Lim Zheng Haur

32023952

FIT2086 Assignment 3

Question 1

1.
```
> # Question 1.1
> # read file
> df = read.csv("fuel.ass3.2022.csv", stringsAsFactors = T)
> # fitting a linear model to the data
> fit <- lm(Comb.FE ~ ., data = df)
> # view summary of data
> summary(fit)

Call:
lm(formula = Comb.FE ~ ., data = df)

Residuals:
    Min      1Q  Median      3Q     Max
-3.7617 -1.0503 -0.0885  0.7359 11.3772

Coefficients:
                           Estimate Std. Error t value Pr(>|t|)
(Intercept)              -2.167e+02  1.550e+02  -1.398 0.162706
Model.Year                1.156e-01  7.679e-02   1.505 0.132856
Eng.Displacement         -1.331e+00  1.861e-01  -7.151 3.22e-12 ***
No.Cylinders              5.730e-03  1.206e-01   0.048 0.962117
AspirationOT             -1.034e-01  1.240e+00  -0.083 0.933569
AspirationSC             -7.990e-01  4.064e-01  -1.966 0.049842 *
AspirationTC             -1.217e+00  2.201e-01  -5.528 5.31e-08 ***
AspirationTS             -1.351e+00  6.720e-01  -2.010 0.044935 *
No.Gears                 -1.940e-01  5.158e-02  -3.760 0.000191 ***
Lockup.Torque.ConverterY -5.621e-01  1.974e-01  -2.847 0.004602 **
Drive.SysA                6.138e-02  2.706e-01   0.227 0.820624
Drive.SysF                1.535e+00  2.930e-01   5.239 2.41e-07 ***
Drive.SysP               -9.766e-01  5.639e-01  -1.732 0.083967 .
Drive.SysR                2.081e-01  2.551e-01   0.816 0.415071
Max.Ethanol              -8.956e-03  6.100e-03  -1.468 0.142704
Fuel.TypeGM               8.096e-01  1.004e+00   0.806 0.420647
Fuel.TypeGP               4.064e-01  2.425e-01   1.676 0.094372 .
Fuel.TypeGPR              8.418e-02  2.458e-01   0.343 0.732106
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 1.688 on 482 degrees of freedom
Multiple R-squared:  0.6571,  Adjusted R-squared:  0.645
F-statistic: 54.34 on 17 and 482 DF,  p-value: < 2.2e-16
```

We consider a predictor as being associated with the target when it has a small p-value

which is suggestive that the data is at odds with the null hypothesis of no association. The predictors that are possibly associated with fuel efficiency are Eng.Displacement, AspirationSC, AspirationTC, AspirationTS, No.Gears, Lockup.Torque.ConverterY, Drive.SysF, Drive.SysP, and Fuel.TypeGP. This is due to all these predictors having a p-value of lower than 0.1. The three variables which appear to be strongest predictors of fuel efficiency are Eng.Displacement, AspirationTC, and Drive.SysF. This is because they have the three lowest p-values among all predictors at 3.22e-12, 5.31e-08, and 2.41e-07 respectively.

2.
```
> # Question 1.2
> # critical value
> a = 0.05
> # number of comparisons
> n = 17
> # adjusted critical value
> a = a/n
> a
[1] 0.002941176
```

In Bonferroni procedure, we adjust the critical value by dividing the original critical value, $a = 0.05$ by the number of comparisons made, $n = 17$. This results in the new critical p-value as $a = 0.002941176$. Hence, as the adjusted critical p-value is lower, our assessment of which predictors are associated would change as there would be lesser predictors with lower p-value then our adjusted critical p-value.

3.
```
> # Question 1.3
> # fitting linear model to only engine displacement
> fit.Eng.Displacement <- lm(Comb.FE ~ Eng.Displacement, data = df)
> fit.Eng.Displacement

Call:
lm(formula = Comb.FE ~ Eng.Displacement, data = df)

Coefficients:
     (Intercept)  Eng.Displacement
          15.283            -1.507

> # fitting liear model to only drive systems
> fit.Drive.Sys <- lm(Comb.FE ~ Drive.Sys, data = df)
> fit.Drive.Sys

Call:
lm(formula = Comb.FE ~ Drive.Sys, data = df)

Coefficients:
(Intercept)   Drive.SysA   Drive.SysF   Drive.SysP   Drive.SysR
     9.1245       1.0142       4.0422      -1.0681       0.1855
```

When engine displacement (in litre) is 0ℓ, the predicted value of fuel efficiency (in kilometres per litre) is 15.283km/ℓ. For every unit increase in engine displacement, the predicted value of fuel efficiency (in kilometres per litre) decreases by 1.507km/ℓ.

A car with a 4-wheel drive system has a predicted value of fuel efficiency (in kilometres per litre) of 9.1245km/ℓ. A car with front-wheel drive system will have an increase of 4.0422km/ℓ of fuel efficiency from a 4-wheel drive car. Resulting in a predicted value of fuel efficiency of 13.1667km/ℓ.

4.
```
> # Question 1.4
> # stepwise selection procedure
> fit.bic <- step(fit, k = log(500), direction = "both")
...
> fit.bic

Call:
lm(formula = Comb.FE ~ Eng.Displacement + Aspiration + No.Gears +
    Lockup.Torque.Converter + Drive.Sys, data = df)

Coefficients:
             (Intercept)  Eng.Displacement   AspirationOT              AspirationSC
                 16.3612           -1.3165         0.1337                   -0.5706
           AspirationTC      AspirationTS       No.Gears  Lockup.Torque.ConverterY
                 -1.0717           -1.3249        -0.1748                   -0.5732
          Drive.SysA        Drive.SysF      Drive.SysP                Drive.SysR
                 0.1934            1.5475        -1.0802                    0.2842
```

$$\begin{aligned}\mathbb{E}[\ \texttt{Comb.FE}\ ] = {} & 16.3612 - 1.3165 \times \texttt{Eng.Displacement} \\ & + 0.1337 \times \texttt{AspirationOT} - 0.5706 \times \texttt{AspirationSC} \\ & - 1.0717 \times \texttt{AspirationTC} - 1.3249 \times \texttt{AspirationTS} \\ & - 0.1748 \times \texttt{No.Gears} - 0.5732 \times \texttt{Lockup.Torque.ConverterY} \\ & + 0.1934 \times \texttt{Drive.SysA} + 1.5475 \times \texttt{Drive.SysF} \\ & - 1.0802 \times \texttt{Drive.SysP} + 0.2842 \times \texttt{Drive.SysP}\end{aligned}$$

5.
```
> # Question 1.5
> # predicting for row 33
> predict(fit.bic, newdata = df[33,], interval = "confidence")
       fit      lwr      upr
33 13.37209 12.99409 13.75009
```

a. **The predicted mean fuel efficiency for this new car is 13.37209km/ℓ. The 95% confidence interval for this prediction is [12.99409, 13.75009]km/ℓ.**

b. **Yes, the new car will have better fuel efficiency than my current car. Using the 95% confidence interval, we could be 95% confident that the lower bound of the mean fuel efficiency for the new car is higher than my current car. Hence, we are 95% confident that the new car has better fuel efficiency than my current car.**

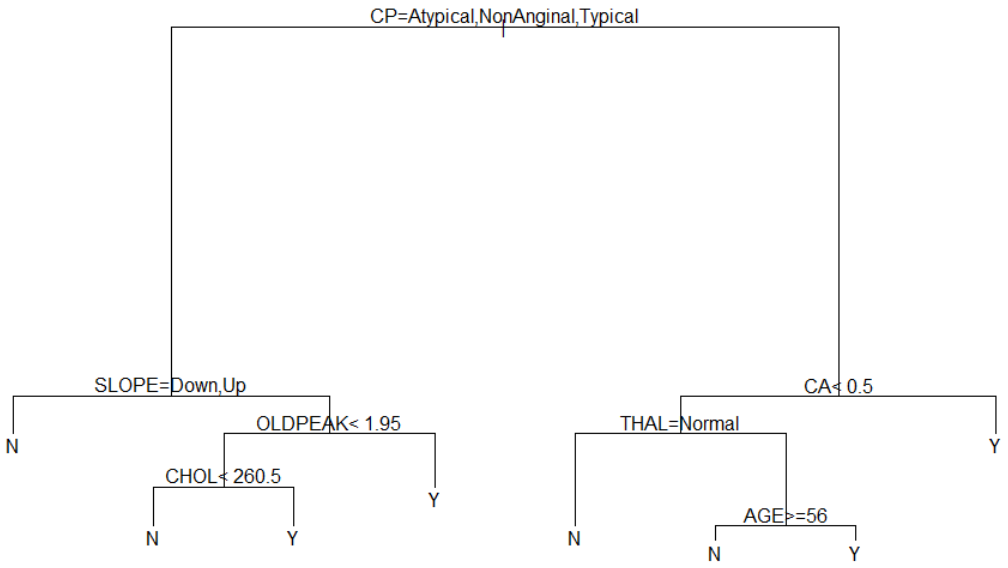
Question 2

1.
```
> # Question 2.1
> # read file
```

```
> heart.train <- read.csv("heart.train.ass3.2022.csv",
stringsAsFactor = T)
> # fitting a decision tree
> tree.heart <- rpart(HD ~ ., heart.train)
> # cross-validation with 10 folds and 5000 repetitions
> cv <- learn.tree.cv(HD ~ .,data = heart.train, nfolds = 10, m =
5000)
> # best tree
> cv$best.tree
n= 210

node), split, n, loss, yval, (yprob)
      * denotes terminal node

 1) root 210 99 N (0.52857143 0.47142857)
   2) CP=Atypical,NonAnginal,Typical 106 23 N (0.78301887
0.21698113)
     4) SLOPE=Down,Up 69  6 N (0.91304348 0.08695652) *
     5) SLOPE=Flat 37 17 N (0.54054054 0.45945946)
      10) OLDPEAK< 1.95 30 10 N (0.66666667 0.33333333)
        20) CHOL< 260.5 21  3 N (0.85714286 0.14285714) *
        21) CHOL>=260.5 9  2 Y (0.22222222 0.77777778) *
      11) OLDPEAK>=1.95 7  0 Y (0.00000000 1.00000000) *
   3) CP=Asymptomatic 104 28 Y (0.26923077 0.73076923)
     6) CA< 0.5 47 23 N (0.51063830 0.48936170)
      12) THAL=Normal 21  4 N (0.80952381 0.19047619) *
      13) THAL=Fixed.Defect,Reversible.Defect 26  7 Y (0.26923077
0.73076923)
        26) AGE>=56 8  3 N (0.62500000 0.37500000) *
        27) AGE< 56 18  2 Y (0.11111111 0.88888889) *
     7) CA>=0.5 57  4 Y (0.07017544 0.92982456) *
```

The variables used in the best tree are CP, SLOPE, OLDPEAK, CHOL, CA, THAL, and AGE. There are 8 leaves (terminal nodes) in the best tree.

2.
```
> # Question 2.2
> # plotting the tree
> plot(cv$best.tree)
> text(cv$best.tree, pretty = 12)
```



The tree predicts that someone has heart disease when:

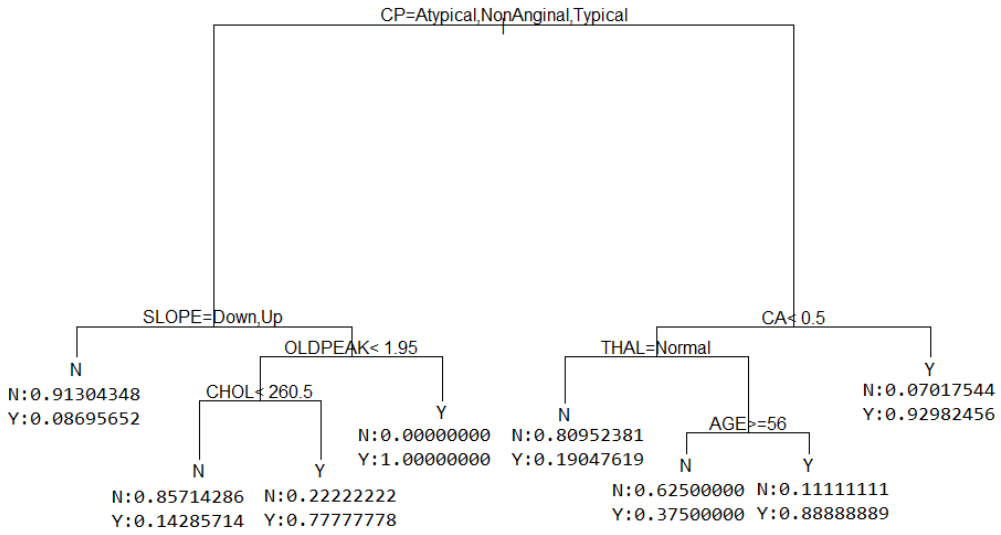
- **They have Atypical or NonAnginal or Typical CP, SLOPE is Flat, OLDPEAK is smaller than 1.95, and CHOL is greater or equal to 260.5.**
- **They have Atypical or NonAnginal or Typical CP, the SLOPE is Flat, and OLDPEAK is greater or equal to 1.95.**
- **They have Asymtomatic CP, CA is smaller than 0.5, THAL is not Normal, and AGE is smaller than 56.**
- **They have Asymtomatic CP, and CA is greater or equal to 0.5.**

The tree predicts someone does not have heart disease when:

- **They have Atypical or NonAnginal or Typical CP, and SLOPE is Down or Up.**
- **They have Atypical or NonAnginal or Typical CP, SLOPE is Flat, OLDPEAK is smaller than 1.95, and CHOL is smaller than 260.5.**
- **They have Asymtomatic CP, CA is smaller than 0.5, and THAL is Normal.**
- **They have Asymtomatic CP, CA is smaller than 0.5, THAL is not Normal and AGE is greater or equal to 56.**

3.
```
> # Question 2.3
> cv$best.tree
n= 210

node), split, n, loss, yval, (yprob)
      * denotes terminal node

 1) root 210 99 N (0.52857143 0.47142857)
   2) CP=Atypical,NonAnginal,Typical 106 23 N (0.78301887
0.21698113)
     4) SLOPE=Down,Up 69  6 N (0.91304348 0.08695652) *
     5) SLOPE=Flat 37 17 N (0.54054054 0.45945946)
      10) OLDPEAK< 1.95 30 10 N (0.66666667 0.33333333)
        20) CHOL< 260.5 21  3 N (0.85714286 0.14285714) *
        21) CHOL>=260.5 9  2 Y (0.22222222 0.77777778) *
      11) OLDPEAK>=1.95 7  0 Y (0.00000000 1.00000000) *
   3) CP=Asymptomatic 104 28 Y (0.26923077 0.73076923)
```

```
      6) CA< 0.5 47 23 N (0.51063830 0.48936170)
       12) THAL=Normal 21  4 N (0.80952381 0.19047619) *
       13) THAL=Fixed.Defect,Reversible.Defect 26  7 Y (0.26923077
0.73076923)
         26) AGE>=56 8  3 N (0.62500000 0.37500000) *
         27) AGE< 56 18  2 Y (0.11111111 0.88888889) *
      7) CA>=0.5 57  4 Y (0.07017544 0.92982456) *
```



4. **Atypical or NonAnginal or Typical CP, and Down or Up SLOPE. The probability of having heart disease is 0.08695652.**

5.
```
> # Question 2.5
> # fitting a logistic regression model
> fullmod <- glm(HD ~ ., data = heart.train, family = binomial)
> # stepwise selection procedure
> fullmod.kic <- step(fullmod, k = 3, direction = "both")
...
> summary(fullmod.kic)

Call:
glm(formula = HD ~ SEX + CP + TRESTBPS + CHOL + OLDPEAK + SLOPE +
    CA, family = binomial, data = heart.train)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.4071  -0.4679  -0.1246   0.4014   2.6684

Coefficients:
              Estimate Std. Error z value Pr(>|z|)
(Intercept)  -7.248413   2.232140  -3.247 0.001165 **
SEXM          1.802856   0.533646   3.378 0.000729 ***
CPAtypical   -2.184817   0.703118  -3.107 0.001888 **
CPNonAnginal -2.599144   0.558932  -4.650 3.32e-06 ***
CPTypical    -2.369844   0.753460  -3.145 0.001659 **
```

```
TRESTBPS      0.021501   0.011787   1.824 0.068140 .
CHOL          0.008167   0.004200   1.944 0.051854 .
OLDPEAK       0.581819   0.260840   2.231 0.025710 *
SLOPEFlat     1.931508   0.994042   1.943 0.052006 .
SLOPEUp       0.206602   1.086994   0.190 0.849257
CA            1.074811   0.285071   3.770 0.000163 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 290.44  on 209  degrees of freedom
Residual deviance: 143.52  on 199  degrees of freedom
AIC: 165.52

Number of Fisher Scoring iterations: 6
```

The final model includes variables SEXM, CPAtypical, CPNonAnginal, CPTypical, TRESTBPS, CHOL, OLDPEAK, SLOPEFlat, SLOPEUp, and CA.

SEX and TRESTBPS is included in the final model but not in the tree estimated by CV. On the other hand, THAL and AGE is included in the tree estimated by CV but not in the final model. Hence, the total variables in both the final model and tree estimated by CV are the same.

In logistic regression, the most important predictor is CPNonAnginal. This is because it has the lowest p-value of 3.32e-06. A low p-value indicates that the data is at odds with the null hypothesis of no association.

6. $$\begin{aligned}\mathbb{E}[\ \texttt{HD}\ ] = &-7.248413 + 1.802856 \times \texttt{SEXM} \\ &- 2.184817 \times \texttt{CPAtypical} - 2.599144 \times \texttt{CPNonAnginal} \\ &- 2.369844 \times \texttt{CPTypical} + 0.021501 \times \texttt{TRESTBPS} \\ &+ 0.008167 \times \texttt{CHOL} + 0.581819 \times \texttt{OLDPEAK} \\ &+ 1.931508 \times \texttt{SLOPEFlat} + 0.206602 \times \texttt{SLOPEUp} \\ &+ 1.074811 \times \texttt{CA}\end{aligned}$$

7. **The greater the number of major vessels coloured by fluoroscopy, the greater the probability of the presence of heart diseases.**

8.
```
> # Question 2.8
> # read file
> heart.test <- read.csv("heart.test.ass3.2022.csv", stringsAsFactor
= T)
> # compute prediction statistics
> my.pred.stats(predict(cv$best.tree, heart.test)[,2],
heart.test$HD)
-----------------------------------------------------------------
Performance statistics:
```

```
Confusion matrix:

    target
pred  N  Y
   N 47 14
   Y  6 25

Classification accuracy = 0.7826087
Sensitivity             = 0.6410256
Specificity             = 0.8867925
Area-under-curve        = 0.8214804
Logarithmic loss        = 87.37257

------------------------------------------------------------------------
> my.pred.stats(predict(fullmod.kic, heart.test, type = "response"),
heart.test$HD)
------------------------------------------------------------------------
Performance statistics:

Confusion matrix:

    target
pred  N  Y
   N 45  8
   Y  8 31

Classification accuracy = 0.826087
Sensitivity             = 0.7948718
Specificity             = 0.8490566
Area-under-curve        = 0.8853411
Logarithmic loss        = 39.43705

------------------------------------------------------------------------
```

The step-wise logistic regression model has a higher classification accuracy of 0.826087 than the tree of 0.7826087. It also has a higher sensitivity of 0.7948718 than the tree of 0.6410256. On the other hand, the tree has a higher specificity of 0.8867925 than the step-wise logistic regression model of 0.8490566. Both the area-under-curve and logarithmic loss also shows that the step-wise logistic regression has a higher accuracy than the tree. Hence, the step-wise logistic regression model is a better model than the tree. This is because the model is used for identifying patients with heart diseases. For this purpose, we require a higher sensitivity model as we prefer to not have any false negatives for disease detection. A higher specificity model has lower false positives which is not our priority at disease detection.

9.

a.
```
> # Question 2.9a
> # predicting using tree
> predict(cv$best.tree, heart.test[10,])
```

```
             N         Y
10 0.07017544 0.9298246
```

$$Odds = \frac{0.9298246}{1 - 0.9298246} = 132.500$$

b.
```
> # Question 2.9b
> # predicting using step-wise logistic regression
> predict(fullmod.kic, heart.test[10,], type = "response")
       10
0.9994987
```

$$Odds = \frac{0.9994987}{1 - 0.9994987} = 1993.813$$

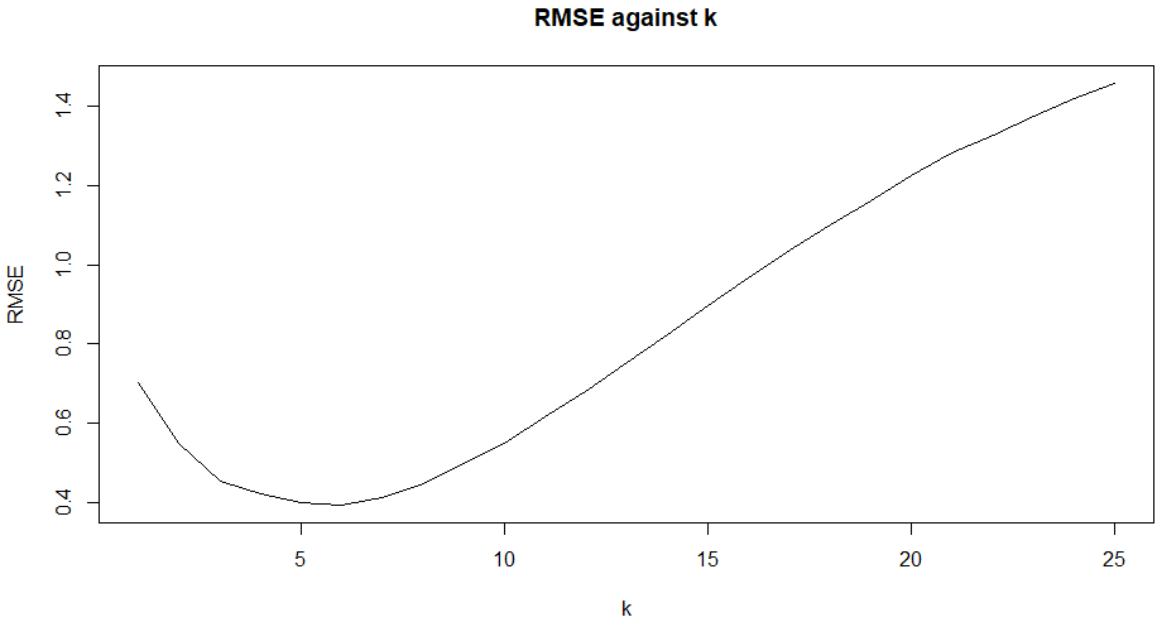
10.
```
> # Question 2.10
> boot.65th <- function(formula, data, indices)
+ {
+   # Create a bootstrapped version of our data
+   d <- data[indices,]
+   # Fit a logistic regression to the bootstrapped data
+   fit <- glm(formula, d, family = binomial)
+   # Compute the prediction for 65th patient and return it
+   return (predict(fit, heart.test[65,], type = "response"))
+ }
> bs.65th <- boot(data = heart.train, statistic = boot.65th, R =
5000, formula = fullmod.kic)
> boot.ci(bs.65th, conf = 0.95, type = "bca")
BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
Based on 5000 bootstrap replicates

CALL :
boot.ci(boot.out = bs.65th, conf = 0.95, type = "bca")

Intervals :
Level       BCa
95%   ( 0.8219,  0.9916 )
Calculations and Intervals on Original Scale
Some BCa intervals may be unstable
> boot.66th <- function(formula, data, indices)
+ {
+   # Create a bootstrapped version of our data
+   d <- data[indices,]
+   # Fit a logistic regression to the bootstrapped data
+   fit <- glm(formula, d, family = binomial)
+   # Compute the prediction for 66th patient and return it
+   return (predict(fit, heart.test[66,], type = "response"))
+ }
```

```
> bs.66th <- boot(data = heart.train, statistic = boot.66th, R =
5000, formula = fullmod.kic)
> boot.ci(bs.66th, conf = 0.95, type = "bca")
BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
Based on 5000 bootstrap replicates

CALL :
boot.ci(boot.out = bs.66th, conf = 0.95, type = "bca")

Intervals :
Level       BCa
95%   ( 0.0285,  0.2408 )
Calculations and Intervals on Original Scale
```

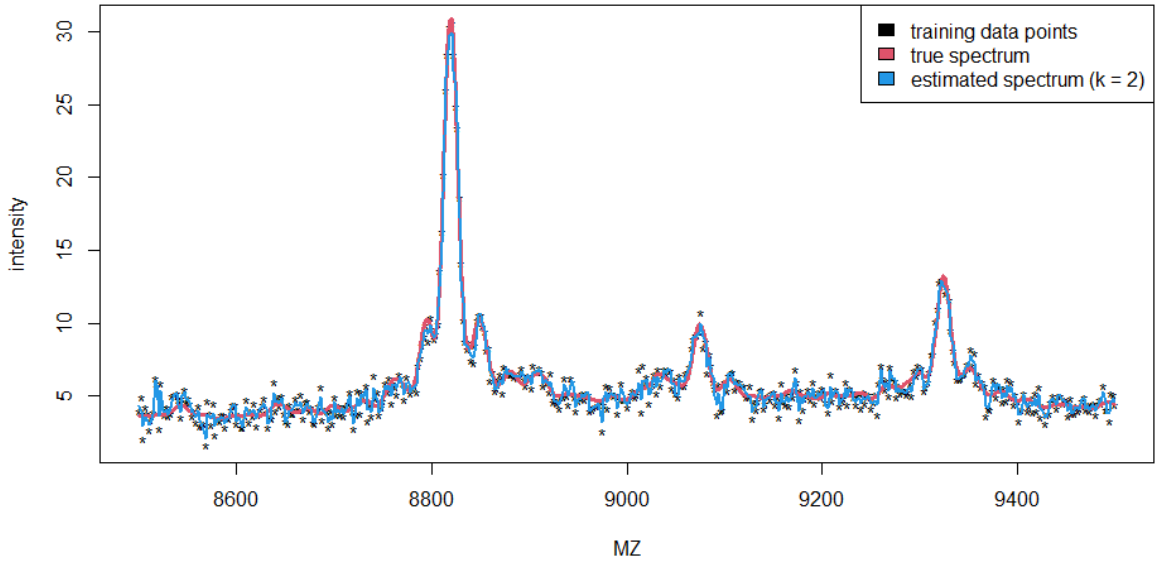
65th patient 95% confidence interval of having a heart disease is [0.8219, 0.9916]
66th patient 95% confidence interval of having a heart disease is [0.0285, 0.2408]
Yes, the confidence interval of both patients suggests that there is a real difference in the population odds of having heart disease between both patients. The 65th patient confidence interval suggests that the patient very likely has a heart disease while on the other hand the 66th patient confidence interval suggests that the patient is not likely to have a heart disease.

Question 3

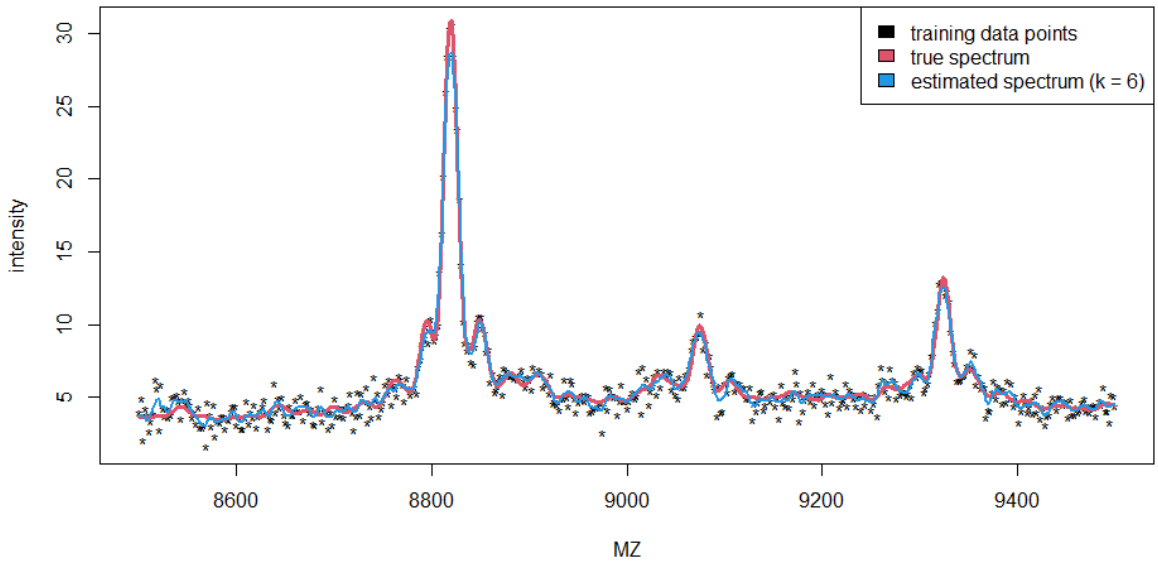
1.
```
> # Question 3.1
> # read files
> ms.truth <- read.csv("ms.truth.2022.csv", stringsAsFactors = T)
> ms.measured <- read.csv("ms.measured.2022.csv", stringsAsFactors =
T)
> # k = 25
> k <- c(1:25)
> # computing rmse
> rmse <- k
> for (i in rmse){
+   ytest.hat <- fitted(kknn(intensity ~ ., ms.measured, ms.truth, k
= i, kernel = 'optimal'))
+   rmse[i] <- sqrt(mean((ytest.hat - ms.truth$intensity)^2))
+ }
> # plotting the graph
> plot(k, rmse, type = 'l', xlab = "k", ylab = "RMSE", main = "RMSE
against k")
```



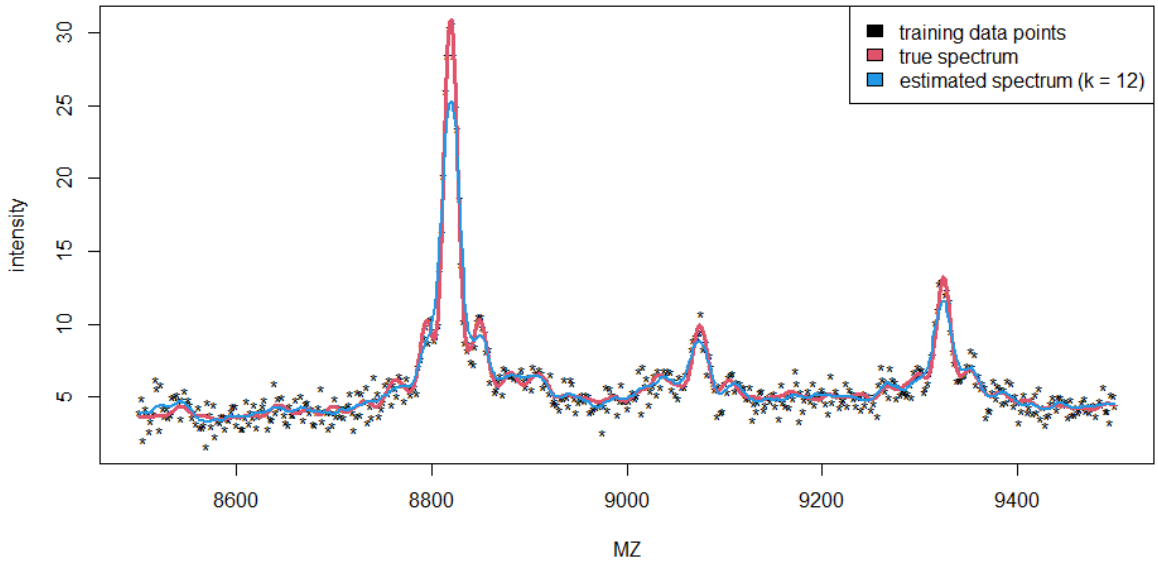
2.
```
> # Question 3.2
> # training data points
> plot(ms.measured, pch = '*', col = 1, xlab = "MZ", ylab =
"intensity")
> # true spectrum
> lines(ms.truth, type = 'l', lwd = 3, col = 2)
> # estimated spectrum (k = 2)
> lines(ms.measured$MZ, fitted(kknn(intensity ~ ., ms.measured,
ms.truth, k = 2, kernel = 'optimal')),
+       type = 'l', lwd = 2, col = 4)
> # legends
> legend(x="topright",legend = c("training data points","true
spectrum","estimated spectrum (k = 2)"), fill=c(1, 2, 4))
```



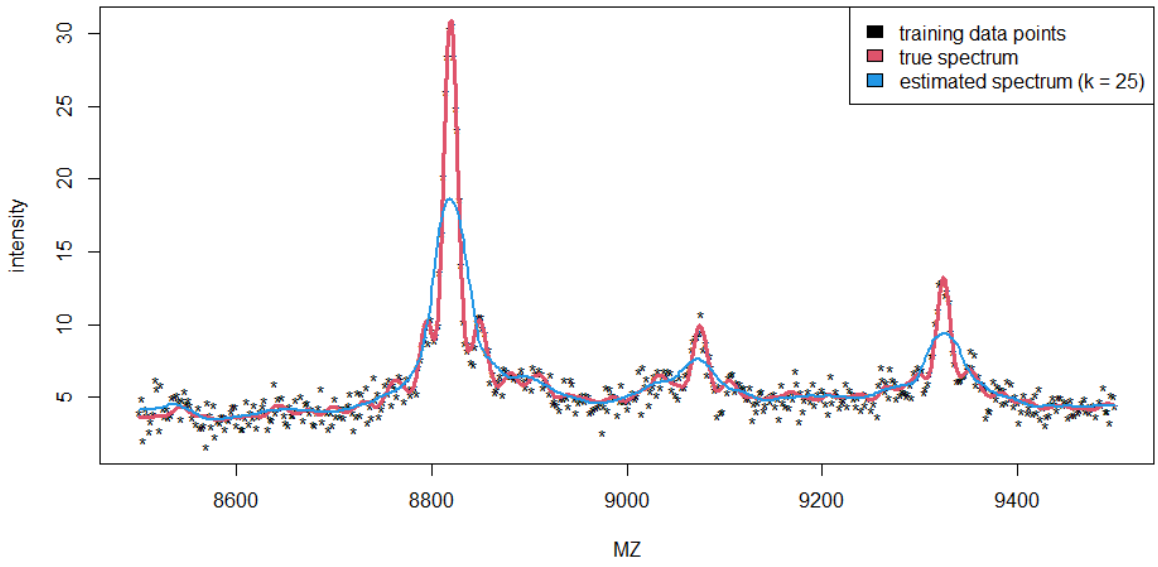
```
> # training data points
> plot(ms.measured, pch = '*', col = 1, xlab = "MZ", ylab =
"intensity")
> # true spectrum
> lines(ms.truth, type = 'l', lwd = 3, col = 2)
> # estimated spectrum (k = 6)
> lines(ms.measured$MZ, fitted(kknn(intensity ~ ., ms.measured,
ms.truth, k = 6, kernel = 'optimal')),
+       type = 'l', lwd = 2, col = 4)
> # legends
> legend(x="topright",legend = c("training data points","true
spectrum","estimated spectrum (k = 6)"), fill=c(1, 2, 4))
```




```
> # training data points
> plot(ms.measured, pch = '*', col = 1, xlab = "MZ", ylab =
"intensity")
> # true spectrum
> lines(ms.truth, type = 'l', lwd = 3, col = 2)
> # estimated spectrum (k = 12)
> lines(ms.measured$MZ, fitted(kknn(intensity ~ ., ms.measured,
ms.truth, k = 12, kernel = 'optimal')),
+       type = 'l', lwd = 2, col = 4)
> # legends
> legend(x="topright",legend = c("training data points","true
spectrum","estimated spectrum (k = 12)"), fill=c(1, 2, 4))
```




```
> # training data points
> plot(ms.measured, pch = '*', col = 1, xlab = "MZ", ylab =
"intensity")
> # true spectrum
> lines(ms.truth, type = 'l', lwd = 3, col = 2)
> # estimated spectrum (k = 25)
> lines(ms.measured$MZ, fitted(kknn(intensity ~ ., ms.measured,
ms.truth, k = 25, kernel = 'optimal')),
+       type = 'l', lwd = 2, col = 4)
> # legends
> legend(x="topright",legend = c("training data points","true
spectrum","estimated spectrum (k = 25)"), fill=c(1, 2, 4))
```



3. **The root-mean-squared-error decreases as k increases from k = 1 until k = 6 where the root-mean-squared error reaches the minimum point. Then, the root-mean-squared-error increases as k increases reaching a maximum point at k = 25 as we only tested k = 1 to 25 for the values of k. Hence, k = 6 provides to most accurate estimate as the root-mean-squared-error is the lowest.**

4. **I do not think that the estimated spectra plotted in Q3.2 achieved our dual aims of providing a smooth, low-noise estimate of the background level as well as accurate estimation of the height of the peaks.**
   - **Using the value of k = 2, the estimated spectrum does not provide a smooth, low-noise estimate of background level but provides an accurate estimation of the height of the peaks.**
   - **Using the value of k = 6, the estimated spectrum provides a smooth, low-noise estimate but provides a fairly inaccurate estimation of the height of the peaks.**
   - **Using the value of k = 12, the estimated spectrum provides a smooth, low-noise estimate but provides an inaccurate estimation of the height of the peaks.**
   - **Using the value of k = 25, the estimated spectrum provides a very smooth, low-noise estimate but provides a very inaccurate estimation of the height of the peaks.**

   Although using the value of k = 6 provides the best estimated spectrum, it does not meet our dual aim of a smooth, low-noise estimate of background level and accurate estimation of the height of the peaks. I think this is due to the k-NN method using it's neighbour values to reduce noise, therefore, it is unable to predict sharp peaks where the values of certain areas are widely distinct from it's neighbours.

5.
```
> # Question 3.5
> # estimate best value of k
> train.kknn(intensity ~ .,data = ms.measured, kmax = 25,kernel =
"optimal")

Call:
train.kknn(formula = intensity ~ ., data = ms.measured, kmax = 25,
kernel = "optimal")

Type of response variable: continuous
minimal mean absolute error: 0.6398525
Minimal mean squared error: 0.6818165
Best kernel: optimal
Best k: 5
> # RMSE of k = 5
> rmse[5]
[1] 0.3986738
> # RMSE of k = 6
> rmse[6]
[1] 0.3919784
```

   The method selects the value of k = 5. It is slightly smaller than the value of k = 6 which we identified in Q3.1 which would minimise the actual mean-squared error. However, it

would then produce an estimate that is more accurate for the height of the peaks, resulting in a better estimation spectrum.

6.
```
> # Question 3.6
> # difference of truth and estimate
> diff <- ms.truth$intensity - fitted(kknn(intensity ~ .,
ms.measured, ms.truth, k = 5, kernel = 'optimal'))
> # standard deviation of difference
> sd(diff)
[1] 0.3986602
```

The estimate of the standard deviation of the sensor/measurement noise that has corrupted our intensity measurements is 0.3987702.

7.
```
> # Question 3.7
> # index of maximum estimated intensity
> ind <- which.max(fitted(kknn(intensity ~ ., ms.measured, ms.truth,
k = 5, kernel = 'optimal')))
> # MZ of maximum estimated intensity
> ms.measured[ind,]
      MZ intensity
160 8818    30.341
```

From the smoothed signal produced by using the value of k found in Q3.5, which is k = 5, the corresponding MZ to the maximum estimated intensity is 8818.

8.
```
> # Question 3.8
> # k = 5
> boot.5 <- function(formula, data, indices)
+ {
+   # Create a bootstrapped version of our data
+   d <- data[indices,]
+   # compute the prediction for MZ = 8818 and return it
+   return (fitted(kknn(formula, d, ms.truth[ind,], k = 5, kernel =
"optimal")))
+ }
> bs.5 <- boot(data = ms.measured, statistic = boot.5, R = 5000,
formula = intensity ~ .)
> # k = 3
> boot.3 <- function(formula, data, indices)
+ {
+   # Create a bootstrapped version of our data
+   d <- data[indices,]
+   # compute the prediction for MZ = 8818 and return it
+   return (fitted(kknn(formula, d, ms.truth[ind,], k = 3, kernel =
"optimal")))
+ }
> bs.3 <- boot(data = ms.measured, statistic = boot.3, R = 5000,
formula = intensity ~ .)
```

```
> # k = 20
> boot.20 <- function(formula, data, indices)
+ {
+   # Create a bootstrapped version of our data
+   d <- data[indices,]
+   # compute the prediction for MZ = 8818 and return it
+   return (fitted(kknn(formula, d, ms.truth[ind,], k = 20, kernel =
"optimal")))
+ }
> bs.20 <- boot(data = ms.measured, statistic = boot.20, R = 5000,
formula = intensity ~ .)
>
> # 95% confidence interval
> boot.ci(bs.5, conf = 0.95, type = "bca")
BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
Based on 5000 bootstrap replicates

CALL :
boot.ci(boot.out = bs.5, conf = 0.95, type = "bca")

Intervals :
Level       BCa
95%   (25.33, 30.57 )
Calculations and Intervals on Original Scale
> boot.ci(bs.3, conf = 0.95, type = "bca")
BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
Based on 5000 bootstrap replicates

CALL :
boot.ci(boot.out = bs.3, conf = 0.95, type = "bca")

Intervals :
Level       BCa
95%   (25.99, 30.66 )
Calculations and Intervals on Original Scale
> boot.ci(bs.20, conf = 0.95, type = "bca")
BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
Based on 5000 bootstrap replicates

CALL :
boot.ci(boot.out = bs.20, conf = 0.95, type = "bca")

Intervals :
Level       BCa
95%   (15.48, 26.30 )
Calculations and Intervals on Original Scale
```

The 95% confidence intervals for when

- **k = 5 : [25.33, 30.57]**
- **k = 3 : [25.99, 30.66]**

- **k = 20 : [15.48, 30.66]**

The sizes of the confidence intervals are greater as k increases. These varying is size is because as k increase, the spectrum becomes smoother to a point where it does not estimate the height of the peaks accurately. Hence, the accuracy will decrease as k increases and the confidence interval will become greater as we are using a less accurate estimation.