Department of Statistics and Data Science at SUSTech

MAT7035: Computational Statistics

Tutorial 12: Bootstrap Methods

A. Parametric Bootstrap

(a) Assumptions:

- θ is an unknown parameter;
- $\mathbf{x} = (X_1, \ldots, X_n)^\top$ is a random sample from a population density $f(x; \theta)$;
- The point estimator of θ is given by $\hat{\theta} = s(\mathbf{x})$, where $s(\mathbf{x})$ is a function of $\mathbf{x}$.

(b) Goal: To obtain bootstrap confidence intervals of θ.

(c) Methods:

- Step 1: Calculate $\hat{\theta} = s(\mathbf{x})$.
- Step 2: Generate a bootstrap sample $\mathbf{x}^* = (X_1^*, \ldots, X_n^*)^\top$ with $\{X_i^*\}_{i=1}^n \overset{\text{iid}}{\sim} f(x; \hat{\theta})$ and calculate the bootstrap replication $\hat{\theta}^* = s(\mathbf{x}^*)$.
- Step 3: Independently repeating this process (i.e., Step 2) G times and get G bootstrap replications $\{\hat{\theta}^*(g)\}_{g=1}^G$.
- Step 4: The standard error, $\mathrm{Se}(\hat{\theta})$, of $\hat{\theta}$ can be estimated by the sample standard deviation of the G replications, i.e.,

$$\widehat{\mathrm{Se}}^*(\hat{\theta}) = \sqrt{\frac{1}{G-1}\sum_{g=1}^G \left[\hat{\theta}^*(g) - \bar{\theta}^*\right]^2},$$

  where $\bar{\theta}^* = \sum_{g=1}^G \hat{\theta}^*(g)/G$.
- Step 5: If $\{\hat{\theta}^*(g)\}_{g=1}^G$ are **approximately normally distributed**, a $100(1-\alpha)\%$ bootstrap CI for θ is

$$\left[\hat{\theta}_l^*, \hat{\theta}_u^*\right] = \left[\bar{\theta}^* - z_{\alpha/2} \cdot \widehat{\mathrm{Se}}^*(\hat{\theta}),\, \bar{\theta}^* + z_{\alpha/2} \cdot \widehat{\mathrm{Se}}^*(\hat{\theta})\right].$$

- Step 6: If the bootstrap replications $\left\{\hat{\theta}^*(g)\right\}_{g=1}^{G}$ are **non-normally distributed**, a $100(1-\alpha)\%$ bootstrap CI for θ is

$$\left[\hat{\theta}_L^*, \hat{\theta}_U^*\right],$$

where $\hat{\theta}_L^*$ and $\hat{\theta}_U^*$ are the $(\alpha/2)G$-th and the $(1-\alpha/2)G$-th **order statistics** of $\left\{\hat{\theta}^*(g)\right\}_{g=1}^{G}$.

B. Non-parametric Bootstrap

(a) Assumptions:

- $\mathbf{x} = (X_1, \ldots, X_n)^\top$ is a random sample from an unknown population cdf F;
- The point estimator of $\theta = T(F)$ is given by $\hat{\theta} = T(\hat{F}_n) = s(\mathbf{x})$, where

— $T(F)$ is a function of F;

— $\hat{F}_n(x) = \frac{1}{n}\sum_{i=1}^{n} I(x_i \leqslant x)$ is the empirical cdf, where $x_1 \leqslant \cdots \leqslant x_n$.

(b) Goal: To obtain bootstrap confidence intervals of θ.

(c) Methods:

- Step 1: Calculate $\hat{\theta} = T(\hat{F}_n) = s(\mathbf{x})$.
- Step 2: Generate a bootstrap sample $\mathbf{x}^* = (X_1^*, \ldots, X_n^*)$ with $\{X_i^*\}_{i=1}^{n} \overset{\text{iid}}{\sim} \hat{F}_n(x)$ and calculate the bootstrap replication $\hat{\theta}^* = s(\mathbf{x}^*)$.
- Steps 3–6 are the same as those in the parametric bootstrap method.

Example T12.1 (Exponential distribution). Suppose that $X_1, \ldots, X_n$ with $n = 300$ is a random sample from an exponential distribution with density $f(x; \lambda) = \lambda \exp(-\lambda x)$ and the mean of the data is $\bar{x} = (1/n)\sum_{i=1}^{n} x_i = 2$. Estimate λ and give a 95% parametric bootstrap confidence interval for λ.

Remark: Note that

$$X_i \sim \text{Exponential}(\lambda) = \text{Gamma}(1, \lambda),$$

we have

$$n\bar{X} = \sum_{i=1}^{n} X_i \sim \text{Gamma}(n, \lambda),$$

so that $\lambda n\bar{X} \sim \text{Gamma}(n, 1)$. Thus, the exact $(1-\alpha)\%$ CI of λ is

$$\left[\frac{\gamma_{\alpha/2}(n,1)}{n\bar{X}}, \frac{\gamma_{1-\alpha/2}(n,1)}{n\bar{X}}\right],$$

where $\gamma_{\alpha/2}(n,1)$ denotes the lower $\alpha/2$ quantile of $X \sim \text{Gamma}(n,1)$ such that

$$\Pr\{X \leqslant \gamma_{\alpha/2}(n,1)\} = \alpha/2.$$

Solution: (a) The MLE of λ is $\hat{\lambda} = 1/\bar{x} = 0.5$.

(b) The exact 95% CI of λ is

$$[\hat{\lambda}_l, \hat{\lambda}_u] = \left[\frac{\gamma_{0.025}(300,1)}{300\bar{x}}, \frac{\gamma_{0.975}(300,1)}{300\bar{x}}\right] = \left[\frac{267.0093}{600}, \frac{334.8846}{600}\right] = [0.44502, 0.55814].$$

(c) We use the bootstrap approach to estimate $\text{Se}(\hat{\lambda})$ and to obtain two BCIs by generating $G = 1000$ bootstrap samples: $\mathbf{x}^*(g) = (X_1^*(g), \ldots, X_n^*(g))$ with $\{X_i^*(g)\}_{i=1}^n \overset{\text{iid}}{\sim} \text{Exponential}(\hat{\lambda})$ for $g = 1, \ldots, G$, and computing 1000 bootstrap replications $\left\{\hat{\lambda}^*(g)\right\}_{g=1}^G$. Numerical results are as follows:

$$\begin{aligned}
\bar{\lambda}^* &= 0.501, \\
\widehat{\text{Se}}^*(\hat{\lambda}) &= 0.028, \\
\left[\hat{\lambda}_l^*, \hat{\lambda}_u^*\right] &= [0.446, 0.556], \\
\left[\hat{\lambda}_L^*, \hat{\lambda}_U^*\right] &= [0.451, 0.558].
\end{aligned}$$

(d) The `R` codes are as follows:

```
mean.std.CI <- function(lasample){
    # Name: mean.std.CI(lasample)
    # Input: lasample = the bootstrap sample, a matrix of G x c
    # Output: the mean, std, two 95% CIs based on column
    G <- dim(lasample)[1]
    lamean <- apply(lasample, 2, mean)
    lastd <- sqrt(apply(lasample, 2, var))
    lal <- lamean - 1.96 * lastd
    lau <- lamean + 1.96 * lastd
    lasort <- apply(lasample, 2, sort)
    indexx <- floor(c(0.025 * G, 0.975 * G))
    laL <- (lasort[indexx[1], ]+ lasort[indexx[1]+1, ])/2
    laU <- (lasort[indexx[2], ]+ lasort[indexx[2]+1, ])/2
    Result <- c(lamean, lastd, lal, lau, laL, laU)
```

```
    return(Result)   }

ExampleT12.1<-function(G){
    # Name: Example12.1(G=1000)
    # Call: c(lamean, lastd, lal, lau, laL, laU) <- mean.std.CI(lasample)
    # G is the bootstrap sample size
    n = 300
    xbar = 2
    lambdahat = 1/xbar
    la.star.sample <- matrix(0, G, 1)
    for(g in 1:G) {
       xstar<-rexp(n,lambdahat)
       lambdastar<-1/mean(xstar)
       la.star.sample[g] <- lambdastar }
    M <- mean.std.CI(la.star.sample)
    lamean <-sprintf("lamean: %.3f", M[[1]])
    lastd <- sprintf("lastd: %.3f", M[[2]])
    CI1<-sprintf("Normality-based BCI for lambda: [%.3f,%.3f]",M[[3]],M[[4]])
    CI2<-sprintf("Non-normality-based BCI for lambda: [%.3f,%.3f]",M[[5]],M[[6]])
    Result = c(lamean, lastd, CI1, CI2)
    return(Result)    }

ExampleT12.1(1000)
```

The above code prints the following output:

```
[1] "lamean: 0.501"
[2] "lastd: 0.028"
[3] "Normality-based BCI for lambda: [0.446, 0.556]"
[4] "Non-normality-based BCI for lambda: [0.451, 0.558]"
```