hw05

陆靖磊 22300680221

2024-11-17

8.4

```r
library(boot)

data <- aircondit[, 1]

# 定义计算 MLE 的函数
calculate_mle <- function(data, indices) {
  mean_value <- mean(data[indices])
  mle_estimate <- 1 / mean_value
  return(mle_estimate)
}

# 执行 Bootstrap 过程
bootstrap_results <- boot(
  data = data,
  statistic = calculate_mle,
  R = 1000
)

print(bootstrap_results)
```

```
##
## ORDINARY NONPARAMETRIC BOOTSTRAP
##
##
## Call:
## boot(data = data, statistic = calculate_mle, R = 1000)
##
##
## Bootstrap Statistics :
##        original       bias    std. error
## t1* 0.00925212 0.001416123 0.004538571
```

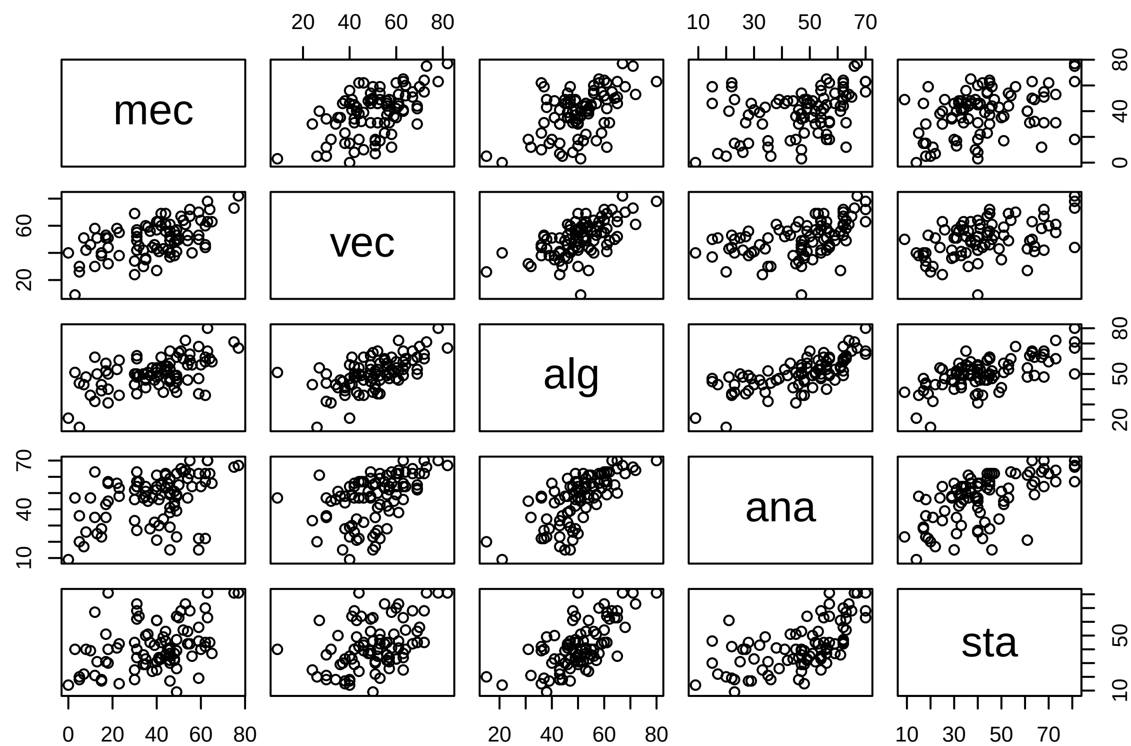
```r
detach(package:boot)
```

8.6

```r
# 获取数据、计算相关系数矩阵、绘制散点图
library(bootstrap)
attach(scor)
cor(scor)
```

```
##             mec       vec       alg       ana       sta
## mec 1.0000000 0.5534052 0.5467511 0.4093920 0.3890993
## vec 0.5534052 1.0000000 0.6096447 0.4850813 0.4364487
## alg 0.5467511 0.6096447 1.0000000 0.7108059 0.6647357
## ana 0.4093920 0.4850813 0.7108059 1.0000000 0.6071743
## sta 0.3890993 0.4364487 0.6647357 0.6071743 1.0000000
```

```r
pairs(scor)
```




```r
# 分别用 bootstrap 方法估计四个相关系数
library(boot)
cor.stat <- function(x, i = 1:NROW(x)) {
    return(cor(x[i, 1], x[i, 2]))
}
x <- as.matrix(scor)
boot(x[, c(1,2)], statistic = cor.stat, R = 1000)
```

```
##
## ORDINARY NONPARAMETRIC BOOTSTRAP
##
##
## Call:
## boot(data = x[, c(1, 2)], statistic = cor.stat, R = 1000)
##
##
```

```
## Bootstrap Statistics :
##      original       bias    std. error
## t1* 0.5534052 -0.002484258  0.07470495
```

```r
boot(x[, c(3,4)], statistic = cor.stat, R = 1000)
```

```
##
## ORDINARY NONPARAMETRIC BOOTSTRAP
##
##
## Call:
## boot(data = x[, c(3, 4)], statistic = cor.stat, R = 1000)
##
##
## Bootstrap Statistics :
##      original       bias    std. error
## t1* 0.7108059 -0.002422725  0.04998886
```

```r
boot(x[, c(3,5)], statistic = cor.stat, R = 1000)
```

```
##
## ORDINARY NONPARAMETRIC BOOTSTRAP
##
##
## Call:
## boot(data = x[, c(3, 5)], statistic = cor.stat, R = 1000)
##
##
## Bootstrap Statistics :
##      original       bias    std. error
## t1* 0.6647357 -0.002508565  0.05880366
```

```r
boot(x[, c(4,5)], statistic = cor.stat, R = 1000)
```

```
##
## ORDINARY NONPARAMETRIC BOOTSTRAP
##
##
## Call:
## boot(data = x[, c(4, 5)], statistic = cor.stat, R = 1000)
##
##
## Bootstrap Statistics :
##      original      bias    std. error
## t1* 0.6071743 -0.00010939  0.06964707
```

```r
detach(scor)
detach(package:bootstrap)
detach(package:boot)
```

8.7

```r
library(bootstrap)
attach(scor)
x <- cov(as.matrix(scor))
e <- eigen(x)
lambda <- e$values
lambda_1_hat=lambda[1]
theta_hat=lambda_1_hat/sum(lambda)
theta_hat
```

```
## [1] 0.619115
```

```r
theta <- function(x, i) {
    y <- as.matrix(x[i, ])
    s <- cov(y)
    e <- eigen(s)
    lambda <- e$values
    max(lambda/sum(lambda))
}
library(boot)
boot(scor, statistic = theta, R = 1000)
```

```
##
## ORDINARY NONPARAMETRIC BOOTSTRAP
##
##
## Call:
## boot(data = scor, statistic = theta, R = 1000)
##
##
## Bootstrap Statistics :
##     original      bias    std. error
## t1* 0.619115 0.004685758  0.04762814
```

估计量为 0.619，偏差为 0.0018，标准误为 0.047

8.A

```r
set.seed(123)

# 设定参数
n <- 30  # sample size
mu <- 0  # true mean
sigma <- 1  # true standard deviation
B <- 1000  # number of bootstrap samples
M <- 1000  # number of Monte Carlo simulations
alpha <- 0.05  # significance level
coverage_normal <- 0
coverage_basic <- 0
coverage_percentile <- 0

for (i in 1:M) {
  # Generate a sample
```

```r
  sample <- rnorm(n, mean = mu, sd = sigma)
  sample_mean <- mean(sample)

  # Bootstrap resampling
  bootstrap_means <- replicate(B, mean(sample(sample, replace = TRUE)))

  # Standard normal bootstrap CI
  se_boot <- sd(bootstrap_means)
  ci_normal <- c(sample_mean - qnorm(1 - alpha/2) * se_boot,
                 sample_mean + qnorm(1 - alpha/2) * se_boot)

  # Basic bootstrap CI
  ci_basic <- 2 * sample_mean - quantile(bootstrap_means, probs = c(1 - alpha/2, alpha/2))

  # Percentile bootstrap CI
  ci_percentile <- quantile(bootstrap_means, probs = c(alpha/2, 1 - alpha/2))

  # Check coverage
  coverage_normal <- coverage_normal + (ci_normal[1] <= mu & mu <= ci_normal[2])
  coverage_basic <- coverage_basic + (ci_basic[1] <= mu & mu <= ci_basic[2])
  coverage_percentile <- coverage_percentile + (ci_percentile[1] <= mu & mu <= ci_percentile[2])
}

# Calculate proportions
coverage_normal <- coverage_normal / M
coverage_basic <- coverage_basic / M
coverage_percentile <- coverage_percentile / M

cat("Coverage Probability (Normal):", coverage_normal, "\n")
```

```
## Coverage Probability (Normal): 0.936
```

```r
cat("Coverage Probability (Basic):", coverage_basic, "\n")
```

```
## Coverage Probability (Basic): 0.938
```

```r
cat("Coverage Probability (Percentile):", coverage_percentile, "\n")
```

```
## Coverage Probability (Percentile): 0.941
```

8.B

```r
library(e1071)

calc_skewness <- function(x) {
  skewness(x)
}

coverage_normal_skew <- 0
coverage_chi_skew <- 0
```

```r
for (i in 1:M) {
  # Generate normal sample
  sample_normal <- rnorm(n, mean = mu, sd = sigma)
  skew_normal <- calc_skewness(sample_normal)

  # Bootstrap resampling for normal
  bootstrap_skew_normal <- replicate(B, calc_skewness(sample(sample_normal, replace = TRUE)))
  ci_percentile_normal <- quantile(bootstrap_skew_normal, probs = c(alpha/2, 1 - alpha/2))
  coverage_normal_skew <- coverage_normal_skew + (ci_percentile_normal[1] <= 0 & 0 <= ci_percentile_norm

  # Generate chi-squared sample
  sample_chi <- rchisq(n, df = 5)
  skew_chi <- calc_skewness(sample_chi)

  # Bootstrap resampling for chi-squared
  bootstrap_skew_chi <- replicate(B, calc_skewness(sample(sample_chi, replace = TRUE)))
  ci_percentile_chi <- quantile(bootstrap_skew_chi, probs = c(alpha/2, 1 - alpha/2))
  coverage_chi_skew <- coverage_chi_skew + (ci_percentile_chi[1] <= skew_chi & skew_chi <= ci_percentile
}

# Calculate proportions
coverage_normal_skew <- coverage_normal_skew / M
coverage_chi_skew <- coverage_chi_skew / M

cat("Coverage Probability for Skewness (Normal):", coverage_normal_skew, "\n")
```

```
## Coverage Probability for Skewness (Normal): 0.951
```

```r
cat("Coverage Probability for Skewness (Chi-squared):", coverage_chi_skew, "\n")
```

```
## Coverage Probability for Skewness (Chi-squared): 1
```