Gov 50: 18. The Bootstrap

Matthew Blackwell

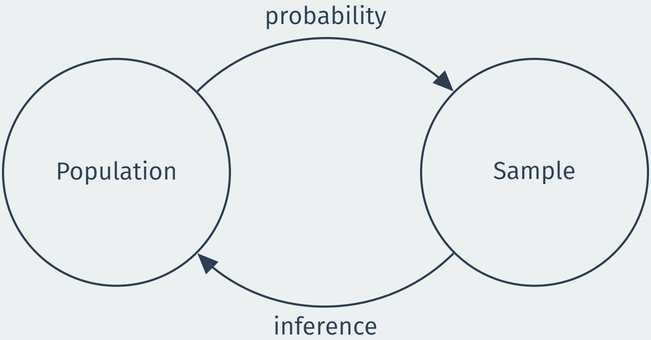
Harvard University

Roadmap

1. Resampling from our sample
2. Confidence intervals
3. Calculating confidence intervals

1/ Resampling from our sample

Where are we?



Can we approximate the **sampling distribution** with our single sample?

American National Election Survey data

Name	Description
`state`	State of respondent
`district`	Congressional district of respondent
`pid7`	Party ID (1=Strong D, 7=Strong R)
`pres_vote`	Self reported vote in 2020
`sci_therm`	0-100 therm score for scientists
`rural_therm`	0-100 therm score for rurual Americans
`favor_voter_id`	1 if respondent thinks voter ID should be required
`envir_doing_more`	1 if respondent thinks gov't should be doing more about climate change

ANES data

```r
library(gov50data)
anes
```

```
## # A tibble: 5,162 x 8
##    state district  pid7 pres_vote sci_therm rural_~1 favor~2
##    <chr>    <dbl> <dbl> <chr>         <dbl>    <dbl>   <dbl>
##  1 ID           2     4 Other            70       60       1
##  2 VA           2     3 Biden           100       75       0
##  3 CO           4     4 Trump            60       90       1
##  4 TX           5     3 Biden            85       85       1
##  5 WI           6     6 Trump            85       70       1
##  6 CA          40     2 Biden            50       50       1
##  7 WI           5     2 Biden           100       70       1
##  8 OR           4     7 Trump            70       50       0
##  9 MA           5     3 Biden            80       70       0
## 10 NV           3     1 Biden            85       40       0
## # ... with 5,152 more rows, 1 more variable:
## #   envir_doing_more <dbl>, and abbreviated variable names
## #   1: rural_therm, 2: favor_voter_id
```

Sample statistic

What is the average thermemeter score for scientists?

```
anes |>
  summarize(mean(sci_therm))
```

```
## # A tibble: 1 x 1
##   `mean(sci_therm)`
##               <dbl>
## 1              80.6
```

Sample statistic

What is the average thermemeter score for scientists?

```
anes |>
  summarize(mean(sci_therm))
```

```
## # A tibble: 1 x 1
##   `mean(sci_therm)`
##               <dbl>
## 1              80.6
```

What is the sampling distribution of this average? We only have this 1 draw!

Notation review

Population: all US adults.

Population parameter: average feeling thermometer score for scientists among all US adults.

Sample: (complicated) random sample of all US adults.

Sample statistic/point estimate: sample average of thermometer scores.

Notation review

Population: all US adults.

Population parameter: average feeling thermometer score for scientists among all US adults.

Sample: (complicated) random sample of all US adults.

Sample statistic/point estimate: sample average of thermometer scores.

Roughly how far our point estimate is likely to be from the truth?

The bootstrap

Mimic sampling from the population by **resampling** many times from the sample itself.

Bootstrap resampling done **with replacement** (same row can appear more than once)

One bootstrap resample

```
boot_1 <- anes |>
  slice_sample(prop = 1, replace = TRUE)
boot_1
```

```
## # A tibble: 5,162 x 8
##    state district  pid7 pres_vote sci_therm rural_~1 favor~2
##    <chr>    <dbl> <dbl> <chr>         <dbl>    <dbl>   <dbl>
##  1 CA          22     3 Biden           100       15       0
##  2 MA           8     1 Biden            85       40       0
##  3 OH          10     1 Biden           100       40       1
##  4 CA          33     2 Biden           100       50       0
##  5 MA           5     1 Biden           100       85       0
##  6 SC           4     5 Trump            50       90       1
##  7 WA           2     3 Biden           100      100       1
##  8 MD           5     6 Trump            50       60       1
##  9 FL           1     2 Biden           100       50       0
## 10 MO           5     7 Trump            85       70       1
## # ... with 5,152 more rows, 1 more variable:
## #   envir_doing_more <dbl>, and abbreviated variable names
## #   1: rural_therm, 2: favor_voter_id
```

Sample mean in the bootstrap sample

```r
boot_1 |>
  summarize(mean(sci_therm))
```

```
## # A tibble: 1 x 1
##   `mean(sci_therm)`
##               <dbl>
## 1              80.8
```

Many bootstrap samples

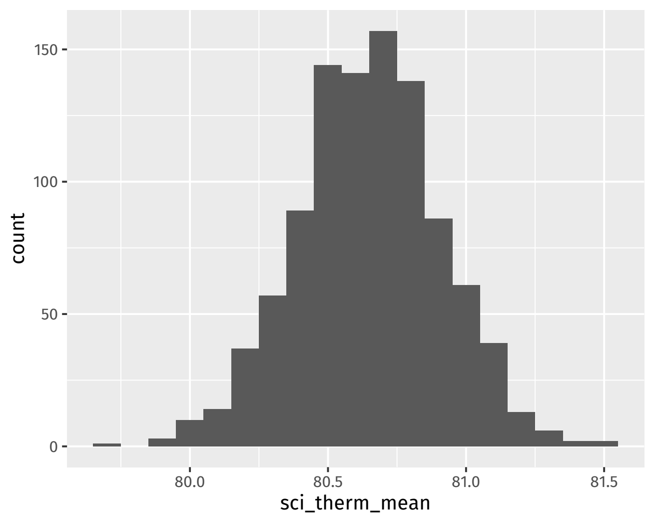
```r
library(infer)
bootstrap_dist <- anes |>
  rep_slice_sample(prop = 1, reps = 1000, replace = TRUE) |>
  group_by(replicate) |>
  summarize(sci_therm_mean = mean(sci_therm))
bootstrap_dist
```

Many bootstrap samples

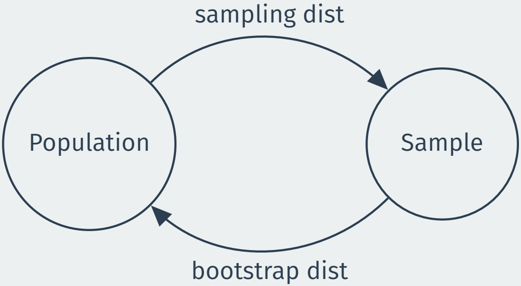
```
## # A tibble: 1,000 x 2
##    replicate sci_therm_mean
##        <int>          <dbl>
##  1         1           80.2
##  2         2           80.7
##  3         3           80.5
##  4         4           80.4
##  5         5           80.3
##  6         6           80.7
##  7         7           80.9
##  8         8           80.7
##  9         9           80.7
## 10        10           80.9
## # ... with 990 more rows
```

Visualizing the bootstrap distribution

```r
bootstrap_dist |>
  ggplot(aes(x = sci_therm_mean)) + geom_histogram(binwidth = 0.1)
```

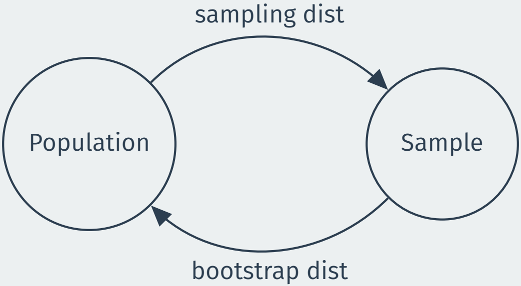


Bootstrap distribution



Bootstrap distribution **approximates** the sampling distribution of the estimator.

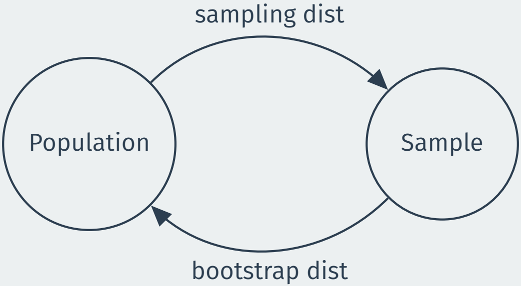
Bootstrap distribution



Bootstrap distribution **approximates** the sampling distribution of the estimator.

Both should have a **similar shape and spread** if sampling from the distribution ≈ bootstrap resampling.

Bootstrap distribution



Bootstrap distribution **approximates** the sampling distribution of the estimator.

Both should have a **similar shape and spread** if sampling from the distribution ≈ bootstrap resampling.

Approximation gets better as sample gets bigger.

Comparing to the point estimate

Given the sampling, not surprising that bootstrap distribution is centered on the point estimate:

```
bootstrap_dist |>
  summarize(mean(sci_therm_mean))
```

```
## # A tibble: 1 x 1
##   `mean(sci_therm_mean)`
##                    <dbl>
## 1                   80.6
```

```
anes |>
  summarize(mean(sci_therm))
```

```
## # A tibble: 1 x 1
##   `mean(sci_therm)`
##               <dbl>
## 1              80.6
```

2/ Confidence intervals

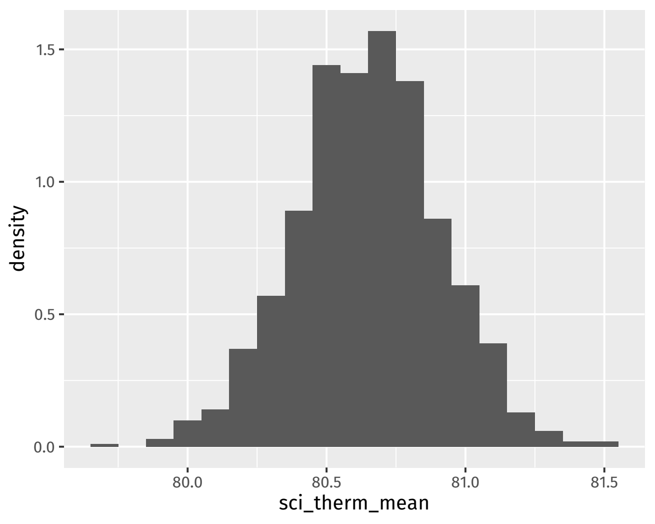
What is a confidence interval?

Point estimate: best single guess about the population parameter. Unlikely to be exactly correct.

Confidence interval: a range of plausible values of the population parameter.

Where is most of the bootstrap distribution?

```
bootstrap_dist |>
  ggplot(aes(x = sci_therm_mean)) +
  geom_histogram(aes(y= ..density..), binwidth = 0.1)
```



Confidence intervals

- Each sample gives a different CI or toss of the ring.

Confidence intervals

- Each sample gives a different CI or toss of the ring.
- Some samples the ring will contain the target (the CI will contain the truth) other times it won't.

Confidence intervals

- Each sample gives a different CI or toss of the ring.
- Some samples the ring will contain the target (the CI will contain the truth) other times it won't.
  - We don't know if the CI for our sample contains the truth!

Confidence intervals

- Each sample gives a different CI or toss of the ring.
- Some samples the ring will contain the target (the CI will contain the truth) other times it won't.
    - We don't know if the CI for our sample contains the truth!
- **Confidence level:** percent of the time our CI will contain the population parameter.

Confidence intervals

- Each sample gives a different CI or toss of the ring.
- Some samples the ring will contain the target (the CI will contain the truth) other times it won't.
  - We don't know if the CI for our sample contains the truth!
- **Confidence level:** percent of the time our CI will contain the population parameter.
  - Number of ring tosses that will hit the target.

Confidence intervals

- Each sample gives a different CI or toss of the ring.
- Some samples the ring will contain the target (the CI will contain the truth) other times it won't.
  - We don't know if the CI for our sample contains the truth!
- **Confidence level:** percent of the time our CI will contain the population parameter.
  - Number of ring tosses that will hit the target.
  - We get to choose, but typical values are 90%, 95%, and 99%

Confidence intervals as occasional liars

The **confidence level** of a CI determine how often the CI will be wrong.

Confidence intervals as occasional liars

The **confidence level** of a CI determine how often the CI will be wrong.

A 95% confidence interval will:

- Tell you the truth in 95% of repeated samples (contain the population parameter 95% of the time)

Confidence intervals as occasional liars

The **confidence level** of a CI determine how often the CI will be wrong.

A 95% confidence interval will:

- Tell you the truth in 95% of repeated samples (contain the population parameter 95% of the time)
- Lie to you in 5% of repeated sample (not contain the population parameter 5% of the time)

Confidence intervals as occasional liars

The **confidence level** of a CI determine how often the CI will be wrong.

A 95% confidence interval will:

- Tell you the truth in 95% of repeated samples (contain the population parameter 95% of the time)
- Lie to you in 5% of repeated sample (not contain the population parameter 5% of the time)

Confidence intervals as occasional liars

The **confidence level** of a CI determine how often the CI will be wrong.

A 95% confidence interval will:

- Tell you the truth in 95% of repeated samples (contain the population parameter 95% of the time)
- Lie to you in 5% of repeated sample (not contain the population parameter 5% of the time)

Can you tell if your particular confidence interval is telling the truth? No!

Percentile method

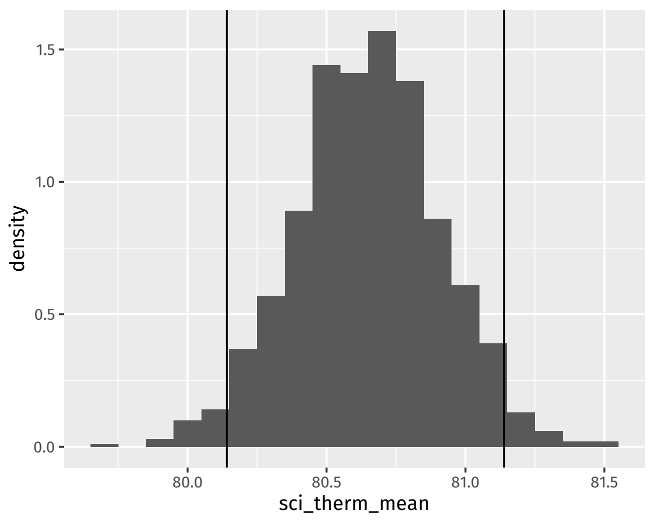
Percentile method: find the middle 95% of the bootstrap distribution.

We can do this by finding the points that the 2.5th percentile and the 97.5th percentile.

```
perc_ci95 <- quantile(bootstrap_dist$sci_therm_mean,
                      probs = c(0.025, 0.975))
perc_ci95
```

```
##  2.5% 97.5%
##  80.1  81.1
```

Visualizing the CI



Width of the interval

What happens if we want the CI to be right more often? Will the width of a 99% confidence interval be wider or narrower?

99% confidence interval

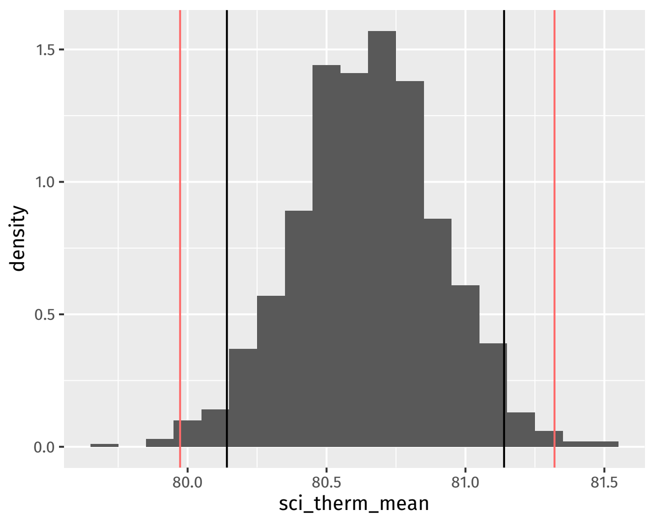
For 99% CI we need to find the middle 99% of the bootstrap distribution.

We can do this by finding the points that the 0.5th percentile and the 99.5th percentile.

```
perc_ci99 <- quantile(bootstrap_dist$sci_therm_mean,
                      probs = c(0.005, 0.995))
perc_ci99
```

```
##   0.5% 99.5%
##   80.0  81.3
```

Visualizing the CIs



3/ Calculating confidence intervals

`infer` package

Possible to use `quantile` to calculate CIs, but `infer` package is a more unified framework for CIs and hypothesis tests.

We'll use a `dplyr`-like approach of chained calls.

Step 1: define an outcome of interest

Start with defining the variable of interest:

```
anes |>
  specify(response = sci_therm)
```

```
## Response: sci_therm (numeric)
## # A tibble: 5,162 x 1
##    sci_therm
##        <dbl>
##  1        70
##  2       100
##  3        60
##  4        85
##  5        85
##  6        50
##  7       100
##  8        70
##  9        80
## 10        85
## # ... with 5,152 more rows
```

Step 2: generate bootstraps

Next `infer` can generate bootstraps with the `generate()` function (similar to `rep_slice_sample()`):

```
anes |>
  specify(response = sci_therm) |>
  generate(reps = 1000, type = "bootstrap")
```

```
## Response: sci_therm (numeric)
## # A tibble: 5,162,000 x 2
## # Groups:   replicate [1,000]
##    replicate sci_therm
##        <int>     <dbl>
##  1         1        50
##  2         1       100
##  3         1        50
##  4         1        60
##  5         1        60
##  6         1        85
##  7         1        90
##  8         1       100
##  9         1        80
## 10         1       100
## # ... with 5,161,990 more rows
```

Step 3: calculate sample statistics

Use `calculate()` to do the `group_by(replicate)` and `summarize` commands in one:

```
boot_dist_infer <- anes |>
  specify(response = sci_therm) |>
  generate(reps = 1000, type = "bootstrap") |>
  calculate(stat = "mean")
```

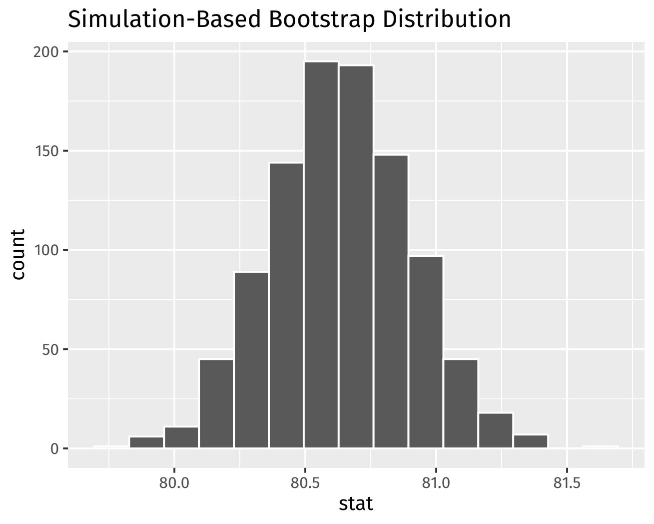
```r
boot_dist_infer
```

```
## Response: sci_therm (numeric)
## # A tibble: 1,000 x 2
##    replicate  stat
##        <int> <dbl>
##  1         1  80.7
##  2         2  80.3
##  3         3  80.5
##  4         4  80.6
##  5         5  80.4
##  6         6  80.9
##  7         7  80.6
##  8         8  80.7
##  9         9  80.9
## 10        10  80.6
## # ... with 990 more rows
```

Step 3(b): visualize the boostrap distribution

`infer` also has a shortcut for plotting called `visualize()`:

```
visualize(boot_dist_infer)
```



Step 4: calculate CIs

Finally we can calculate the CI using the percentile method with `get_confidence_interval()`:

```
perc_ci_95 <- boot_dist_infer |>
  get_confidence_interval(level = 0.95, type = "percentile")
perc_ci_95
```

```
## # A tibble: 1 x 2
##   lower_ci upper_ci
##      <dbl>    <dbl>
## 1     80.1     81.2
```

Step 4(b): visualize CIs

```
visualize(boot_dist_infer) +
  shade_confidence_interval(endpoints = perc_ci_95)
```