STATISTICAL METHODS FOR DATA SCIENCE Mini-Project 5

Duo Group #23

Members: Hima Sri Tipirineni

Nithin Pingili

Contribution of each team member:

Hima Sri and Nithin worked together to complete both the questions. Collaborated to learn R and then worked on plotting the scatter plots, qq plots, boxplots and histograms for the given questions. Also worked on finding the bootstrap estimates and confidence intervals. Both worked together to answer the questions and report all the findings. Hima Sri wrote R code and annotated the code and Nithin worked to check the accuracy of the R code and added the observations.

1. Consider the data stored in bodytemp-heartrate.csv on eLearning, containing measurements of body temperature and heart rate for 65 male (gender = 1) and 65 female (gender = 2) subjects.

(a) Do males and females differ in mean body temperature? Answer this question by performing an appropriate analysis of the data, including an exploratory analysis.

Sol :

Summary for the male body temperature :

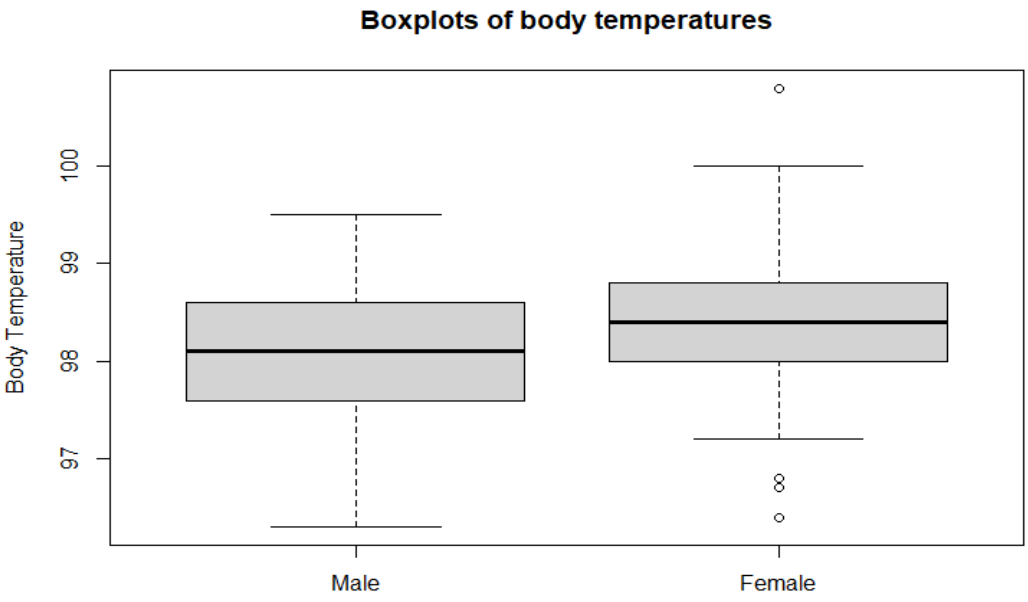
Min	1st Qu.	Median	Mean	3rd Qu.	Max
96.3	97.6	98.1	98.1	98.6	99.5

Summary for the Female body temperature :

Min	1st Qu.	Median	Mean	3rd Qu.	Max
96.40	98.00	98.40	98.39	98.80	100.80

From the summary statistics, we can see that the mean body temperature of female is slightly greater than that of the male body temperature. Also, the mean and median of the male body temperature is similar, which might imply that the distribution may be symmetric.

```
>
> #### Question 1 ####
>
> ### Reading the body temperature and heart rate data from the csv data ###
>
> bodytemp_hrate = read.csv("C:\\Users\\hxt210018\\Downloads\\6313_Prob\\HW\\bodytemp-heartrate.csv");
>
> ## Separating the male and female data ##
>
> male_data = subset(bodytemp_hrate,bodytemp_hrate$gender == 1)
> female_data = subset(bodytemp_hrate, bodytemp_hrate$gender == 2)
>
> ## Getting the summary for the male_data and female data ##
>
> summary(male_data$body_temperature)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
   96.3    97.6    98.1    98.1    98.6    99.5
> summary(female_data$body_temperature)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  96.40   98.00   98.40   98.39   98.80  100.80
>
> ## Drawing the boxplots ##
>
> boxplot(male_data$body_temperature,female_data$body_temperature, main = "Boxplots of body temperatures", names = c('Male','Female'),
+         ylab = "Body Temperature")
>
```



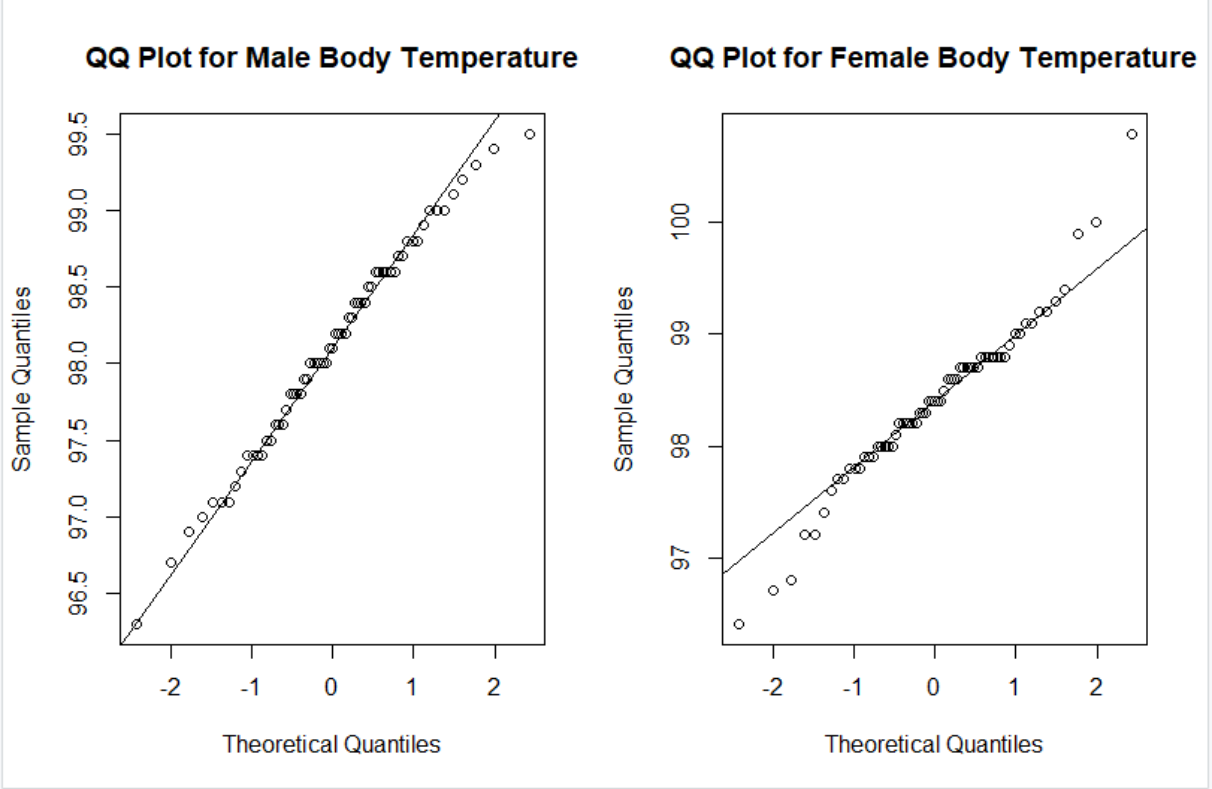
Conclusions from the boxplot :

1. Median body temperature of male is less than that of the median of the female body temperature.
2. There are four outliers in the boxplot of the female body temperatures which suggest that there is more variability in values than male. Hence, we cannot assume equal variances.
3. There are no outliers in the male body temperature boxplot.

4. The Interquartile Range for the body temperature of men is more than the IQR of the body temperature of female.

Now, lets us draw the QQ plots for the male and female body temperatures.

```
>
>
> ## Drawing the QQ plots for the body temperatures ##
>
> par(mfrow = c(1,2))
>
> qqnorm(male_data$body_temperature, main = 'QQ Plot for Male Body Temperature')
> qqline(male_data$body_temperature)
>
> qqnorm(female_data$body_temperature, main = 'QQ Plot for Female Body Temperature')
> qqline(female_data$body_temperature)
>
```



Observations from the QQ Plots :

As the values fall in almost same straight line, we can make normality assumptions for both the distributions.

Let M be the body temperatures of the male, F be the body temperature of the female. Let m1 estimate the population mean of male body temperatures μ_m and f1 estimate the population mean of the female body temperatures μ_f

Null Hypothesis : Difference between the mean of the body temperature of men and women is equal is 0. i.e m1 – f1 = 0

Alternative Hypothesis : Difference between the mean of the body temperature of men and women is not equal to 0. i.e m1 – f1 ≠ 0.

Let us treat the two samples as independent samples with unequal variances coming from approximately normal distributions, hence we can use t-distribution with Satterthwaite's approximation to get the confidence interval.

Hence, constructing the confidence interval using t-test.

```
> ### Calculating confidence intervals using the t-test for the body temperture ###
> t.test(male_data$body_temperature, female_data$body_temperature,alternative = 'two.sided',var.equal = FALSE)

        Welch Two Sample t-test

data:  male_data$body_temperature and female_data$body_temperature
t = -2.2854, df = 127.51, p-value = 0.02394
alternative hypothesis: true difference in means is not equal to 0
95 percent confidence interval:
 -0.53964856 -0.03881298
sample estimates:
mean of x mean of y
 98.10462  98.39385

>
```

The confidence interval obtained is [-0.53964856 ,-0.03881298].The p-value is 0.02394

Since 0 does not lie in the confidence interval , and also the p-value is less than 0.05(i.e α value), We **reject the Null hypothesis**.

Hence, it is concluded that the there is difference between the mean body temperatures of male and female.

(b) Do males and females differ in mean heart rate? Answer this question by performing an appropriate analysis of the data, including an exploratory analysis.

Summary for the male heart rate :

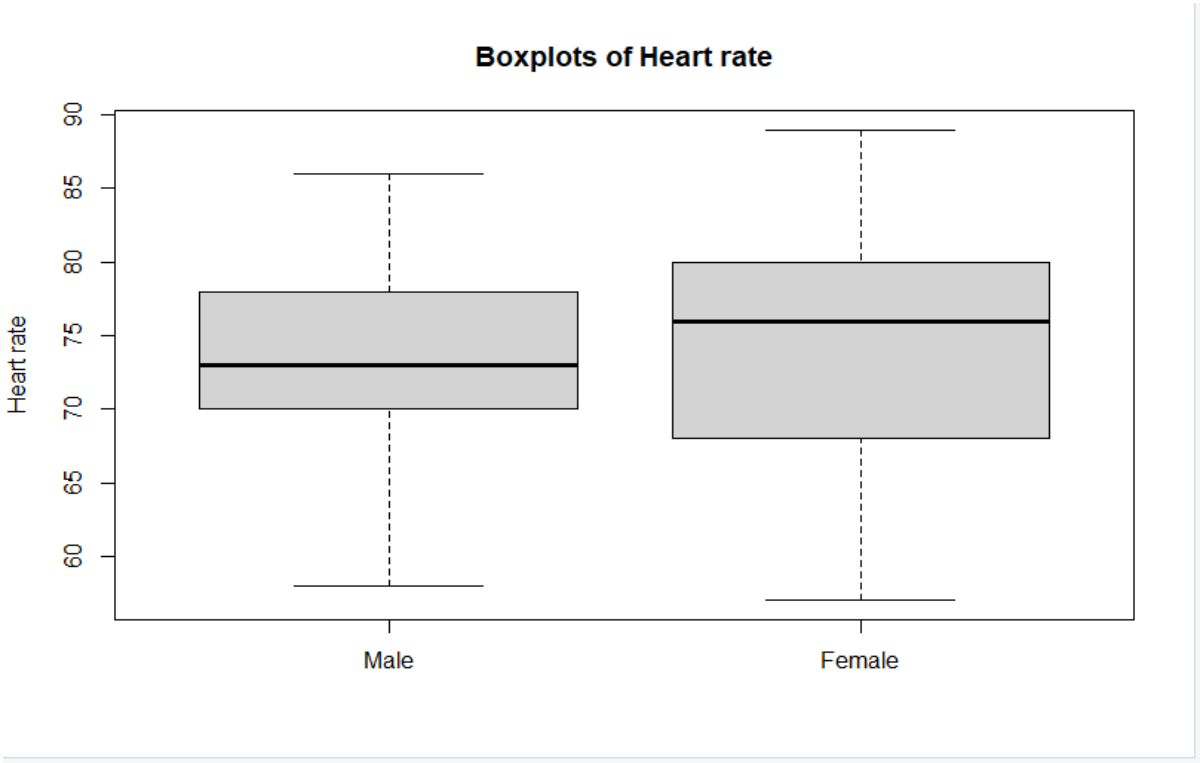
Min	1st Qu.	Median	Mean	3rd Qu.	Max
58.00	70.00	73.00	73.37	78.00	86.00

Summary for the Female heart rate :

Min	1st Qu.	Median	Mean	3rd Qu.	Max
57.00	68.00	76.00	74.15	80.00	89.00

From the summary statistics, we can see that the mean heart rate of female is slightly greater than that of the mean heart rate of male.

```
>
> #### Question 1 part (b) ####
>
> ## Getting the summary for the male_data and female data heart_rate ##
>
> summary(male_data$heart_rate)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  58.00   70.00   73.00   73.37   78.00   86.00
> summary(female_data$heart_rate)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  57.00   68.00   76.00   74.15   80.00   89.00
>
> ## Drawing the boxplots ##
>
> boxplot(male_data$heart_rate,female_data$heart_rate, main = "Boxplots of Heart rate", names = c('Male','Female'),
+         ylab = "Heart rate")
>
```

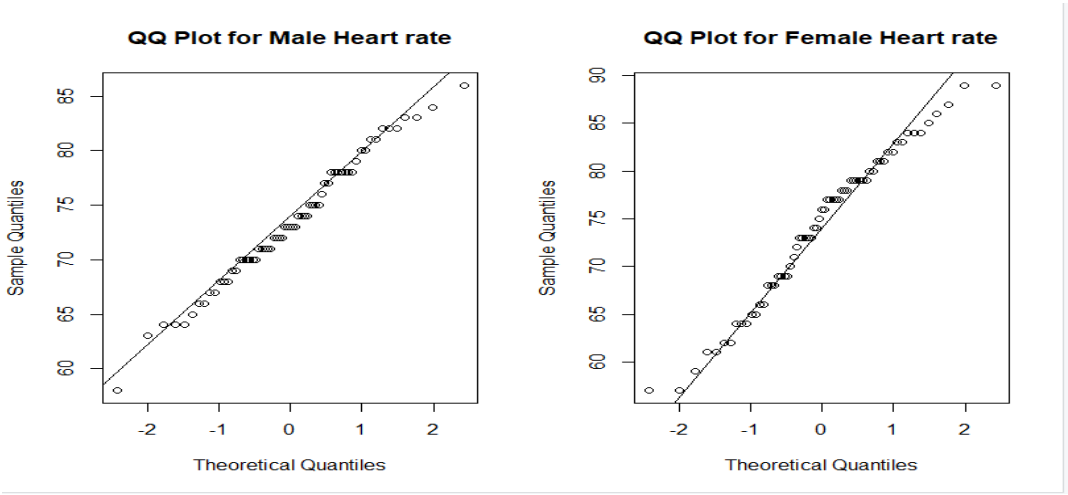


Conclusions from boxplot :

1. Median heart rate of male is less than that of the median of the female heart rate.
2. There are no outliers in the male and female heart rate boxplots.
3. The Interquartile Range for the heart rate of female is more than the IQR of the body temperature of male.
4. The heart rate values among the women seems to be more stretched out than those of the men heart rate values. Hence, we cannot assume equal variances.

Constructing the QQ plots

```
>
> ## Drawing the QQ plots for the Heart rate ##
>
> par(mfrow = c(1,2))
>
> qqnorm(male_data$heart_rate, main = 'QQ Plot for Male Heart rate')
> qqline(male_data$heart_rate)
>
> qqnorm(female_data$heart_rate, main = 'QQ Plot for Female Heart rate')
> qqline(female_data$heart_rate)
>
>
> |
```



Observations from QQ-plot :

As the values fall in almost same straight line, we can make normality assumptions for both the distributions.

Let M_h be the heart rate of the male, F_h be the heart rate of the female. Let mh1 estimate the population mean of male heart rate values μ_{mh} and fh1 estimate the population mean of the female body temperatures μ_{fh}

Null hypothesis: Difference between the mean of the heart rate of men and women is equal is 0. i.e mh1 – fh1 = 0

Alternative hypothesis: Difference between the mean of the heart rate values of men and women is not equal to 0. i.e mh1 – fh1 ≠ 0.

Let us treat the two samples as independent samples with unequal variances coming from approximately normal distributions, hence we can use t-distribution with Satterthwaite's approximation to get the confidence interval.

Hence, constructing the confidence interval using t-test.

```
> ### Calculating confidence intervals using the t-test for the heart rate  ###
> t.test(male_data$heart_rate, female_data$heart_rate,alternative = 'two.sided',var.equal = FALSE)

        Welch Two Sample t-test

data:  male_data$heart_rate and female_data$heart_rate
t = -0.63191, df = 116.7, p-value = 0.5287
alternative hypothesis: true difference in means is not equal to 0
95 percent confidence interval:
 -3.243732  1.674501
sample estimates:
mean of x mean of y
 73.36923  74.15385

> |
```

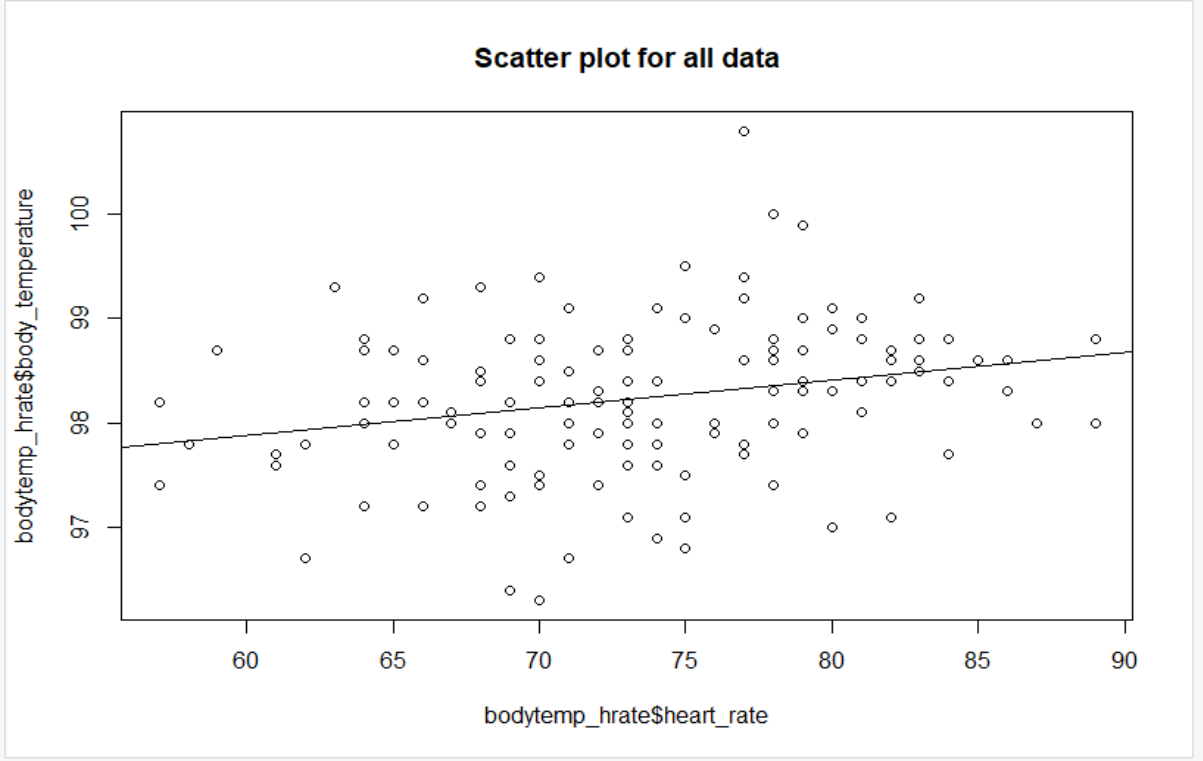
The confidence interval obtained is [-3.243732, 1.674501]. The p-value is 0.5287

Since 0 lies in the confidence interval, and also the p-value is greater than 0.05(i.e α value), We **accept the Null hypothesis**.

Hence, it is concluded that there is no difference between the mean heart rates of male and female.

(c) Is there a linear relationship between body temperature and heart rate? Does this relationship depend on gender? Answer these questions by performing an appropriate analysis of the data, including an exploratory analysis.

Sol :



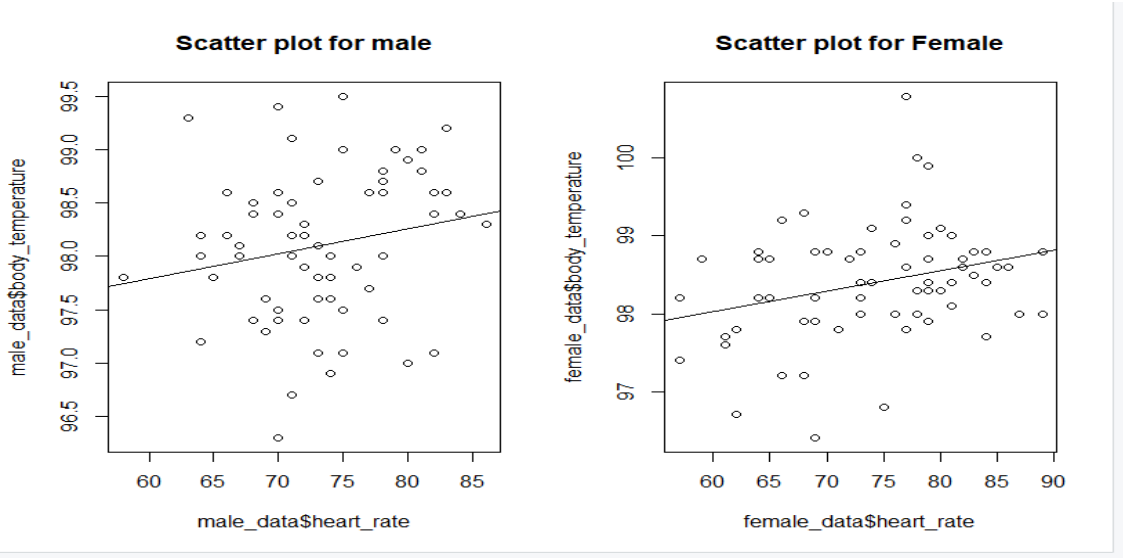
From the scatter plot above, it seems there is a slight linear relationship between heart rate and body temperature upon considering all samples from male and female combined.

We compute the linear model and summary for the linear model and below is the data obtained.

The correlation between the body temperature and heartrate of the combined data is 0.2536564.

```
>
> ## Question 1 Part (c) ##
>
> ##Finding the correlation between the body temperature and heart rate values irrespective of gender ##
>
> par(mfrow=c(1,1))
> cor(bodytemp_hrate$body_temperature, bodytemp_hrate$heart_rate)
[1] 0.2536564
>
> plot(bodytemp_hrate$heart_rate, bodytemp_hrate$body_temperature, pch = 1, main = 'Scatter plot for all data')
> abline(lm(bodytemp_hrate$body_temperature ~ bodytemp_hrate$heart_rate))
>
> linear_model = lm(bodytemp_hrate$body_temperature ~ bodytemp_hrate$heart_rate)
>
> print(linear_model)

Call:
lm(formula = bodytemp_hrate$body_temperature ~ bodytemp_hrate$heart_rate)

Coefficients:
              (Intercept)  bodytemp_hrate$heart_rate
                 96.30675                   0.02633

> summary(linear_model)

Call:
lm(formula = bodytemp_hrate$body_temperature ~ bodytemp_hrate$heart_rate)

Residuals:
     Min       1Q   Median       3Q      Max
-1.85017 -0.39999  0.01033  0.43915  2.46549

Coefficients:
                          Estimate Std. Error t value Pr(>|t|)
(Intercept)              96.306754   0.657703 146.429  < 2e-16 ***
bodytemp_hrate$heart_rate  0.026335   0.008876   2.967  0.00359 **
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.712 on 128 degrees of freedom
Multiple R-squared:  0.06434,   Adjusted R-squared:  0.05703
F-statistic: 8.802 on 1 and 128 DF,  p-value: 0.003591
```

Scatter plots for male and female separately



The scatterplot for male shows that there is greater variability in the points and points are farther away from the line. The scatterplot for female shows less variability and points are closer to the line than for male scatterplot.

```
>
> ## Finding the correlation between the body temperature and heart rate values for male and female ##
>
> cor(male_data$body_temperature,male_data$heart_rate)
[1] 0.1955894
>
> cor(female_data$body_temperature, female_data$heart_rate)
[1] 0.2869312
>
```

For males, the correlation between body temperature and heart rate is 0.1955894. The value shows that there is weak linear relationship between body temperature and heart rate. The correlation value for female is 0.2869312. For female, the linear relationship between body temperature and heart rate is also weak. However, the linear relationship is greater than for male.

We compute the linear model and summary for the linear model and below is the data obtained.

```
>
> par(mfrow = c(1,2))
>
> plot(male_data$heart_rate, male_data$body_temperature, pch = 1, main = 'Scatter plot for male')
> abline(lm(male_data$body_temperature ~ male_data$heart_rate))
>
> linear_model = lm(male_data$body_temperature ~ male_data$heart_rate)
>
> print(linear_model)

Call:
lm(formula = male_data$body_temperature ~ male_data$heart_rate)

Coefficients:
         (Intercept)  male_data$heart_rate
            96.39789               0.02326

> summary(linear_model)

Call:
lm(formula = male_data$body_temperature ~ male_data$heart_rate)

Residuals:
     Min       1Q   Median       3Q      Max
-1.72624 -0.49603  0.05291  0.48766  1.43659

Coefficients:
                    Estimate Std. Error t value Pr(>|t|)
(Intercept)         96.39789    1.08154  89.130   <2e-16 ***
male_data$heart_rate  0.02326    0.01469   1.583    0.118
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.6907 on 63 degrees of freedom
Multiple R-squared:  0.03826,   Adjusted R-squared:  0.02299
F-statistic: 2.506 on 1 and 63 DF,  p-value: 0.1184

>
```

```
>
>
> plot(female_data$heart_rate, female_data$body_temperature, pch = 1, main = 'Scatter plot for Female')
> abline(lm(female_data$body_temperature ~ female_data$heart_rate))
>
> linear_model = lm(female_data$body_temperature ~ female_data$heart_rate)
>
> print(linear_model)

Call:
lm(formula = female_data$body_temperature ~ female_data$heart_rate)

Coefficients:
           (Intercept)  female_data$heart_rate
              96.44211                 0.02632

> summary(linear_model)

Call:
lm(formula = female_data$body_temperature ~ female_data$heart_rate)

Residuals:
    Min      1Q  Median      3Q     Max
-1.8582 -0.3635 -0.0582  0.4576  2.3312

Coefficients:
                      Estimate Std. Error t value Pr(>|t|)
(Intercept)           96.44211    0.82576 116.792   <2e-16 ***
female_data$heart_rate  0.02632    0.01107   2.377   0.0205 *
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.7179 on 63 degrees of freedom
Multiple R-squared:  0.08233,   Adjusted R-squared:  0.06776
F-statistic: 5.652 on 1 and 63 DF,  p-value: 0.02048

> |
```

2) The goal of this exercise to see how large n should be for the large-sample and the (parametric) bootstrap percentile method confidence intervals for the mean of an exponential population to be accurate. To be specific, let X1, . . . , Xn represent a random sample from an exponential (λ) distribution. Note that this distribution is skewed and its mean is μ = 1/λ. We can construct two confidence intervals for μ — one the large-sample z-interval (interval 1) and the other a (parametric) bootstrap percentile method interval (interval 2). We would like to investigate their accuracy, i.e., how close their estimated coverage probabilities are to the assumed nominal level of confidence, for various combinations of (n, λ). This investigation will focus on $1 - \alpha = 0.95$, λ = 0.01, 0.1, 1, 10 and n = 5, 10, 30, 100. Thus, we have a total of $4 * 4 = 16$ combinations of (n, λ) to investigate.

(a) For a given setting, compute Monte Carlo estimates of coverage probabilities of the two intervals by simulating appropriate data, using them to construct the two confidence intervals, and repeating the process 5000 times.

Sol :

To simulate the Monte Carlo estimates of coverage probabilities of the two intervals, we have created the following functions :

Checkz_func : takes n and lambda as inputs , simulates a sample, constructs an interval and returns whether the true mean exists within the confidence interval.

Zproportion : takes n and lambda as inputs, calls the checkz_func 5000 times and calculates the coverage probabilities.

Checkb_func : takes n and lambda as inputs , it calls the myFunc_mean function 1000 times, constructs an interval and returns whether the true mean exists within the confidence interval.

Bproportion : takes n and lambda as inputs, constructs a parametric initial bootstrap sample and calls checkb_func 5000 times and calculates the coverage probabilities.

```
> ### Question 2 (a) ###
>
> ## Creating function to check if true mean exists within the confidence interval ##
> checkz_func = function(n,lambda) {
+   u = rexp(n,lambda)
+
+   lower_bound = mean(u) - qnorm(0.975) * sd(u) / sqrt(n)
+   upper_bound = mean(u) + qnorm(0.975) * sd(u) / sqrt(n)
+
+   sm = 1/lambda
+
+   if(upper_bound > sm  & lower_bound < sm){
+     return (1)
+   }
+   else
+   {
+     return (0)
+   }
+ }
>
> ## calling the function 5000 times and checking the probability ##
> zproportion = function(n, lambda) {
+   values = replicate(5000, checkz_func(n, lambda))
+   no_ones = values[which(values == 1)]
+   return (length(no_ones)/5000)
+ }
>
>
> ## checking for n = 10 and lambda = 0.1 ##
> zproportion(10, 0.1)
[1] 0.8724
>
```

```
>
> ## Creating a function to return the mean ##
> myFunc_mean = function(n, lambda){
+   u = rexp(n, lambda)
+   return (mean(u))
+ }
>
> ## Calls the myFunc_mean 1000 times and forms the confidence intervals and returns whether the true mean is present in the constru
cted interval ##
>
>
> checkb_func = function(n, lambda){
+   u = rexp(n, lambda)
+   sm = 1/lambda
+   lambda_temp = 1/mean(u)
+
+   values = replicate(1000,myFunc_mean(n,lambda_temp))
+   bounds = sort(values)[c(25, 975)]
+
+   if(bounds[2] > sm & bounds[1] < sm){
+     return (1)
+   }
+   else
+   {
+     return (0)
+   }
+ }
>
> ## Creating a function for the parametric bootstrap sample and calls the checkb_func 5000 times to calculate the coverage probabil
ities ##
>
> bproportion = function(n, lambda){
+   values = replicate(5000, checkb_func(n, lambda))
+   no_ones = values[which(values == 1)]
+   return (length(no_ones)/5000)
+ }
>
> ## Checking the bproportion for n = 10 and lambda = 0.1 ##
>
> bproportion(10, 0.1)
[1] 0.9234
```

For (n, λ) combination of (10, 0.1), 0.8724 is the coverage probability of Z interval and 0.9234 is the coverage probability of bootstrap interval

(b) Repeat (a) for the remaining combinations of (n, λ). Present an appropriate summary of the results.

```
>
> ## Question 2 (b) ###
>
> ### For various values of n and lambda calculating the zproportion and bproportion ###
>
> n_values = c(5,10,30,100)
> lambda_values = c(0.01, 0.1, 1, 10)
>
> n_len = length(n_values)
> lambda_len = length(lambda_values)
>
> zMatrix = matrix(NA, nrow = n_len, ncol = lambda_len)
> bMatrix = matrix(NA, nrow = n_len, ncol = lambda_len)
>
> for(i in 1:n_len){
+   for(j in 1:lambda_len){
+     zMatrix[i,j]= zproportion(n_values[i],lambda_values[j])
+     bMatrix[i,j] = bproportion(n_values[i],lambda_values[j])
+   }
+ }
>
> zMatrix
       [,1]   [,2]   [,3]   [,4]
[1,] 0.8078 0.8096 0.8064 0.8272
[2,] 0.8688 0.8694 0.8798 0.8758
[3,] 0.9124 0.9176 0.9168 0.9222
[4,] 0.9402 0.9398 0.9402 0.9384
>
> bMatrix
       [,1]   [,2]   [,3]   [,4]
[1,] 0.8982 0.8948 0.8844 0.8966
[2,] 0.9122 0.9184 0.9266 0.9220
[3,] 0.9394 0.9388 0.9448 0.9352
[4,] 0.9474 0.9442 0.9474 0.9428
>
>
```

coverage probabilities of Z interval

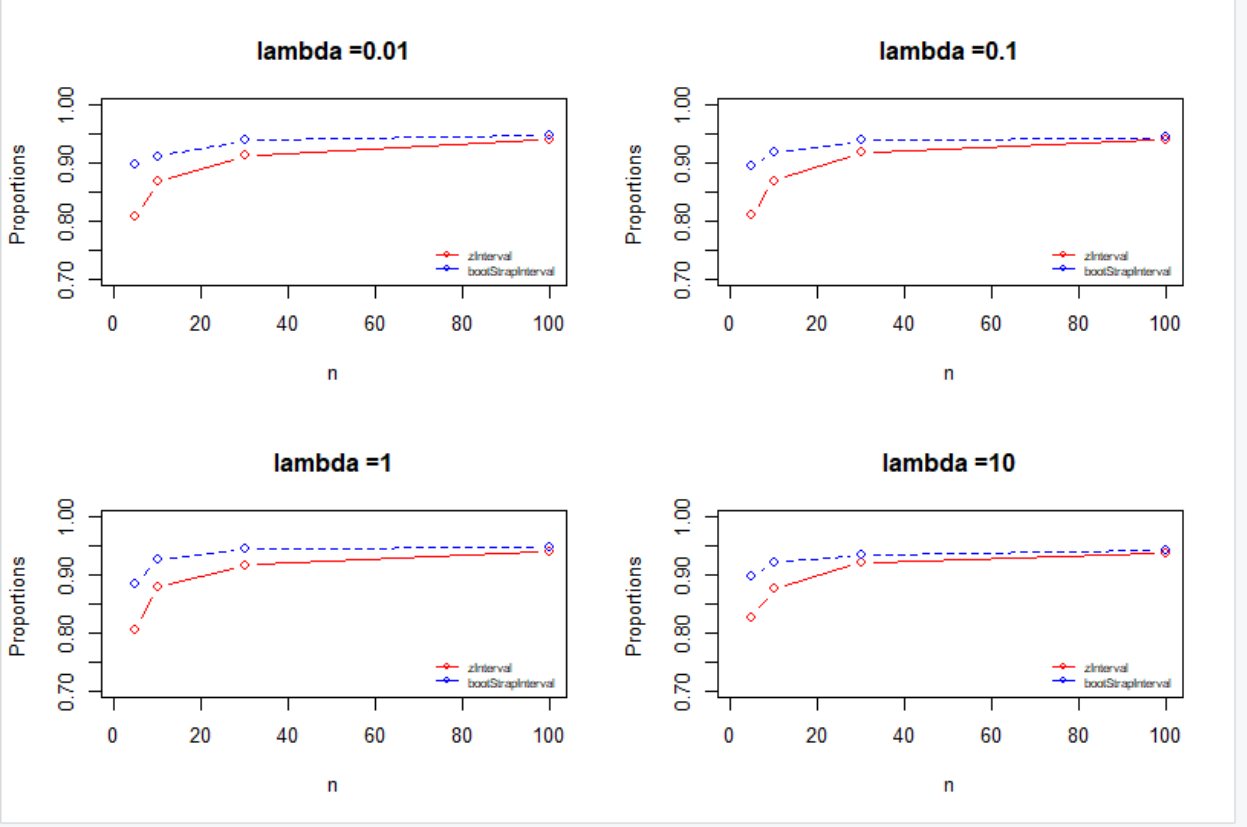
	λ = 0.01	λ = 0.1	λ = 1	λ = 10
n = 5	0.8078	0.8096	0.8064	0.8272
n = 10	0.8688	0.8694	0.8798	0.8758
n = 30	0.9124	0.9176	0.9168	0.9222
n = 100	0.9402	0.9398	0.9402	0.9384

coverage probabilities of bootstrap interval

	λ = 0.01	λ = 0.1	λ = 1	λ = 10
n = 5	0.8982	0.8948	0.8844	0.8966
n = 10	0.9122	0.9184	0.9266	0.9220
n = 30	0.9394	0.9388	0.9448	0.9352
n = 100	0.9474	0.9442	0.9474	0.9428

Graphically representing the data , we get the below graphs :

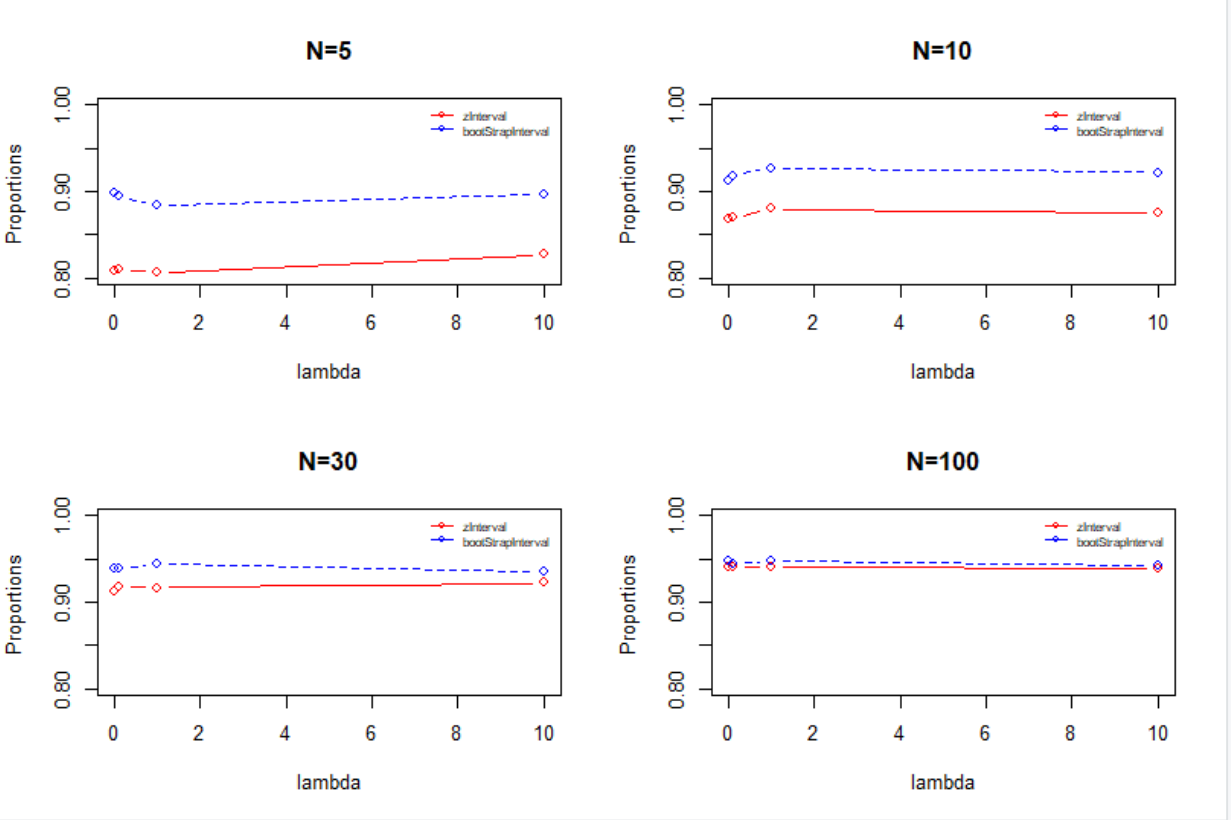
Figure - 1



The red line represents zinterval proportions whereas the blue line represents bootstrap interval proportions. The graphs are drawn varying n keeping lambda fixed

```
>
> ## Plotting the results graphically ##
>
> par(mfrow = c(2,2))
> for(i in 1:lambda_len){
+   plot(n_values,zMatrix[,i],type = 'b',lty = 1,xlab = 'n', ylab = 'Proportions ',col ='red',xlim=c(1,100),ylim=c(0.7,1),main =past
e0("lambda =",lambda_values[i]))
+   lines(n_values, bMatrix[,i],lty = 2,col = 'blue',type = 'b')
+   legend("bottomright",legend = c("zInterval","bootStrapInterval"),col = c('red','blue'),text.col = c('black','black'),lty = 1, pc
h =
+            1, inset = 0.01,ncol = 1, cex = 0.6,bty = 'n')
+ }
>
```

Figure - 2



The red line represents zinterval proportions whereas the blue line represents bootstrap interval proportions. The graphs are drawn varying lambda values keeping n fixed

```
>
> par(mfrow = c(2,2))
> for(i in 1:n_len){
+   plot(lambda_values,zMatrix[i,],type = 'b',lty = 1,xlab = 'lambda', ylab = 'Proportions ',col ='red',xlim=c(0.01,10),ylim=c(0.8,
1),main =paste0("N=",n_values[i]))
+   lines(lambda_values, bMatrix[i,],lty = 2,col = 'blue',type = 'b')
+   legend("topright",legend = c("zInterval","bootStrapInterval"),col = c('red','blue'),text.col = c('black','black'),lty = 1, pch =
+            1, inset = 0.01,ncol = 1, cex = 0.6,bty = 'n')
+ }
>
```

(c) Interpret all the results. Be sure to answer the following questions: In case of the large-sample interval, how large n is needed for the interval to be accurate? Likewise, in case of the bootstrap interval, how large n is needed for the interval to be accurate? Do these answers depend on λ? Can we say that one method is more accurate than the other? Which interval would you recommend? Provide justification for all your conclusions.

Sol :

From the graphs in the figure 1, we can observe the following :

(i) The graphs for z-interval and bootstrap interval do not change a lot even when lambda values changes. By this, we can conclude that the coverage probabilities do not depend on the value of the lambda.
(ii) The coverage probabilities we get from the z interval are less than those we get from the bootstrap method.

From the graphs in the figure 2, we can observe the following :

(i) The coverage probabilities depend on the value of n as we can observe the changes in the graphs with the change in the n value.
(ii) When n is large (i.e n = 100) the coverage probabilities for the large sample z interval are as accurate as the coverage probabilities obtained from the bootstrap method.
(iii) From n = 30 onwards, the coverage probabilities from zinterval method are higher.
(iv) But even for low values of n , the bootstrap method coverage probabilities are more accurate.
(v) Considering all the observations, bootstrap method coverage probabilities are usually higher for any combination of n and lambda than those for the large sample z interval method.

Since, bootstrap method gives accurate probabilities for the lower values of n as well, we would recommend using the bootstrap method.

(d) Do your conclusions in (c) depend on the specific values of λ that were fixed in advance? Explain.

Sol :

The conclusions in (c) does not depend on the the specific values of lambda because lambda is a parameter of the population distribution. Also, the conclusions about the convergence of the sampling methods shouldn't depend on the population parameters.

Section -2

R codes for the 2 questions :

```
#### Question 1 ####

### Reading the body temperature and heart rate data from the csv data ###

bodytemp_hrate = read.csv("C:\\Users\\hxt210018\\Downloads\\6313_Prob\\HW\\bodytemp-heartrate.csv");

## Separating the male and female data ##

male_data = subset(bodytemp_hrate,bodytemp_hrate$gender == 1)

female_data = subset(bodytemp_hrate, bodytemp_hrate$gender == 2)

## Getting the summary for the male_data and female data ##

summary(male_data$body_temperature)

summary(female_data$body_temperature)

## Drawing the boxplots ##

boxplot(male_data$body_temperature,female_data$body_temperature, main = "Boxplots of body temperatures", names = c('Male','Female'), ylab = "Body Temperature")

## Drawing the QQ plots for the body temperatures ##

par(mfrow = c(1,2))

qqnorm(male_data$body_temperature, main = 'QQ Plot for Male Body Temperature')

qqline(male_data$body_temperature)

qqnorm(female_data$body_temperature, main = 'QQ Plot for Female Body Temperature')

qqline(female_data$body_temperature)

### Calculating confidence intervals using the t-test for the body temperture ###

t.test(male_data$body_temperature, female_data$body_temperature,alternative = 'two.sided',var.equal = FALSE)

#### Question 1 part (b) ####

## Getting the summary for the male_data and female data heart_rate ##

summary(male_data$heart_rate)

summary(female_data$heart_rate)
```

```
## Drawing the boxplots ##

boxplot(male_data$heart_rate,female_data$heart_rate, main = "Boxplots of Heart rate",
names = c('Male','Female'), ylab = "Heart rate")

## Drawing the QQ plots for the Heart rate ##

par(mfrow = c(1,2))

qqnorm(male_data$heart_rate, main = 'QQ Plot for Male Heart rate')

qqline(male_data$heart_rate)

qqnorm(female_data$heart_rate, main = 'QQ Plot for Female Heart rate')

qqline(female_data$heart_rate)

### Calculating confidence intervals using the t-test for the heart rate  ###

t.test(male_data$heart_rate, female_data$heart_rate,alternative = 'two.sided',var.equal =
FALSE)

## Question 1 Part (c) ##

##Finding the correlation between the body temperature and heart rate values irrespective of
gender ##

par(mfrow=c(1,1))

cor(bodytemp_hrate$body_temperature, bodytemp_hrate$heart_rate)

plot(bodytemp_hrate$heart_rate, bodytemp_hrate$body_temperature, pch = 1, main =
'Scatter plot for all data')

abline(lm(bodytemp_hrate$body_temperature ~ bodytemp_hrate$heart_rate))

    linear_model = lm(bodytemp_hrate$body_temperature ~ bodytemp_hrate$heart_rate)

print(linear_model)

summary(linear_model)

## Finding the correlation between the body temperature and heart rate values for male and
female ##

cor(male_data$body_temperature,male_data$heart_rate)

cor(female_data$body_temperature, female_data$heart_rate)

## Drawing the scatter plots for the body temperatures and heart rate values for male and
female ##
```

```
par(mfrow = c(1,2))
plot(male_data$heart_rate, male_data$body_temperature, pch = 1, main = 'Scatter plot for male')
abline(lm(male_data$body_temperature ~ male_data$heart_rate))
linear_model = lm(male_data$body_temperature ~ male_data$heart_rate)
print(linear_model)
summary(linear_model)
plot(female_data$heart_rate, female_data$body_temperature, pch = 1, main = 'Scatter plot for Female')
abline(lm(female_data$body_temperature ~ female_data$heart_rate))
linear_model = lm(female_data$body_temperature ~ female_data$heart_rate)
print(linear_model)
summary(linear_model)
### Question 2 (a) ###
## Creating function to check if true mean exists within the confidence interval ##
checkz_func = function(n,lambda) {
  u = rexp(n,lambda)
 lower_bound = mean(u) - qnorm(0.975) * sd(u) / sqrt(n)
  upper_bound = mean(u) + qnorm(0.975) * sd(u) / sqrt(n)
   sm = 1/lambda
   if(upper_bound > sm  & lower_bound < sm){
    return (1)
  }
  else
  {
    return (0)
  }
}
```

```
## calling the function 5000 times and checking the probability ##
zproportion = function(n, lambda) {
  values = replicate(5000, checkz_func(n, lambda))
  no_ones = values[which(values == 1)]
  return (length(no_ones)/5000)
}
## checking for n = 10 and lambda = 0.1 ##
zproportion(10, 0.1)
## Creating a function to return the mean ##
myFunc_mean = function(n, lambda){
  u = rexp(n, lambda)
  return (mean(u))
}
## Calls the myFunc_mean 1000 times and forms the confidence intervals and returns whether
the true mean is present in the constructed interval ##
checkb_func = function(n, lambda){
  u = rexp(n, lambda)
  sm = 1/lambda
  lambda_temp = 1/mean(u)
  values = replicate(1000,myFunc_mean(n,lambda_temp))
  bounds = sort(values)[c(25, 975)]
  if(bounds[2] > sm & bounds[1] < sm){
    return (1)
  }
  else
  {
    return (0)
```

```
  }
}
## Creating a function for the parametric bootstrap sample and calls the checkb_func 5000 times to calculate the coverage probabilities ##
bproportion = function(n, lambda){
  values = replicate(5000, checkb_func(n, lambda))
  no_ones = values[which(values == 1)]
  return (length(no_ones)/5000)
}
## Checking the bproportion for n = 10 and lambda = 0.1 ##
bproportion(10, 0.1)
## Question 2 (b) ###
### For various values of n and lambda calculating the zproportion and bproportion ###
n_values = c(5,10,30,100)
lambda_values = c(0.01, 0.1, 1, 10)
n_len = length(n_values)
lambda_len = length(lambda_values)
zMatrix = matrix(NA, nrow = n_len, ncol = lambda_len)
bMatrix = matrix(NA, nrow = n_len, ncol = lambda_len)
for(i in 1:n_len){
  for(j in 1:lambda_len){
    zMatrix[i,j]= zproportion(n_values[i],lambda_values[j])
    bMatrix[i,j] = bproportion(n_values[i],lambda_values[j])
  }
}
zMatrix
bMatrix
```

```
## Plotting the results graphically ##

par(mfrow = c(2,2))

for(i in 1:lambda_len){

  plot(n_values,zMatrix[,i],type = 'b',lty = 1,xlab = 'n', ylab = 'Proportions ',col
='red',xlim=c(1,100),ylim=c(0.7,1),main =paste0("lambda =",lambda_values[i]))

  lines(n_values, bMatrix[,i],lty = 2,col = 'blue',type = 'b')

  legend("bottomright",legend = c("zInterval","bootStrapInterval"),col = c('red','blue'),text.col =
c('black','black'),lty = 1, pch =

        1, inset = 0.01,ncol = 1, cex = 0.6,bty = 'n')

}

par(mfrow = c(2,2))

for(i in 1:n_len){

  plot(lambda_values,zMatrix[i,],type = 'b',lty = 1,xlab = 'lambda', ylab = 'Proportions ',col
='red',xlim=c(0.01,10),ylim=c(0.8,1),main =paste0("N=",n_values[i]))

  lines(lambda_values, bMatrix[i,],lty = 2,col = 'blue',type = 'b')

  legend("topright",legend = c("zInterval","bootStrapInterval"),col = c('red','blue'),text.col =
c('black','black'),lty = 1, pch =

        1, inset = 0.01,ncol = 1, cex = 0.6,bty = 'n')

}
```