statComp_hw5

凌浩东

11/27/2021

7.3

Obtain a bootstrap t confidence interval estimate for the correlation statistic in Example 7.2

```
set.seed(42)
boot.t.ci <- function(x, B = 500, R = 100, level = .95, statistic){
  #compute the bootstrap t CI
  x <- as.matrix(x); n <- nrow(x)
  stat <- numeric(B); se <- numeric(B)
  boot.se <- function(x, R, f) {
    #local function to compute the bootstrap
    #estimate of standard error for statistic f(x)
    x <- as.matrix(x); m <- nrow(x)
    th <- replicate(R, expr = {
      i <- sample(1:m, size = m, replace = TRUE)
      f(x[i, ])
    })
    return(sd(th))
  }
  for (b in 1:B) {
    j <- sample(1:n, size = n, replace = TRUE)
    y <- x[j, ]
    stat[b] <- statistic(y)
    se[b] <- boot.se(y, R = R, f = statistic)
  }
  stat0 <- statistic(x)
  t.stats <- (stat - stat0) / se
  se0 <- sd(stat)
  alpha <- 1 - level
  Qt <- quantile(t.stats, c(alpha/2, 1-alpha/2), type = 1)
  names(Qt) <- rev(names(Qt))
  CI <- rev(stat0 - Qt * se0)
}

data <- cbind(law$LSAT, law$GPA)
stat <- function(data) {
  cor(data[,1], data[,2])
}
ci <- boot.t.ci(data, statistic = stat, B = 2000, R = 200)
print(ci)
```

```
##       2.5%      97.5%
## -0.2959033  0.9939044
```

The bootstrap t confidence interval estimate with 5% level for correlation is $[-0.296, 0.994]$.

7.4

Refer to the air-conditioning data set aircondit provided in the boot package. The 12 observations are the times in hours between failures of airconditioning equipment. Assume that the times between failures follow an exponential model $Exp(\lambda)$. Obtain the MLE of the hazard rate $lambda$ and use bootstrap to estimate the bias and standard error of the estimate.

$$\mathcal{L} = \prod_{i=1}^{n} \lambda exp\left(-\lambda x_i\right)$$

$$l = log(\mathcal{L}) = -\lambda \sum_{i=1}^{n} x_i + nlog(\lambda)$$

$$\frac{\partial l}{\partial \lambda} = -\sum_{i=1}^{n} x_i + \frac{n}{\lambda} = 0$$

$$\Rightarrow \hat{\lambda} = \frac{n}{\sum_{i=1}^{n} x_i}$$

```
data <- aircondit$hours
lambda.hat <- length(data) / sum(data)
lambda.hat
```

```
## [1] 0.00925212
```

thus MLE for λ is 0.009.

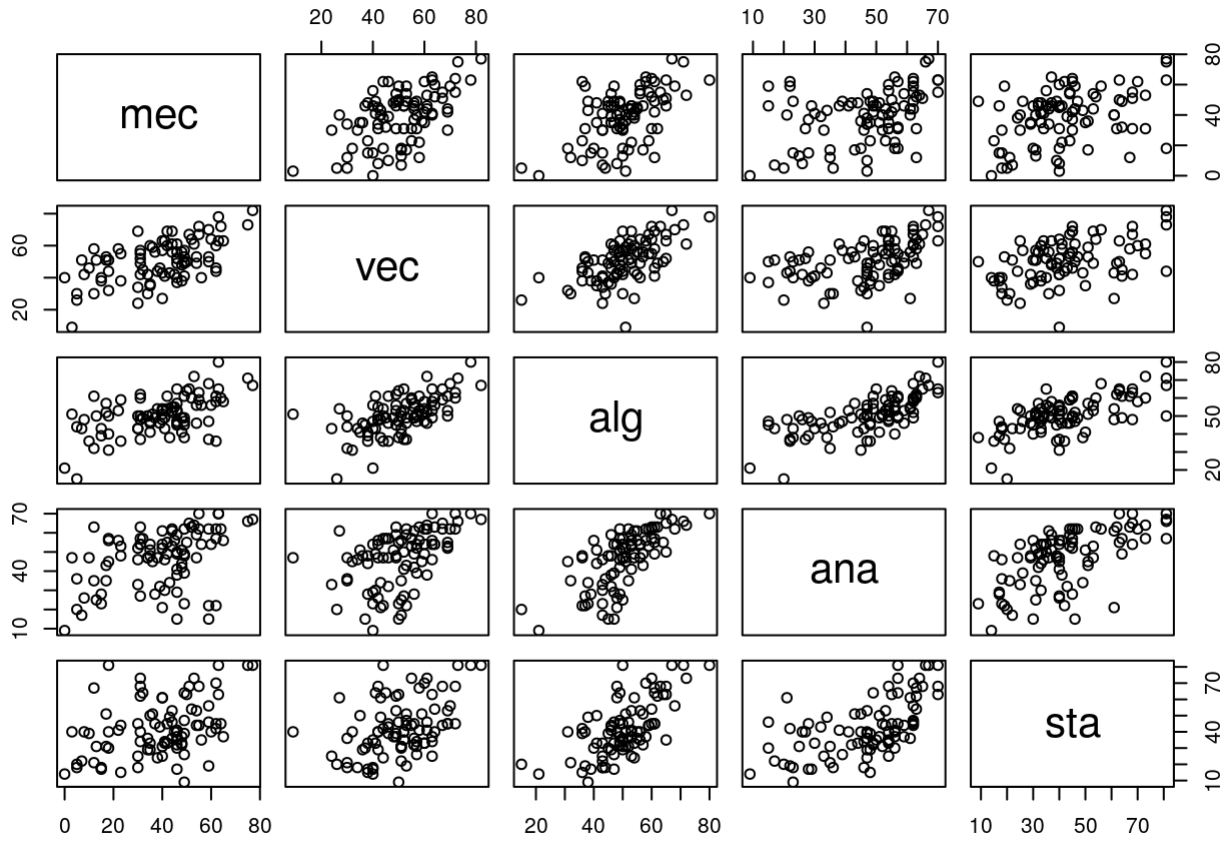
```
set.seed(42)
stat <- function(data, i) {
  length(data[i]) / sum(data[i])
}
boot(data, statistic = stat, R=2000)
```

```
##
## ORDINARY NONPARAMETRIC BOOTSTRAP
##
##
## Call:
## boot(data = data, statistic = stat, R = 2000)
##
##
## Bootstrap Statistics :
##       original      bias    std. error
## t1* 0.00925212 0.001329367 0.004256292
```

The bias and the standard error of the estimates are 0.0013 and 0.0043.

7.6

```
data <- scor
pairs(data)
```




```
set.seed(42)
r <- function(x, i) {
  cor(x[i,1], x[i,2])
}
boot(cbind(data$mec, data$vec), statistic = r, R = 2000)
boot(cbind(data$vec, data$alg), statistic = r, R = 2000)
boot(cbind(data$alg, data$ana), statistic = r, R = 2000)
boot(cbind(data$ana, data$sta), statistic = r, R = 2000)
```

The standard errors for the estimators are listed below:

correlation	std.error
$\hat{\rho}_{12}$	0.07492908
$\hat{\rho}_{23}$	0.06885137
$\hat{\rho}_{34}$	0.04895375
$\hat{\rho}_{45}$	0.06783836

7.7

```
set.seed(42)
stat <- function(x, i) {
  sigma.hat <- cov(x[i,])
  lambda.hat <- eigen(sigma.hat)$values
  lambda.hat[1] / sum(lambda.hat)
}
boot(data, statistic = stat, R = 2000)
```

```
##
## ORDINARY NONPARAMETRIC BOOTSTRAP
##
##
## Call:
## boot(data = data, statistic = stat, R = 2000)
##
##
## Bootstrap Statistics :
##     original      bias    std. error
## t1* 0.619115 0.002337701  0.04788141
```

The bias and standard errors are 0.002337701 and 0.04788141.

7.A

```
set.seed(42)
stat <- function(x, i) {
  mean(x[i])
}

myFun <- function(generate.Fun, stat, theta){
  n <- 100 # number of data
  m <- 100 # number of Monte Carlo replicates
  alpha <- c(.025, .975)
  std.norm.ci.left.miss <- 0
  std.norm.ci.right.miss <- 0
  basic.ci.left.miss <- 0
  basic.ci.right.miss <- 0
  percentile.ci.left.miss <- 0
  percentile.ci.right.miss <- 0

  for (i in 1:m){
    if (generate.Fun == "rnorm")
      data <- rnorm(n)
    if (generate.Fun == "rchisq")
      data <- rchisq(n, df=5)
    boot.obj <- boot(data, statistic = stat, R=2000)
    std.norm.ci.left <- boot.obj$t0 + qnorm(alpha[1]) * sd(boot.obj$t)
    std.norm.ci.right <- boot.obj$t0 + qnorm(alpha[2]) * sd(boot.obj$t)
    basic.ci.left <- 2*boot.obj$t0 -quantile(boot.obj$t, alpha[2], type=1)
    basic.ci.right <- 2*boot.obj$t0 -quantile(boot.obj$t, alpha[1], type=1)
    percentile.ci.left <- quantile(boot.obj$t, alpha[1], type=6)
    percentile.ci.right <- quantile(boot.obj$t, alpha[2], type=6)
    if ( std.norm.ci.left > theta)
      std.norm.ci.left.miss <- std.norm.ci.left.miss + 1
    if ( std.norm.ci.right < theta)
      std.norm.ci.right.miss <- std.norm.ci.right.miss + 1
    if ( basic.ci.left > theta)
      basic.ci.left.miss <- basic.ci.left.miss + 1
    if ( basic.ci.right < theta)
      basic.ci.right.miss <- basic.ci.right.miss + 1
    if ( percentile.ci.left > theta)
      percentile.ci.left.miss <- percentile.ci.left.miss + 1
    if (percentile.ci.right < theta)
      percentile.ci.right.miss <- percentile.ci.right.miss + 1
  }

  coverage.prob.norm <- 1 - std.norm.ci.left.miss/m - std.norm.ci.right.miss/m
  coverage.prob.basic <- 1 - basic.ci.left.miss/m - basic.ci.right.miss/m
  coverage.prob.percentile <- 1 - percentile.ci.left.miss/m - percentile.ci.right.miss/m

  output <- matrix(c(coverage.prob.norm, std.norm.ci.left.miss/m, std.norm.ci.right.mis
s/m,
         coverage.prob.basic, basic.ci.left.miss/m, basic.ci.right.miss/m,
         coverage.prob.percentile, percentile.ci.left.miss/m, percentile.ci.right.miss/
m),
         nrow = 3)
  output <- as.data.frame(output, row.names = c("coverage rate", "miss on the left", "mi
```

```
ss on the right"))
  names(output) <- c("normal","basic","percentile")
  output
}
myFun("rnorm", stat, 0)
```

```
##                   normal basic percentile
## coverage rate       0.94  0.95       0.95
## miss on the left    0.03  0.03       0.03
## miss on the right   0.03  0.02       0.02
```

7.B

```
set.seed(42)
sk <- function(x, i) {
  #computes the sample skewness coeff.
  xbar <- mean(x[i])
  m3 <- mean((x[i] - xbar)^3)
  m2 <- mean((x[i] - xbar)^2)
  return( m3 / m2^1.5 )
}

myFun("rnorm", sk, 0)
```

```
##                   normal basic percentile
## coverage rate       0.90  0.89       0.91
## miss on the left    0.04  0.05       0.04
## miss on the right   0.06  0.06       0.05
```

For normal populations, result shown above.

For chi square distributions, result shown below.

```
set.seed(42)
myFun("rchisq", sk, sqrt(8/5))
```

```
##                   normal basic percentile
## coverage rate       0.81  0.75        0.8
## miss on the left    0.01  0.09        0.0
## miss on the right   0.18  0.16        0.2
```