Bootstrapping Tutorial

Michael Kummer

Introduction

This tutorial explores the concept of bootstrapping in statistics, using R to perform bootstrapping to estimate the confidence intervals of regression coefficients. Bootstrapping is a resampling technique used to approximate the sampling distribution of an estimator by sampling with replacement from the original data.

Setup

First, we'll load the required libraries:

```
require(data.table)
```

```
## Loading required package: data.table
```

```
## Warning: package 'data.table' was built under R version 4.1.3
```

```
require(sandwich)
```

```
## Loading required package: sandwich
```

```
require(lmtest)
```

```
## Loading required package: lmtest
```

```
## Warning: package 'lmtest' was built under R version 4.1.3
```

```
## Loading required package: zoo
```

```
## Warning: package 'zoo' was built under R version 4.1.3
```

```
##
## Attaching package: 'zoo'
```

```
## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric
```

```
require(stargazer)
```

```
## Loading required package: stargazer
```

```
##
## Please cite as:
```

```
##  Hlavac, Marek (2022). stargazer: Well-Formatted Regression and Summary Statistics Tables.
```

```
##  R package version 5.2.3. https://CRAN.R-project.org/package=stargazer
```

Create Population Data

We start by creating a synthetic dataset to represent a population of 25,000 individuals with two independent variables and a dependent variable generated by a linear model.

```r
set.seed(1984)
n1 <- 25000
x1 <- rnorm(n = n1, mean = 0, sd = 3)  # Independent variable 1
x2 <- rnorm(n = n1, mean = 0, sd = 4)  # Independent variable 2
e <- rnorm(n = n1, mean = 0, sd = 2)   # Error term
y <- 2 + 3*x1 + 4*x2 + e               # Dependent variable
sorting <- rnorm(n = n1, mean = 0, sd = 2)   # Variable for sorting and sampling

# Creating and organizing the data table
dt.population <- data.table(y, x1, x2, sorting)
dt.population <- dt.population[order(sorting)]  # Randomizing data
```

Sampling and Initial Regression

Now, we draw a random sample from the population and perform a regression analysis.

```r
ssize <- 2000  # Sample size
r.sample.rows <- sample(1:nrow(dt.population), size = ssize)
r.sample <- dt.population[r.sample.rows, ]
ols1 <- lm(y ~ x1 + x2, data = r.sample)
summary(ols1)
```

```
##
## Call:
## lm(formula = y ~ x1 + x2, data = r.sample)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -7.9418 -1.3424  0.0165  1.3364  6.1312
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  2.02431    0.04346   46.58   <2e-16 ***
## x1           3.02483    0.01428  211.81   <2e-16 ***
## x2           3.99298    0.01080  369.72   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.942 on 1997 degrees of freedom
## Multiple R-squared:  0.9893, Adjusted R-squared:  0.9893
## F-statistic: 9.246e+04 on 2 and 1997 DF,  p-value: < 2.2e-16
```

Bootstrapping

To understand the variability of our estimate, we'll perform bootstrapping by repeatedly resampling and recalculating the regression coefficient.

```r
bootreps <- 5  # Number of bootstrap repetitions for demonstration
boot.resultsvec <- numeric(bootreps)

for (j in 1:bootreps) {
  r.sample.rows <- sample(1:nrow(dt.population), size = ssize)
  r.sample <- dt.population[r.sample.rows, ]
  olsboot <- lm(y ~ x1 + x2, data = r.sample)
  boot.resultsvec[j] <- coef(olsboot)["x1"]
```

```
}
boot.resultsvec
```

```
## [1] 3.006677 2.998252 3.003211 2.997287 3.000090
```

Full Bootstrap Procedure

For a more robust estimation, we increase the number of repetitions.

```
bootreps <- 1000
boot.resultsvec <- numeric(bootreps)

for (j in 1:bootreps) {
  r.sample.rows <- sample(1:nrow(dt.population), size = ssize)
  r.sample <- dt.population[r.sample.rows, ]
  olsboot <- lm(y ~ x1 + x2, data = r.sample)
  boot.resultsvec[j] <- coef(olsboot)["x1"]
}

# Sorting results and extracting confidence intervals
boot.resultsvec <- sort(boot.resultsvec)
conf.low <- boot.resultsvec[round(bootreps*0.025)]
conf.up <- boot.resultsvec[round(bootreps*0.975)]

# Display the 95% confidence interval
list(conf.low = conf.low, conf.up = conf.up)
```

```
## $conf.low
## [1] 2.973118
##
## $conf.up
## [1] 3.031781
```

Conclusion

The output shows the 2.5th and 97.5th percentiles as the boundaries of the 95% confidence interval for the coefficient of `x1`. This bootstrap method provides an empirical distribution of the estimator allowing us to quantify the uncertainty around our estimate.

Feel free to compare this to the asymptotic values derived from normal theory, which should be very close if the number of bootstrap repetitions is sufficiently large.

Clarification - Why heck?

Note that in this tutorial, you might wonder why we did all of this Bootstrapping if we could simply use the errors we got from the OLS-package?

The reason is simple: Bootstrapping essentially `ALWAYS'' works. Also when you don't know the`theoretical'' standard errors.

Last Note

In cases where you have too little data to comfortably resample a subset, consider using sampling with replacement from the entire dataset. This approach adheres more closely to the theoretical foundations of the bootstrap method.