Statistical Computing Project 2

Mario Ibanez

February 19, 2016

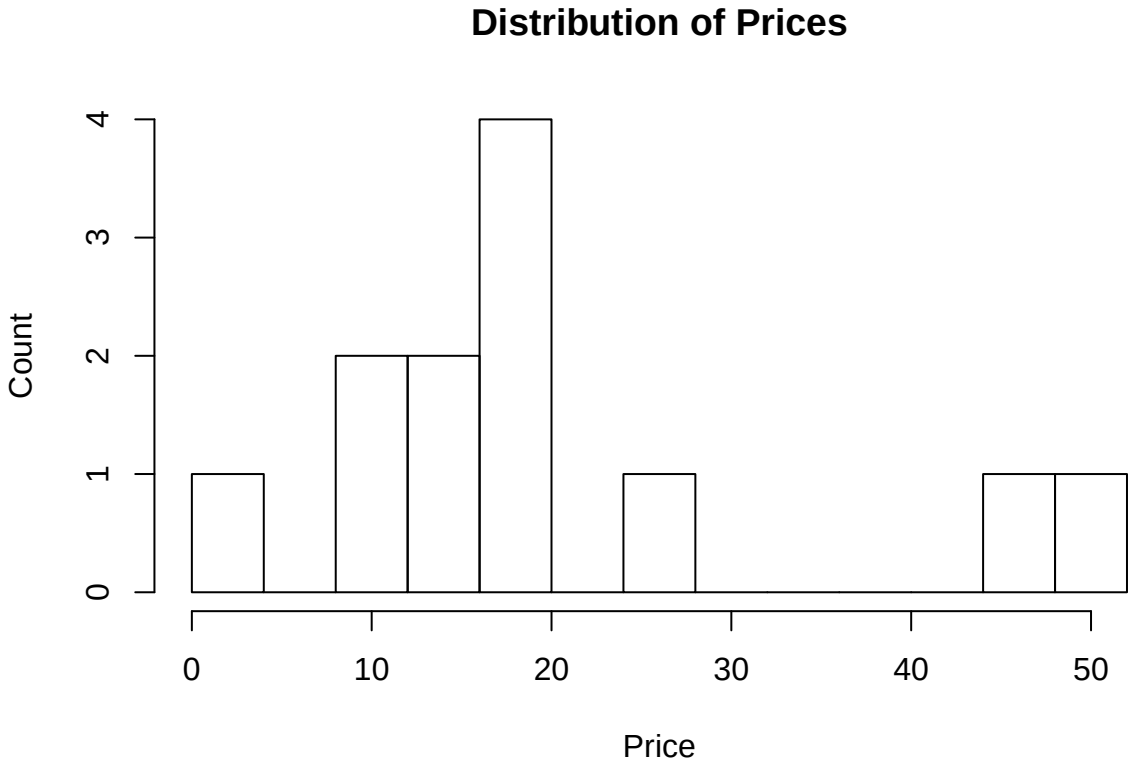
Introduction

In this report, bootstrapping is used to estimate the population mean of (recent) haircut prices for students at the University of North Florida. The percentile method is used as well as the bootstrap-t method to create two 90% confidence intervals for the population mean haircut price.

Discussion

The idea of the bootstrap method is that the sample distribution, under necessary assumptions, can be assumed to be a good approximation of the population distribution. Since it is not possible to repeatedly sample from the population in order to discover the distribution of various statistics, repeated sampling is instead done on the sample data. Ideally, the sample would be a representative sample from the population of interest. The data for this report is below:

Index	Price
1	17.00
2	16.49
3	45.00
4	25.00
5	12.00
6	18.00
7	13.00
8	20.00
9	10.00
10	0.00
11	15.00
12	50.00



Two data points are a possible cause of concern as they cause the distribution to be very right-skewed. The haircut price at index 3 is the haircut price of a student at a different university. In general, most university students are similar, if compared to other segments of the population. The price at index 12 is the recent haircut price of a faculty member at the university. Another possible issue is that the data was taken from a survey of a single class. However, aside from the two data points mentioned, it is reasonable to assume that in terms of haircut spending, this class is representative of the university population in general.

Bootstrap Algorithm

From the sample of 12 haircut prices, 10^5 bootstrap samples of size 12 are generated, with replacement. A simple way to perform this sampling would be to generate 1.2×10^6 (which is 12 times 10^5) random numbers from the discrete uniform distribution with support [1, 12] and treat these as indices for the haircut price in the data set. It would then be simply a matter of partitioning this list into 10^5 sets of 12 points. This is basically the approach used in this report used to generate the bootstrapped samples.

Below is a small piece of the bootstrapped sample data set:

1	2	3	4	5	6	7	8	9	10	11	12
16.49	20	20.00	20	15.00	20	17.00	45	20.00	13.00	10.00	13
25.00	50	25.00	15	25.00	25	45.00	45	25.00	25.00	16.49	17
45.00	0	13.00	15	0.00	17	18.00	25	25.00	13.00	45.00	0
45.00	25	50.00	0	13.00	20	25.00	20	25.00	13.00	10.00	18
45.00	0	17.00	25	10.00	13	16.49	13	18.00	0.00	45.00	15
15.00	17	25.00	17	45.00	10	25.00	13	17.00	13.00	16.49	15
17.00	0	16.49	13	12.00	17	25.00	10	50.00	18.00	16.49	13
45.00	15	12.00	25	16.49	15	16.49	15	16.49	16.49	16.49	13

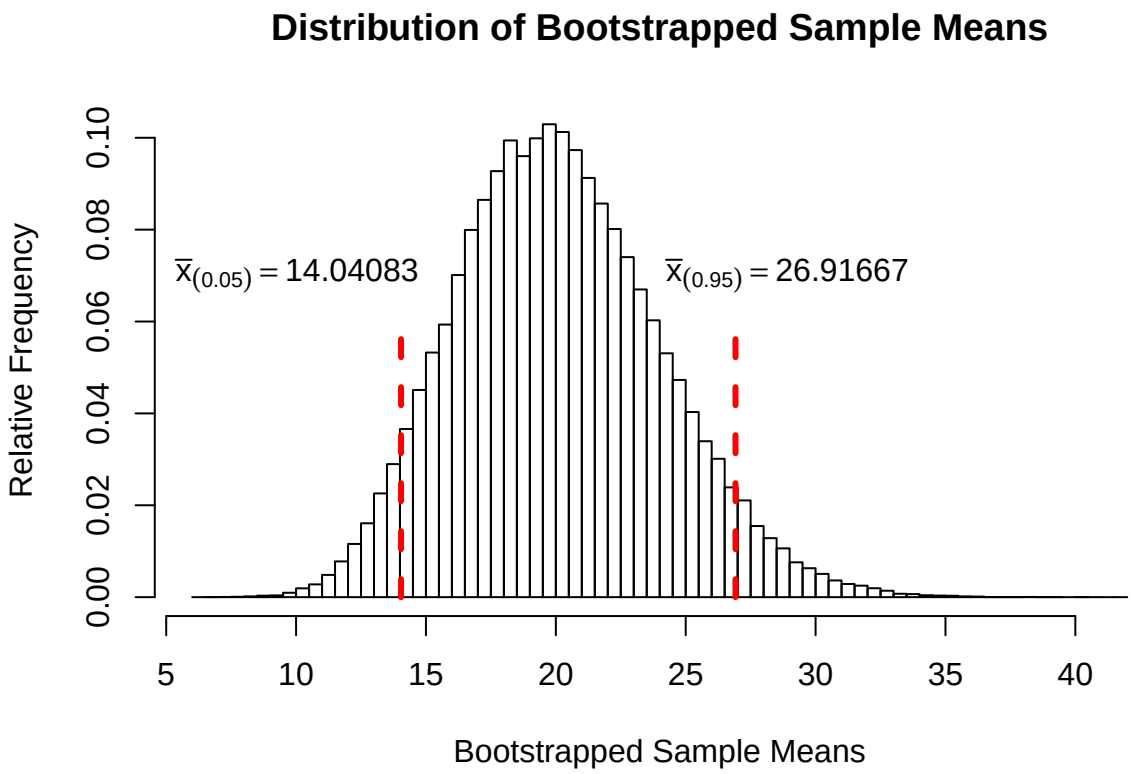
Below is the corresponding information that was found for each row of the data presented above:

Sample Mean	Standard Error	t-statistic
19.12417	8.842640	-0.3917497
28.20750	11.824333	2.3681241
18.00000	15.338603	-0.4797262
22.00000	13.993505	0.4643638
18.12417	14.400426	-0.4811110
19.04083	9.299463	-0.4035477
17.33167	11.863620	-0.8153923
18.53750	8.913389	-0.6166425

These are just small views of the data that was generated as these data sets are each 10^5 rows in length.

Percentile Method

The percentile method to create a 90% confidence interval is very simple. In the sorted list of bootstrapped sample means, the values at the 5th percentile and the 95th percentile form the left and right end points of the interval, respectively. This histogram below shows the distribution of the 10^5 bootstrapped sample means:



So, the 90% confidence interval for the population mean of recent haircut prices is:

$$[14.0408333, 26.9166667]$$

Bootstrap-t Method

The bootstrap-t method is similar to the percentile method, but the 5th and 95th percentile of the distribution of the t^* distribution are used to create the interval. Explicitly, the confidence interval is:

$$[\overline{x} - t^*_{(0.95)}\hat{se}_{\overline{x}},\ \overline{x} - t^*_{(0.05)}\hat{se}_{\overline{x}}]$$

where

$$\overline{x} = 20.12417$$

$$\overline{x}^*_i = \text{ mean of the } i^{th} \text{ bootstrapped sample}$$

$$\hat{se}_{\overline{x}} = \frac{s}{\sqrt{n}} = 4.094652$$

$$\hat{se}_{\overline{x}^*_i} = \frac{s^*_i}{\sqrt{n}} = \text{ standard error of } i^{th} \text{ bootstrapped sample}$$

$$t^* = \frac{\overline{x}^* - \overline{x}}{\hat{se}_{\overline{x}^*}}$$

Note that t^* is calculated for each bootstrapped sample.

A potential issue

One potential issue is that due to the fact that we are resampling with replacement, there is the chance that the value $\hat{se}_{\overline{x}^*}$ could be equal to zero, in the even that the same number is sampled 12 times in a bootstrap sample. However, the possibility of generating a bootstrapped sample containing the same number repeated 12 times is:

$$\frac{1}{12^{11}} = 1.3458799 \times 10^{-12}$$

Even with the number of bootstrap samples being used in this report, 10^5, the chance that this "error" does not occur is:

$$\left(1 - \frac{1}{12^{11}}\right)^{10^5} = 0.9999999$$

In other words, it is not an issue that needs to be dealt with.

Number of unique bootstrapped samples

Given n unique items, the number of ways that k items can be chosen from them, allowing for repetition and disregarding order is:

$$\binom{n+k-1}{n-1} = \binom{n+k-1}{k}$$

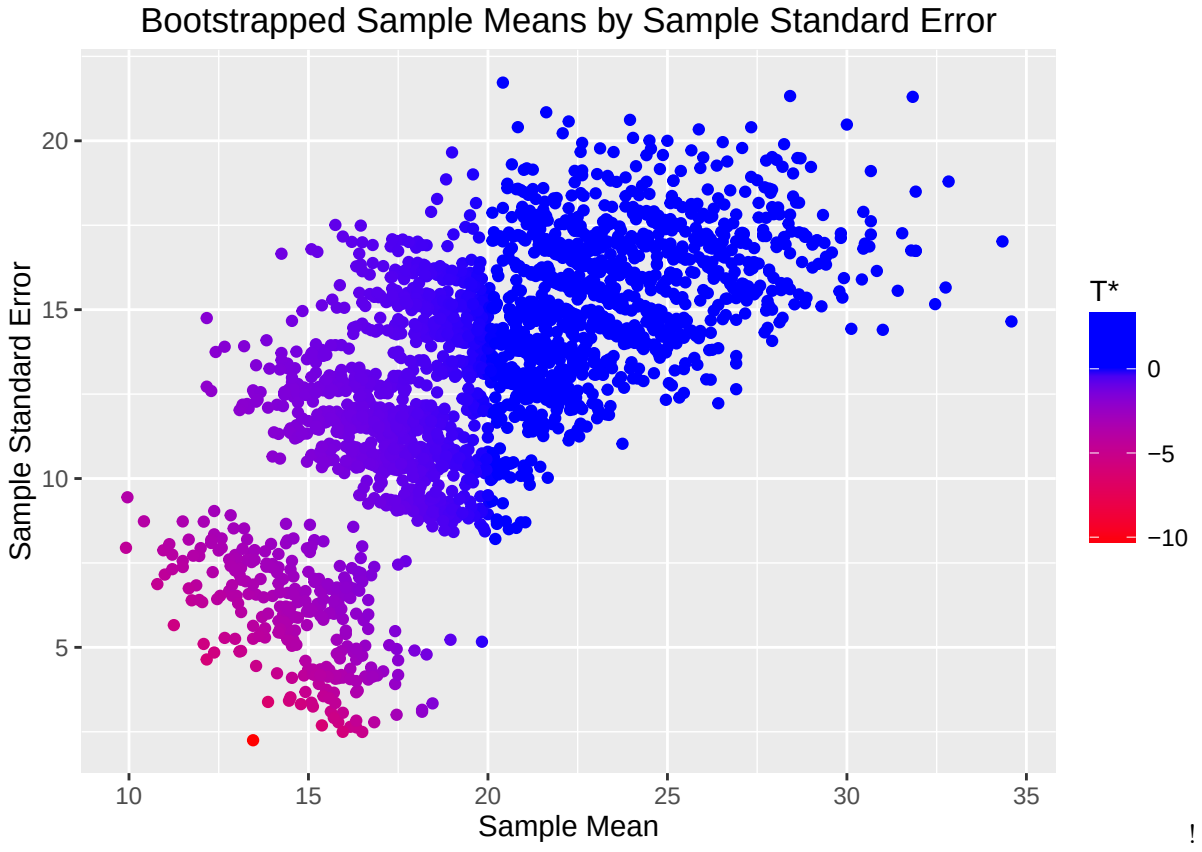
With $n = k = 12$, the number of unique bootstrapped samples is:

$$\binom{23}{12} = 1.352078 \times 10^6$$

It would be fairly easy to attempt and create each possible bootstrapped sample, but it does not serve an important purpose to the question of creating a confidence interval for the population mean.

Distribution of the sampling statisic

Before calculating the confidence interval, there are a few interesting things to look at. One way to visualize the bootstrapped data is the scatter plot below, where the bootstrapped sample standard errors plotted against their sample means, and the points are colored by their t^* statistic:



!

Thus, the 90% confidence interval is:

$$\begin{aligned}[\overline{x} - t^*_{(0.95)}\hat{se}_{\overline{x}},\ \overline{x} - t^*_{(0.05)}\hat{se}_{\overline{x}}] &= [20.124 - (1.446)(4.094652),\ 20.124 - (-3.227)(4.094652)] \\ &= [14.203, 33.337]\end{aligned}$$

Conclusion

The percentile method gave a 90% confidence interval of [14.041, 26.917] and the bootstrap-t method gave a confidence interval of [14.203, 33.337]. Both methods gave approximately the same left endpoint, but the two methods gave very different right endpoints. This was partly due to the fact that the distribution of t^* was skewed left, which created a larger right endpoint in the interval.

Appendix (Code)

```
# Set seed
set.seed(1234)

# Load packages and data
library(ggplot2)
library(knitr)
haircuts <- c(17, 16.49, 45,
              25, 12, 18,
              13, 20, 10,
              0, 15, 50)

# Print data
kable(data.frame(Index = 1:12,
                 Price = haircuts))

# Function to calculate statistics
statistic_generator <- function(x, theta_hat){
  mean_x <- mean(x)
  se_x <- sd(x)/sqrt(length(x))
  return(c(mean_x,
           sd(x),
           (mean_x - theta_hat)/se_x))
}

# Original sample's mean and standard error
x_bar <- mean(haircuts)
s_x_bar <- sd(haircuts)/sqrt(length(haircuts))

# Number of bootstrapped samples
B <- 10^5
```

```r
# Print histogram of original sample
hist(haircuts, breaks = seq.int(from = 0, to = 52, by = 4),
     main = "Distribution of Prices",
     xlab = "Price",
     ylab = "Count")


# Generate bootstrapped samples
boot_strapped_haircuts <- matrix(data = sample(x = haircuts, size = 12*B, replace = TRUE),
                                 ncol = 12, byrow = TRUE)

# Calculate mean, standard error, and t for each
boot_statistics <- matrix(apply(X = boot_strapped_haircuts, MARGIN = 1,
                                FUN = statistic_generator, theta_hat = x_bar),
                          ncol = 3, byrow = TRUE)

# Assign to variable names
sample_means <- boot_statistics[, 1]
sample_stan_errors <- boot_statistics[, 2]
t_star <- boot_statistics[, 3]
sorted_means <- sort(sample_means)
sorted_t_star <- sort(t_star)

# Print two tables
kable(data.frame(boot_strapped_haircuts[1:8, ]), col.names = sapply(1:12, paste))
kable(data.frame(boot_statistics[1:8, ]),
      col.names = c("Sample Mean", "Standard Error", "t-statistic"))

# Print histogram
hist(sample_means, breaks = seq.int(floor(min(sample_means)-1),
                                    ceiling(max(sample_means+1)),
                                    by = .5),
     main = "Distribution of Bootstrapped Sample Means",
     ylab = "Relative Frequency",
     xlab = "Bootstrapped Sample Means",
     freq = FALSE)
segments(x0 = c(sorted_means[0.05*B], sorted_means[0.95*B]),
         x1 = c(sorted_means[0.05*B], sorted_means[0.95*B]),
         y0 = c(0, 0),
         y1 = c(0.06, 0.06),
         col = "red",
         lwd = 3,
         lty = 2)
text(sorted_means[0.95*B]+2, 0.07, labels = expression(bar(x) [(0.95)] == 26.91667))
text(sorted_means[0.05*B]-4, 0.07, labels = expression(bar(x) [(0.05)] == 14.04083))
```

```r
# Print scatter plot
sequence <- seq.int(from = 1, to = B, by = 40)
ggplot()+
  geom_point(aes(x = sample_means[sequence], y = sample_stan_errors[sequence],
                 colour = t_star[sequence]))+
  scale_colour_gradient2(low = "red", mid = "blue", high = "blue")+
  ggtitle("Bootstrapped Sample Means by Sample Standard Error")+
  labs(x="Sample Mean", y="Sample Standard Error", size = 20, colour = expression("T*"))+
  theme_grey(base_size = 11)

# Print histogram
left <- round(sorted_t_star[1]-1)
right <- round(sorted_t_star[B]+1)
hist(t_star, breaks = seq.int(left*101, right*101, by = 15)/100,
     freq = FALSE,
     main = "Distribution of t*",
     xlab = "t*",
     xlim = c(-6, 4))
segments(x0 = c(sorted_t_star[0.05*B], sorted_t_star[0.95*B]),
         y0 = c(0, 0),
         x1 = c(sorted_t_star[0.05*B], sorted_t_star[0.95*B]),
         y1 = c(0.2, 0.2),
         col = "red",
         lwd = 3,
         lty = 2)
text(sorted_t_star[0.95*B]+1, 0.22, labels = expression(t [(0.95)] == 1.446))
text(sorted_t_star[0.05*B]-0.5, 0.22, labels = expression(t [(0.05)] == -3.227))
```