Boostrap

Khanin Sisaengsuwanchai

2022-06-07

Create a bootstrap function in R.

Parameters - *B*: The number of bootstrap replications to be done (default of 1000). - *ci*: Confidence level for the interval (default 95%).

```
# Define a function to calculate confidence interval with t distribution because
# we don't know the population parameters
conf_interv = function(data, index, ci) {

   # Select data based on given index
   data = data[index]

   sample.n = length(data)   # Number of sample
   sample.mean = mean(data) # Calculate sample mean
   sample.se   = sd(data)/sqrt(sample.n) # Calculate sample standard errors

   # Calculate t-score
   alpha = 1-ci # Confidence interval
   t.score = qt(p=alpha/2, df=sample.n-1, lower.tail=FALSE)

   margin.error <- t.score * sample.se  # Calculate margin of errors
   bound.lower <- sample.mean - margin.error # Lower bound
   bound.upper <- sample.mean + margin.error # Upper bound

   conf_limits = data.frame("lower_bound" = bound.lower, "upper_bound" = bound.upper)
   return(conf_limits)
}

# Create a function that do bootstrap as defined params
boost_ci = function(Y, B=1000, ci=0.95) {
  # 1. if ci > 1 or ci < 0, returns errors
  # 2. Define max B

  i = 1

  # Define an empty dataframe to keep the confidence interval in each loop
  boost_ci_df = data.frame(lower_bound = numeric(0), upper_bound = numeric(0))

  while( i <= B){
    boost = sample(length(Y), length(Y), replace = T)

    # Collect confidence interval
```

```
    boost_ci_df = rbind(boost_ci_df, conf_interv(Y, boost, ci=ci))
    i = i + 1

  }

  return(boost_ci_df)
}
```

(a) Using the baby weights dataset, construct a bootstrap 95% confidence interval for the mean infant birth weight. Compare this to the ordinary confidence interval.

```
load(url("http://www.stodden.net/StatData/KaiserBabies.rda"))


set.seed(1)

# 1. Calculate ordinary confidence interval on birth weight
ord_ci.bw = conf_interv(infants$bwt, 1:length(infants$bwt), ci=0.95)
ord_ci.bw
```

```
##   lower_bound upper_bound
## 1    118.5592    120.5945
```

```
# 2. Generate boostrap dataset and return confidence interval
boost_ci_df.bw = boost_ci(infants$bwt, ci=0.95)
mean(boost_ci_df.bw$lower_bound) # mean of lower bound
```

```
## [1] 118.5384
```

```
mean(boost_ci_df.bw$upper_bound) # mean of upper bound
```

```
## [1] 120.572
```

```
# Ordinary lower bound is 118.5592 and boostrap lower bound is 118.5384,
# which is 0.021 more than boostrap lower bound.
# Ordinary upper bound is 120.5945 and boostrap upper bound is 120.572,
# which is 0.0225 more than boostrap upper bound.
```

(b) Repeat the bootstrap interval from (a) 9 more times, using $B = 1000$ each time. Do you think $B = 1000$ is big enough for this problem? Why or why not? You may want to empirically test your ideas by running your function with B » 1000.

```
# Repeat bootstrap 10 more times with B=100 and ci=0.95

# 1
boost_ci_df.bw = boost_ci(infants$bwt, B=1000)
print('#1')
```

```
## [1] "#1"
```

```r
mean(boost_ci_df.bw$lower_bound) # mean of lower bound
```

```
## [1] 118.5466
```

```r
mean(boost_ci_df.bw$upper_bound) # mean of upper bound
```

```
## [1] 120.5793
```

```r
# 2
boost_ci_df.bw = boost_ci(infants$bwt, B=1000)
print('#2')
```

```
## [1] "#2"
```

```r
mean(boost_ci_df.bw$lower_bound) # mean of lower bound
```

```
## [1] 118.5252
```

```r
mean(boost_ci_df.bw$upper_bound) # mean of upper bound
```

```
## [1] 120.5598
```

```r
# 3
boost_ci_df.bw = boost_ci(infants$bwt, B=1000)
print('#3')
```

```
## [1] "#3"
```

```r
mean(boost_ci_df.bw$lower_bound) # mean of lower bound
```

```
## [1] 118.5875
```

```r
mean(boost_ci_df.bw$upper_bound) # mean of upper bound
```

```
## [1] 120.6191
```

```r
# 4
boost_ci_df.bw = boost_ci(infants$bwt, B=1000)
print('#4')
```

```
## [1] "#4"
```

```r
mean(boost_ci_df.bw$lower_bound) # mean of lower bound
```

```
## [1] 118.5762
```

```r
mean(boost_ci_df.bw$upper_bound) # mean of upper bound
```

```
## [1] 120.6094
```

```r
# 5
boost_ci_df.bw = boost_ci(infants$bwt, B=1000)
print('#5')
```

```
## [1] "#5"
```

```r
mean(boost_ci_df.bw$lower_bound) # mean of lower bound
```

```
## [1] 118.5631
```

```r
mean(boost_ci_df.bw$upper_bound) # mean of upper bound
```

```
## [1] 120.5979
```

```r
# 6
boost_ci_df.bw = boost_ci(infants$bwt, B=1000)
print('#6')
```

```
## [1] "#6"
```

```r
mean(boost_ci_df.bw$lower_bound) # mean of lower bound
```

```
## [1] 118.566
```

```r
mean(boost_ci_df.bw$upper_bound) # mean of upper bound
```

```
## [1] 120.6015
```

```r
# 7
boost_ci_df.bw = boost_ci(infants$bwt, B=1000)
print('#7')
```

```
## [1] "#7"
```

```r
mean(boost_ci_df.bw$lower_bound) # mean of lower bound
```

```
## [1] 118.5255
```

```r
mean(boost_ci_df.bw$upper_bound) # mean of upper bound
```

```
## [1] 120.561
```

```r
# 8
boost_ci_df.bw = boost_ci(infants$bwt, B=1000)
print('#8')
```

```
## [1] "#8"
```

```r
mean(boost_ci_df.bw$lower_bound) # mean of lower bound
```

```
## [1] 118.5518
```

```r
mean(boost_ci_df.bw$upper_bound) # mean of upper bound
```

```
## [1] 120.5857
```

```r
# 9
boost_ci_df.bw = boost_ci(infants$bwt, B=1000)
print('#9')
```

```
## [1] "#9"
```

```r
mean(boost_ci_df.bw$lower_bound) # mean of lower bound
```

```
## [1] 118.5603
```

```r
mean(boost_ci_df.bw$upper_bound) # mean of upper bound
```

```
## [1] 120.5968
```

```r
# 10
boost_ci_df.bw = boost_ci(infants$bwt, B=1000)
print('#10')
```

```
## [1] "#10"
```

```r
mean(boost_ci_df.bw$lower_bound) # mean of lower bound
```

```
## [1] 118.5747
```

```r
mean(boost_ci_df.bw$upper_bound) # mean of upper bound
```

```
## [1] 120.6084
```

```r
## Try with B >> 1000

set.seed(1)

# B = 10,000

boost_ci_df = boost_ci(infants$bwt, B=10000, ci=0.95)
mean(boost_ci_df$lower_bound) # mean of lower bound
```

```
## [1] 118.5541
```

```r
mean(boost_ci_df$upper_bound) # mean of upper bound
```

```
## [1] 120.5883
```

```r
# B = 30,000
boost_ci_df = boost_ci(infants$bwt, B=30000, ci=0.95)
mean(boost_ci_df$lower_bound) # mean of lower bound
```

```
## [1] 118.5595
```

```r
mean(boost_ci_df$upper_bound) # mean of upper bound
```

```
## [1] 120.5937
```

```r
# B = 1000
# lower bound is 118.5384
# upper bound is 120.572

# B = 10000
# lower bound is 118.5541
# upper bound is 120.5883

# B = 50000
# lower bound is 118.5595
# upper bound is 120.5937

# According to the results, I believe having B = 1000 is not big enough because
# When we increase B, the lower bound and upper bound still increase and they become
# stable with only third digits differences. However, I choose B=1000 due to computationally
# limitation.
```

(c) Compute bootstrap ($B = 1000$) and regular 95% confidence intervals for for mean house price in the SFHousing data.

```r
# Load house data
load(url("https://www.stodden.net/StatData/SFHousing.rda"))

set.seed(1)

# 1. Calculate ordinary confidence interval on housing price
ord_ci.p = conf_interv(housing$price, 1:length(housing$price), ci=0.95)
ord_ci.p
```

```
##   lower_bound upper_bound
## 1    600699.5      603300
```

```r
# 2. Generate boostrap dataset and return confidence interval
boost_ci_df.hou = boost_ci(housing$price, B=1000, ci=0.95)
mean(boost_ci_df.hou$lower_bound) # mean of lower bound
```

```
## [1] 600705.4
```

```r
mean(boost_ci_df.hou$upper_bound) # mean of upper bound
```

```
## [1] 603305
```

(d) For which dataset does the difference in the bootstrap and regular confidence intervals appear greatest? Is there anything about these datssets that might lead you to predict that for one of them, the parametric intervals might be better than bootstrap intervals, or vice versa?

```r
# To calculate the difference in the bootstrap and regular confidence intervals,
# I will use percent differences to illustrate the results.

# 1. Kaiser dataset
# Calculate percent difference of regular confidence interval and boostrap
diff_lower.kai = abs(ord_ci.bw$lower_bound-mean(boost_ci_df.bw$lower_bound))*100/ord_ci.bw$lower_bound
diff_lower.kai
```

```
## [1] 0.01310211
```

```r
diff_upper.kai = abs(ord_ci.bw$upper_bound-mean(boost_ci_df.bw$upper_bound))*100/ord_ci.bw$upper_bound
diff_upper.kai
```

```
## [1] 0.01147113
```

```r
# 2. SF Housing dataset
# Calculate percent difference of regular confidence interval and boostrap
diff_lower.hou = abs(ord_ci.p$lower_bound-mean(boost_ci_df.hou$lower_bound))*100/ord_ci.p$lower_bound
diff_lower.hou
```

```
## [1] 0.0009898389
```

```r
diff_upper.hou = abs(ord_ci.p$upper_bound-mean(boost_ci_df.hou$upper_bound))*100/ord_ci.p$upper_bound
diff_upper.hou
```

```
## [1] 0.0008248393
```

```r
# According to the results, the Kaiser dataset has the largest percent differences compared to SFhousin
# The situation where parametric intervals might be better than bootstrap intervals is when
# we already have a lot of data says more than 100,000 rows, so it is no need to boostrap.
```