Bootstrapping: Percentile Method & Bootstrap-t

Computer-Intensive in Statistics Project II Report

Scott Williams

1 Introduction

1.1 Goal of the Project

A data set was constructed by asking fifteen students at University of North Florida how much they spent on their last haircut. We want to provide a 90% confidence interval for the average number of dollars that a student at UNF spends on a haircut. To do this, we will be using a resampling procedure called bootstrapping. Specifically, we will be using two different non-parametric bootstrapping methods called the percentile method and bootstrap-t.

2 Approach

2.1 Programming Environment & Preparation

To perform both the percentile method and bootstrap-t, we will be using a Python 3 environment. The necessary dependencies for the code are the following Python libraries: math, numpy, and matplotlib. Before any bootstrapping methods take place, we first need to define an array that contains our data set and a variable that represents how many bootstrap samples we desire. We will then define a function that performs both bootstrapping methods and returns their respective 90% confidence intervals.

Before moving on to any code, let's conceptually define what is meant by the variables that represent our data set and how many bootstrap samples we desire.

Let our original data be represented by the following array:

$$\textbf{Original Data} = \begin{bmatrix} x_1 & x_2 & x_3 & \dots & x_n \end{bmatrix}$$

Bootstrapping is a resampling procedure that allows for sampling with replacement. In other words, we allow for repeat entries in our new sample. To make this more clear, let's consider a

simple example. Say we have the original data set $\begin{bmatrix}1 & 2 & 3\end{bmatrix}$. After sampling with replacement, we could potentially get the new data set $\begin{bmatrix}2 & 2 & 1\end{bmatrix}$. This new data set will be our bootstrap sample.

In general, let's define a bootstrap sample to be:

$$\textbf{Bootstrap Sample} = \begin{bmatrix}x_1^* & x_2^* & x_3^* & \dots & x_n^*\end{bmatrix}$$

In this study, we want to generate B bootstrap samples where B = 500.

2.2 Bootstrap Function

We will be using the function titled bootstrap to perform both bootstrapping methods at the same time. It is important that we perform both methods in the same function since we will be relying on random number generation to generate the indices that we will use to sample from the original data to then build our bootstrap samples. Doing both methods in the same function ensures that we construct the 90% confidence intervals using the same bootstrap samples. If we had split this process into two different functions, the bootstrap samples our confidence interval would be based on for percentile method would be completely different bootstrap samples from the ones used in bootstrap-t. This would make any kind of meaningful comparison between the two methods difficult. However, since we are doing this process all in one function, we know that both methods will be based on the sample bootstrap samples which will allow us to compare the results between the two methods. Our function bootstrap takes the data, the number of bootstrap samples B, and α our significance level as parameters. In this study, the data is the fifteen reported dollar amounts that the UNF students spent on their last haircut, B = 500, and $\alpha = 0.1$.

2.3 Percentile Method

We construct a 90% confidence interval using the percentile method with the following algorithm:

1. Pass in original data = $\begin{bmatrix}x_1 & x_2 & \dots & x_{15}\end{bmatrix}$, B = 500, and $\alpha = 0.1$ to the function.
2. Since we want to find a confidence interval for the average number of dollars, propose $\overline{X} = \frac{\sum_{n=1}^{15} x_n}{15}$ as the estimator.
3. Get a bootstrap sample $\begin{bmatrix}x_1^* & x_2^* & \dots & x_{15}^*\end{bmatrix}$.
4. Compute the estimator $\overline{X}^* = \frac{\sum_{n=1}^{15} x_n^*}{15}$.
5. Repeat steps 3 and 4 B times.
6. Rank the B $\overline{X}^*$'s in ascending order.
7. Find $\left(\overline{X}^*_{(\lfloor B*\frac{\alpha}{2}\rfloor+1)},\ \overline{X}^*_{(\lfloor B*(1-\frac{\alpha}{2})\rfloor+1)}\right) = \left(\overline{X}^*_{26},\ \overline{X}^*_{476}\right)$ where the subscripts of $\overline{X}^*$ indicate the index of the ranked $\overline{X}^*$'s. This will be our confidence interval.

2.4 Bootstrap-t

We construct a 90% confidence interval using bootstrap-t with the following algorithm:

1. Pass in original data = $\begin{bmatrix} x_1 & x_2 & \dots & x_{15} \end{bmatrix}$, B = 500, and $\alpha = 0.1$ to the function.
2. Since we want to find a confidence interval for the average number of dollars, propose $\overline{X} = \frac{\sum_{n=1}^{15} x_n}{15}$ as the estimator.
3. Based on the original data, compute the sample mean $\overline{X}$, the sample variance $S^2 = \frac{\sum_{n=1}^{15}(x_n - \overline{X})^2}{14}$, and the standard error $S_{\bar{X}} = \sqrt{\frac{S^2}{15}}$.
4. Get B bootstrap samples $\begin{bmatrix} x_1^* & x_2^* & \dots & x_{15}^* \end{bmatrix}$.
5. For each bootstrap sample compute the estimator $\overline{X}_b^* = \frac{\sum_{n=1}^{15} x_n^*}{15}$, the sample variance $S^2_{\bar{X}_b^*} = \frac{\sum_{n=1}^{15}(x_n^* - \overline{X}_b^*)^2}{14}$, the standard error $S_{\bar{X}_b^*} = \sqrt{\frac{S^2_{\bar{X}_b^*}}{15}}$, and the T test statistic value $T^* = \frac{\overline{X}_b^* - \overline{X}}{S_{\bar{X}_b^*}}$ where the subscript b indicates the bth bootstrap sample.
6. Rank the B T^*'s in ascending order.
7. In order to construct our confidence interval, we need to find $t_{\frac{\alpha}{2}} = t_{0.05} = T^*_{(\lfloor B*(1-\frac{\alpha}{2})\rfloor+1)} = T^*_{476}$ and $t_{1-\frac{\alpha}{2}} = t_{0.95} = T^*_{(\lfloor B*\frac{\alpha}{2}\rfloor+1)} = T^*_{26}$. Where the subscripts of T^* indicate the indices of the ranked T^* values.
8. Construct confidence interval by calculating $\left(\overline{X} - t_{0.05} * S_{\bar{X}},\ \overline{X} - t_{0.95} * S_{\bar{X}}\right)$

2.5 Implementation in Python

All code and output will be included in section 5 titled Figures and Python Code. However, it should be explained how the code works. The above algorithms for percentile method and bootstrap-t were implemented in the bootstrap function using the following steps:

1. Calculate the sample mean, sample variance, and standard error for the original data.
2. Create an array called estimates that will store the mean of each bootstrap sample.
3. Create an array called T that will store the T test statistic values.
4. Use a nested for loop to obtain the bootstrap samples. The outer for loop will run B times while the inner for loop will run 15 times since 15 is the length of our data set.
5. In the outer for loop, define an array called boot that will store the bootstrap sample.

6. In the inner for loop, generate a random integer from 0 to 14 (since Python indexes from 0). Use that random integer to pull the value from the corresponding index of the original data and append it to boot. This process is done 15 times and yields a bootstrap sample.

7. Outside of the inner for loop, calculate the mean of the bootstrap sample, the sample variance of the bootstrap sample, the standard error of the bootstrap sample, and the T test statistic value of the bootstrap sample.

8. Append the mean of the bootstrap sample to the estimates array and the T test statistic value to the T array.

9. The process ends and returns to the beginning of the outer for loop for the next iteration. Since the outer for loop runs B times, B bootstrap samples will be obtained.

10. Find the percentile method confidence interval by doing the following steps.

11. Sort the estimates array.

12. Find the lower and upper bounds of the confidence interval as detailed in section 2.3 step 7.

13. Define the variable confidence_interval and store the lower and upper bounds as a tuple.

14. Find the bootstrap-t confidence interval by doing the following steps.

15. Sort the T array.

16. Find the t values as detailed in section 2.4 step 7.

17. Construct the confidence interval as detailed in section 2.4 step 8.

18. Define the variable confidence_interval_t and store the lower and upper bounds as a tuple.

19. Finish the entire bootstrapping process by returning both confidence intervals

3 Performing Percentile Method & Bootstrap-t on Haircut Data

3.1 Results

Recall, in this study we want to construct 90% confidence intervals for the average number of dollars that a student at UNF spends on a haircut. To do this we use a process called bootstrapping. Specifically, we perform both percentile method and bootstrap-t on the data set obtained by asking UNF students how much they spent on their last haircut. We accomplish this by using 500 bootstrap samples and a significance level $\alpha = 0.1$. In Python, we pass the data set, the number of bootstrap samples B = 500, and $\alpha = 0.1$ into the bootstrap function and assign the output as CI_p and CI_t.

These two variables stand for confidence interval percentile and confidence interval bootstrap-t respectively. After our code is finished running, we print the two variables and obtain the following confidence intervals:

$$CI\_p \;=\; \Big(40.6,\; 75.86666666666666\Big)$$

$$CI\_t \;=\; \Big(38.42676993891435,\; 79.42093766874939\Big)$$

We interpret these confidence intervals to mean that we are 90% confident that the average number of dollars that a student at UNF spends on a haircut falls between 40.6 and 75.86666666666666 based on percentile method and 38.42676993891435 and 79.42093766874939 based on bootstrap-t. Both intervals contain the mean of our data $\overline{X} = 56.4$.

4 Conclusion

4.1 Final Thoughts

Since bootstrap-t is a more robust method, it was surprising to see the 90% confidence interval be wider than the more straightforward percentile method's 90% confidence interval. We have high confidence at 90% and a more narrow interval using percentile method which is ideally what we want to obtain.[1] However, upon further research it was found that for small samples, the percentile method consistently undercovers in its confidence interval.[2] Furthermore, we should actually expect the bootstrap-t confidence interval to be wider because in using this method, the coverage of the confidence interval is increased.[3] It is important to keep in mind that skewness may negatively impact the confidence interval obtained using bootstrap-t by making it too wide.[3] Plotting a histogram of our data, we observe that it does somewhat resemble a right skewed distribution since the mean is to the right of the peak.[4 & 5] It is possible that our confidence interval obtained from bootstrap-t is being impacted by this. In conclusion, although skewness may be impacting the confidence interval from bootstrap-t, we have obtained results consistent with the expected outcome from both of the methods performed.

5 Figures and Python Code

Importing the necessary libraries

In [1]:
```
# Importing the necessary libraries
import math
import numpy as np
import matplotlib.pyplot as plt
```

Defining our data set

In [2]:
```
# Creating our data set
data = [0, 65, 25, 0, 25, 35, 50, 36, 44, 170, 87, 96, 100, 65, 48]

# Since we are always using 500 as the number of bootstrap samples wanted, define a variable
B = 500
```

Defining a function that will give us the confidence intervals based on both the percentile method and bootstrap-t method

In [3]:
```
def bootstrap(data, B, alpha):
    # Calculating the mean of our original sample
    sample_mean = sum(data)/len(data)
    # Calculating the standard error of our original sample
    sample_var = sum([(values - sample_mean)**2 for values in data])/(len(data) - 1)
    sample_std_err = math.sqrt(sample_var/len(data))
    # Creating an array that will store the value of the estimator
    estimates = []
    # Creating an array that will store the T test statistics
    T = []
    # Obtaining B bootstrap samples
    for i in range(B):
        # Creating an array which will store the bootstrap sample
        boot = []
        for j in range(len(data)):
            # Define a variable that will give a random index to pull from for sampling with replacement
            index = np.random.randint(len(data))
            # Obtaining a bootstrap sample with replacement
            boot.append(data[index])
        # Appending the estimate for the current bootstrap sample to the array estimates
        estimates.append(sum(boot)/len(boot))
        # Calculating the sample variance
        var = sum([(values - estimates[i])**2 for values in boot])/(len(boot) - 1)
        # Calculating the standard error
        std_err = math.sqrt(var/len(boot))
        # Calculating the T test statistic
        T.append((estimates[i] - sample_mean)/std_err)

    # Ranking the estimates
    estimates.sort()
```

```
    # Calculating the lower and upper bound indices for the confidence interval
    lower = math.floor(B * alpha/2) + 1
    upper = math.floor(B * (1 - alpha/2)) + 1

    # Putting the confidence interval together so it can be returned as a tuple
    confidence_interval = (estimates[lower], estimates[upper])

    # Ranking the T test statistics
    T.sort()

    # Using percentile method to find indices for t values
    upper_t = math.floor(B * alpha/2) + 1
    lower_t = math.floor(B * (1 - alpha/2)) + 1

    # Defining the correct t values
    t_lower = T[lower_t]
    t_upper = T[upper_t]

    # Finding the confidence interval
    lower_bound = sample_mean - t_lower*sample_std_err
    upper_bound = sample_mean - t_upper*sample_std_err
    confidence_interval_t = (lower_bound, upper_bound)

    return confidence_interval, confidence_interval_t
```

Obtaining the confidence intervals

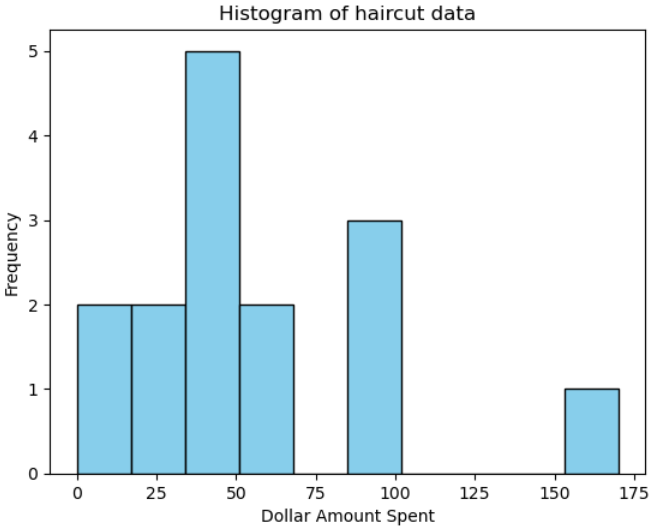
In [4]:
```
print("The mean of the data set is: " + str(sum(data)/len(data)))
```

```
The mean of the data set is: 56.4
```

In [5]:
```
# Obtaining the 90% confidence interval by the percentile method
CI_p, CI_t = bootstrap(data, B, 0.1)
print("The confidence interval obtained with the percentile method is: " + str(CI_p))
print("The confidence interval obtained with the bootstrap-t method is: " + str(CI_t))
```

```
The confidence interval obtained with the percentile method is: (40.6, 75.86666666666666)
The confidence interval obtained with the bootstrap-t method is: (38.42676993891435, 79.42093766874939)
```

In [6]:
```
plt.figure(1)
plt.hist(data, color='skyblue', edgecolor='black')
# Adding title, x label, and y label
plt.title('Histogram of haircut data')
plt.xlabel('Dollar Amount Spent')
plt.ylabel('Frequency')
plt.show()
```


Histogram of haircut data
Frequency
Dollar Amount Spent
0
1
2
3
4
5
0
25
50
75
100
125
150
175

References

1. Ismay C, Kim A, McConville K. Chapter 8 Bootstrapping and Confidence Intervals | Statistical Inference via Data Science.; 2024. Accessed March 2, 2024. https://moderndive.com/8-confidence-intervals.html#ci-width

2. Dransfield B, Brightwell B. Bootstrap confidence intervals- Principles. influentialpoints.com. Published 1999. Accessed March 2, 2024. https://influentialpoints.com/Training/bootstrap_confidence_intervals -principles-properties-assumptions.htm

3. Dransfield B, Brightwell B. How to: Calculate Bootstrap confidence intervals. influentialpoints.com. Published 1999. Accessed March 2, 2024. https://influentialpoints.com/Training/bootstrap_confidence_intervals.htm

4. Siegel A, Wagner M. Skewed Distribution - an overview | ScienceDirect Topics. www.sciencedirect.com. Published 2022. Accessed March 2, 2024. https://www.sciencedirect.com/topics/mathematics/skewed-distribution

5. Glen S. Skewed Distribution: Definition, Examples. Statistics How To. Published 2022. Accessed March 2, 2024. https://www.statisticshowto.com/probability-and-statistics/skewed-distribution/