Math 300 NTI Lesson 25

Constructing Confidence Intervals

Professor Bradley Warner

July, 2022

Contents

Objectives

1. Construct bootstrap percentile and standard error confidence intervals for a single mean or median using the `infer` package.

Reading

Chapter 8.4

Lesson

Remember that you will be running this more like a lab than a lecture. You want them using `R` and answering questions. Have them open the notes rmd and work through it together.

Work through the learning check LC 8.5.

- We will be using the `infer` package to create the bootstrap confidence intervals. Last lesson we used `rep_sample_n()`. The `infer` package will be used for inference in the rest of the course.
- The `infer` package gives a framework to think about and conduct inference. It makes hypothesis testing and confidence interval construction more structured and puts computational resources at the center versus mathematical tools.
- The `infer` package uses the verbs `specify()`, `generate()`, `calculate()`, and `visualize()` to complete the construction process.

Libraries

```
library(tidyverse)
library(moderndive)
library(infer)
```

Review

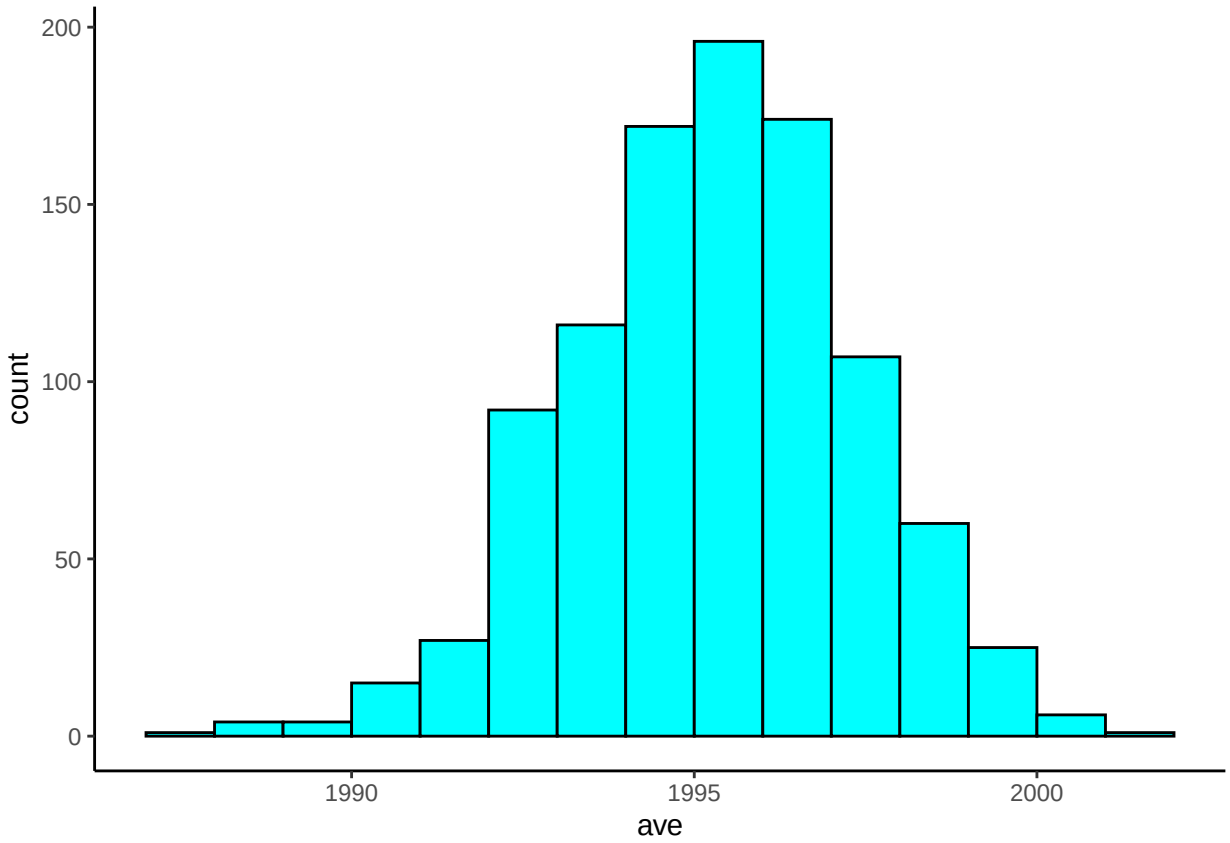
As a review, let's construct the bootstrap distribution of the sample mean for our pennies data. This requires use to use our skills on wrangling data frames.

```
head(pennies_sample)
```

```
## # A tibble: 6 x 2
##      ID  year
##   <int> <dbl>
## 1     1  2002
## 2     2  1986
## 3     3  2017
## 4     4  1988
## 5     5  2008
## 6     6  1983
```

```
set.seed(52249)
bootstrap_dist<-pennies_sample %>%
  rep_sample_n(size = 50, replace = TRUE, reps = 1000) %>%
  group_by(replicate) %>%
  summarize(ave=mean(year))
```

```
bootstrap_dist %>%
  ggplot(aes(x=ave)) +
  geom_histogram(binwidth = 1, color = "black", boundary = 1990, fill = "cyan") +
  theme_classic()
```



Now let's obtain a 95% confidence interval using the percentile method.

```
bootstrap_dist %>%
  summarize(center=mean(ave),
            lower=quantile(ave,probs=0.025),
            upper=quantile(ave,probs=.975))
```

```
## # A tibble: 1 x 3
##   center lower upper
##    <dbl> <dbl> <dbl>
## 1  1995. 1991. 1999.
```

Now use the standard error method.

```
bootstrap_dist %>%
  summarize(center=mean(ave),
            lower=center-sd(ave)*qnorm(.975),
            upper=center+sd(ave)*qnorm(.975))
```

```
## # A tibble: 1 x 3
##   center lower upper
##    <dbl> <dbl> <dbl>
## 1  1995. 1991. 1999.
```

Steps from the `infer` package

- First specify the response variable, and explanatory variables if present. We like to use the `formula` option.

```
pennies_sample %>%
  specify(formula=year~NULL)
```

```
## Response: year (numeric)
## # A tibble: 50 x 1
##     year
##    <dbl>
##  1  2002
##  2  1986
##  3  2017
##  4  1988
##  5  2008
##  6  1983
##  7  2008
##  8  1996
##  9  2004
## 10  2000
## # ... with 40 more rows
```

It is similar to using `select()` but note the meta data has also changed.

- Generate replicates

```
pennies_sample %>%
  specify(formula=year~NULL) %>%
  generate(reps = 1000, type = "bootstrap")
```

```
## Response: year (numeric)
## # A tibble: 50,000 x 2
## # Groups:   replicate [1,000]
##    replicate  year
##        <int> <dbl>
##  1         1  2000
##  2         1  1996
##  3         1  1986
##  4         1  1985
##  5         1  2006
##  6         1  1986
##  7         1  1990
##  8         1  1982
##  9         1  1978
## 10         1  2004
## # ... with 49,990 more rows
```

Compare this code with that using `rep_sample_n()`.

- Find the sample statistic for each replicate.

```
pennies_sample %>%
  specify(formula=year~NULL) %>%
  generate(reps = 1000, type = "bootstrap") %>%
  calculate(stat = "mean")
```

```
## Response: year (numeric)
## # A tibble: 1,000 x 2
##    replicate  stat
##        <int> <dbl>
##  1         1 1999.
##  2         2 1995.
##  3         3 1996.
##  4         4 1994.
##  5         5 1993.
##  6         6 1998.
##  7         7 1994.
##  8         8 1996.
##  9         9 1996.
## 10        10 1995.
## # ... with 990 more rows
```

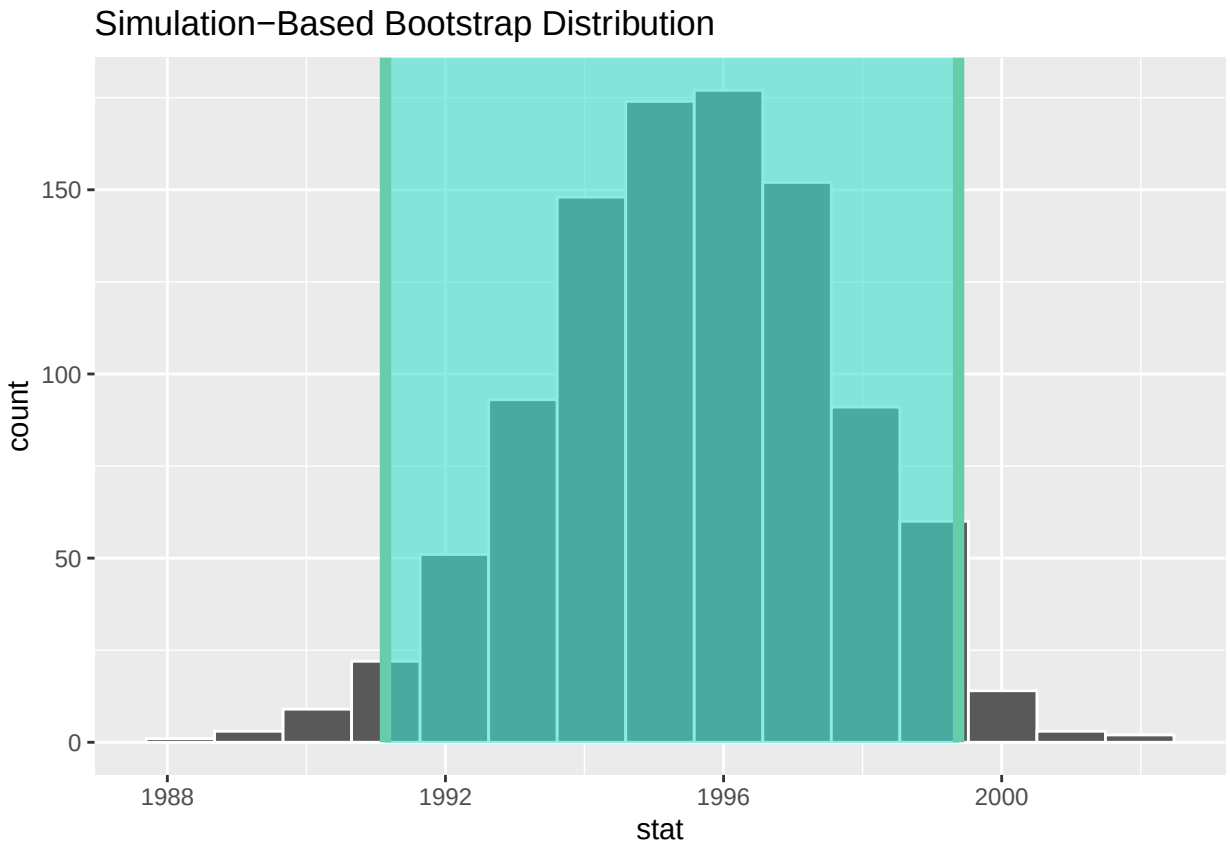
- Using results of bootstrap distribution

```
boot_dist_mean <- pennies_sample %>%
  specify(formula=year~NULL) %>%
  generate(reps = 1000, type = "bootstrap") %>%
  calculate(stat = "mean")
```

```
(percentile_ci <- boot_dist_mean %>%
  get_confidence_interval(level = 0.95, type = "percentile"))
```

```
## # A tibble: 1 x 2
##   lower_ci upper_ci
##      <dbl>    <dbl>
## 1    1991.    1999.
```

```
# Visualize the results
visualize(boot_dist_mean) +
  shade_confidence_interval(endpoints = percentile_ci)
```



Or if we want the standard error method.

```
(mean_pennies <- pennies_sample %>%
  summarize(ave=mean(year)) %>%
  pull())
```

```
## [1] 1995.44
```

```
(standard_error_ci <- boot_dist_mean %>%
  get_confidence_interval(type = "se", point_estimate = mean_pennies))
```

```
## Using 'level = 0.95' to compute confidence interval.
```

```
## # A tibble: 1 x 2
##   lower_ci upper_ci
##      <dbl>    <dbl>
## 1    1991.    2000.
```

LC 8.5 (Objective 1)

(LC8.5) Construct a 95% confidence interval for the *median* year of minting of *all* US pennies? Use the percentile method and, if appropriate, then use the standard-error method.

Solution:

Using the percentile method:

```
set.seed(539)
bootstrap_distribution <- pennies_sample %>%
  specify(formula = year ~ NULL) %>%
  generate(reps = 1000, type="bootstrap") %>%
  calculate(stat = "median")
```

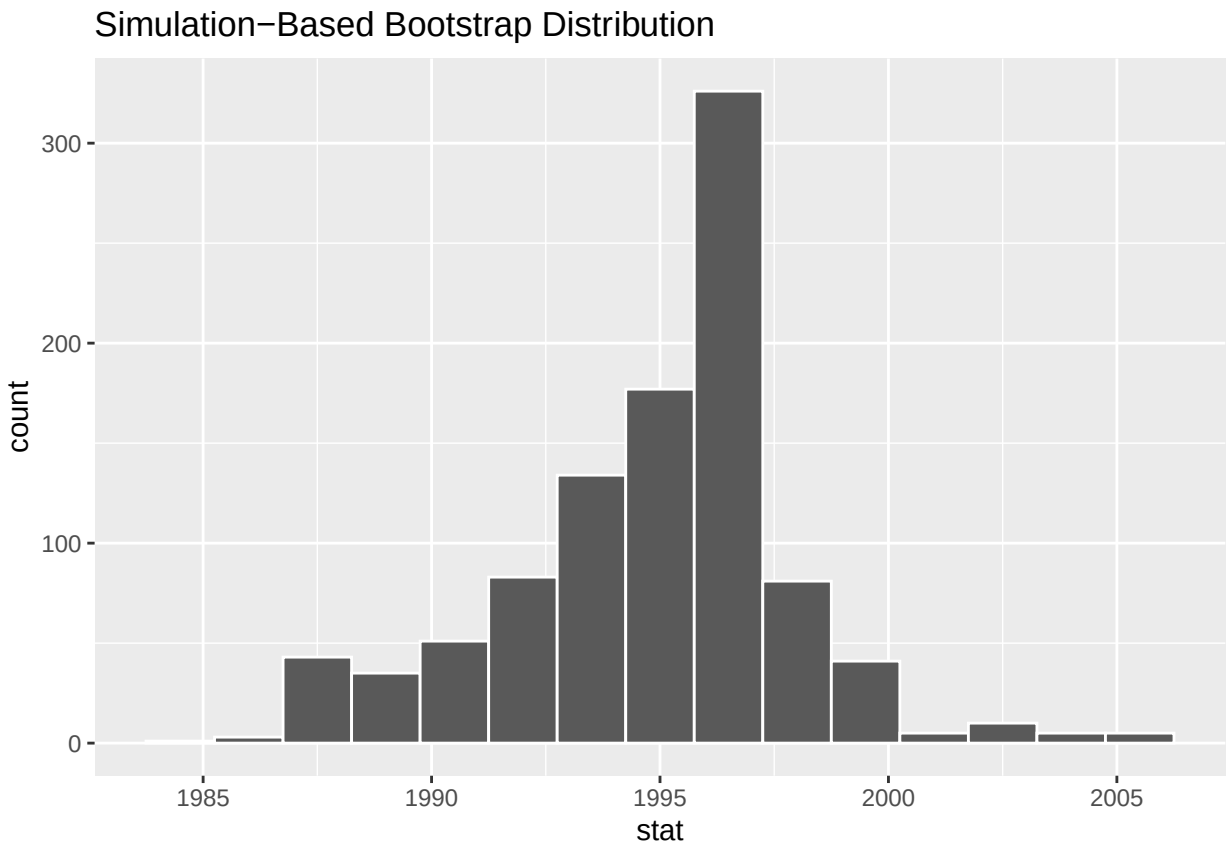
```
percentile_ci <- bootstrap_distribution %>%
  get_confidence_interval(level = 0.95, type = "percentile")
```

```
percentile_ci
```

```
## # A tibble: 1 x 2
##   lower_ci upper_ci
##      <dbl>    <dbl>
## 1     1988    2000.
```

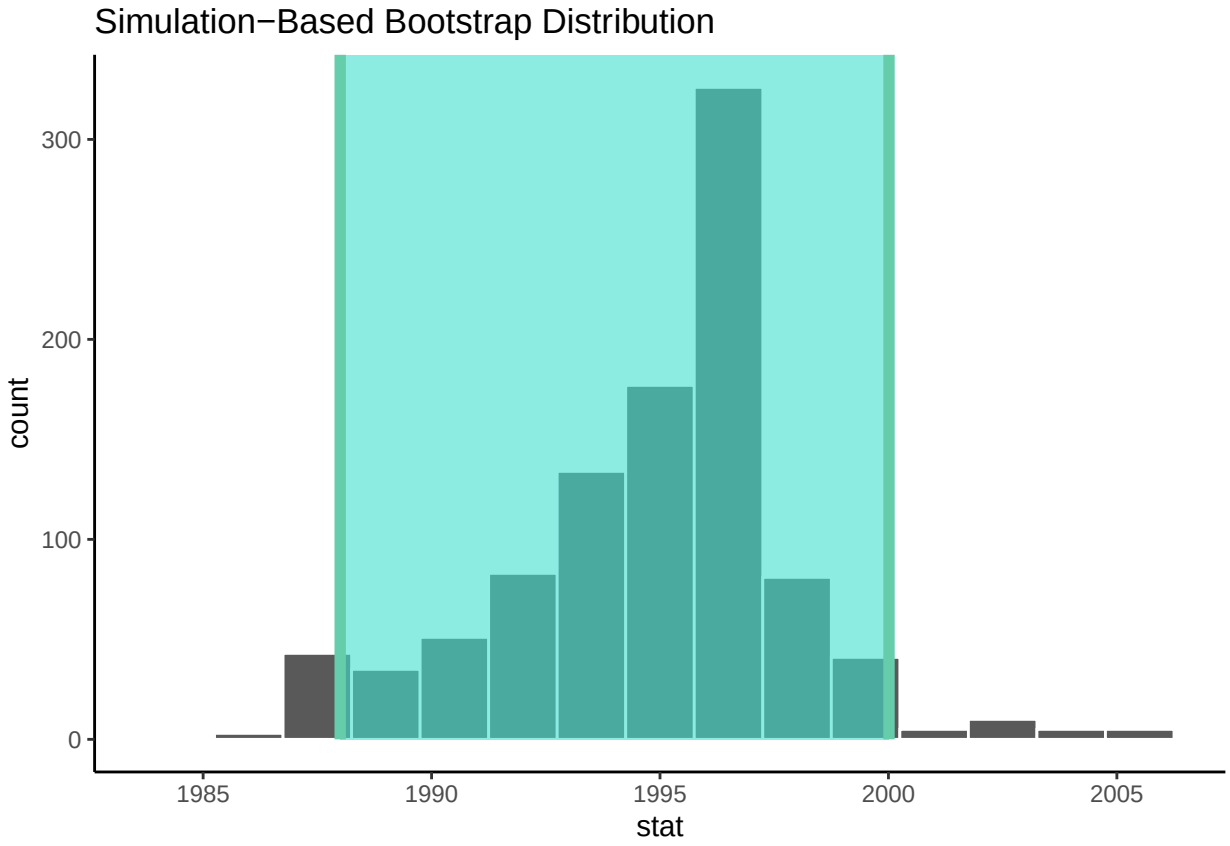
The standard-error method is not appropriate, because the bootstrap distribution is not bell-shaped:

```
visualize(bootstrap_distribution)
```



Let's visualize the interval.

```
visualize(bootstrap_distribution) +
  shade_confidence_interval(endpoints = percentile_ci) +
  theme_classic()
```



Documenting software

- File creation date: 2022-07-06
- R version 4.1.3 (2022-03-10)
- `tidyverse` package version: 1.3.1
- `moderndive` package version: 0.5.4
- `infer` package version: 1.0.2