Pre-midterm Section

Hikaru Yamagishi

3/7/2019

Today in section, we reviewed the bootstrap. First, conceptually; then in R code.

This code contains two ways to go about bootstrapping: We can define a function or use a `for` loop. The first may be more helpful for the near future.

Bootstrapping: Concepts

- Plug-in principle
- ECDF
- "Pull yourself up by your bootstraps"

Remember that we are trying to characterize the uncertainty of our estimator.

Bootstrapping: 4 steps

First, let us walk through the steps of bootstrapping.

We will talk through each step and figure out the code for **one** resampling iteration first.

We will start with one variable. Remember Pset 2? What was the setting then? Remember to keep population and sample straight.

Data

Now let's assume that this example data `df_eg1` is our sample data ($n = 100$):

```
# Let's say this is our sample data
df_eg1 <- sample(x = c(56:59), size = 100, replace = TRUE)
head(df_eg1)
```

```
## [1] 59 57 56 58 59 58
```

4 steps

1. From your sample, take a *with replacement* sample of size n.

```
# Define the number of samples you want to draw
n_data <- length(df_eg1)

# Sample
samp <- sample(x = df_eg1, size = n_data, replace = TRUE)
samp
```

```
##   [1] 58 56 58 58 59 57 59 56 56 56 58 58 56 56 59 59 56 56 59 56 56 56 59
##  [24] 56 57 59 59 56 59 57 56 59 57 59 59 58 57 59 58 57 56 56 58 57 56 57
##  [47] 56 57 58 58 59 59 57 59 57 58 56 58 56 56 58 56 56 57 58 57 57 57 57
##  [70] 56 58 59 59 58 57 58 57 59 59 58 56 56 57 59 58 59 58 56 56 58 58 58
##  [93] 56 56 57 57 58 56 57 59
```

2. Calculate what your estimate would be using the bootstrap sample from step 1.

If the test statistic of interest was the mean, we would run:

```
mean(samp)
```

```
## [1] 57.39
```

3. Repeat step 1 and step 2 many times.

We now have a collection of bootstrap estimates.

4. Calculate the standard deviation of the bootstrap distribution of our estimator.

We take the bootstrap estimates obtained in step 3 and run the `sd()` function.

Example 2: Full example

Now suppose there are multiple variables, X, Y, and Z.

Suppose that our sample statistic of interest is the sum of the mean of X, mean of Y, and mean of Z. Now we will try the whole bootstrapping procedure, including the iteration part.

Data

Let's use this toy data.

```
set.seed(503)

data_sample <- data.frame(
  x = rnorm(n = 1000, mean = 3, sd = 1),
  y = rnorm(n = 1000, mean = 4, sd = 2),
  z = rnorm(n = 1000, mean = 6, sd = 3)
)
```

Define the function

Let's define our function `estimator`:

```
estimator <- function(sample) {mean(sample$x) + mean(sample$y) + mean(sample$z)}
estimator(sample = data_sample)
```

```
## [1] 13.02346
```

Bootstrapping using a user-defined function

1. From your sample, take a *with replacement* sample of size n. This time, let's define this as a function. Notice how we sample different variables. *discussion*

```
draw_bootstrap_sample <- function() {
  data_sample[sample(nrow(data_sample), replace = TRUE), ]
}
```

Addendum: `samp.int` is a bare interface version of `sample`. *discussion*

2. Calculate what your estimate would be using the bootstrap sample from step 1.
3. Repeat step 1 and step 2 many times.

```
n_iter <- 1000
bootstrap_est <- replicate(n_iter, estimator(draw_bootstrap_sample()))
```

We now have a collection of bootstrap estimates.

4. Calculate the standard deviation of the bootstrap distribution of our estimator (standard error)

```
est_se_est <- sd(bootstrap_est)
est_se_est
```

```
## [1] 0.1141754
```

Example using `for` loop

Suppose we have a function that will give us three values for each of the three variables.

Define function

Don't worry about what this is, we just want to illustrate how we may do this with three outputs (as discussed in class).

```
estimator2 <- function(sample) {
  c(mean(sample$x), mean(sample$y)/mean(sample$x), mean(sample$z)/mean(sample$x))
}
estimator2(sample = data_sample)
```

```
## [1] 2.987772 1.327488 2.031432
```

Using a `for` loop

```
# Number of sample data rows
nrow_samp <- nrow(data_sample)

# Number of iterations
n_iter = 1000

# Store the three coefficients from each iteration in a row of this matrix.
boot <- matrix(NA, nrow = n_iter, ncol = 3)

# For loop
for(i in 1:n_iter){
  samp_data <- data_sample[sample(nrow_samp, replace = T),]
  boot[i,] <- estimator2(samp_data)
}
```

```r
# Take sd() of each column to return a bootstrapped SE on each estimate.
apply(boot, MARGIN = 2, FUN = sd)
```

```
## [1] 0.03052673 0.02578081 0.03783804
```