Code ▾

Foundations for Inference

4.1 Key Concepts

- **Statistical inference** deals with the quality of parameter estimates; that is, we use *sampled* data to estimate *population* parameters and wish to know how good these estimates are.
- An estimate that is a single value is called a **point estimate** (e.g., the sample mean, $\bar{x}$, is a point estimate for the population mean, μ).
- **Sampling distribution:** consider taking repeated samples of the same size and constructing point estimates from these samples. The estimates will likely vary from one sample to another, creating a sampling distribution.
    - We think of any given point estimate as being drawn from this sampling distribution.
    - For our purposes, we'll assume that the sampling distribution is centered at the true parameter value. A point estimate whose sampling distribution satistifes this assumption is called **unbiased**.
    - The standard deviation of the sampling distribution is called the **standard error**. It quantifies the typical uncertainty associated with a point estimate.
    - When the point estimate is the sample mean, we have a formula for the standard error. Suppose that a sample of n independent observations is taken from a population with standard deviation σ. Then the standard error is given by

      $$SE = \frac{\sigma}{\sqrt{n}}$$

      In practice, we don't have access to σ, so we use the standard deviation of our sample, s, instead. Our estimate of SE is therefore $s/\sqrt{n}$.
- **Confidence intervals:** suppose your point estimate is unbiased and comes from an approximately normal sampling distribution. Then the confidence interval is given by

  $$\text{point estimate} \pm z^* SE,$$

  where z^* corresponds to the confidence level and SE denotes the standard error. We call $z^* SE$ the **margin of error**.
    - ***Interpretation:*** consider taking repeated samples of the same size. For each sample, get a point estimate and construct a (say) 95% confidence interval using the above formula. Then approximately 95% of those intervals will include the true parameter.
    - ***Language:*** in this case, we would say "We are 95% confident that the population parameter is between [*insert confidence interval here*]"
- **Central Limit Theorem:** if your sample is sufficiently large and the data are not heavily skewed, then the sampling distribution of $\bar{x}$ is approximately normal.
    - The sample mean also happens to be unbiased. Therefore, we can construct confidence intervals for $\bar{x}$ using the formula above.
- **Hypothesis testing** is concerned with assessing competing claims. Usually the **null hypothesis** (H_0) represents the skeptical perspective, while the **alternative hypothesis** (H_A) represents the new perspective.

  - A hypothesis test can be **one-sided** or **two-sided.** For instance, a one-sided test might have hypotheses $H_0 : \mu = 10$ and $H_A : \mu > 10$, while a two-sided test might have hypotheses $H_0 : \mu = 10$ and $H_A : \mu \neq 10.$
  - Based on the results of our test, we will either *reject the null hypothesis* or *fail to reject the null hypothesis*.
  - Of course we may make an error in deciding between H_0 and H_A. These errors are classified as either **Type 1** (rejecting H_0 when it is actually true) or **Type 2** (failing to reject H_0 when H_A is actually true).
  - The **significance level**, α, of a hypothesis test is the probability of making a Type 1 error. Typically we choose α to be 0.05 or 0.01 (e.g., if we set $\alpha = 0.05$, then we could use a 95% confidence interval for our hypothesis test).
- **Using p-values for hypothesis testing:** the p-value is the probability of obtaining a summary statistic that is as or more extreme than the summary statistic we obtained from our current data set, given that the null hypothesis is true. This summary statistic is called a **test statistic**.
  - The smaller the p-value, the more evidence we have to reject the null hypothesis.
  - We reject the null hypothesis whenever the p-value is less than or equal to our chosen significance level, α.
  - For a one-sided hypothesis test, the area where we reject H_0 is in *one* of the tails. For a two-sided hypothesis test, the area where we reject H_0 is spread over *both* tails.

4.2 More on Bootstrap

Let's return to our example from chapter 3 – remember Penelope the cow? Recall that we had 17,184 NPR listeners' guesses of how much Penelope weighed. These responses formed our population. We were interested in estimating μ, the average of the 17,184 guesses. So we took a sample of size $n = 1000$ and repeatedly sampled from the sample (i.e., performed bootstrap) to build a distribution of sample means. Let's copy our code from sections 3.2 and 3.3: read in the data, calculate μ, take an "original sample", and perform bootstrap.

Hide

```
# Read in guesses of the cow's weight
cow.data <- read.csv(file = "cow_data.csv", header = TRUE)
# Calculate population mean
mu <- mean(cow.data$Guess)
```

Hide

```
# Get original sample
set.seed(5) # for reproducibility
orig.sample <- sample(cow.data$Guess, size = 1000)
```

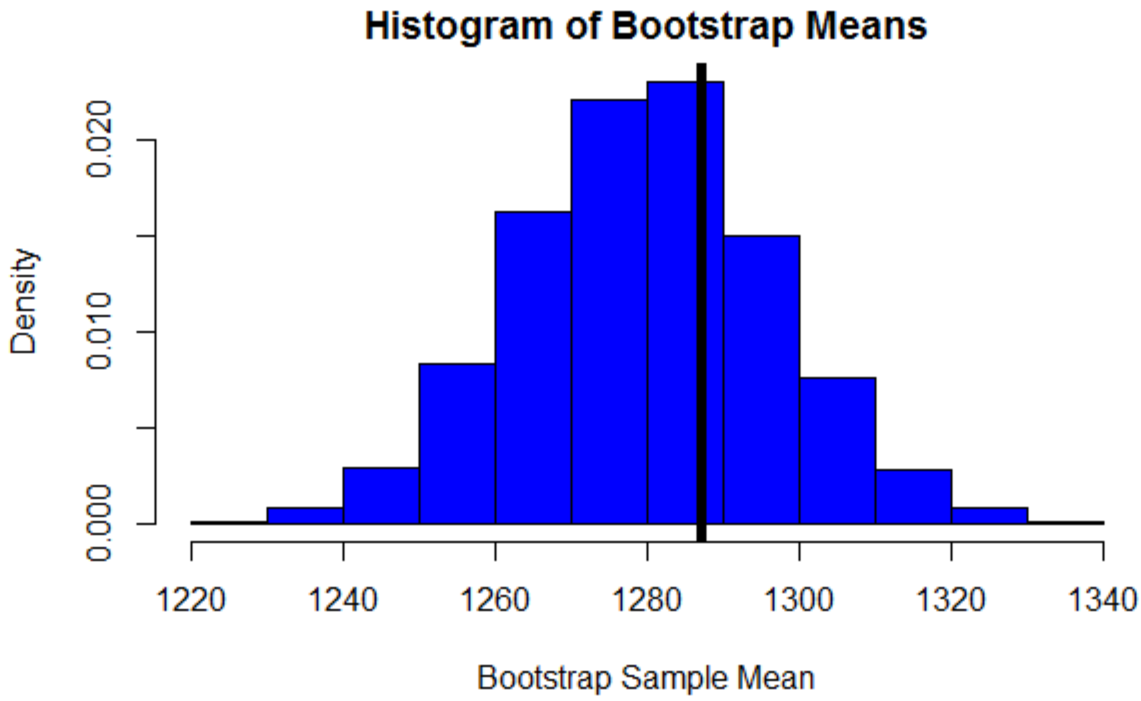
Hide

```
# Function to get bootstrap means
# Input: original.sample = the original sample
#        k = number of replications
# Output: vector of sample means
get.bootstrap.means <- function(original.sample, k) {

  # Get size of original sample
  n <- length(original.sample)

  # Create vector to store sample means
  sample.means <- c()

  # Repeat k times
  for(i in 1:k) {
    # Get bootstrap sample
    boot.sample <- sample(original.sample, size = n, replace = TRUE)
    # Calculate and store sample mean
    sample.means[i] <- mean(boot.sample)
  }

  return(sample.means)
}
```

We'll call the function `get.bootstrap.means` to get 5,000 bootstrap sample means. Then we'll plot a histogram of these means and mark the location of the true parameter value: $\mu = 1,287$ lbs.

Hide

```
# Set seed for reproducibility
set.seed(5)
# Get 5,000 sample means
boot.means <- get.bootstrap.means(orig.sample, 5000)
# Plot distribution of the bootstrap means, along with the true parameter value
hist(boot.means, freq = FALSE, xlab = "Bootstrap Sample Mean", ylab = "Density",
main = "Histogram of Bootstrap Means", col = "blue")
abline(v = mu, lwd = 5)
```



We see that μ – marked by a vertical line – is contained within the histogram. In fact, μ is close to the center of the distribution rather than towards the tails.

Let's explore this concept of "sitting squarely within the histogram" a little further. We'll calculate the middle 95% of the bootstrap distribution and mark this interval with a horizontal red line.

Hide

```
# The middle 95% of the distribution is given by (left.end, right.end), where left.end is the value below which 2.5% of the data falls, and right.end is the value below which 97.5% of the data falls.
left.end <- quantile(boot.means, probs = 0.025)
right.end <- quantile(boot.means, probs = 0.975)
print( paste( "The middle 95% of the bootstrap means is given by the interval (", left.end, ",", right.end, ")", sep = ""  ) )
```

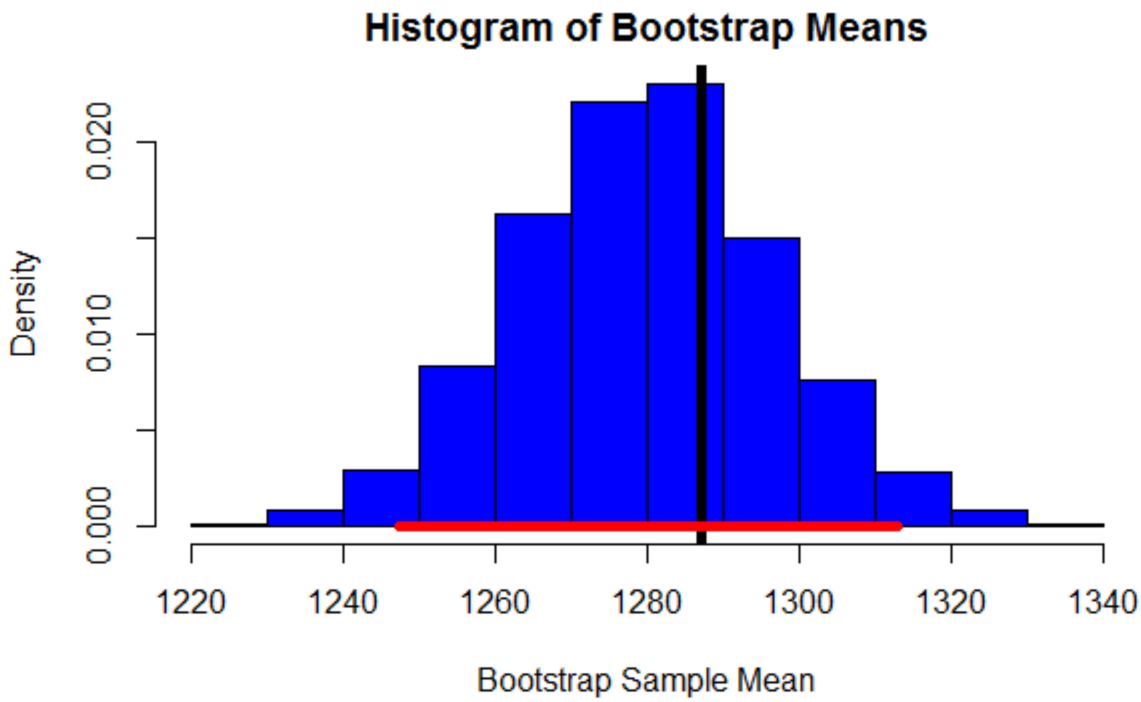
```
[1] "The middle 95% of the bootstrap means is given by the interval (1247.31581875,1312.831703)"
```

Hide

```
# Plot distribution of the bootstrap means, along with the true parameter value
hist(boot.means, freq = FALSE, xlab = "Bootstrap Sample Mean", ylab = "Density", main = "Histogram of Bootstrap Means", col = "blue")
abline(v = mean(cow.data$Guess), lwd = 5)
```

Hide

```
# Mark the middle 95% of the distribution with a horizontal red line
segments( x0 = left.end, y0 = 0, x1 = right.end, y1 = 0, col = "red", lwd = 5 )
```



We see that the middle 95% of bootstrap means lie in the interval (1247 lbs, 1312 lbs) and our true parameter value, $\mu = 1,287$ lbs, lies neatly in this interval.

Will this always be the case? That is, will μ always sit in the middle 95% of the bootstrap distribution? Let's repeat these steps to find out:

1. Draw an "original" sample of size $n = 1000$.
2. Using this original sample, draw $k = 5,000$ bootstrap samples and compute the sample mean of each. This will yield a distribution of 5,000 bootstrap means. Find the middle 95% of this distribution.
3. Repeat steps 1 and 2 a total of 100 times. The result will be 100 intervals. We're going to calculate how many of these intervals contain μ.

Hide

```
# WARNING: the following code is slow; don't be alarmed if it takes a little whil
e to run
# We will get 100 intervals
num.repeats <- 100
# Create vectors to store the left and right endpoints of the intervals
left.ends <- c()
right.ends <- c()
# Set seed for reproducibility
set.seed(7)
# Repeat 100 times...
for(i in 1:num.repeats) {

  # Get original sample
  orig.sample <- sample( cow.data$Guess, size = 1000 )

  # Get 5,000 bootstrap means, yielding a distribution
  boot.means <- get.bootstrap.means( orig.sample, k = 5000 )

  # Find middle 95% of the distribution and store endpoints
  left.ends[i] <- quantile( boot.means, probs = 0.025 )
  right.ends[i] <- quantile( boot.means, probs = 0.975 )

}
```

Let's display the first 5 intervals to see if they seem reasonable:

Hide

```
# Print the middle 95% of the first five bootstrap distributions
for(i in 1:5) {

  print( paste( "Interval number ", i, " is (", left.ends[i], ", ", right.ends
[i], ")", sep = "") )

}
```

```
[1] "Interval number 1 is (1262.263661, 1349.836954875)"
[1] "Interval number 2 is (1245.04824596765, 1311.61730375)"
[1] "Interval number 3 is (1261.392939785, 1336.640199)"
[1] "Interval number 4 is (1279.18397022212, 1374.850590135)"
[1] "Interval number 5 is (1262.7644115, 1354.89714625)"
```

Notice that all five contain μ. Now we'll calculate for how many of the 100 intervals this is the case.

Hide

```
# Create data frame with two columns, Left and Right, containing the left and right endpoints, respectively, of each interval
intervals <- data.frame( Left = left.ends, Right = right.ends )
# Calculate how many of the 100 intervals contain the true parameter value
good.indices <- which( intervals$Left < mu & intervals$Right > mu )
print( paste( length(good.indices), "of the 100 intervals contain the true parameter value" ) )
```

```
[1] "95 of the 100 intervals contain the true parameter value"
```

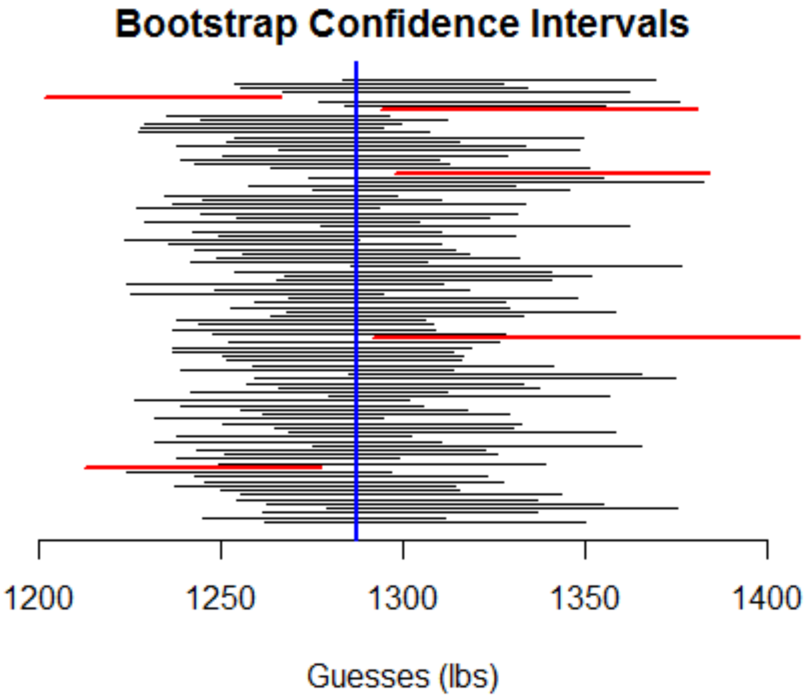
We see that 95 of the intervals contain μ and 5 do not. Below we've plotted each of these intervals, coloring the good ones black and the bad ones red. The true parameter value is shown in blue.

Hide

```
# Create blank plot
plot(0, type = "n", xlim = c(1175, 1425), ylim = c(1,100), yaxt = "n", xlab = "Guesses (lbs)", ylab = NA, main = "Bootstrap Confidence Intervals", bty = "n")
# Add line segment depicting each of the 100 intervals.  Color the "correct" intervals black and the "incorrect" intervals red.
segments( x0 = intervals$Left[good.indices], y0 = good.indices, x1 = intervals$Right[good.indices], y1 = good.indices, col = "black" )
```

Hide

```
bad.indices <- setdiff(1:100, good.indices)
segments( x0 = intervals$Left[bad.indices], y0 = bad.indices, x1 = intervals$Right[bad.indices], y1 = bad.indices, col = "red", lwd = 2 )
# Draw vertical line at the true parameter value
abline( v = mu, col = "blue", lwd = 2 )
```



If we were to repeat this process (notice that you'll have to change the seed), we might not always get 95 good intervals. Sometimes we'll get 94 or 96 or 97, but the number of μ-containing intervals will always be close to 95. What we've shown is that *when we repeatedly compute the middle 95% of a bootstrap distribution, these intervals contain our true parameter value about 95% of the time*.

If we were to use the middle 80% of the bootstrap distribution instead, we would find that ~80% of our intervals contain μ. The same goes for any other percentage we choose (except 100%).

Does this remind you of anything?

Hopefully it does. We have shown that you can compose a 95% confidence interval for μ by taking the middle 95% of a distribution of bootstrap means.

4.3 Confidence Intervals in Practice: Thanksgiving Spending

Each year the Thanksgiving weekend is one of the busiest shopping weekends of the year. Suppose we're interested in the average amount that Americans spend during this period – call it μ. We'll estimate μ using a sample that was taken in 2009, in which 436 randomly-selected Americans were asked how much money they spent in the week following Thanksgiving. We're going to use this data to practice two ways of constructing confidence intervals for μ.

4.3.1 Analytical Confidence Intervals

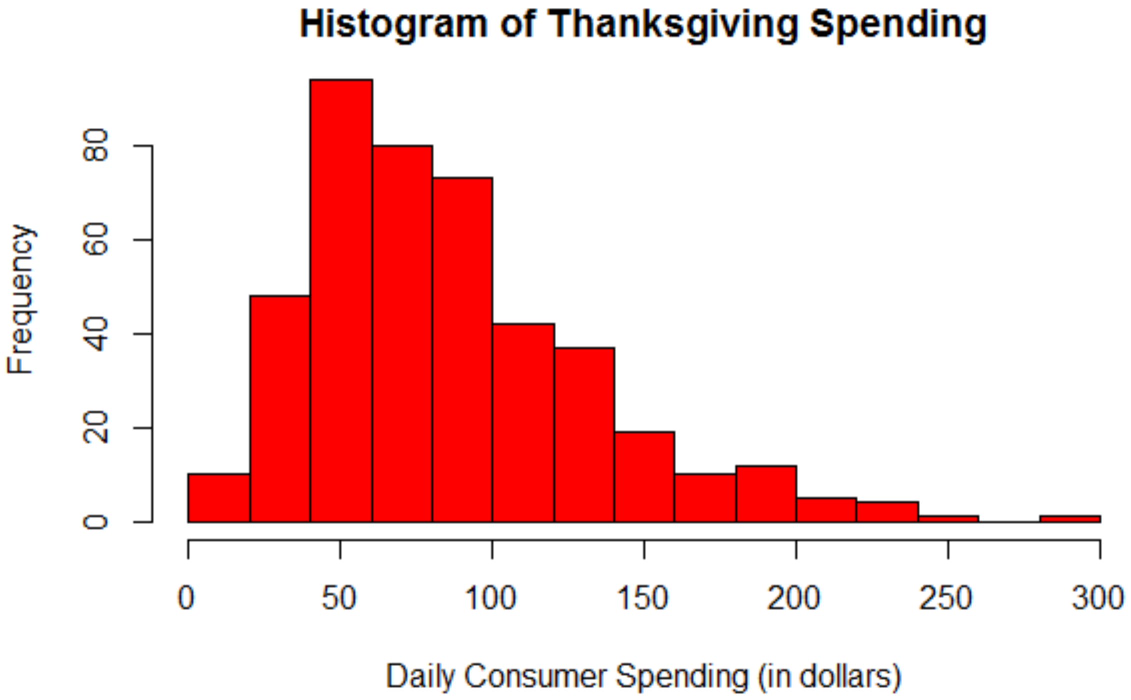
First let's build a confidence interval using the method discussed in the course textbook. We'll load the data and make a histogram of Thanksgiving expenditures.

Hide

```
# Load the data, which is included in an R library that comes with your textbook
library(openintro)
data(tgSpending)
```

Hide

```
# Plot histogram of consumer spending
hist(tgSpending$spending, xlab = "Daily Consumer Spending (in dollars)", main = "
Histogram of Thanksgiving Spending", col = "red")
```



From the histogram, we see that most Americans spent between \$50 and \$100 per day, although some spent much more (up to \$300). The distribution is right-skewed, but this isn't a problem since our sample size is quite large ($n = 436$). Given that the skew condition is satisfied, and that we have a large sample of presumably independent observations, the sample mean should be approximately normally-distributed. This sample mean is $\bar{x} = \$84.71$:

Hide

```
# Compute sample mean
x.bar <- mean(tgSpending$spending)
x.bar
```

```
[1] 84.70677
```

The same conditions ensure that we can estimate the standard error ($\sigma/\sqrt{n}$) by replacing the population standard deviation, σ, with the sample standard deviation, s. The standard error in this case is $SE = \$2.25$:

Hide

```
# Get number of individuals in sample
n <- length(tgSpending$spending)
# Get sample standard deviation
s <- sd(tgSpending$spending)
# Compute (estimate of) standard error
SE <- s/sqrt(n)
SE
```

```
[1] 2.247468
```

Now that we have the sample mean and the standard error, let’s construct a 95% confidence interval for the average consumer spending. Recall that this confidence interval is given by

$$\bar{x} \pm 1.96 \times SE,$$

which in this case turns out to be $(\$80.31, \$89.11)$:

Hide

```
# Compute lower end of confidence interval
lower <- x.bar - 1.96*SE
# Compute upper end of confidence interval
upper <- x.bar + 1.96*SE
# Print interval
print( paste( "The 95% confidence interval is (", lower, ", ", upper, ")", sep =
"" ) )
```

```
[1] "The 95% confidence interval is (80.3017276066506, 89.1118026611223)"
```

We are 95% confident that the average holiday spending among all Americans was between $80.31 and $89.11.

4.3.2 Bootstrap Confidence Intervals

Notice that in the above example, we assume that $\bar{x}$ is approximately normally distributed. We justified this assumption by checking several conditions, namely that the sample is large, the observations are independent, and the population distribution isn’t strongly skewed. But sometimes these conditions might not hold. For instance, the histogram for the sample of Thanksgiving spending is right-skewed. We disregarded this fact because our sample size is sufficiently large, but what if it weren’t? We would like to have some way of forming confidence intervals that *doesn’t depend on our making assumptions about the population distribution*.

Can you think of a solution?

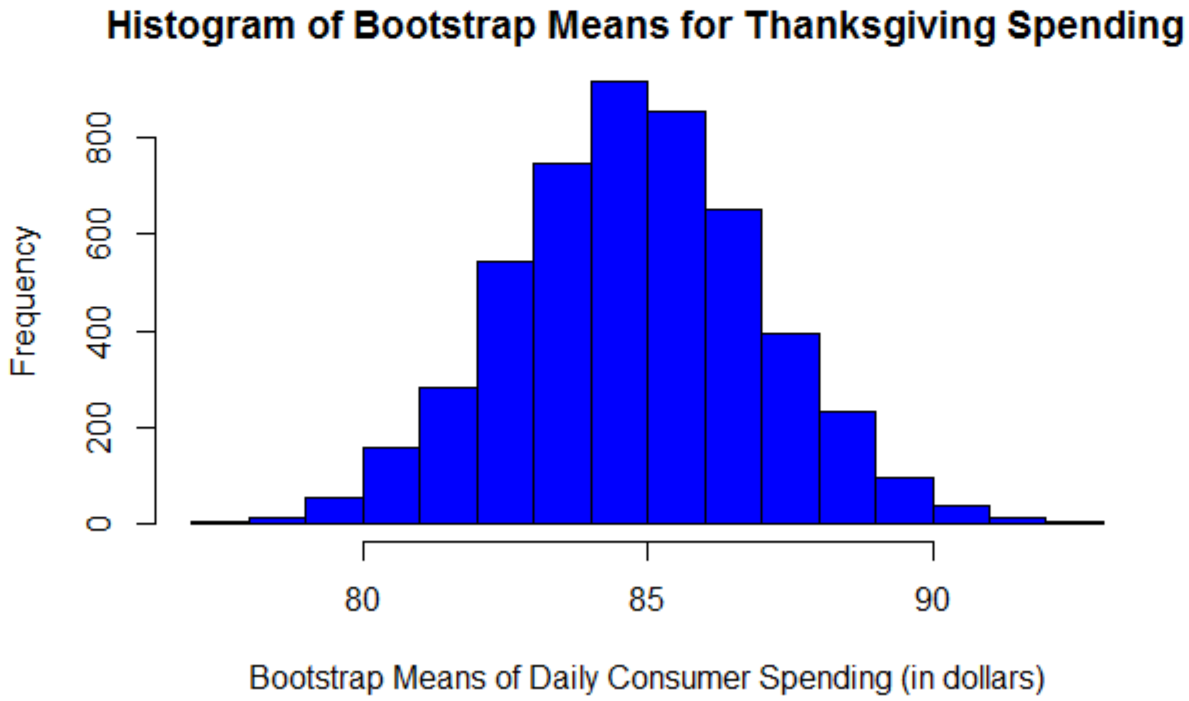
Remember that in our cow guessing example (see Section 4.2), we constructed a 95% confidence interval for the population mean by taking the middle 95% of the distribution of bootstrap sample means.

And we didn't have any assumptions on the data! It turns out that bootstrap is a powerful tool for constructing confidence intervals when we're unable or unwilling to make assumptions about our population.

Let's form a 95% boostrap confidence interval for average Thanksgiving spending.

Hide

```
set.seed(5)
# Get 5,000 bootstrap means using our original sample of 436 Americans' expenditu
res
boot.means <- get.bootstrap.means(tgSpending$spending, 5000)
# Plot histogram of the sample means
hist( boot.means, xlab = "Bootstrap Means of Daily Consumer Spending (in dollar
s)", main = "Histogram of Bootstrap Means for Thanksgiving Spending", col = "blu
e"  )
```



We'll calculate the left and right endpoints of the middle 95% of this distribution:

Hide

```
left.end <- quantile(boot.means, probs = 0.025)
right.end <- quantile(boot.means, probs = 0.975)
print( paste( "A 95% confidence interval is (",  left.end, ", ", right.end, ")",
sep = "" ) )
```

```
[1] "A 95% confidence interval is (80.4078826936107, 89.1317148834295)"
```

We are 95% confident that average Thanksgiving spending among all Americans is between $80.41 and

$89.13. Notice how close this is to the confidence interval we calculated using the book's method: ($80.31, $89.11). The advantage here was that we didn't need to make assumptions about the underlying population (e.g., regarding skewness). That being said, we can't throw caution to the wind. Bootstrap confidence intervals are only successful when we have a large original sample (n large) and can perform many bootstrap replications (k large). Furthermore, we don't want to use bootstrap to estimate quantities like the minimum or maximum value in the population. These quantities depend on rare elements which we're unlikely to obtain in our original sample, let alone in bootstrap samples.

Sources

Sections 4.2-4.3 were adapted from Sections 13.2-13.3 of *Computational and Inferential Thinking: The Foundations of Data Science* (https://www.inferentialthinking.com/chapters/13/2/Bootstrap).