Jared Yu

Computational Statistics

Problem Set 8

NOTE: The code for the assignment is at the end under the Appendix.

1. Posted on the course Blackboard site is the data set CVdata.txt containing 2 lists, X and Y, each of length 100. When written as an ordered pair, (x, y), x is a sample observation and $y = f(x)$, the observed value of the density at x. In this problem we will use balanced half-sampling to predict the error of fitting the curve (X, Y) with a normal density, $N(\mu, 2)$. As our error metric, we will use $R(y, g) = |y - g|$.
    a. Plot the (X, Y) and use the MLE $\hat{\mu}$ to fit $f(x)$ with a normal distribution. Clearly state the fitted function g and find the apparent error, $E_{\hat{P}_{Y|X}}$.

First $\hat{\mu}$ needs to be calculated.

$$X \sim \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left\{-\frac{(x-\mu)^2}{2\sigma^2}\right\}; \theta = (\mu, \sigma^2 = 2)^\top$$

$$L(\theta) = f(x_1, \cdots, x_n; \mu, \sigma^2) = \prod_{i=1}^{n} \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left\{-\frac{(x_i-\mu)^2}{2\sigma^2}\right\}$$

$$= \frac{1}{(2\pi\sigma^2)^{n/2}} \exp\left\{-\frac{\sum_{i=1}^{n}(x_i-\mu)^2}{2\sigma^2}\right\}$$

$$l(\theta) = \ln L(\theta) = -\frac{n}{2}\ln(2\pi\sigma^2) - \frac{\sum_{i=1}^{n}(x_i-\mu)^2}{2\sigma^2}$$

$$\frac{\partial l(\theta)}{\partial \mu} = \frac{\sum_{i=1}^{n}(x_i-\mu)}{\sigma^2} \stackrel{set\ to}{=} 0 \Rightarrow \sum_{i=1}^{n} x_i = n\mu \Rightarrow \hat{\mu} = \frac{\sum_{i=1}^{n} x_i}{n} = \bar{x}$$

Then, the function $g_{X,Y}(x)$ can be written as,

$$g_{X,Y}(x) = \frac{1}{2\sqrt{\pi}} \exp\left\{\frac{-(x-2)^2}{4}\right\}.$$

The plot of (X, Y) with the fitted line is below in Figure 1.

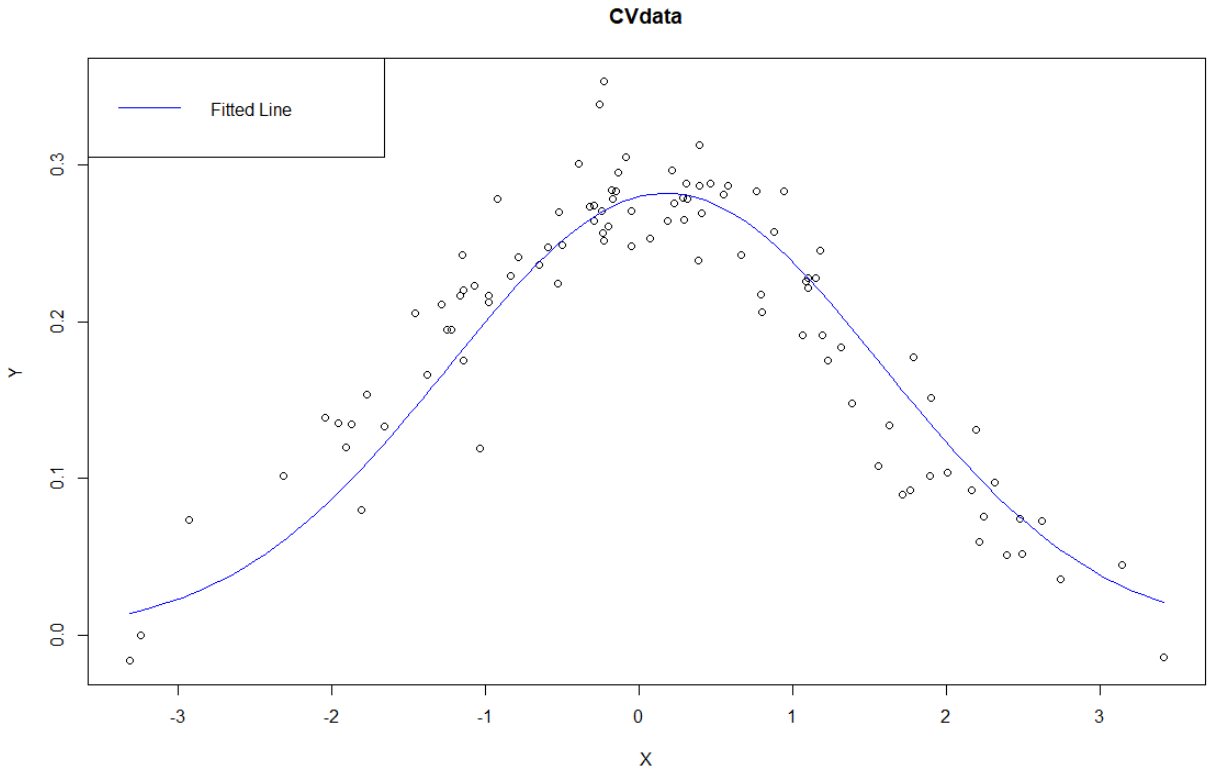

Figure 1 Plot of (X,Y) from CVdata. The blue line is the fitted line g.

Given the error metric, $R(y, g)$, then the apparent error can be written as follows,

$$E_{\hat{P}_{Y|X}}\left(R\left(Y_0, g_{X,Y}(x_0)\right)\right) = \frac{1}{n}\sum_{i=1}^{n} R\left(y_i, g_{X,Y}(x_i)\right) = \frac{1}{n}\sum_{i=1}^{n} |y_i - g_i|.$$

Using R, the apparent error is roughly $\boxed{0.0252.}$

b. Partition the data set into $S_1 = \{x_1, \cdots, x_{50}\}$ and $S_2 = \{x_{51}, \cdots, x_{100}\}$ and find $E_{\tilde{P}_{Y|X}}$ (Eq. 12.5 in Gentle).

The formula for $E_{\tilde{P}_{Y|X}}$ is as follows:

$$E_{\tilde{P}_{Y|X}}\left(R\left(Y_0, g_{X,Y}(X_0)\right)\right) = \frac{1}{n}\left(\sum_{i\in S_2} R\left(y_i, g_{1X,Y}(x_i)\right) + \sum_{i\in S_1} R\left(y_i, g_{2X,Y}(x_i)\right)\right),$$

where

$$g_{1X,Y}(x) = \frac{1}{2\sqrt{\pi}}\exp\left\{\frac{-(x-\bar{x}_1)^2}{4}\right\} \text{ and } g_{2X,Y}(x) = \frac{1}{2\sqrt{\pi}}\exp\left\{\frac{-(x-\bar{x}_2)^2}{4}\right\}.$$

The values $\bar{x}_1$ and $\bar{x}_2$ come from the mean of $S1$ and $S2$ respectively. Using R, the apparent error after partitioning the dataset into two parts is roughly $\boxed{0.0310.}$

c. Discuss your results.

In the problem we are given a dataset and a distribution that it came from. The problem with the distribution is that it has two parameters where one is known and the other is unknown. Therefore, to estimate the value of the unknown parameters, MLE is used to obtain an estimate based off the sample data. The next step is to utilize an error metric, in this case the L_1 norm, to gauge how well the fitted line is to the data itself. The difficulty is that since the same dataset was used to create the fitted line itself, there's some contradiction in the logic of trying to understand how good the fit is.

The practical solution since there's only one set of data is to try and partition the data using balanced half-sampling. In this case, the dataset is partitioned in to two halves and the fit is recalibrated on each of these two subsets. Then, to re-gauge the error metric, the opposite datasets will be placed into each of the other's model fit to perform a balanced half-sampling. Then a combined type of 'average' of these two values is determined to be the final adjusted error metric. The trade-off of this process would be that each of the separate model fits would themselves be less accurate since there's half of the data being used to fit an individual model.

In this case, it's expected that the balanced half-sampling would produce a larger error in comparison to the error derived from the complete dataset. The reason being that there's less of a case of overfit in the error metric. This result is seen in the above problems, where the error metric for the complete dataset is approximately 0.252 and the error metric from balanced half-sampling is approximately 0.0310. The results thus match what is understood from the theory and so there's no issue.

2. (a) For $r = n$, show that the jackknife variance estimate, $\widehat{V(T)}_J$ (equation 12.11) in Gentle), can be expressed as

$$\frac{n-1}{n}\sum_{j=1}^{n}\left(T_{(-j)} - \bar{T}_{(\cdot)}\right)^2.$$

The following is from equation 12.11 in the text:

$$\widehat{V(T)}_J = \frac{\sum_{j=1}^{r}\left(T_j^* - J(T)\right)^2}{r(r-1)}.$$

Then substituting $T_j^* = rT - (r-1)T_{(-j)}$ and $J(T) = rT - (r-1)\bar{T}_{(\cdot)}$:

$$= \frac{\sum_{j=1}^{r}\left\{\left[rT - (r-1)T_{(-j)}\right] - \left[rT - (r-1)\bar{T}_{(\cdot)}\right]\right\}^2}{r(r-1)}$$

$$= \frac{\sum_{j=1}^{r}\{(r-1)\bar{T}_{(\cdot)} - (r-1)T_{(-j)}\}^2}{r(r-1)} = \frac{\sum_{j=1}^{r}\{(r-1)(\bar{T}_{(\cdot)} - T_{(-j)})\}^2}{r(r-1)}$$

$$= \frac{\sum_{j=1}^{r}(r-1)^2(\bar{T}_{(\cdot)} - T_{(-j)})^2}{r(r-1)} = \frac{(r-1)}{r}\sum_{j=1}^{r}(\bar{T}_{(\cdot)} - T_{(-j)})^2 = \frac{(r-1)}{r}\sum_{j=1}^{r}(T_{(-j)} - \bar{T}_{(\cdot)})^2$$

Then, setting $r = n$, there is the following:

$$\frac{n-1}{n}\sum_{j=1}^{n}(T_{(-j)} - \bar{T}_{(\cdot)})^2 \blacksquare$$

(b) Again, for $r = n$, show that

$$\widehat{V(T)}_J \leq \frac{\sum_{j=1}^{n}(T_j^* - T)^2}{n(n-1)}.$$

Given that $r = n$, the above can be written as follows:

$$\widehat{V(T)}_J = \frac{\sum_{j=1}^{r}(T_j^* - J(T))^2}{r(r-1)} = \frac{\sum_{j=1}^{n}(T_j^* - J(T))^2}{n(n-1)} \leq \frac{\sum_{j=1}^{n}(T_j^* - T)^2}{n(n-1)}$$

This can be further simplified to:

$$\sum_{j=1}^{n}(T_j^* - J(T))^2 \leq \sum_{j=1}^{n}(T_j^* - T)^2$$

These two equations are both similar in that they're both a function of some value we can call x. For example, they can both be rewritten as follows:

$$f(x) = \sum_{j=1}^{n}(T_j^* - x)^2$$

Next, to try and find the optimum of this function, the first and second derivatives will be found.

$$f'(x) = 2\sum_{j=1}^{n}(T_j^* - x)(-1) \overset{set\ to}{\cong} 0$$

$$\rightarrow \sum_{j=1}^{n}(T_j^* - x) = 0$$

$$\rightarrow \sum_{j=1}^{n}T_j^* - nx = 0$$

$$\rightarrow \hat{x} = \frac{\sum_{j=1}^{n}T_j^*}{n} = \bar{T}^* = J(T)$$

$$f''(x) = \left(-2\sum_{j=1}^{n}\left(T_j^* - x\right)\right)'$$

$$= -2\frac{\partial}{\partial x}\left(\sum_{j=1}^{n} T_j^* - nx\right) = 2n > 0 \text{ (given that } n > 0)$$

Therefore, it can be said that $J(T)$ is the minimum of the function $f(x) = \sum_{j=1}^{n}\left(T_j^* - x\right)^2$. From this it follows that $\widehat{V(T)}_J \leq \frac{\sum_{j=1}^{n}\left(T_j^* - T\right)^2}{n(n-1)}$ when $r = n$. ■

3. The statistic

$$b_2 = \frac{\sum(y_i - \bar{y})^4}{(\sum(y_i - \bar{y})^2)^2}$$

is sometimes used to decide whether a least squares estimator is appropriate (otherwise, a robust method may be used).

a. What is the jackknife estimate of the standard deviation of b_2?

Letting the statistic $T = b_2$, the variance $\widehat{V(T)}_J$ becomes

$$\widehat{V(b_2)}_J = \frac{\sum_{j=1}^{r}\left(b_{2_j}^* - J(b_2)\right)^2}{r(r-1)},$$

where $b_{2_j}^* = rb_2 - (r-1)b_{2_{(-j)}}$, $J(b_2) = rb_2 - (r-1)\bar{b}_{2_{(\cdot)}}$, and $\bar{b}_{2_{(\cdot)}} = \frac{1}{r}\sum_{j=1}^{r} b_{2_{(-j)}}$.

Then the estimate of the standard deviation of b_2 becomes

$$\boxed{\widehat{se(b_2)}_J = \sqrt{\frac{\sum_{j=1}^{r}\left(b_{2_j}^* - J(b_2)\right)^2}{r(r-1)}}}.$$

b. Posted on the course Blackboard site is the data set Jackknife.txt containing 100 observations from a $N(0,1)$ distribution. Use this sample to calculate b_2 and the jackknife estimate of the standard deviation for the cases $k = 1$ and $k = 5$.

$$\boxed{b_2 \approx 0.026700}$$

	$k = 1$	$k = 5$
$\widehat{se(b_2)}_J$	≈ 0.003714	≈ 0.003693

c. Discuss the performance of the jackknife estimators found in (b). Be specific and use any techniques you feel are appropriate. (For example: You could draw several more samples from $N(0,1)$ and use them to obtain another estimate of the standard deviation.)

In the above use of jackknife, it shows that the estimated standard deviation (standard error) of the statistic is approximately 0.003714 for $k = 1$ and approximately 0.003693 for $k = 5$, while the value of b_2 itself was calculated to be approximately 0.026700. The corresponding standard errors for different values of k show that the statistic b_2 can be practical since the standard errors are small relative to the value of the statistic itself. If for example the standard errors were significantly larger, such as 1.5, then it could be problematic since the statistic itself is so small already. However, after doing some searching online, I couldn't find much information about the statistic b_2 and so it's difficult to make conclusive judgements about the statistic itself. In other words, I'm not sure if in this case a suitable value for b_2 would be anywhere from -5 to 5, which would make a standard error such as 1.5 still valid.

Following the example, two different new samples were created by adding $1{,}000$ random samples from the $N(0,1)$ distribution in one case and $10{,}000$ random samples in another. The following table shows the calculated values of b_2 for each of the two cases. In the two cases, therefore the total sample sizes were $1{,}100$ and $10{,}1000$ respectively.

$\#\ Added\ samples$	$+1{,}000$	$+10{,}000$
b_2	≈ 0.268828	≈ 2.517476

Furthermore, different values of k were also tested. The below table shows the $\widehat{se(b_2)}_J$ for the different values of k. The tested values of k were all chosen such that all the r groups would be the same size.

	$k = 1$	$k = 5$	$k = 10$	$k = 20$
$\widehat{se(b_2)}_J\ (+1{,}000)$	≈ 0.792275	≈ 0.354963	≈ 0.251572	≈ 0.178710
$\widehat{se(b_2)}_J\ (+10{,}000)$	2.653616	1.186968	≈ 0.839521	≈ 0.593926

In these tests it seems apparent that from the larger number of samples, the value of the standard error will vary greatly depending on the value of k. This is an interesting result, since it shows how important each of the pseudovalues possible are, since they're calculated as if they were independent. By grouping them into larger clusters, the ability for the groups of pseudovalues to appear independent holds better. Previously, in part (b), it seemed possible that this same pattern held, but the difference it made with a sample size of 100 was quite negligible.

It seems then that by looking at the variance of the statistic b_2 using jackknife, the statistic can be said to be appropriate for the data. When using an appropriate k, the standard error of b_2 will remain within a range that appears to be relatively small enough compared to the value of b_2 itself such that the use of b_2 is not made irrelevant.

Appendix

```
### 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-ret
urn-vectors
    Z <- R_list
  } else {
    # Reference: https://stackoverflow.com/questions/652136/how-can-i-remove-an-element-from-a
```

```r
-list
    Z <- R_list[-j]
  }

  # Reference: https://stackoverflow.com/questions/14924935/using-r-convert-data-frame-to-simp
le-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
```

```r
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
```