Assessment

Vishal

2022-12-14

Section B

B.1

(1) Let's consider,

p = probability that light bulb comes from line A$\mathrm{p_A}$ = probability that the light bulb has a lifetime equal to or greater than

We need to calculate the probability that the light bulb is manufactured from line B, provided it's lifetime is less than 2 years. i.e. α.

Let's assume that $\mathrm{p_D}$ = probability that light bulb has life time less than 2 years.

So, $\alpha = \mathbb{P}(B|D)$ i.e. probability that the light bulb is made in line B, given that it's lifetime is less than 2 years.

We know that,

$$\alpha = \mathbb{P}(B|D) \Rightarrow \frac{\mathbb{P}(B \cap D)}{\mathbb{P}(D)}$$

$$\Rightarrow \frac{\mathbb{P}(B)\mathbb{P}(D|B)}{\mathbb{P}(A)\mathbb{P}(D|A) + \mathbb{P}(B)\mathbb{P}(D|B)}$$

$$\Rightarrow \boxed{\frac{(1-\mathrm{p})(1-\mathrm{p_B})}{\mathrm{p}(1-\mathrm{p_A}) + (1-\mathrm{p})(1-\mathrm{p_B})}} \tag{1}$$

(2) We have,

$$\mathrm{p_A} = 0.99\mathrm{p_B} = 0.5\mathrm{p} = 0.1$$

And we need to calculate α. Using equation (1), we get,

$$\Rightarrow \frac{(1-\mathrm{p})(1-\mathrm{p_B})}{\mathrm{p}(1-\mathrm{p_A}) + (1-\mathrm{p})(1-\mathrm{p_B})} \Rightarrow \frac{(1-0.1)(1-0.5)}{0.1(1-0.99) + (1-0.1)(1-0.5)} \Rightarrow \frac{0.9 * 0.5}{0.1 * 0.01 + 0.9 * 0.5} \Rightarrow \frac{0.45}{0.451} \Rightarrow \boxed{0.9977827}$$

```r
n <- 100000
 df <- data.frame(Line = sample(c(0, 1), 100000, prob = c(0.1, 0.9), replace = T)) %>%
  mutate(LessThan2Years = case_when(Line == 0 ~sample(c(0, 1), n, prob = c(0.99, 0.01), replace = TRUE)
                                    (Line == 1 ~sample(c(0, 1), n, prob = c(0.5, 0.5), replace = TRUE))
```

(3)

```r
lessThan2YearsDf <- df %>%
  filter(LessThan2Years == 1)

lessThan2YearsLine1 <- lessThan2YearsDf %>%
  filter(Line == 1) %>%
  count()

alpha = lessThan2YearsLine1 / length(lessThan2YearsDf$Line)
print(round(alpha, 5))
```

(4)

```
##         n
## 1 0.99773
```

The estimated value of α is 0.99773.

B.2

Let,

$$X = \text{continuous random variable} X = 0, \text{product gets on the shelf immediately after it is sold out}$$

We have PDF given as,

$$\mathrm{p}_\lambda(x) = \begin{cases} ae^{-\lambda(x-b)} & x \geq b \\ 0 & \text{if } x < b \end{cases}$$

where,

$$b > 0 \text{ is a known constant} \lambda > 0 \text{ is a parameter of the distribution}$$

The given probability density function is a **memoryless** form of the exponential distribution function meaning, the distribution of time between restocking the shelves is the same as at time zero.
So the function is similar as,

$$\mathrm{p}_\lambda(x) = \begin{cases} ae^{-\lambda(x-b)} & x > 0 \\ 0 & otherwise \end{cases}$$

Integrating this function, we will get,

$$\mathrm{p}_\lambda(x) = -\frac{a\mathrm{e}^{-\lambda\cdot(x-b)}}{\lambda}$$

$$\Rightarrow \frac{a\cdot\left(\mathrm{e}^{b\lambda}-1\right)}{\lambda} \tag{1}$$

(1)

To derive the value of a, we need to integrate the function while the value is 1.
From equation (1), we have,

$$\int_b^\infty \mathrm{p}_\lambda(x) = \int_b^\infty ae^{-\lambda(x-b)} \text{substituting } u = -\lambda\cdot(x-b) \rightarrow \frac{du}{dx} = -\lambda \rightarrow dx = \frac{1}{\lambda}du = -\frac{a}{\lambda}\int e^u du \text{now solving: } \int e^u du \text{Applying}$$

2

Population mean:

We'll be using integration by parts to find the population mean.

$$EX = \int_b^\infty ae^{-\lambda(x-b)}xdx$$

$$\Rightarrow aeb^\lambda \int_b^\infty xe^{-\lambda x}dx \Rightarrow \frac{a\cdot(b\lambda+1)}{\lambda^2} \text{Substituting } a=\lambda, \Rightarrow \frac{\lambda\cdot(b\lambda+1)}{\lambda^2} \Rightarrow \boxed{\frac{b\lambda+1}{\lambda}}$$

Standard Deviation:

$$\sigma^2 = E(X-\mu)^2 \Rightarrow E(X-E(X)^2) \Rightarrow E\left(X-\left(\frac{b\lambda+1}{\lambda}\right)^2\right) \Rightarrow \int_b^\infty \left(z-\frac{b\lambda+1}{\lambda}\right)^2 ae^{-\lambda(z-b)}dz \Rightarrow \left[-\frac{a\cdot\left(\lambda^2\cdot(z-b)^2+1\right)\mathrm{e}^{-\lambda\cdot}}{\lambda^3}\right.$$

(3) Cumulative Distribution function:

$$\int_{-\infty}^x f(t)dt = \int_{-\infty}^b f(t)dt + \int_b^x f(t)dt \Rightarrow \int_{-\infty}^b 0dt + \int_b^x ae^{-\lambda(t-b)}dt \Rightarrow \left[-\frac{a\mathrm{e}^{-\lambda\cdot(t-b)}}{\lambda}\right]_b^x \Rightarrow \frac{a\cdot\left(1-\mathrm{e}^{b\lambda-x\lambda}\right)}{\lambda} \text{Since } a=\lambda, \Rightarrow \boxed{1-$$

Quantile function:

The quantile function $\mathrm{Q_X}(p)$ is defined as the smallest x such that $F_X(x) = p$

$$Q_X(p) = min\{x \in \mathbb{R} | F_X(x) = p\} \tag{a}$$

Thus we have $Q_X(p) = -\infty,$ if $p = 0$. When $p > 0,$ it holds that ,

$$Q_X(p) = F_X^{-1}(x) \tag{b}$$

Arranging the probability density function we get,

$$p = 1-\mathrm{e}^{\lambda(x-b)} \Rightarrow 1-p = \mathrm{e}^{\lambda(x-b)} \Rightarrow \ln(1-p) = \lambda(x-b) \Rightarrow \ln(1-p) = \lambda x - \lambda b \Rightarrow \ln(1-p)+\lambda b = \lambda x \Rightarrow \boxed{x = \frac{\ln(1-p)+\lambda b}{\lambda}}$$

(4) Maximum Likelihood Estimate:

$$\lambda_{MLE} = \prod_{i=1}^n f_X(x_i;\lambda) = \prod_{i=1}^n \{ae^{-\lambda(x_i-b)}\} L(\lambda) = ae^{-n\lambda((x_1+x_2+\ldots+x_n)-b)} \log L(\lambda) = \log(a) - \lambda\sum_{i=1}^n xlog(e) + \lambda b\log(e) \text{Substitutin}$$

```
supermarketDataDf <- read.csv("supermarket_data_EMATM0061.csv")
head(supermarketDataDf)
```

```
##   TimeLength
## 1   304.1401
## 2   504.4332
## 3   388.3975
## 4   326.7841
## 5   310.8762
## 6   340.0900
```

(5) To calculate the MLE, we know that log MLE for the given function is $\frac{1}{b-\bar{x}}$.

```
b = 300
MLE = 1 / (mean(supermarketDataDf$TimeLength) - b)
MLE
```

```
## [1] 0.01988426
```

```
compute_median <- function(df, indices, col_name) {
  sub_sample <- slice(df, indices) %>%
    pull(all_of(col_name))
  return(median(sub_sample, na.rm = T))
}
```

```
results <- boot(data = supermarketDataDf, statistic = compute_median, col_name = "TimeLength", R = 1000

boot.ci(boot.out = results, type = "perc", conf = 0.95)
```

(6)

```
## BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
## Based on 10000 bootstrap replicates
##
## CALL :
## boot.ci(boot.out = results, conf = 0.95, type = "perc")
##
## Intervals :
## Level     Percentile
## 95%   (333.6, 337.8 )
## Calculations and Intervals on Original Scale
```

```
b = 0.01
lambda_0 = 2
```

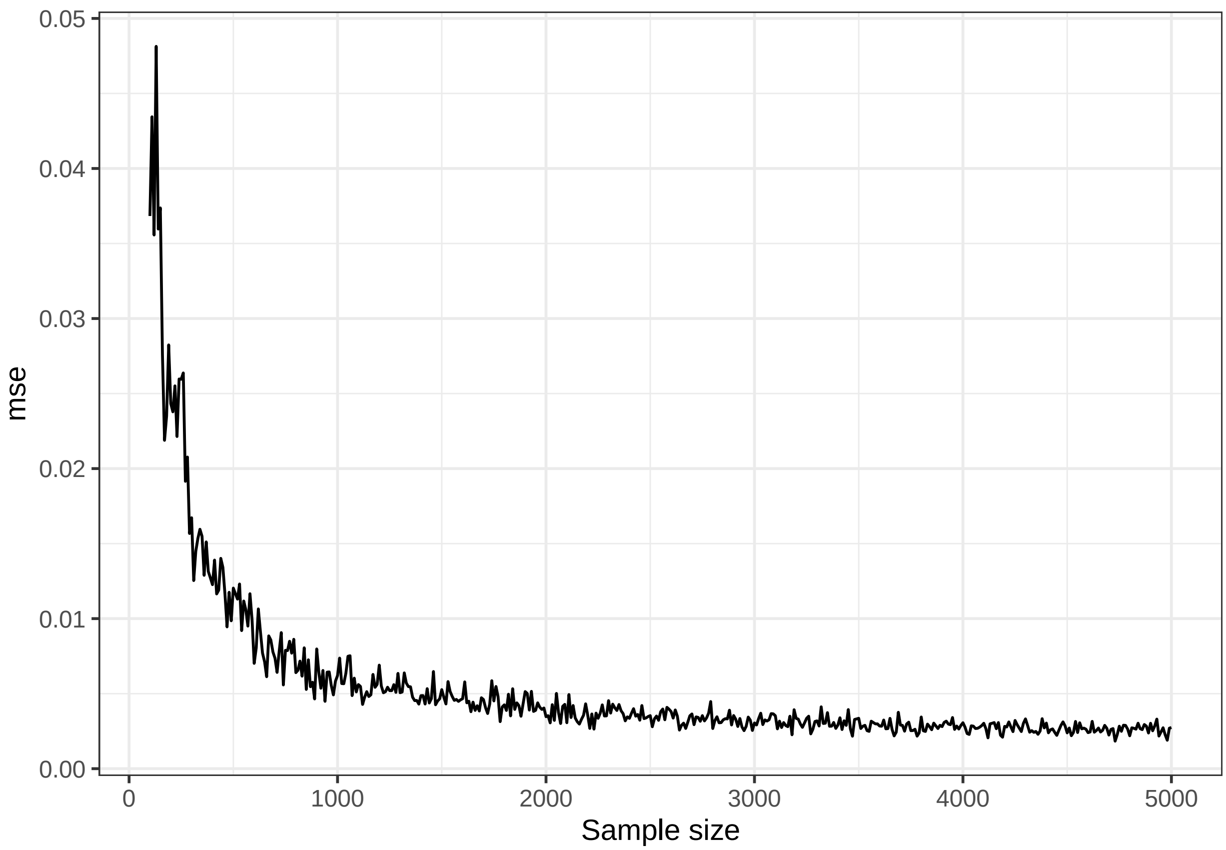
```r
sample_size_seq <- seq(100, 5000, 10)
num_trials_per_size <- 100

compute_mle <- function(x, b) {
  return (1 / (mean(x) - b))
}

df <- crossing(sample_size = sample_size_seq, trials = seq(num_trials_per_size)) %>%
  mutate(samples = map(sample_size, ~rexp(.x, lambda_0))) %>%
  mutate(lambda_mle = map_dbl(samples, ~compute_mle(.x, b)))


df_mse <- df %>%
  group_by(sample_size) %>%
  summarise(mse = mean((lambda_mle - lambda_0) ^ 2))

ggplot(df_mse) +
  geom_line(aes(x = sample_size, y = mse)) +
  theme_bw() +
  xlab('Sample size')
```



(7)

B.3

(1) Probability Mass Function:

The probability mass function $\mathrm{p_X} : \mathbb{R} \to [0,1]$ is defined by$\mathrm{p_X}(x) := P_X(\{x\}) = \mathbb{P}(X = x)$where $\mathrm{p_X}$ is the distribution of X.

$a =$ number of red balls in a bag$b =$ number of blue balls in a bag$a+b =$ total number of balls in a bag$X =$ number of red b

As we are drawing only two balls from the bag, our random variable can take on 2 possible values. From the drawing of 2 red balls and 0 blue ball, we can end up with possible value of 2. Whereas, if we draw 1 red ball and 1 blue ball, we will end up the possible value of 0 for our random variable X.

Thus we can have $X = 0$ and $X = 2$.

Now we should calculate the probability of each possible value of X. There are total $a + b$ balls, and we are picking 2 without replacement. Thus, there are $\binom{a+b}{2}$ total ways to draw 2 balls without replacement from the bag. There are a red balls and b blue balls. The probabilities are calculated as follows:

For $X = 0 :$ We will have 1 red ball and 1 blue ball. Thus $P(X = 0) = \dfrac{\binom{a}{1}\binom{b}{1}}{\binom{a+b}{2}}$For $X = 2 :$ We will have 2 red balls and 0

$$\sum_x P_x(X) = P(X=0)+P(X=2)+P(X=-2) \Rightarrow \frac{\binom{a}{1}\binom{b}{1}}{\binom{a+b}{2}} + \frac{\binom{a}{2}\binom{b}{0}}{\binom{a+b}{2}} + \frac{\binom{a}{0}\binom{b}{2}}{\binom{a+b}{2}} \Rightarrow \frac{\binom{a}{1}\binom{b}{1} + \binom{a}{2}\binom{b}{0} + \binom{a}{0}\binom{b}{2}}{\binom{a+b}{2}} \Rightarrow$$

We have our values of $P(X = x)$ for all possible X, and thus we have our probability mass function describing the distribution of X.

(2) Expectation:

$$E(X) = \mu_x = \sum_{\text{all } x_i} (x_i)P(x_i)E(X) = (0)*\left(\frac{\frac{a!}{(a-1)!} * \frac{b!}{(b-1)!}}{\frac{(a+b)!}{2(a+b-2)!}}\right)+(2)*\left(\frac{\frac{a!}{2(a-2)!}}{\frac{(a+b)!}{2(a+b-2)!}}\right)-(2)*\left\{\frac{\frac{b!}{2(b-2)!}}{\frac{(a+b)!}{2(a+b-2)!}}\right\}E(X) = (2)*\left(\frac{\frac{a!}{2(a-2)!}}{\frac{(a+b)!}{2(a+b-2)!}}\right)-$$

(3) Variance:

$$\sigma^2 = Var(X) = \sum_x [x^2 \cdot p(x)] - \Big[\sum_x x \cdot p(x)\Big]^2 \boxed{\sum_x x^2.p(x) = (4) * \left(\frac{\frac{a!}{2(a-2)!}}{\frac{(a+b)!}{2(a+b-2)!}}\right) + (4) * \left(\frac{\frac{b!}{2(b-2)!}}{\frac{(a+b)!}{2(a+b-2)!}}\right)} \Big[\sum_x x \cdot p(x)\Big]^2 = \Big\{(2)*\Big(\frac{\frac{}{2(}}{\frac{(}{2(a}}$$

```
compute_expectation_X <- function(a, b) {
  xpx <- (2 * choose(a, 2) / choose(a + b, 2)) - (2 * choose(b, 2) / choose(a + b, 2))
  return(xpx)
}

compute_variance_X <- function(a, b) {
  variance <-
     (((4 * choose(a, 2) / choose(a + b, 2)) + (4 * choose(b, 2) / choose(a + b, 2))) -
    (((2 * (choose(a, 2) / choose(a + b, 2))) ^ 2) - (8 * (choose(a, 2) / choose(a + b, 2)) * (choose(b
    ((2 * (choose(b, 2) / choose(a + b, 2))) ^ 2)))

  return(variance)
}

compute_variance_X(3, 5)
```

(4)

```
## [1] 1.607143
```

(5) We know that, X can take only 2 values i.e. 0 and 2. Although, we are not considering the event where we pick 2 blue balls and 0 red balls.

$$\bar{X} = \frac{1}{n}\sum_{i=1}^{n} X$$

Since, X takes only two values 0 and 2 with equal chance of occurrence, if we repeat the experiment n times, the mean will be equal to 1.
Therefore, the expectation will be,

$$Expectation = E(\bar{X}) = (1) * \left(\frac{\frac{a!}{2(a-2)!}}{\frac{(a+b)!}{2(a+b-2)!}}\right)$$

(6)

$Variance = \sigma^2_{\bar{X}} = \dfrac{\sum_i \sigma_i^2 + 2\sum_i \sum_{j<i} cov(X_i, X_j)}{n^2}$where n = number of independent copies of XSince all X_i are independen

```
sample_Xs <- function(a, b, n) {
  prob_0 <- (choose(a, 1) * choose(b, 1)) / choose(a + b, 2)
  prob_2 <- (choose(a, 2) * choose(b, 0)) / choose(a + b, 2)

  prob_neg2 = (choose(a, 0) * choose(b, 2)) / choose(a + b, 2)
  # prob_neg2: Probability of choosing 0 red balls and 2 blue balls
```

```r
  samples <- sample(c(0, 2, -2), size = n, prob = c(prob_0, prob_2, prob_neg2), replace = T)
  return(samples)
}

table(sample_Xs(3, 5, 100))
```

(7)

```
##
## -2  0  2
## 32 56 12
```

```r
a = 3
b = 5
n = 100000
Samples <- sample_Xs(a, b, n)

Expectation <- compute_expectation_X(a, b)
Variance <- compute_variance_X(a, b)

sampleMean <- mean(Samples)
sampleVariance <- var(Samples)

print(Expectation)
```

(8)

```
## [1] -0.5
```

```r
print(Variance)
```

```
## [1] 1.607143
```

```r
print(sampleMean)
```

```
## [1] -0.501
```

```r
print(sampleVariance)
```

```
## [1] 1.601895
```

The expectation calculated using the function `compute_expectation_X` is -0.5 whereas the sample mean is -0.501. As it is seen these two values are almost equal. That is, the sample mean and the expectation of the sample are similar.

Additionally, the variance that we got using the function `compute_variance_X` is 1.6071429 and it is equal to the the variance of the sample generated and calculated using the `var` function in R i.e. 1.601895.

```
mu_0 <- Expectation
n = 900
sigma_0 <- sqrt(Variance / n)
a = 3
b = 5

num_trials <- 20000

simStudyDf <- data.frame(trials = rep(n, num_trials)) %>%
  mutate(samples = map(trials, ~sample_Xs(a, b, .x))) %>%
  mutate(sampleMean = map_dbl(samples, mean))
```

(9)

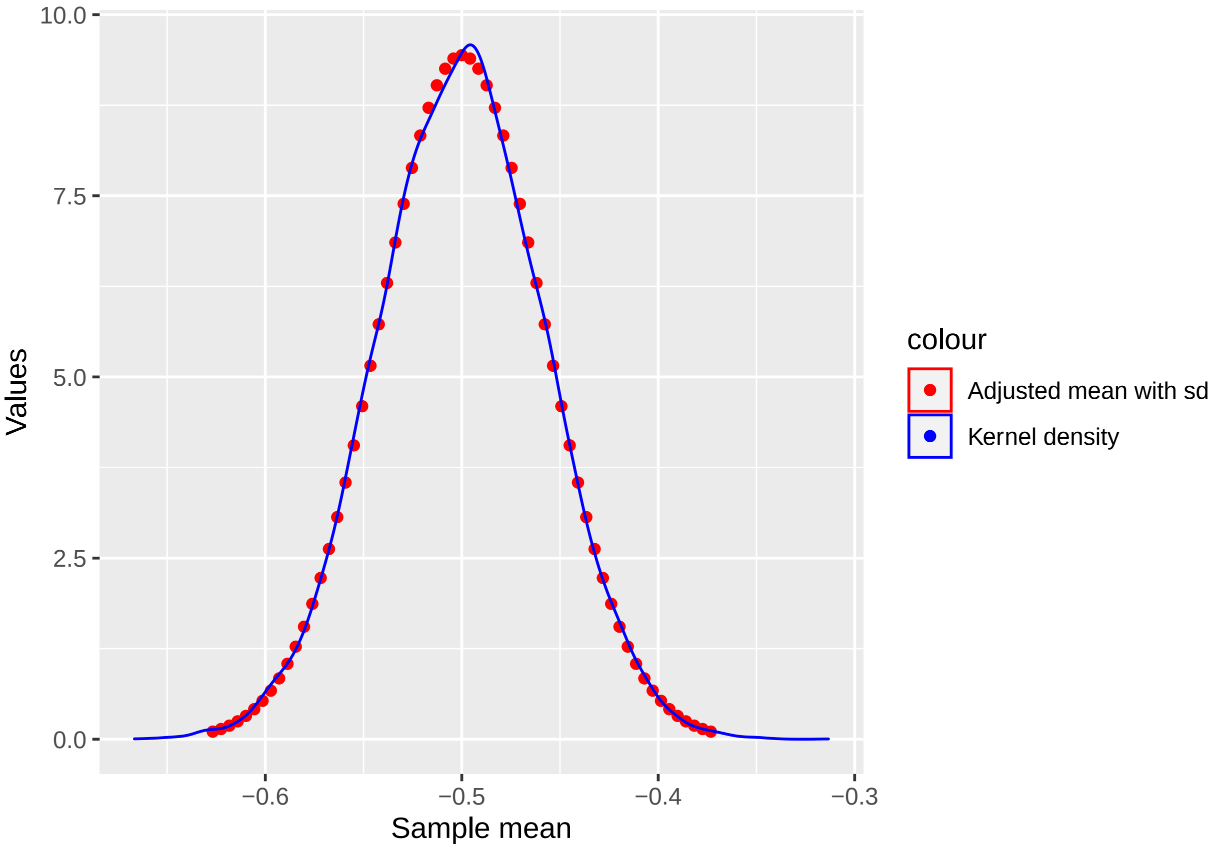
```
mu_minus_sigma_calc <- function(mu, sig) {
  return(mu - sig)
}

densityPlotDf <- data.frame(sigma = seq(-3 * sigma_0, 3 * sigma_0, 0.1 * sigma_0)) %>%
  mutate(newSeq = map_dbl(sigma, ~mu_minus_sigma_calc(mu_0, .x))) %>%
  mutate(GuassianDistr = map_dbl(.x = newSeq, ~dnorm(.x, mean = mu_0, sd = sigma_0)))
```

(9)

```
color <- c("Adjusted mean with sd" = "red", "Kernel density" = "blue")

ggplot(densityPlotDf) +
  geom_point(aes(x = newSeq, y = GuassianDistr, color = "Adjusted mean with sd")) +
  geom_density(data = simStudyDf, aes(x = sampleMean, color = "Kernel density")) +
  ylab('Values') +
  xlab("Sample mean") +
  scale_color_manual(values = color)
```



(10)

(11) The Kernel density of $\bar{X}$ follows the same curve as the scatter plot plotted with function $f_{\mu,\sigma}$ which is a Gaussian distribution.

In our simulation study we had a small standard deviation i.e., a very narrow peak in our plot. And that is why, it coincides with the normal distribution plot. Although, it is slightly smoother and more flexible than the normal distribution, since it is a non-parametric method that does not rely on any assumptions about the underlying distribution.