STAT 428: Homework 5: Chapter 7: Jackknife and Bootstrap

Du, Yuting, yutingd3

Table of Contents

Please refer to the [**detailed homework policy document**] on Course Page for information about homework formatting, submission, and grading.

Exercise 1

Bootstrap (Normal).

Perform the following tasks:

1. Generate a sample of size 100 from `N(10,1)` distribution.

```
n = 100
x = rnorm(n, 10, 1)
```

2. Estimate the mean of observations in this sample.

```
m = mean(x)
```

3. Use Bootstrap to estimate the standard error of this mean.

```
B = 10000
Tboot <- numeric(B)
for (b in 1 : B) {
  xb <- sample(x, n, replace=TRUE)
  Tboot[b] <- mean(xb)
```

```
}
(se <- sd(Tboot))

## [1] 0.1058
```

4. What should be (theoretically) and is (practically according to your experiment) the relation between this Bootstrap estimate and the actual standard deviation of the distribution you sampled from?

```
norm_sd <- 1 / 5
(se_theory <- norm_sd / sqrt(n))

## [1] 0.02
```

5. Calculate a 95% confidence interval for the estimate of mean (based on bootstrap).

```
zval <- qnorm(.975)
(lower <- m - zval * se)

## [1] 9.789

(upper <- m + zval * se)

## [1] 10.2
```

Exercise 2

Jackknife (Normal).

Consider the same sample that you generated in Exercise 1.1 and the same estimator you used in Exercise 1.2. Find the jackknife estimate of the bias of this estimator. (You may choose to guess the answer instead of performing the jackknife procedure, in that case EXPLAIN clearly why your guess makes sense.)

```
## [1] -0.0002688  0.0228957
```

The bias is about -0.0002536.

Exercise 3

Bootstrap and Jackknife Error Comparison Consider the following (simulated) months until various batteries of the same type burn out:

```
Ex3 <- c(2.228, 2.051, 1.683, 3.285, 1.219, 2.879, 2.976, 2.112, 2.357,
2.425, 1.255, 2.562, 0.829, 2.581, 2.340, 3.043, 0.684, 1.810, 2.529,
0.700)
```

1. We want to estimate the probability that these batteries burn out in less than 2 months, so $\theta = P(X < 2)$. Calculate $\hat{\theta}$.

```
count = 0
for(val in Ex3){
  if(val < 2) count = count + 1
```

```
}

theta.hat = count / length(Ex3)
theta.hat

## [1] 0.35
```

2. What is the standard error and bias of your estimator? Estimate the standard error and bias using bootstrapping.

```
library(SimDesign)
B = 10000
Tboot <- numeric(B)
for (b in 1 : B) {
  xb <- sample(Ex3, length(Ex3), replace=TRUE)

  count = 0
  for(val in xb){
    if(val < 2) count = count + 1
  }

  Tboot[b] <- count / length(Ex3)
}
(se <- sd(Tboot))

## [1] 0.1056

(bs <- bias(Tboot))

## [1] 0.3483
```

3. Calculate the standard error and bias using Jackknife.

```
library(SimDesign)
B = 10000
x <- numeric()
for (b in 1 : B) {
  x[b] <- sample(Ex3, length(Ex3), replace=TRUE)#the data
}

theta.hat = mean(x<2)#the estimate
theta.hat.jack = numeric()

for(i in 1: B){
  xi = x[-i]
  theta.hat.jack[i] = mean(xi<2)
}

#Calculate standard error
sumsq=sum((theta.hat.jack-mean(theta.hat.jack))^2)
sqrt((n-1)/n)*sqrt(sumsq)
```

```
## [1] 0.004732

#Calculate bias
(n-1)*(mean(theta.hat.jack)-theta.hat)

## [1] 0
```

4. What if you wanted to calculate the median time to burn out instead? Estimate the median, the standard error of the median estimator, and the bias of the median estimator using a resampling method.

```
library(SimDesign)
B = 10000
Tboot <- numeric()
for (b in 1 : B) {
  xb <- sample(Ex3, length(Ex3), replace=TRUE)
  Tboot[b] <- median(xb)
}
(se <- sd(Tboot))

## [1] 0.2086

(bs <- bias(Tboot))

## [1] 2.231
```

Exercise 4

Consider the data stored in cw on the weight of 30 chicks after eating two different diets for ten days. Chicks 1-20 had diet 1, and chicks 21-30 had diet 2.

```
cw<-ChickWeight[ChickWeight$Time==10&ChickWeight$Diet %in% c("1","2"),]
cw<-droplevels(cw)
```

Using a permutation test, determine whether the mean weights of the chicks on the two diets are different. Clearly show your steps, with at least comments to explain what you are doing (you should be doing this anyway, but just a reminder). What do you conclude?

```
x = cw["weight"][cw["Diet"] == 1]
y = cw["weight"][cw["Diet"] == 2]

# First we find the p-value for the two-sample t statistic by referring
to thet-distribution with n+m-2 df
t.test(x,y,var.equal=TRUE)

##
##  Two Sample t-test
##
## data:  x and y
## t = -1.7, df = 27, p-value = 0.1
## alternative hypothesis: true difference in means is not equal to 0
```

```
## 95 percent confidence interval:
##  -33.998   3.103
## sample estimates:
## mean of x mean of y
##     93.05    108.50

# Now we'll let ϑˆ = |t| to test null hypothesis
# of equal mean against a two sided alternative,
# and use the randomization distribution for the
# p-value
x = c(cw["weight"][cw["Diet"] == 1])
y = c(cw["weight"][cw["Diet"] == 2])

B = 1000 #The number of bootstrap samples to take
z = c(x,y)
nu = 1:length(z)
reps=numeric(B)
t0=t.test(x,y,var.equal=FALSE)$statistic
for(i in 1:B){
  perm=sample(nu,size=length(x),replace=FALSE)
  x1=z[perm]
  y1=z[-perm]
  reps[i]=abs(t.test(x1,y1,var.equal=FALSE)$statistic)
}
mean(c(t0,reps)>=t0)

## [1] 1
```

So, the two means are quite similar.

Exercise 5

Do exercise 7.3 from the book.

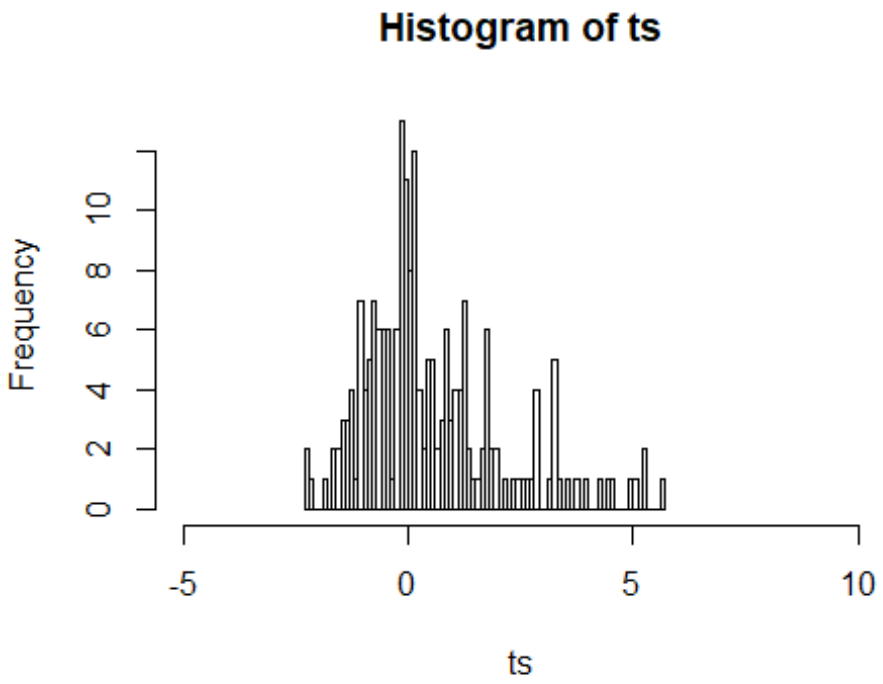
7.3 Obtain a bootstrap t confidence interval estimate for the correlation statistic in Example 7.2 (law data in bootstrap).

```
library("bootstrap")
B = 200
n = nrow(law)

theta.hat = cor(law$LSAT, law$GPA)
theta.hats.b = numeric(B)

ts = numeric(B)

for (b in 1:B) {
  i = sample(x = 1:n, size = n, replace = TRUE)
  law.b = law[i,]
  theta.hats.b[b] = cor(law.b$LSAT, law.b$GPA)
```

```r
  sd.theta.hats.b = numeric(B)

  for(b2 in 1:B) {
    i2 = sample(x = 1:n, size = n, replace = TRUE)
    law.b2 = law.b[i2,]
    sd.theta.hats.b[b2] = cor(law.b2$LSAT, law.b2$GPA)
  }

  se.b = sd(sd.theta.hats.b)

  ts[b] = (theta.hats.b[b] - theta.hat) / se.b
}

alpha = 0.05
ts.ordered = sort(ts)

qs = quantile(ts.ordered, probs = c(alpha/2, 1-alpha/2))

se.hat = sd(theta.hats.b)

(CI = c(theta.hat - qs[2]*se.hat, theta.hat - qs[1]*se.hat))
```

```
##  97.5%   2.5%
## 0.1269 1.0103
```

```r
hist(ts, breaks = 100, xlim = c(-5, 10))
```



Exercise 6

Do exercise 7.10 from the book.

7.10 In Example 7.18, leave-one-out (n-fold) cross validation was used to select the best fitting model. Repeat the analysis replacing the Log-Log model with a cubic polynomial model. Which of the four models is selected by the cross validation procedure? Which model is selected according to maximus adjusted R^2?

```
library(DAAG); attach(ironslag)

n <- length(magnetic) #in DAAG ironslag
e1 <- e2 <- e3 <- e4 <- numeric(n)
# for n-fold cross validation
# fit models on leave-one-out samples
for (k in 1:n) {
y <- magnetic[-k]
x <- chemical[-k]
J1 <- lm(y ~ x)
yhat1 <- J1$coef[1] + J1$coef[2] * chemical[k]
e1[k] <- magnetic[k] - yhat1

J2 <- lm(y ~ x + I(x^2) + I(x^3))
yhat2 <- J2$coef[1] + J2$coef[2] * chemical[k] +
J2$coef[3] * chemical[k]^2 + J2$coef[4] * chemical[k]^3
e2[k] <- magnetic[k] - yhat2

J3 <- lm(log(y) ~ x)
logyhat3 <- J3$coef[1] + J3$coef[2] * chemical[k]
yhat3 <- exp(logyhat3)
e3[k] <- magnetic[k] - yhat3

J4 <- lm(log(y) ~ log(x))
logyhat4 <- J4$coef[1] + J4$coef[2] * log(chemical[k])
yhat4 <- exp(logyhat4)
e4[k] <- magnetic[k] - yhat4
}

c(mean(e1^2), mean(e2^2), mean(e3^2), mean(e4^2))

## [1] 19.56 18.18 18.44 20.45
```

According to the estimates for prediction error,the cubic model is best fit for data. Then it's the exponential model, then the linear model. The Log-log model is the worst.

Exercise 7

Do exercise 7.11 from the book.

7.11 In Example 7.18, leave-one-out (n-fold) cross validation was used to select the best fitting model. Use leave-two-out cross validation to compare the models.

```
library(DAAG);
attach(ironslag)
n <- length(magnetic) #in DAAG ironslag
e1 <- e2 <- e3 <- e4 <- numeric(n*(n-1)) # 'leave two out' has n(n-1)
combinations

for (i in 1:n){
  for (j in i:n){
    if (i != j){
      y=magnetic[c(-i,-j)]
      x=chemical[c(-i,-j)]

      J1 <- lm(y ~ x)
      yhat11 <- J1$coef[1] + J1$coef[2] * chemical[i]
      yhat12 <- J1$coef[1] + J1$coef[2] * chemical[j]
      e1[(i-1)*n+j] <- sqrt((magnetic[i] - yhat11)^2+(magnetic[j] -
yhat12)^2)

      J2 <- lm(y ~ x + I(x^2))
      yhat21 <- J2$coef[1] + J2$coef[2] * chemical[i] +
      J2$coef[3] * chemical[i]^2
      yhat22 <- J2$coef[1] + J2$coef[2] * chemical[j] +
      J2$coef[3] * chemical[j]^2
      e2[(i-1)*n+j] <- sqrt((magnetic[i] - yhat21)^2+(magnetic[j] -
yhat22)^2)

      J3 <- lm(log(y) ~ x)
      logyhat31 <- J3$coef[1] + J3$coef[2] * chemical[i]
      logyhat32 <- J3$coef[1] + J3$coef[2] * chemical[j]
      yhat31 <- exp(logyhat31)
      yhat32 <- exp(logyhat32)
      e3[(i-1)*n+j] <- sqrt((magnetic[i] - yhat31)^2+(magnetic[j] -
yhat32)^2)

      J4 <- lm(log(y) ~ log(x))
      logyhat41 <- J4$coef[1] + J4$coef[2] * log(chemical[i])
      logyhat42 <- J4$coef[1] + J4$coef[2] * log(chemical[j])
      yhat41 <- exp(logyhat41)
      yhat42 <- exp(logyhat42)
      e4[(i-1)*n+j] <- sqrt((magnetic[i] - yhat41)^2+(magnetic[j] -
yhat42)^2)
    }
  }
}
# estimates for prediction error
c(mean(e1^2), mean(e2^2), mean(e3^2), mean(e4^2))
```

```
## [1] 19.57 17.87 18.45 20.47
```

According to the estimates for prediction error,the quadratic model is best fit for data. Then it's the exponential model, then the linear model. The Log-log model is the worst.

Exercise 8

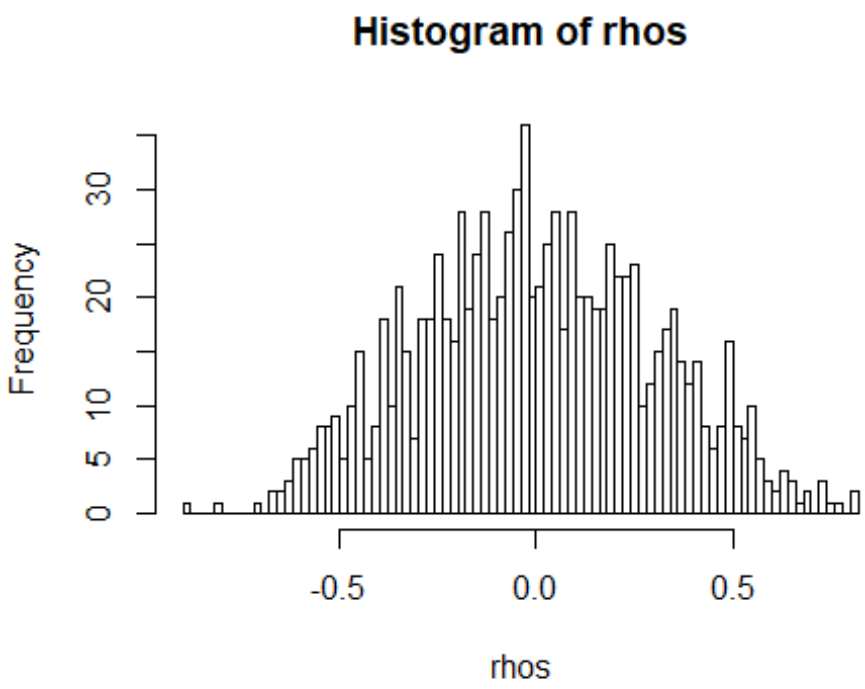
Permutation Test for Spearman Correlation Coefficient
You can test for the independence of two random variable using the Spearman Correlation coefficient r^2 (which relies on ranks rather than the actual values of X and Y) using the following test statistic:

$t = r\sqrt{\frac{n-2}{1-r^2}} \sim t_{n-2}$, or the Student's t-distribution with $n-2$ degrees of freedom under the null hypothesis (that X and Y are independent). Now, do exercise 8.2 in the book, using the t-statistic above. Use the `Petal.Length`, `Petal.Width`, `Sepal.Width`, and `Sepal.Length` variables from `iris` to test your function and compare it to `cor.test`, like mentioned in 8.2. (In other words, do pairwise tests).

8.2 Implement the bivariate Spearman rank correlation test for independence [255] as a permutation test. The Spearman rank correlation test statistic can be obtained from function `cor` with `method="spearman"`. Compare the achieved significance level of the permutation test with the p-value reported by `cor.test` on the same samples.

```
soybean = chickwts$weight[chickwts$feed=="soybean"]
linseed = chickwts$weight[chickwts$feed=="linseed"]
n = length(soybean)
m = length(linseed)

tmp = min(n, m)
soybean = sort(soybean[1:tmp])
linseed = sort(linseed[1:tmp])

zs = c(soybean, linseed)
spearman.cor.test = cor.test(x = soybean, y = linseed, method =
"spearman")

B = 1000
k = length(zs)

rhos = numeric(B)

for (b in 1:B) {
  i = sample(1:k, k/2, replace = FALSE)
  xs = zs[i]
  ys = zs[-i]
  rhos[b] = cor(x = xs, y = ys, method = "spearman")
}
```

```r
hist(rhos, breaks = 100)
```




```r
(theta.hat = spearman.cor.test$estimate)
```

```
## rho
##   1
```

```r
spearman.cor.test$p.value
```

```
## [1] 0
```

```r
(p.hat = mean(abs(rhos) > abs(theta.hat)))
```

```
## [1] 0
```

```r
(alpha = 0.05)
```

```
## [1] 0.05
```

```r
# p.hat < alpha, thus H0 rejected.
```