ps8\_rmd

Jared Yu

March 26, 2020

### Problem 1  
cvdataX <- scan(file.choose()) # Load variables  
cvdataY <- scan(file.choose())  
n <- length(cvdataX)  
  
# part (a)  
# Plot the (X,Y)  
plot(cvdataX, cvdataY, main = 'CVdata',  
 xlab = 'X', ylab = 'Y') # plot data  
mu\_hat <- mean(cvdataX) # MLE mu  
x\_seq <- seq(from = range(cvdataX)[1], to = range(cvdataX)[2],  
 length.out = 100) # Generate x range  
lines(x\_seq, # Plot fitted line  
 dnorm(x = x\_seq, mean = mu\_hat, sd = sqrt(2)),  
 col = 'blue')  
legend("topleft", legend = 'Fitted Line', lty = 1, col = 'blue')  
  
g <- function(x) { # g function  
 dnorm(x = x, mean = mu\_hat, sd = sqrt(2))  
 #(1 / (2 \* sqrt(pi))) \* exp((-1 / 4) \* (x - mu\_hat)^2)  
}  
R <- function(y, g) { # R function  
 abs(y - g)  
}  
  
sum(R(y = cvdataY, g = g(cvdataX))) / n # Apparent error  
  
# part (b)  
S1 <- cvdataX[1:50]; S2 <- cvdataX[51:100] # Initialize variables  
S1\_y <- cvdataY[1:50]; S2\_y <- cvdataY[51:100]  
xbar\_1 <- mean(S1); xbar\_2 <- mean(S2)  
  
g1 <- function(x, xbar = xbar\_1) { # Create g\_1X,Y(x)  
 (1 / (2 \* sqrt(pi))) \* exp((-1 / 4) \* (x - xbar)^2)  
}  
g2 <- function(x, xbar = xbar\_2) { # Create g\_2X,Y(x)  
 (1 / (2 \* sqrt(pi))) \* exp((-1 / 4) \* (x - xbar)^2)  
}  
  
(1 / n) \* sum(R(y = S2\_y, g = g1(x = S2)) +  
 R(y = S1\_y, g = g2(x = S1))) # Partitioned apparent error  
  
### Problem 3  
# part (b)  
jackknife <- scan(file.choose()) # Load data  
b2 <- function(Y) {  
 y\_bar <- mean(Y)  
 sum((Y - y\_bar)^4) /  
 ((sum((jackknife - y\_bar)^2))^2)  
}  
b2(Y = jackknife) # 0.02669995  
  
b2\_minus\_j <- function(R\_list, j = 0) {  
 if (j == 0) { # Remove jth group, if j=0 remove none of the groups  
 # Reference: https://stackoverflow.com/questions/1335830/why-cant-rs-ifelse-statements-return-vectors  
 Z <- R\_list  
 } else {  
 # Reference: https://stackoverflow.com/questions/652136/how-can-i-remove-an-element-from-a-list  
 Z <- R\_list[-j]  
 }  
  
 # Reference: https://stackoverflow.com/questions/14924935/using-r-convert-data-frame-to-simple-vector  
 # Calculate b2 with jth group removed  
 Z <- as.vector(unlist(Z), mode = 'numeric')  
 z\_bar <- mean(Z)  
 Z\_b2 <- sum((Z - z\_bar)^4) / ((sum((Z - z\_bar)^2))^2)  
   
 return(Z\_b2)  
}  
  
J <- function(R\_j) {  
 r <- length(R\_j)  
 b2\_bar <- (1 / r) \*  
 sum(sapply(1:r,  
 function(x) { b2\_minus\_j(R\_list = R\_j, j = x) }))  
 jackknifed\_stat <- r \* b2 - (r - 1) \* b2\_bar  
 return(jackknifed\_stat)  
}  
  
se\_jack <- function(Y = jackknife, # Vector of values  
 k = 1) { # Size of each group  
 n <- length(Y) # Initialize variables  
 r <- n / k  
   
 # Reference: https://stackoverflow.com/questions/3318333/split-a-vector-into-chunks-in-r  
 # Split Y into r groups R = (r\_1, r\_2, ..., r\_r)^T  
 r\_groups <- split(Y, cut(seq\_along(Y), r, labels = FALSE))  
   
 jackknifed\_T <- J(R\_j = r\_groups) # J(T)  
 T\_stat <- b2\_minus\_j(R\_list = r\_groups, j = 0) # T  
  
 numer <- sum( # numerator  
 sapply(1:r, function(x) {  
 T\_j\_star <- r \* T\_stat - (r - 1) \*  
 b2\_minus\_j(R\_list = r\_groups, j = x)  
 (T\_j\_star - jackknifed\_T)^2  
 })  
 )  
 denom <- (r \* (r - 1)) # denominator  
 se\_JT <- sqrt(numer / denom)  
  
 return(se\_JT)  
}  
  
se\_jack(Y = jackknife, k = 1) # 0.003714044  
se\_jack(Y = jackknife, k = 5) # 0.003692711  
  
# part (c)  
se\_jack(Y = jackknife, k = 10) # 0.003362023  
se\_jack(Y = jackknife, k = 20) # 0.004758237  
  
plot(density(jackknife))  
additional\_samples <- rnorm(n = 1e4, mean = 0, sd = 1)  
jackknife\_2 <- c(jackknife, additional\_samples)  
plot(density(jackknife\_2))  
b2(jackknife\_2) # 2.517476  
  
se\_jack(Y = jackknife\_2, k = 1) # 2.653616  
se\_jack(Y = jackknife\_2, k = 5) # 1.186968  
se\_jack(Y = jackknife\_2, k = 10) # 0.8395211  
se\_jack(Y = jackknife\_2, k = 20) # 0.5939255  
  
additional\_samples2 <- rnorm(n = 1e3, mean = 0, sd = 1)  
jackknife\_3 <- c(jackknife, additional\_samples2)  
plot(density(jackknife\_3))  
b2(jackknife\_3) # 0.2688279  
  
se\_jack(Y = jackknife\_3, k = 1) # 0.7922754  
se\_jack(Y = jackknife\_3, k = 5) # 0.3549629  
se\_jack(Y = jackknife\_3, k = 10) # 0.2515717  
se\_jack(Y = jackknife\_3, k = 20) # 0.1787098