Computational Statistics. Chapter 6: Bootstrapping. Solution of exercises

Thierry Denoeux

2024-03-21

```
set.seed(20240321)
```

Exercise 1

Question a

Data generation:

```
n<-50
rate<-0.5
x<-rexp(n,rate=rate)
```

The cdf of the exponential distribution is $F(x) = 1 - \exp(-\theta x)$. The equation $F(m) = 0.5$ gives us $m = \log(2)/\theta$. Here, we get

```
mtrue<-log(2)/rate
print(mtrue)
```

```
## [1] 1.386294
```

Question b

```
bootstrap <- function(x,B){
    n<-length(x)
    X<-matrix(0,B,n)
    for(b in (1:B)) X[b,]<-sample(x,size=n,replace=TRUE)
    return(X)
}
```

Question c

For our dataset, the sample median is

```
mhat<-median(x)
print(mhat)
```

```
## [1] 1.358023
```

To estimate the standard error of $\widehat{m}$, we generate B bootstrap samples and we compute the standard deviation of the bootstrap estimates:

```
B<- 1000
X<-bootstrap(x,B)  # generation of B bootstrap samples
med<-apply(X,1,median)  # calculation of the B bootstrap estimates
se_B<- sd(med)  # calculation of the standard deviation
print(se_B)
```

```
## [1] 0.3053775
```

Question d

The bounds of the $1-\alpha$ bootstrap percentile confidence interval are simply the $\alpha/2$ and $1-\alpha/2$ empirical quantiles of the B bootstrap estimates:

```
alpha<- 0.05
CI1<-quantile(med,c(alpha/2,1-alpha/2))
print(CI1)
```

```
##     2.5%    97.5%
## 0.861640 2.076756
```

We note that we have used only $B = 1000$ bootstrap datasets to avoid lengthy calculations, but it would be desirable to use $B \geq 10^4$.

Question e

To compute bootstrap estimates of the standard error of $\widehat{m}$, we need to bootstrap each bootstrap sample. We will use only $B_1 = 50$ pseudo-datasets in the inner bootstrap loop:

```
B1 <- 50
se<-vector("numeric",B)
for(b in 1:B){
    X1<- bootstrap(X[b,],B1)  # Bootstrapping bootstrap sample b
    med_star<-apply(X1,1,median)
    se[b]<- sd(med_star)
    }
```

We can then compute the quantiles of the approximately pivotal quantity

$$Z^* = \frac{\widehat{m}^* - \widehat{m}}{\widehat{\mathsf{se}}^*}$$

```
q<-quantile((med-mhat)/se,c(alpha/2,1-alpha/2))
CI2<- c(mhat-q[2]*se_B,mhat-q[1]*se_B)
print(CI2)
```

```
##     97.5%      2.5%
## 0.6129133 2.1892261
```

Question f

To estimate the coverage probabilities of the two confidence intervals above, we will generate $N = 100$ datasets and compute the corresponding confidence intervals. The coverage probability is estimated by the

proportion of intervals containing the true value of the parameter. We give 95% confidence intervals on coverage probabilities using function `binom.confint` in package `binom`.

```
N<-100
k1<-0
k2<-0
for(i in 1:N){
  x<-rexp(n,rate=rate)  # Sample generation
  X<-bootstrap(x,B)  # bootstrapping
  med<-apply(X,1,median)  # computation of the  estimates
  se_B<- sd(med)
  CI1<-quantile(med,c(alpha/2,1-alpha/2))  # percentile confidence interval
  # k1 is incremented if the CI contains the true median
  if((CI1[1]<=mtrue)&(mtrue<=CI1[2])) k1<-k1+1
  mhat<-median(x)
  for(b in 1:B){
    X1<- bootstrap(X[b,],B1)  # Bootstrapping bootstrap sample b
    med_star<-apply(X1,1,median)
    se[b]<- sd(med_star)
  }
  q<-quantile((med-mhat)/se,c(alpha/2,1-alpha/2))
  CI2<- c(mhat-q[2]*se_B,mhat-q[1]*se_B)  # Bootstrap t interval
  if((CI2[1]<=mtrue)&(mtrue<=CI2[2])) k2<-k2+1
}
library(binom)
binom.confint(x=c(k1,k2),n=N,method="exact")
```

```
##   method  x   n mean     lower     upper
## 1  exact 94 100 0.94 0.8739701 0.9776651
## 2  exact 91 100 0.91 0.8360177 0.9580164
```

Question g

To use parametric bootstrap, we change function `bootstrap` to

```
bootstrap_param <- function(x,B){
  n <- length(x)
  X <- matrix(rexp(B*n,rate=1/mean(x)),B,n)
  return(X)
}
```

We then repeat the previous steps:

We generate B bootstrap samples and we compute the standard deviation of the bootstrap estimates:

```
B<- 1000
X<-bootstrap_param(x,B)  # generation of B bootstrap samples
med<-apply(X,1,median)  # calculation of the B bootstrap estimates
se_B<- sd(med)  # calculation of the standard deviation
print(se_B)
```

```
## [1] 0.329218
```

Bootstrap percentile interval:

```
alpha<- 0.05
CI1<-quantile(med,c(alpha/2,1-alpha/2))
```

```r
print(CI1)
```

```
##     2.5%    97.5%
## 1.082314 2.344471
```

Bootstrap t interval:

```r
B1<-50
se<-vector("numeric",B)
for(b in 1:B){
    X1<- bootstrap_param(X[b,],B1)  # Bootstrapping bootstrap sample b
    med_star<-apply(X1,1,median)
    se[b]<- sd(med_star)
    }
q<-quantile((med-mhat)/se,c(alpha/2,1-alpha/2))
CI2<- c(mhat-q[2]*se_B,mhat-q[1]*se_B)
print(CI2)
```

```
##    97.5%     2.5%
## 1.067364 2.336239
```

Estimation of coverage probabilities:

```r
N<-100
k1<-0
k2<-0
for(i in 1:N){
  x<-rexp(n,rate=rate)  # Sample generation
  X<-bootstrap_param(x,B)  # bootstrapping
  med<-apply(X,1,median)  # computation of the  estimates
  se_B<- sd(med)
  CI1<-quantile(med,c(alpha/2,1-alpha/2))  # percentile confidence interval
  # k1 is incremented if the CI contains the true median
  if((CI1[1]<=mtrue) & (mtrue<=CI1[2])) k1 <- k1+1
  mhat<-median(x)
  for(b in 1:B){
    X1<- bootstrap_param(X[b,],B1)  # Bootstrapping bootstrap sample b
    med_star<-apply(X1,1,median)
    se[b]<- sd(med_star)
  }
  q<-quantile((med-mhat)/se,c(alpha/2,1-alpha/2))
  CI2<- c(mhat-q[2]*se_B,mhat-q[1]*se_B)  # Bootstrap t interval
  if((CI2[1]<=mtrue)&(mtrue<=CI2[2])) k2 <- k2+1
}
binom.confint(x=c(k1,k2),n=N,method="exact")
```

```
##   method  x   n mean     lower     upper
## 1  exact 99 100 0.99 0.9455406 0.9997469
## 2  exact 73 100 0.73 0.6319837 0.8139336
```

Exercise 2

Question a

We first read and transform the data, and create a new data frame:

```
data <- read.table("/Users/Thierry/Documents/R/Data/Compstat/investment.txt", header=TRUE)
y<- data$Invest/(data$CPI*10)
time<- 1:15
GNP1<-data$GNP/(data$CPI*10)
data1<-data.frame(Invest=y,time, GNP=GNP1,data$Interest,data$Inflation)
```

We then perform linear regression on this dataset using function `lm`, and compute 95% confidence intervals on the coefficients under the assumptions of the standard linear regression model:

```
reg<- lm(Invest~.,data=data1)
CI.norm<-confint(reg, level = 0.95)
print(CI.norm,2)
```

```
##                   2.5 %   97.5 %
## (Intercept)    -0.6292 -0.38890
## time           -0.0209 -0.01229
## GNP             0.5504  0.79018
## data.Interest  -0.0051  0.00023
## data.Inflation -0.0029  0.00300
```

Question b

We first load package `boot`:

```
library("boot")
```

We then write the function for the calculation of the bootstrapped statistics, to be provided to function `boot`. The first argument of this function must be the dataset, and the second argument must be a vector of indices of any bootstrap sample for which the statistics will be computed:

```
reg_invest<-function(data1,ii){
  reg<- lm(Invest~.,data=data1[ii,])
  return(reg$coefficients)
}
```

We then call function `boot` and compute the percentile and BC_a confidence intervals using function `boot.ci`:

```
bt<- boot(data1, reg_invest, R=1000)
CI.cases.perc<-matrix(0,5,2)
CI.cases.bca<-matrix(0,5,2)
for(j in 1:5){
  CI<-boot.ci(bt, conf = 0.95, index=j,type = c("perc","bca"))
  CI.cases.perc[j,]=CI$percent[4:5]
  CI.cases.bca[j,]=CI$bca[4:5]
}
```

We print the two sets of confidence intervals side by side:

```
print(cbind(CI.cases.perc,CI.cases.bca),2)
```

```
##         [,1]     [,2]    [,3]    [,4]
## [1,] -0.6066 -0.33925 -0.6528 -0.3835
## [2,] -0.0206 -0.00999 -0.0214 -0.0115
## [3,]  0.5003  0.76882  0.5307  0.8051
## [4,] -0.0068  0.00016 -0.0054  0.0028
## [5,] -0.0039  0.00577 -0.0045  0.0048
```

Question c

We first compute the residuals:

```
fit<- fitted(reg)
residuals<-residuals(reg)
```

We put the design matrix in a matrix `X` that will be useful later:

```
X<-model.matrix(reg)
```

The following function computes the statistics that will be passed to function `boot`:

```
reg_invest1<-function(residuals,ii,fit,X){
  y<-fit+residuals[ii]  # construction of pseudo-responses
  reg<- lm(y~X-1) # regression on the pseudo responses
  return(reg$coefficients)
}
```

Finally, we compute the bootstrap percentile and BC_a confidence intervals by passing function `reg_invest1` to `boot` and then using function `boot.ci`:

```
bt1<- boot(residuals, reg_invest1, R=1000,fit=fit,X=X)
CI.res.perc<-matrix(0,5,2)
CI.res.bca<-matrix(0,5,2)
for(j in 1:5){
  CI<-boot.ci(bt1, conf = 0.95, index=j,type = c("perc","bca"))
  CI.res.perc[j,]=CI$percent[4:5]
  CI.res.bca[j,]=CI$bca[4:5]
}
```

We print the two sets of confidence intervals side by side:

```
print(cbind(CI.res.perc,CI.res.bca),2)
```

```
##         [,1]     [,2]    [,3]     [,4]
## [1,] -0.5977 -0.43086 -0.5976 -0.43055
## [2,] -0.0198 -0.01386 -0.0197 -0.01372
## [3,]  0.5942  0.75979  0.5933  0.75915
## [4,] -0.0042 -0.00064 -0.0041 -0.00055
## [5,] -0.0021  0.00212 -0.0020  0.00219
```

Exercise 3

Question a

```
data <- read.table("/Users/Thierry/Documents/R/Data/Compstat/wages.txt",
                   header=TRUE)
attach(data)
plot(output,wage)
```