Middle East and Northern Africa | 5.38605
Central and Eastern Europe      | 5.37069
Southeastern Asia               | 5.33889
Southern Asia                   | 4.56329
Sub-Saharan Africa              | 4.13642
```

There's a wide range of values between the different regions, but is this difference due to anything other than random chance? Using an A/B test, let's compare the distributions of happiness scores between Western Europe and Sub-Saharan Africa to see if there is a statistically significant difference between the two regions.

Question 2.1. We want to run an A/B Test to see if the happiness scores in Western Europe are higher than scores in Sub-Saharan Africa. In the cell below, define the null hypothesis, alternative hypothesis, and test statistic for your test.

```
BEGIN QUESTION
name: q2_1
```

SOLUTION:

Null Hypothesis: Happiness scores in Western Europe and Sub-Saharan Africa come from the same distribution. Any difference in their distributions is due to random chance.

Alternative Hypothesis: Happiness scores in Western Europe are higher than happiness scores in Sub-Saharan Africa.

Test Statistic: The difference in mean happiness scores between Western Europe and Sub-Saharan Africa

Question 2.2. Create a table `we_ssa_happiness` that contains data from `happiness_scores` for the `Western Europe` and `Sub-Saharan Africa` regions.

Hint: You might want to look at the different predicates that you can use with the `.where()` table method on the Python reference page

```
BEGIN QUESTION
name: q2_2
```

[17]:
```
regions_of_interest = make_array("Sub-Saharan Africa", "Western Europe")
we_ssa_happiness = happiness_scores.where("Region", are.
↪contained_in(regions_of_interest)) # SOLUTION
we_ssa_happiness
```

[17]:
```
Country     | Region         | Happiness Rank | Happiness Score | Lower
Confidence Interval | Upper Confidence Interval | Economy (GDP per Capita) |
Family   | Health (Life Expectancy) | Freedom | Trust (Government Corruption) |
Generosity
Denmark     | Western Europe | 1              | 7.526           | 7.46
| 7.592                     | 1.44178                    | 1.16374 | 0.79504
| 0.57941 | 0.44453                         | 0.36171
Switzerland | Western Europe | 2              | 7.509           | 7.428
| 7.59                      | 1.52733                    | 1.14524 | 0.86303
| 0.58557 | 0.41203                         | 0.28083
Iceland     | Western Europe | 3              | 7.501           | 7.333
| 7.669                     | 1.42666                    | 1.18326 | 0.86733
| 0.56624 | 0.14975                         | 0.47678
Norway      | Western Europe | 4              | 7.498           | 7.421
| 7.575                     | 1.57744                    | 1.1269  | 0.79579
| 0.59609 | 0.35776                         | 0.37895
Finland     | Western Europe | 5              | 7.413           | 7.351
| 7.475                     | 1.40598                    | 1.13464 | 0.81091
| 0.57104 | 0.41004                         | 0.25492
Netherlands | Western Europe | 7              | 7.339           | 7.284
| 7.394                     | 1.46468                    | 1.02912 | 0.81231
| 0.55211 | 0.29927                         | 0.47416
Sweden      | Western Europe | 10             | 7.291           | 7.227
| 7.355                     | 1.45181                    | 1.08764 | 0.83121
| 0.58218 | 0.40867                         | 0.38254
Austria     | Western Europe | 12             | 7.119           | 7.045
| 7.193                     | 1.45038                    | 1.08383 | 0.80565
| 0.54355 | 0.21348                         | 0.32865
Germany     | Western Europe | 16             | 6.994           | 6.93
| 7.058                     | 1.44787                    | 1.09774 | 0.81487
| 0.53466 | 0.28551                         | 0.30452
Belgium     | Western Europe | 18             | 6.929           | 6.861
| 6.997                     | 1.42539                    | 1.05249 | 0.81959
| 0.51354 | 0.26248                         | 0.2424
… (49 rows omitted)
```

Question 2.3. Define a function `calculate_test_statistic` that finds the difference between the mean happiness score in Western Europe and the mean happiness score in Sub-Saharan Africa in the table being passed through as the `tbl` argument. Use this function to set `observed_happiness_difference` to the observed test statistic in the `we_ssa_happiness` table.

```
BEGIN QUESTION
name: q2_3
```

[21]:
```
def calculate_test_statistic(tbl):
    # BEGIN SOLUTION
    grouped_table = tbl.select("Region", "Happiness Score").group("Region", np.
 ↪mean)
    ssa_mean_happiness = grouped_table.where("Region", "Sub-Saharan Africa").
 ↪column(1).item(0)
    we_mean_happiness = grouped_table.where("Region", "Western Europe").
 ↪column(1).item(0)
    test_statistic = we_mean_happiness - ssa_mean_happiness
    return test_statistic
    # END SOLUTION

observed_happiness_difference = calculate_test_statistic(we_ssa_happiness) #␣
 ↪SOLUTION
observed_happiness_difference
```

[21]:
```
2.5492456140350868
```

Question 2.4. Define a function `shuffle_regions` that simulates one random permutation of the `we_ssa_happiness` table. The function should take no arguments and return a copy `we_ssa_happiness` where the values of `Region` have been shuffled.

```
BEGIN QUESTION
name: q2_4
```

[24]:
```
def shuffle_regions():
    # BEGIN SOLUTION
    shuffled_regions = we_ssa_happiness.sample(we_ssa_happiness.num_rows,␣
 ↪with_replacement = False).column("Region")
    we_ssa_happiness_with_shuffled_regions = we_ssa_happiness.
 ↪with_column("Region", shuffled_regions)
    return we_ssa_happiness_with_shuffled_regions
    # END SOLUTION

shuffle_regions()
```

[24]:
```
Country     | Region             | Happiness Rank | Happiness Score | Lower
Confidence Interval | Upper Confidence Interval | Economy (GDP per Capita) |
Family   | Health (Life Expectancy) | Freedom | Trust (Government Corruption) |
Generosity
Denmark     | Sub-Saharan Africa | 1              | 7.526           | 7.46
```

```
| 7.592                       | 1.44178                     | 1.16374 | 0.79504
| 0.57941 | 0.44453                         | 0.36171
Switzerland | Western Europe     | 2                | 7.509              | 7.428
| 7.59                        | 1.52733                     | 1.14524 | 0.86303
| 0.58557 | 0.41203                         | 0.28083
Iceland     | Sub-Saharan Africa | 3                | 7.501              | 7.333
| 7.669                       | 1.42666                     | 1.18326 | 0.86733
| 0.56624 | 0.14975                         | 0.47678
Norway      | Western Europe     | 4                | 7.498              | 7.421
| 7.575                       | 1.57744                     | 1.1269  | 0.79579
| 0.59609 | 0.35776                         | 0.37895
Finland     | Sub-Saharan Africa | 5                | 7.413              | 7.351
| 7.475                       | 1.40598                     | 1.13464 | 0.81091
| 0.57104 | 0.41004                         | 0.25492
Netherlands | Western Europe     | 7                | 7.339              | 7.284
| 7.394                       | 1.46468                     | 1.02912 | 0.81231
| 0.55211 | 0.29927                         | 0.47416
Sweden      | Sub-Saharan Africa | 10               | 7.291              | 7.227
| 7.355                       | 1.45181                     | 1.08764 | 0.83121
| 0.58218 | 0.40867                         | 0.38254
Austria     | Sub-Saharan Africa | 12               | 7.119              | 7.045
| 7.193                       | 1.45038                     | 1.08383 | 0.80565
| 0.54355 | 0.21348                         | 0.32865
Germany     | Western Europe     | 16               | 6.994              | 6.93
| 7.058                       | 1.44787                     | 1.09774 | 0.81487
| 0.53466 | 0.28551                         | 0.30452
Belgium     | Sub-Saharan Africa | 18               | 6.929              | 6.861
| 6.997                       | 1.42539                     | 1.05249 | 0.81959
| 0.51354 | 0.26248                         | 0.2424
… (49 rows omitted)
```

Question 2.5. Create an array `happiness_differences` that contains 1000 simulated values of your test statistic.

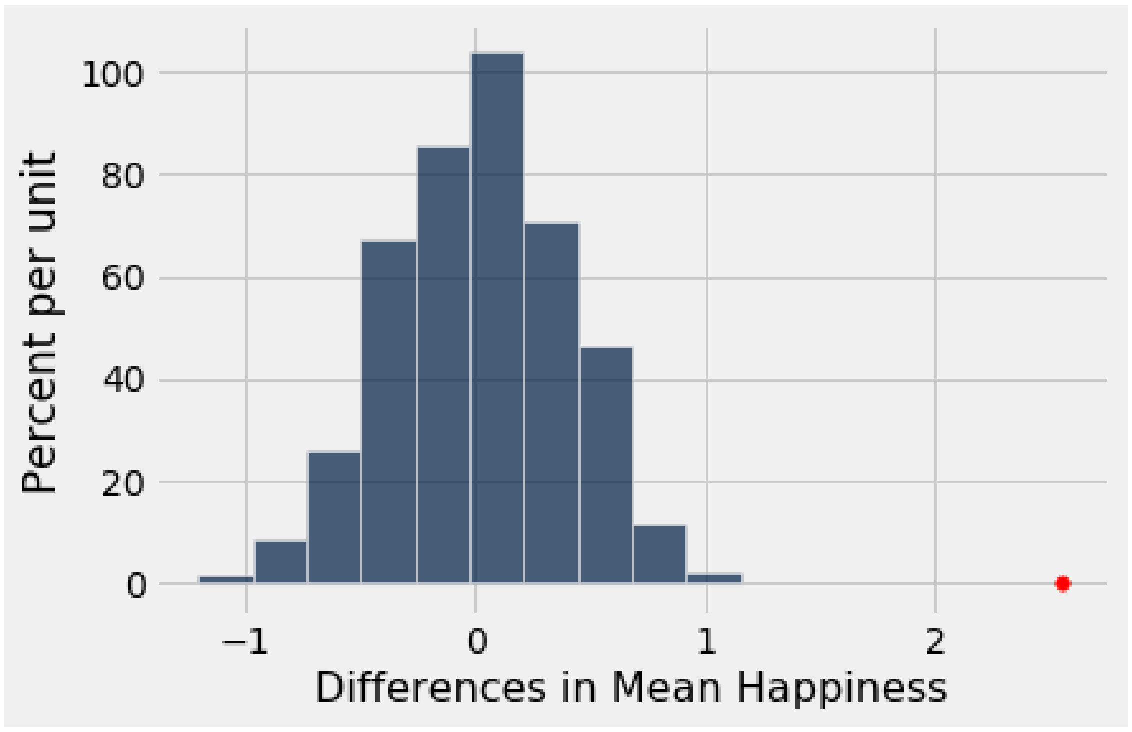
```
BEGIN QUESTION
name: q2_5
```

[28]:
```
happiness_differences = make_array() # SOLUTION

# BEGIN SOLUTION
for i in np.arange(1000):
    one_resample = shuffle_regions()
    one_test_statistic = calculate_test_statistic(one_resample)
    happiness_differences= np.append(happiness_differences, one_test_statistic)
# END SOLUTION
```

Run the cell below to view a histogram of your test statistics plotted with the observed test statistic.

[31]:
```
Table().with_column("Differences in Mean Happiness", happiness_differences).
 ↪hist()
plt.scatter(observed_happiness_difference , 0, color='red', s=30, zorder=2);
plt.show();
```



Question 2.6. Use your simluated statistics to calculate the p-value of your test. Make sure that this number is consistent with what you observed in the histogram above.

```
BEGIN QUESTION
name: q2_6
```

[32]:
```
p_value_2 = np.count_nonzero(happiness_differences >=␣
 ↪observed_happiness_difference) # SOLUTION
p_value_2
```

[32]:
```
0
```

Question 2.7. What can you conclude the distributions of happiness score in Western Europe and Sub-Saharan Africa? Explain your answer using the results of your hypothesis test. Assume a p-value cutoff of 0.05

```
BEGIN QUESTION
name: q2_7
```

SOLUTION: The p-value is less than our cutoff, so the data is more consistent with the alternative hypothesis. The distribution of happiness scores in Western Europe is higher than the distribution of happiness scores in Sub-Saharan Africa.

5 3. Bootstrapping and Confidence Intervals

We're often interested in estimating a population parameter by calculating a statistic from a sample. However, sampling will always result in some degree of uncertainty. A sample captures a small subset of the population, so we can never be sure that a statistic (computed from only one sample) captures the complete truth.

We can learn more about the population by going back and taking more samples. However, sampling is a very costly process and we don't always have the resources to collect more samples from the population. Instead, when we have a large representative sample of the population, we can create new samples by sampling with replacement from the original sample in a process called *bootstrapping*. By the law of averages, the distribution of the original sample is likely to resemble the population, and the distributions of all the "resamples" are likely to resemble the original sample. Therefore, the distributions of all the resamples are likely to resemble the population as well.

We calculate one statistic for each of our new bootstrap samples. By combining all of them, we can produce a confidence interval for the population parameter. We use this interval to estimate the parameter to some degree of confidence (typically 95%). A confidence level of 95% means that we can be 95% confident that the process resulted in a "good" interval.

In other words, before we start the process of collecting a sample of data from the population and creating a new confidence interval, there is a 95% chance that the interval will contain the population parameter. If we were to create many new confidence intervals, around 95% of those intervals will contain the population parameter.

It's important to note that when a confidence interval has already been generated, it has either a 0% or 100% chance of containing the true population parameter. This is because the true population parameter isn't random: it's unknown, but it's fixed.

Let's use bootstrapping to see how much the top two countries differ in their happiness scores. Run the cell below to view the happiness scores and 95% confidence interval bounds for Denmark and Switzerland.

[33]:
```
denmark = happiness_scores.where("Country", "Denmark")
switzerland = happiness_scores.where("Country", "Switzerland")

happiness_scores.take(np.arange(2)).select(0, 3, 4, 5)
```

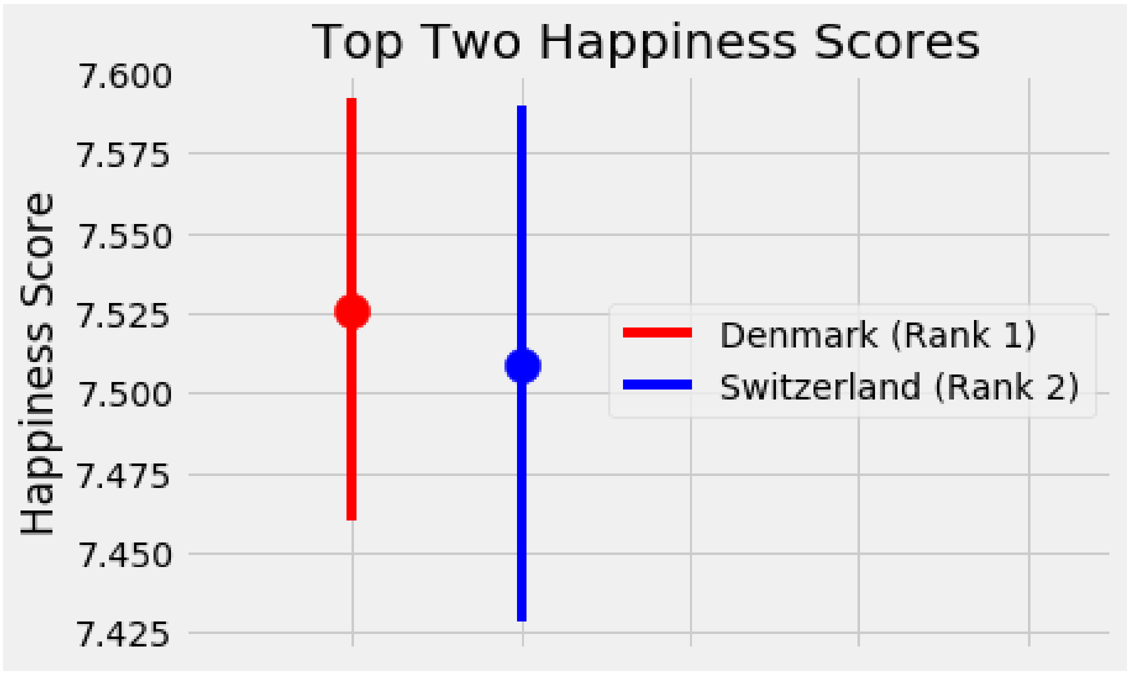
[33]:
```
Country     | Happiness Score | Lower Confidence Interval | Upper Confidence
Interval
Denmark     | 7.526           | 7.46                      | 7.592
Switzerland | 7.509           | 7.428                     | 7.59
```

The happiness scores are very similar, and we see that the confidence intervals overlap. Run the cell below to view the scores and confidence intervals plotted side-by-side: you don't have to understand how the plotting code works.

[34]:
```
denmark_cis = make_array(denmark.column("Lower Confidence Interval").item(0),\
                         denmark.column("Upper Confidence Interval").item(0))

switzerland_cis = make_array(switzerland.column("Lower Confidence Interval").
 ↪item(0),\
                         switzerland.column("Upper Confidence Interval").
 ↪item(0))

plt.plot([0, 0], denmark_cis, label = "Denmark (Rank 1)", color='red')
plt.scatter(0, denmark.column("Happiness Score").item(0), color='red', s=200,␣
 ↪zorder=2)
plt.plot([1, 1], switzerland_cis, label = "Switzerland (Rank 2)", color='blue')
plt.scatter(1, switzerland.column("Happiness Score").item(0), color='blue',␣
 ↪s=200, zorder=2)
plt.legend(loc = "right")
plt.xlim(-1,4.5)
plt.ylabel("Happiness Score")
plt.title("Top Two Happiness Scores")
plt.tick_params(
    axis='x',
    which='both',
    bottom=False,
    top=False,
    labelbottom=False)
```



Visually, the intervals look very similar, but we can't definitively say that there's no difference in their happiness scores without running a hypothesis test. If we decrease the width of the confidence interval, we can get a tighter bound on what the mean happiness score in Denmark could be.

Question 3.1. How can we reduce the width of a 95% confidence interval of sample means?

```
BEGIN QUESTION
name: q3_1
```

SOLUTION: Decrease the standard deviation of sample means by increasing sample size.

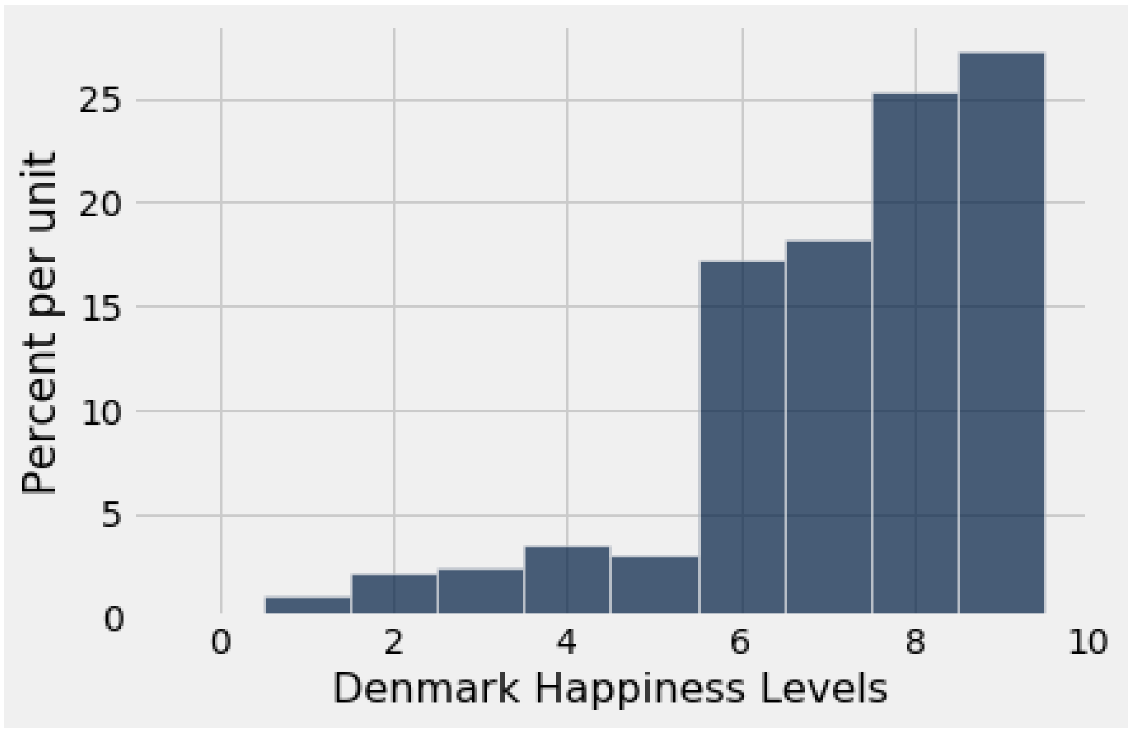
To see if Denmark actually has a higher happiness score than Switzerland, pretend we went out and created a random sample that contains more Danish residents than the original study. We recorded their responses in the table `denmark_scores`. Run the cell below to see data from our new sample.

[35]:
```
denmark_scores = Table.read_table("denmark_scores.csv")
print("Happiness Score (Mean Happiness Level) of Sample:", np.
 ↪mean(denmark_scores.column(0)))
print("Sample Size:", denmark_scores.num_rows)
denmark_scores
```

```
Happiness Score (Mean Happiness Level) of Sample: 7.525892857142857
Sample Size: 13440
```

[35]:
```
Denmark Happiness Levels
5
10
10
6
10
4
9
6
9
10
… (13430 rows omitted)
```

[36]:
```
denmark_scores.hist(0, bins=np.arange(-.5, 10.5, 1))
```



Using this new sample, we'll create new bootstrap samples to produce a 95% confidence interval of Denmark's mean happiness scores. We can use this interval to test the following hypotheses at a 0.05 significant level:

> **Null Hypothesis:** The mean happiness score for Denmark and Switzerland is the same. Any difference in happiness score is due to random chance.
>
> **Alternative Hypothesis:** The mean happiness score for Denmark is different than the mean happiness score for Switzerland.

If the confidence interval of Denmark's mean happiness scores contains Switzerland's mean happiness score (7.509), the data are more consistent with the null hypothesis. If the confidence interval of Denmark's mean happiness scores does not contain Switzerland's mean happiness score (7.509), the data are more consistent with the alternative hypothesis.

Question 3.2. Define a function `bootstrap_sample` that generates a bootstrap sample from `tbl`, a one column table that looks like `denmark_scores`. The function should return the mean happiness score from the bootstrap sample.

Call your function once on `denmark_scores` to make sure it works.

```
BEGIN QUESTION
name: q3_2
```

[37]:
```
def bootstrap_sample(tbl):
    # BEGIN SOLUTION
    bootstrap_sample = tbl.sample(tbl.num_rows, with_replacement = True)
    return np.mean(bootstrap_sample.column(0))
    # END SOLUTION

bootstrap_sample(denmark_scores)
```

```
[37]: 7.549181547619048
```

Question 3.3. Create an array `denmark_means` that contains the mean happiness scores from 1000 different bootstrap samples.

```
BEGIN QUESTION
name: q3_3
```

```
[40]: repetitions = 1000 # SOLUTION

denmark_means = make_array() # SOLUTION

# BEGIN SOLUTION
for i in np.arange(repetitions):
    one_difference = bootstrap_sample(denmark_scores)
    denmark_means = np.append(denmark_means, one_difference)
# END SOLUTION
```

Question 3.4. Create a 95% confidence interval of mean happiness scores. Assign `bootstrap_ci_left_end` to the lower bound of the interval and `bootstrap_ci_right_end` to the upper bound of the interval.

```
BEGIN QUESTION
name: q3_4
```

```
[43]: bootstrap_ci_left_end = percentile(2.5, denmark_means) # SOLUTION
bootstrap_ci_right_end = percentile(97.5, denmark_means) # SOLUTION
bootstrap_ci_left_end, bootstrap_ci_right_end
```

```
[43]: (7.495535714285714, 7.558705357142857)
```

Now suppose that we didn't have access to Python and can't create any bootstrap samples. However, since the happiness score is the sample mean of our happiness levels, we know that they are normally distributed under the Central Limit Theorem. We also know that in a normal distribution, approximately 95% of values are contained within 2 standard deviations of the mean. Using this information, we can still construct a 95% confidence interval with only the sample mean and standard deviation using this formula:

$$\text{Lower bound of 95 percent CI} = \text{Sample Mean} - 2 * \text{SD of Sample Means}$$

$$\text{Upper bound of 95 percent CI} = \text{Sample Mean} + 2 * \text{SD of Sample Means}$$

Where the formula for the standard deviation of sample means is given as the following:

$$\text{SD of Sample Means} = \frac{\text{SD of Population}}{\sqrt{\text{Sample Size}}}$$

When we have a large random sample that is representative of our population, the standard deviation of the sample gives a good estimate for the standard deviation of the population. Therefore, we can use the following formula to approximate the standard deviation of sample means if we don't know the population standard deviation.

$$\text{SD of Sample Means} \approx \frac{\text{SD of Sample}}{\sqrt{\text{Sample Size}}}$$

Question 3.5. Create a 95% confidence interval of the happiness score (mean happiness level) for Denmark. You may not use any form of resampling or simulation to find your interval bounds. Your confidence interval should be very similar to the one you found in Question 3.4.

```
BEGIN QUESTION
name: q3_5
```

```
[47]: sample_mean = np.mean(denmark_scores.column(0))
      sample_sd = np.std(denmark_scores.column(0))
      sd_of_sample_mean = sample_sd / (denmark_scores.num_rows)**0.5 # SOLUTION
      left_end = sample_mean - 2 * sd_of_sample_mean # SOLUTION
      right_end = sample_mean + 2 * sd_of_sample_mean # SOLUTION
      left_end, right_end
```

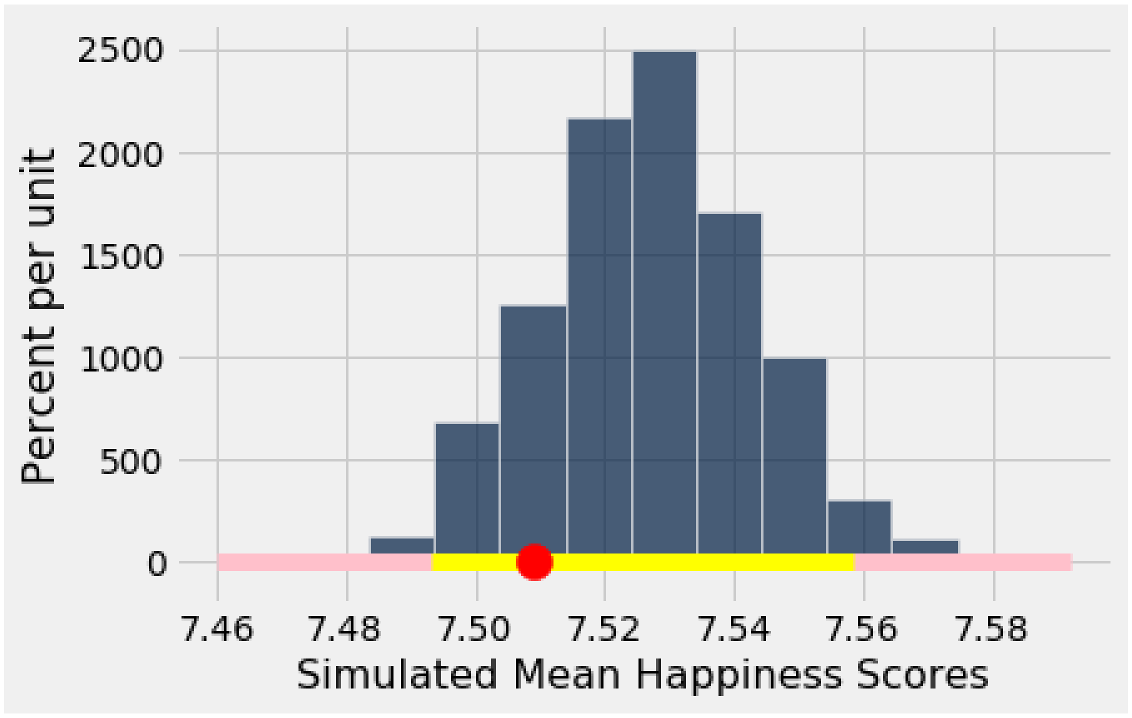
```
[47]: (7.493090232263674, 7.55869548202204)
```

```
[48]: # TEST
      type(left_end) in set([float, np.float32, np.float64])
```

```
[48]: True
```

Run the cell below to plot the simulated mean happiness scores for Denmark and the observed mean happiness score for Switzerland: * The original confidence interval for Denmark's happiness scores from `happiness_scores` is plotted as a pink line. * The confidence interval created by our bootstrap procedure is plotted as a yellow line. * The happiness score for Switzerland is plotted as a red dot.

```
[51]: Table().with_column("Simulated Mean Happiness Scores", denmark_means).hist()
      plt.plot(make_array(denmark.column("Lower Confidence Interval").item(0),\
                          denmark.column("Upper Confidence Interval").item(0)), \
               make_array(0, 0), color='pink', lw=7, zorder=1)
      plt.plot(make_array(left_end, right_end), make_array(0, 0), color='yellow',␣
       ↪lw=7, zorder=1)
      plt.scatter(switzerland.column("Happiness Score").item(0), 0, color='red',␣
       ↪s=200, zorder=2);
```



Question 3.6. Interpret the results of the hypothesis test by assigning `happiness_ci_statements` to an array of integer(s) that correspond to the true statement(s).

1. The data are more consistent with the alternative hypothesis, since our confidence interval is contained inside the original study's confidence interval.
2. The data are more consistent with the null hypothesis, since our confidence interval contains Switzerland's happiness score.
3. The happiness levels of individual Danish residents in our sample are normally distributed.
4. The mean happiness levels (happiness scores) of our bootstrapped samples are normally distributed

```
BEGIN QUESTION
name: q3_6
```

[52]:
```
happiness_ci_statements = make_array(2, 4)
```

Congratulations! You've completed Lab 9, wrapping up the statistical inference portion of the course.

Be sure to - **run all the tests** (the next cell has a shortcut for that), - **Save and Checkpoint** from the `File` menu, - **run the last cell to submit your work**, - and ask one of the staff members to check you off.

[ ]:
```
# For your convenience, you can run this cell to run all the tests at once!
import os
```

```python
print("Running all tests...")
_ = [ok.grade(q[:-3]) for q in os.listdir("tests") if q.startswith('q')]
print("Finished running all tests.")
```

[ ]:
```python
# Run this cell to submit your work *after* you have passed all of the test␣
↪cells.
# It's ok to run this cell multiple times. Only your final submission will be␣
↪scored.
_ = ok.submit()
```