Assignment 3 Rui Qin 30874157

Question 1

Question 1A

```
housing <- read.csv("housing.2023.csv")

model <- lm(medv ~ crim + zn + indus + chas + nox + rm + age + dis + rad + tax +
ptratio + lstat, data = housing)

summary(model)

> summary(model)

Call:

lm(formula = medv ~ crim + zn + indus + chas + nox + rm + age +

    dis + rad + tax + ptratio + lstat, data = housing)


Residuals:

     Min       1Q   Median       3Q      Max

-17.9480  -2.7966  -0.5589   1.5896  26.2270


Coefficients:

              Estimate Std. Error t value Pr(>|t|)

(Intercept)  34.054337   7.568558   4.499 1.07e-05 ***

crim         -0.115818   0.041915  -2.763 0.006174 **

zn            0.018561   0.021190   0.876 0.381961

indus        -0.011274   0.087587  -0.129 0.897691

chas          4.163521   1.299647   3.204 0.001544 **

nox         -16.722652   6.154586  -2.717 0.007071 **

rm            4.501521   0.688705   6.536 3.83e-10 ***

age           0.001457   0.020603   0.071 0.943690

dis          -1.163294   0.315727  -3.684 0.000284 ***

rad           0.291680   0.112473   2.593 0.010096 *

tax          -0.012387   0.006284  -1.971 0.049871 *

ptratio      -0.960017   0.199722  -4.807 2.73e-06 ***

lstat        -0.480698   0.079723  -6.030 6.26e-09 ***

---

Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1


Residual standard error: 5.164 on 237 degrees of freedom

Multiple R-squared:  0.7089,      Adjusted R-squared:  0.6942

F-statistic:  48.1 on 12 and 237 DF,  p-value: < 2.2e-16
```

Based on the summary of the linear regression model we can know:

- Predictors Associated with Median House Value:
    - rm: This predictor also has a highly significant impact on median house value with a p-value much less than 3.83e-10 (***).
    - dis: This predictor is significant with a p-value of 0.0003 (***),
    - ptratio: ptratio is 2.73e-06 with a very low p-value (2.73e-06).
    - lstat: The coefficient for lstat is -0.48 with a low p-value (6.26e-09).
    - Top 3 Strongest Predictors: rm, lstat, ptratio.

Question 1B

Adjust Significance Level:

- we have 12 predictor variables, and $\alpha = 0.05$
    - Adjusted significance level (α/p): 0.05 / 12 = 0.0042

Based on the p-value summary, only these variables are lower than 0.0042,

- chas: 0.001544
- rm: 3.83e-10
- dis: 0.000284
- ptratio: 2.73e-06
- lstat: 6.26e-09

Other variables with higher P-values were no longer considered statistical after applying the Bonferroni correction, and these five predictors correlated with the median home value.

Question 1C

Effect of Per-capita Crime Rate (crim):

- For every unit increase in per-capita crime rate, the median house price decreases by $115.82 (coefficient: -0.115818).
- This means that higher crime rates are related to lower median house prices in the suburbs, and areas with lower crime rates tend to have higher property values.

Effect of Having Frontage on the Charles River (chas):

- If a suburb has a frontage on the Charles River, the median house price is higher by approximately $4,163.52 (coefficient: 4.163521), compared to suburbs that do not have Charles River frontage.
- This indicates that suburbs located along the Charles River tend to have higher median house prices compared to those that do not have this feature.

Question 1D

```
final_model <- step(model, direction = "both", k = log(nrow(housing)))
summary(final_model)
> summary(model)

Call:
lm(formula = medv ~ crim + zn + indus + chas + nox + rm + age +
    dis + rad + tax + ptratio + lstat, data = housing)

Residuals:
     Min       1Q   Median       3Q      Max
-17.9480  -2.7966  -0.5589   1.5896  26.2270

Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept)  34.054337   7.568558   4.499 1.07e-05 ***
crim         -0.115818   0.041915  -2.763 0.006174 **
zn            0.018561   0.021190   0.876 0.381961
indus        -0.011274   0.087587  -0.129 0.897691
chas          4.163521   1.299647   3.204 0.001544 **
nox         -16.722652   6.154586  -2.717 0.007071 **
rm            4.501521   0.688705   6.536 3.83e-10 ***
age           0.001457   0.020603   0.071 0.943690
dis          -1.163294   0.315727  -3.684 0.000284 ***
rad           0.291680   0.112473   2.593 0.010096 *
tax          -0.012387   0.006284  -1.971 0.049871 *
ptratio      -0.960017   0.199722  -4.807 2.73e-06 ***
lstat        -0.480698   0.079723  -6.030 6.26e-09 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 5.164 on 237 degrees of freedom
Multiple R-squared:  0.7089,      Adjusted R-squared:  0.6942
F-statistic:  48.1 on 12 and 237 DF,  p-value: < 2.2e-16
```

The final regression equation is:

- medv = 34.054337 - 0.115818 * crim + 0.018561 * zn - 0.011274 * indus + 4.163521 * chas - 16.722652 * nox + 4.501521 * rm + 0.001457 * age - 1.163294 * dis + 0.291680 * rad - 0.012387 * tax - 0.960017 * ptratio - 0.480698 * lstat

Question 1E

Based on the regression equation, the council could:

- Lower crime rate
- Planning more residential land
- Promote waterfront properties and preserve Charles River
- Reduced nitrogen oxide concentration and address air quality
- Encourage building additional rooms in homes
- Improve highway accessibility
- Tax reduction
- Hire more teachers and improve education
- Attracting residents of higher socioeconomic status

Question 1F

```
predicted_price <- predict(model, data.frame(
    crim = 0.04741,
    zn = 0,
    indus = 11.93,
    chas = 0,
    nox = 0.573,
    rm = 6.03,
    age = 80.8,
    dis = 2.505,
    rad = 1,
    tax = 273,
    ptratio = 21,
    lstat = 7.88
  ), interval = "confidence", level = 0.95)
print(predicted_price)
> print(predicted_price)
       fit      lwr      upr
1 21.64175 19.44955 23.83396
```

Based on the result we know that the new suburb price may be $21641.75. The 95% confidence interval for this prediction is between approximately $19449.55 and $23833.96.

Question 1G

```
housing$rm_dis_interaction <- housing$rm * housing$dis

interaction_model <- lm(medv ~ crim + zn + indus + chas + nox + rm + age + dis + rad
+ tax + ptratio + lstat + rm_dis_interaction, data = housing)

summary(interaction_model

Call:

lm(formula = medv ~ crim + zn + indus + chas + nox + rm + age +

    dis + rad + tax + ptratio + lstat + rm_dis_interaction, data = housing)


Residuals:

     Min       1Q   Median       3Q      Max

-13.9970  -2.6186  -0.5712   1.6650  25.7453


Coefficients:

                     Estimate Std. Error t value Pr(>|t|)

(Intercept)         62.961194   9.441846   6.668 1.82e-10 ***

crim                -0.127609   0.040189  -3.175 0.001697 **

zn                  -0.012191   0.021278  -0.573 0.567230

indus               -0.025792   0.083875  -0.308 0.758728

chas                 4.488579   1.245624   3.603 0.000383 ***

nox                -20.971488   5.956830  -3.521 0.000517 ***

rm                   0.499458   1.066573   0.468 0.640014

age                  0.012635   0.019855   0.636 0.525180

dis                 -8.986039   1.666728  -5.391 1.69e-07 ***

rad                  0.297185   0.107642   2.761 0.006218 **

tax                 -0.011865   0.006015  -1.972 0.049720 *

ptratio             -1.036977   0.191813  -5.406 1.57e-07 ***

lstat               -0.513869   0.076611  -6.707 1.45e-10 ***

rm_dis_interaction   1.227802   0.257263   4.773 3.19e-06 ***

---

Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1


Residual standard error: 4.942 on 236 degrees of freedom

Multiple R-squared:  0.7345,      Adjusted R-squared:  0.7199

F-statistic: 50.23 on 13 and 236 DF,  p-value: < 2.2e-16
```

The coefficient of the rm_dis_interaction is 1.228, and it has a very low p-value ($p < 0.001$). This indicates that there is an interaction effect between the number of rooms and the distance to employment centres.

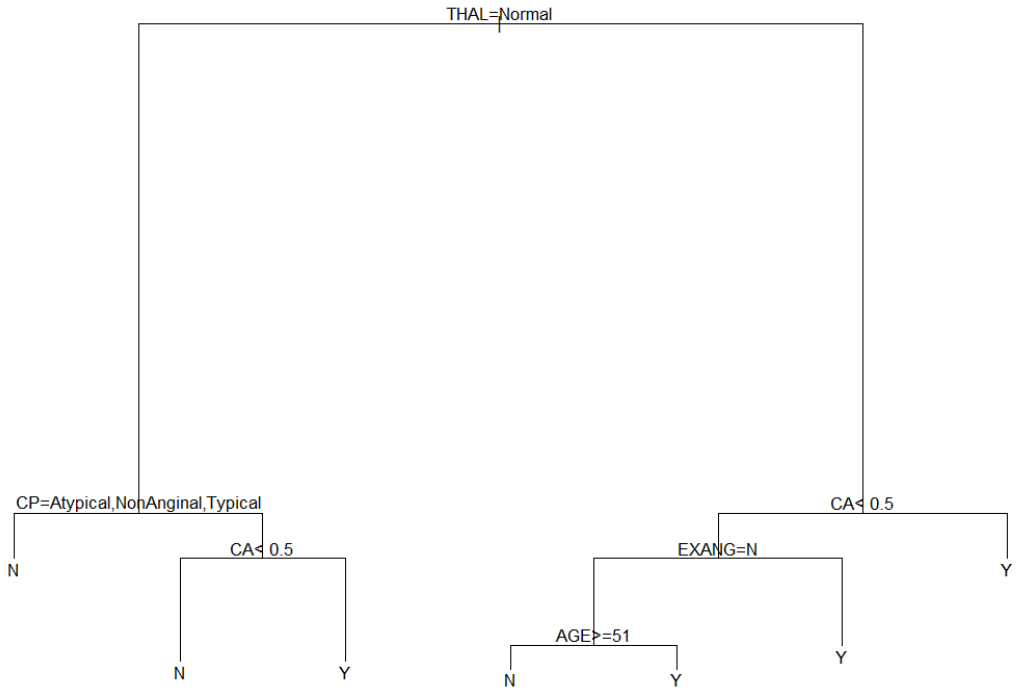
Question 2

Question 2A

```
source("my.prediction.stats.R")
source("wrappers.R")
library(pROC)
library(tree)
library(rpart)
heart.train <- read.csv("heart.train.2023.csv")

#Q2.1
cv_heart = learn.tree.cv(HD ~ ., data = heart.train, nfolds=10, m=5000)
best_tree <- cv_heart$best.tree
plot(cv_heart$best.tree)
text(cv_heart$best.tree)
best_tree
> best_tree
n= 260

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

 1) root 260 125 N (0.51923077 0.48076923)
   2) THAL=Normal 140  34 N (0.75714286 0.24285714)
     4) CP=Atypical,NonAnginal,Typical 95  12 N (0.87368421 0.12631579) *
     5) CP=Asymptomatic 45  22 N (0.51111111 0.48888889)
      10) CA< 0.5 28   7 N (0.75000000 0.25000000) *
      11) CA>=0.5 17   2 Y (0.11764706 0.88235294) *
   3) THAL=Fixed.Defect,Reversible.Defect 120  29 Y (0.24166667 0.75833333)
     6) CA< 0.5 53  24 Y (0.45283019 0.54716981)
      12) EXANG=N 31  10 N (0.67741935 0.32258065)
        24) AGE>=51 20   3 N (0.85000000 0.15000000) *
        25) AGE< 51 11   4 Y (0.36363636 0.63636364) *
      13) EXANG=Y 22   3 Y (0.13636364 0.86363636) *
     7) CA>=0.5 67   5 Y (0.07462687 0.92537313) *
```

Based on the output, the best tree has 7 leaves, and the variables used in the best tree: THAL, CP, CA, EXANG, AGE.

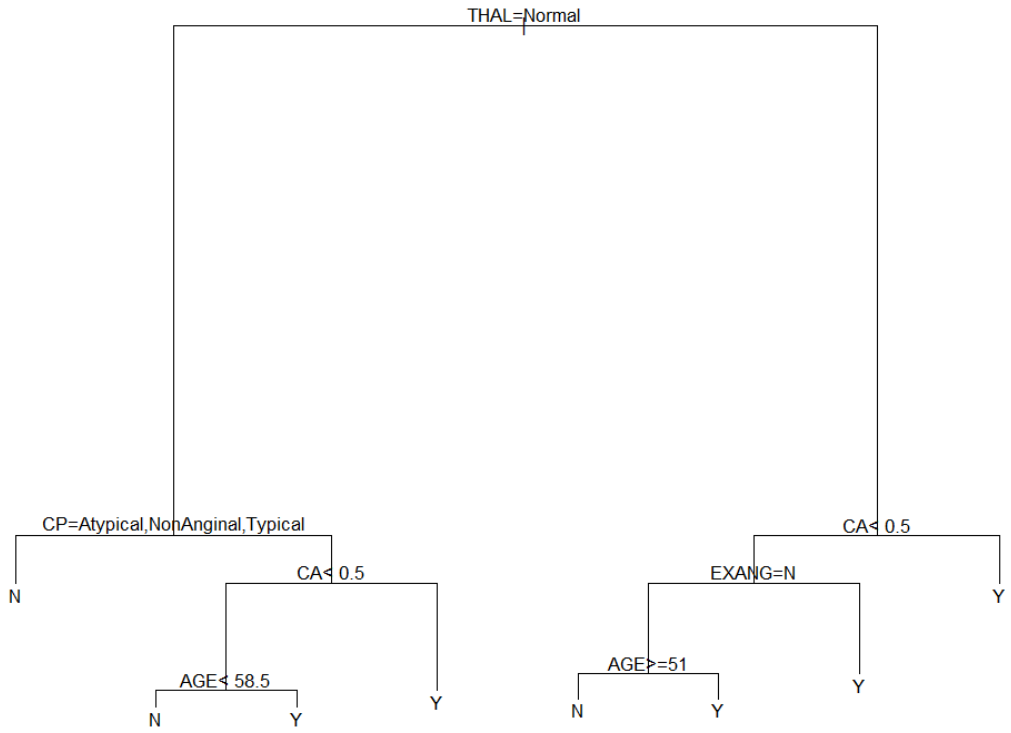
Question 2B



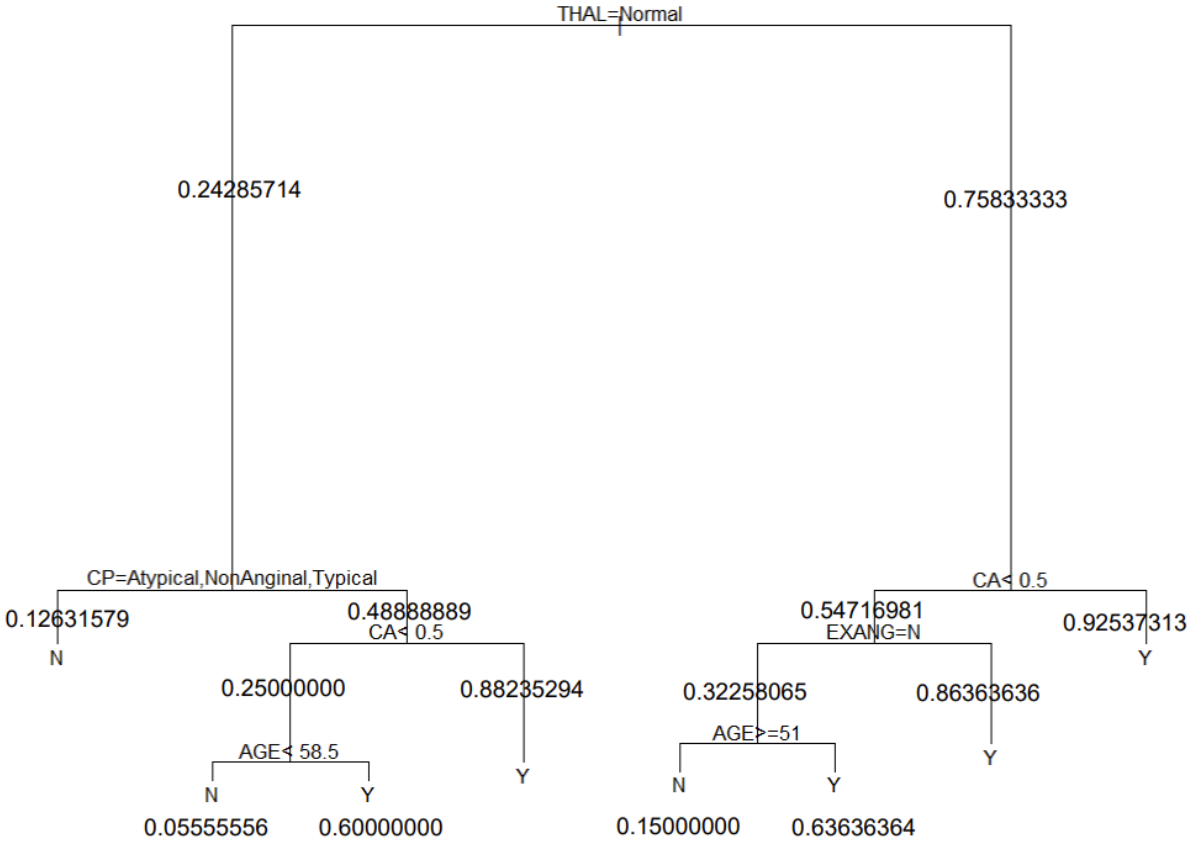
- Thallium Scanning (THAL):
    - If THAL is normal and Chest Pain Type (CP) is Atypical, NonAnginal, or Typical, the patient is likely to not have heart disease.
    - If THAL is normal, CP is Asymptomatic, and the Number of Major Vessels Colored by Flourosopy (CA) is smaller than 0.5, the patient is likely to not have heart disease.
    - If THAL is normal, CP is Asymptomatic, and CA is larger than 0.5, the patient may have heart disease.
- Thallium Scanning (THAL) not normal:
- If CA is not smaller than 0.5, the patient is likely to have heart disease.
- If CA is smaller than 0.5, we further examine Exercise Induced Angina (EXANG):
    - If EXANG is yes, the patient may have heart disease.
    - If EXANG is no, we then consider the patient's age:
        - If age is greater than or equal to 51, the patient is likely to not have heart disease.
        - If the age is less than 51, the patient may have heart disease.

Question 2C

```
#Q2.3
heart_tree = rpart(HD~., heart.train)
heart_tree
plot(heart_tree)
text(heart_tree, pretty=12)
> heart_tree
n= 260

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

 1) root 260 125 N (0.51923077 0.48076923)
   2) THAL=Normal 140  34 N (0.75714286 0.24285714)
     4) CP=Atypical,NonAnginal,Typical 95  12 N (0.87368421 0.12631579) *
     5) CP=Asymptomatic 45  22 N (0.51111111 0.48888889)
      10) CA< 0.5 28   7 N (0.75000000 0.25000000)
        20) AGE< 58.5 18   1 N (0.94444444 0.05555556) *
        21) AGE>=58.5 10   4 Y (0.40000000 0.60000000) *
      11) CA>=0.5 17   2 Y (0.11764706 0.88235294) *
   3) THAL=Fixed.Defect,Reversible.Defect 120  29 Y (0.24166667 0.75833333)
     6) CA< 0.5 53  24 Y (0.45283019 0.54716981)
      12) EXANG=N 31  10 N (0.67741935 0.32258065)
        24) AGE>=51 20   3 N (0.85000000 0.15000000) *
        25) AGE< 51 11   4 Y (0.36363636 0.63636364) *
      13) EXANG=Y 22   3 Y (0.13636364 0.86363636) *
     7) CA>=0.5 67   5 Y (0.07462687 0.92537313) *
```



The annotated plot of the tree:



Question 2D

For people whose Thallium Scanning (THAL) is not normal, k by Flourosopy (CA) is larger than 0.5 is a high risk of getting heart disease (0.92537).

Question 2E

```
logistic_model <- glm(as.factor(heart.train$HD)~., data = heart.train, family =
binomial)

stepwise_model  <- step(logistic_model, direction="both", k=log(nrow(heart.train)),
trace = 0)

summary(stepwise_model)

> summary(stepwise_model)


Call:

glm(formula = as.factor(heart.train$HD) ~ CP + THALACH + OLDPEAK +

    CA + THAL, family = binomial, data = heart.train)


Coefficients:

                       Estimate Std. Error z value Pr(>|z|)

(Intercept)             2.740517   1.480858   1.851  0.06422 .

CPAtypical             -1.185881   0.549552  -2.158  0.03094 *

CPNonAnginal           -1.890318   0.446996  -4.229 2.35e-05 ***

CPTypical              -1.853046   0.628142  -2.950  0.00318 **

THALACH                -0.023493   0.009215  -2.550  0.01078 *

OLDPEAK                 0.576266   0.204136   2.823  0.00476 **

CA                      1.098536   0.250277   4.389 1.14e-05 ***

THALNormal             -0.325278   0.747767  -0.435  0.66356

THALReversible.Defect   1.459413   0.767118   1.902  0.05711 .

---

Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1


(Dispersion parameter for binomial family taken to be 1)


    Null deviance: 360.05  on 259  degrees of freedom

Residual deviance: 194.09  on 251  degrees of freedom

AIC: 212.09


Number of Fisher Scoring iterations: 6
```

logistic regression model has the following values: THAL, CA, OLDPEAK, CP

Compared with the tree model, the logistic regression model has OLDPEAK, but lacks EXANG and AGE, and others remain consistent

The predictor "CA" (number of major vessels coloured by fluoroscopy) has the largest coefficient magnitude (1.098536) which means it is the most important

Question 2F

P(HD = Y) =

2.741−1.186×CPAtypical−1.890×CPNonAnginal−1.853×CPTypical−0.0235×THALACH+0.576×OLDPEAK +1.099×CA−0.325×THALNormal+1.459×THALReversible.Defect

Question 2G

```
#Q2.7

heart.test = read.csv('heart.test.2023.csv')

my.pred.stats(predict(stepwise_model, heart.test, type='response'),
as.factor(heart.test$HD))

my.pred.stats(predict(best_tree, heart.test)[,2], as.factor(heart.test$HD))
```

Performance statistic for logistic regression model:

Performance statistics:

Confusion matrix:

	target	
pred	N	Y
N	98	18
Y	11	73

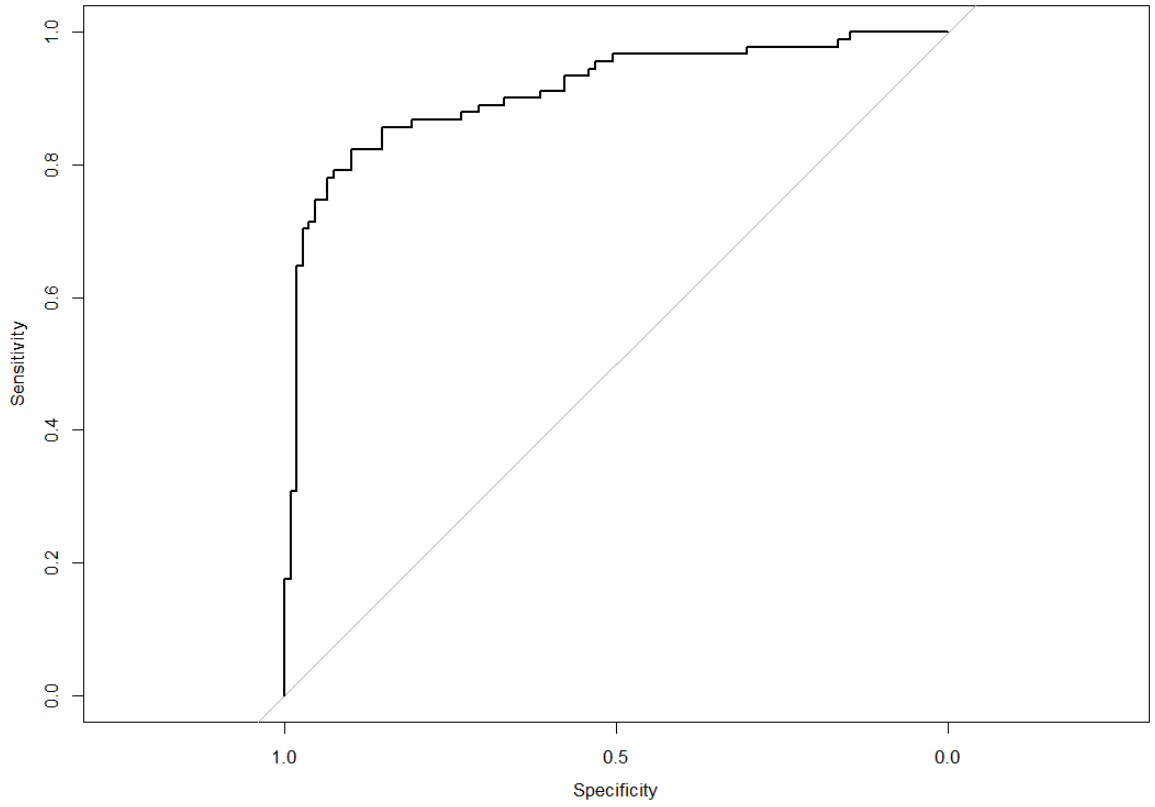
Classification accuracy = 0.855

Sensitivity = 0.8021978

Specificity = 0.8990826

Area-under-curve = 0.9107773

Logarithmic loss = 72.81979



Performance statistic for tree

Performance statistics:

Confusion matrix:

	target	
pred	N	Y
N	96	11
Y	13	80

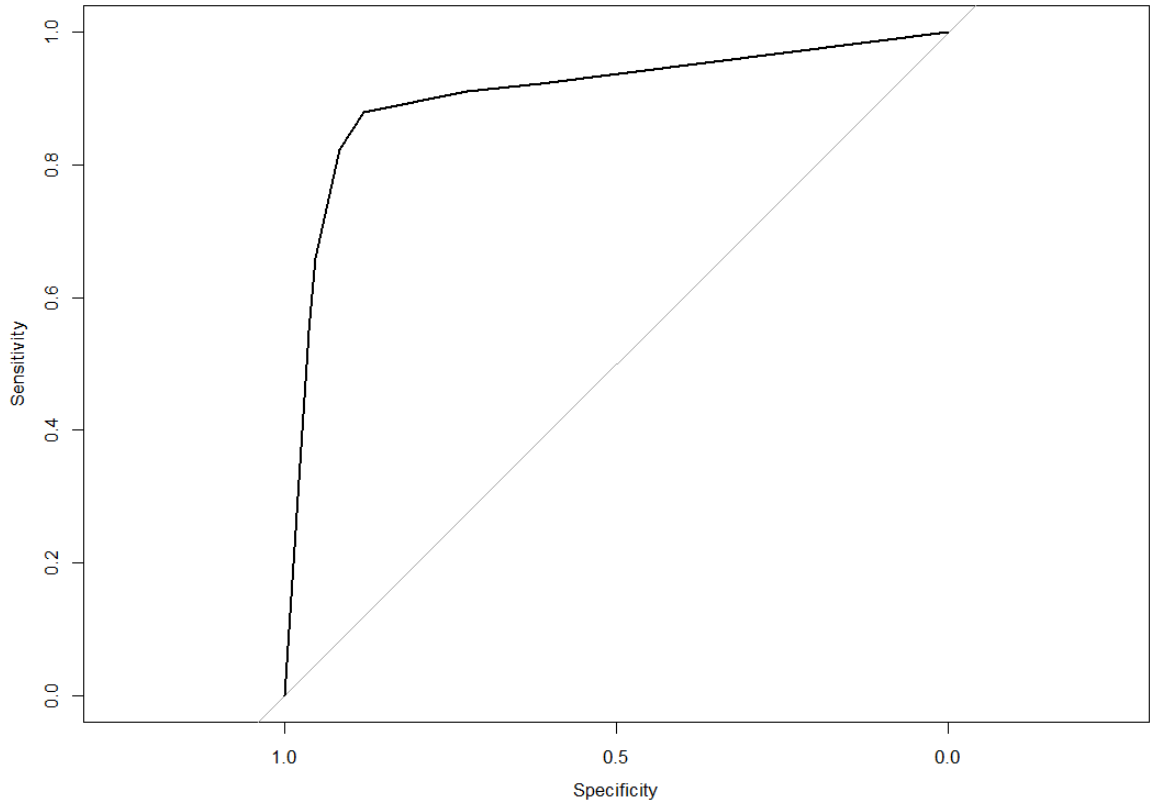
Classification accuracy = 0.88

Sensitivity = 0.8791209

Specificity = 0.8807339

Area-under-curve = 0.9058373

Logarithmic loss = 70.55278



Logistic Regression Model:

- Classification Accuracy: 85.5%
- Sensitivity (True Positive Rate): 80.2%
- Specificity (True Negative Rate): 89.9%
- AUC: 91.1%
- Logarithmic Loss: 72.82

Decision Tree:

- Classification Accuracy: 88%
- Sensitivity (True Positive Rate): 87.9%
- Specificity (True Negative Rate): 88.1%
- AUC: 90.6%
- Logarithmic Loss: 70.55

The decision tree is slightly better at sensitivity and logarithmic loss, and the logistic regression model has a higher specificity and AUC.

In the case of medicine, we may avoid giving unnecessary treatments to patients, so avoiding false positives (specificity) is more important. Therefore, the logistic regression model might be a better choice.

Question 2H

```
#Q2.8
patient_69 <- heart.test[69,]


predicted_prob_tree <- predict(best_tree, patient_69)
predicted_prob_tree
predicted_prob_tree[2]/predicted_prob_tree[1]


odds_logistic <- predict(stepwise_model, patient_69, type = "response")
odds_logistic
odds_logistic / (1 - odds_logistic)

> predicted_prob_tree
           N         Y
69 0.1363636 0.8636364
> predicted_prob_tree[2]/predicted_prob_tree[1]
[1] 6.333333
> odds_logistic
       69
0.9463509
> odds_logistic / (1 - odds_logistic)
      69
17.63966
```

Tree:

- The probability of the patient having heart disease predicted by the tree is 0.8636364
- The probability that the patient does not have heart disease predicted by the tree is 0.1363636
    - predicted odds is 6.333333

logistic regression model:

- The probability of the patient having heart disease predicted by the logistic regression model is 0.9463509
- predicted odds is 17.63966

Both models predict that the 69th patient has a high likelihood of having heart disease, with the logistic regression model assigning an even higher probability compared to the tree model, which indicates greater confidence in the prediction.

Question 2I

```
#Q2.9

library(boot)

boot.auc <- function(formula, data, indices)

{

  d = data[indices,]


  fit = glm(formula, d, family=binomial)


  target = patient_69

  rv = predict(fit, target, type="response")

  return(rv)


}


bs_logic <- boot(data=heart.train, statistic=boot.auc, R=5000, formula=as.factor(HD)
~ CP + THALACH + OLDPEAK + CA + THAL)

boot.ci(bs_logic, conf=0.95, type="bca")

> boot.ci(bs_logic, conf=0.95, type="bca")

BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS

Based on 5000 bootstrap replicates


CALL :

boot.ci(boot.out = bs_logic, conf = 0.95, type = "bca")


Intervals :

Level       BCa

95%   ( 0.8285,  0.9842 )

Calculations and Intervals on Original Scale
```

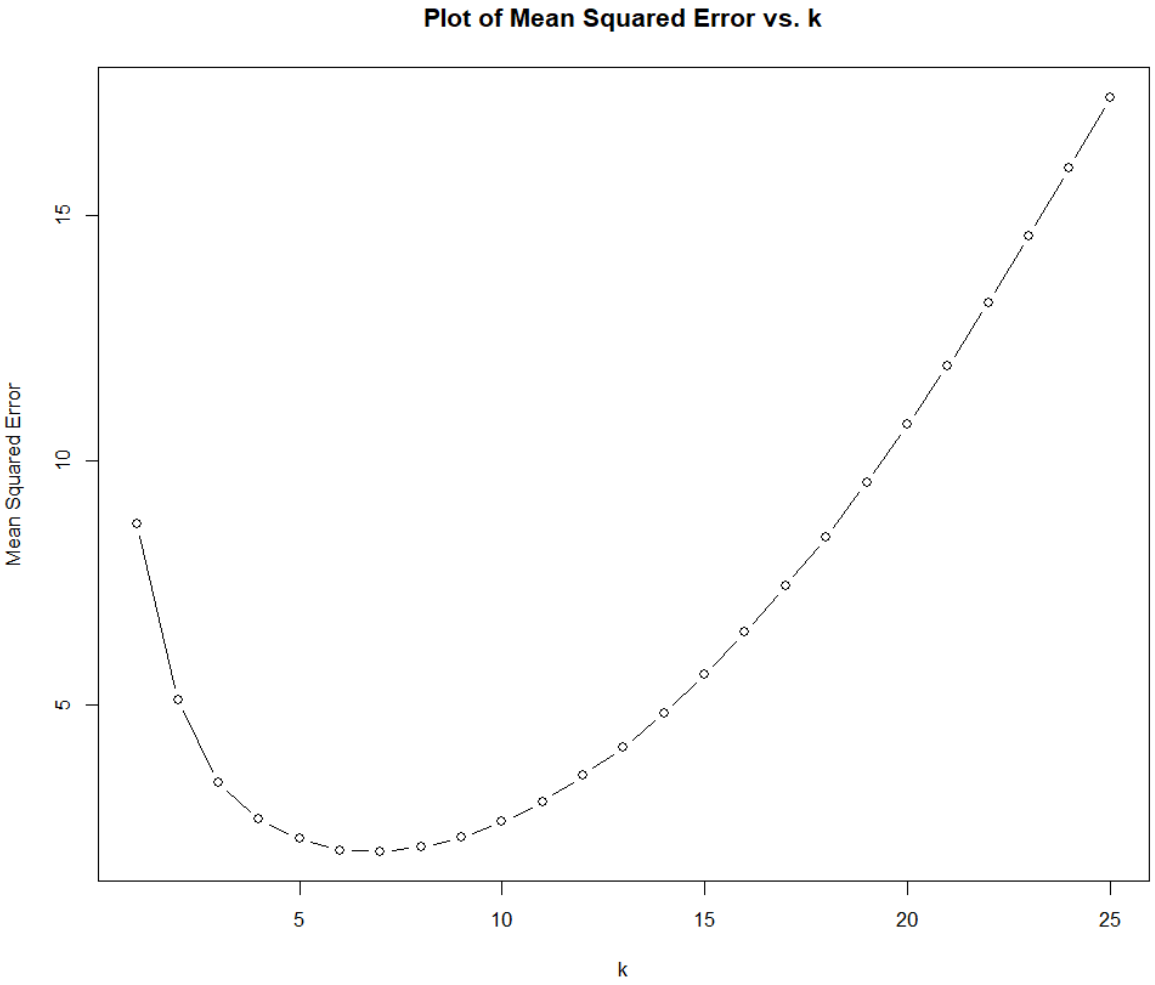
The computed bootstrap confidence interval for the probability of having heart disease for the 69th patient is (0.8285, 0.9842) with a 95% confidence level. The logistic regression model's predicted probability for the patient was approximately 0.9464 (94.6%), and the tree model's predicted probability was approximately 0.8636 (86.4%), which both fall within the confidence interval (0.8285, 0.9842).

Question 3

Question 3A

```
#Q3

library(kknn)

library(ggplot2)

ms.truth <- read.csv("ms.truth.2023.csv")

ms.measured <- read.csv("ms.measured.2023.csv")


mse_values <- numeric(25)


for (k in 1:25) {


  knn_model <- kknn(intensity ~ MZ, train = ms.measured, test = ms.truth, k = k,
kernel = "optimal")


  predictions <- predict(knn_model)


  mse <- mean((predictions - ms.truth$intensity)^2)


  mse_values[k] <- mse

}

mse_values

plot(1:25, mse_values, type = "b", xlab = "k", ylab = "Mean Squared Error", main =
"Plot of Mean Squared Error vs. k")

> mse_values

 [1]  8.704256  5.104779  3.410489  2.656165  2.262812  2.021296  2.004127  2.084660
2.286621

[10]  2.608518  3.012139  3.553871  4.124015  4.838148  5.619558  6.482609  7.436011
8.422623

[19]  9.547819 10.733335 11.927679 13.234540 14.597129 15.985650 17.420855
```

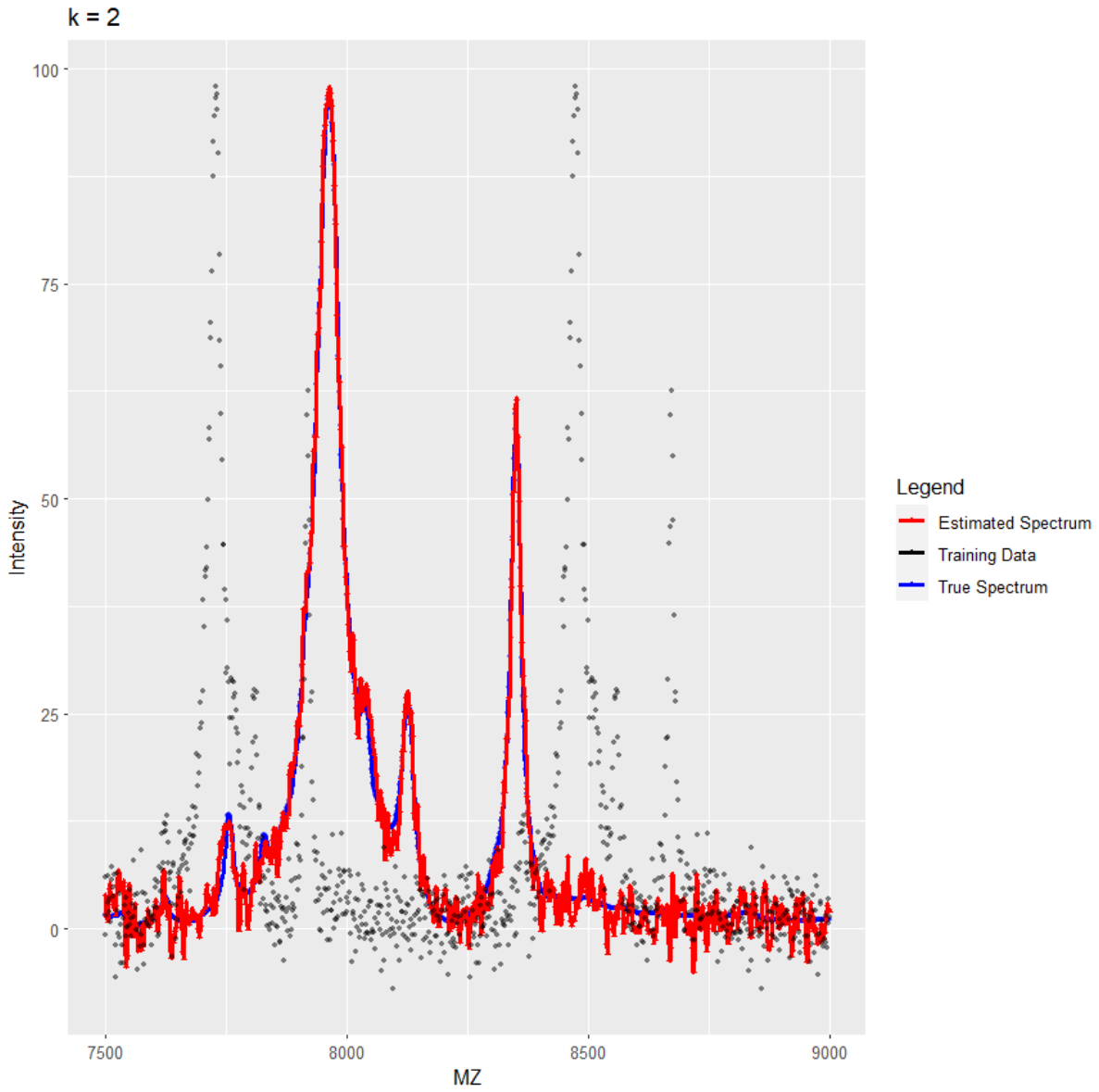
Here is the plot of errors against the values of k:


Plot of Mean Squared Error vs. k
Mean Squared Error
5
10
15
5
10
15
20
25
k

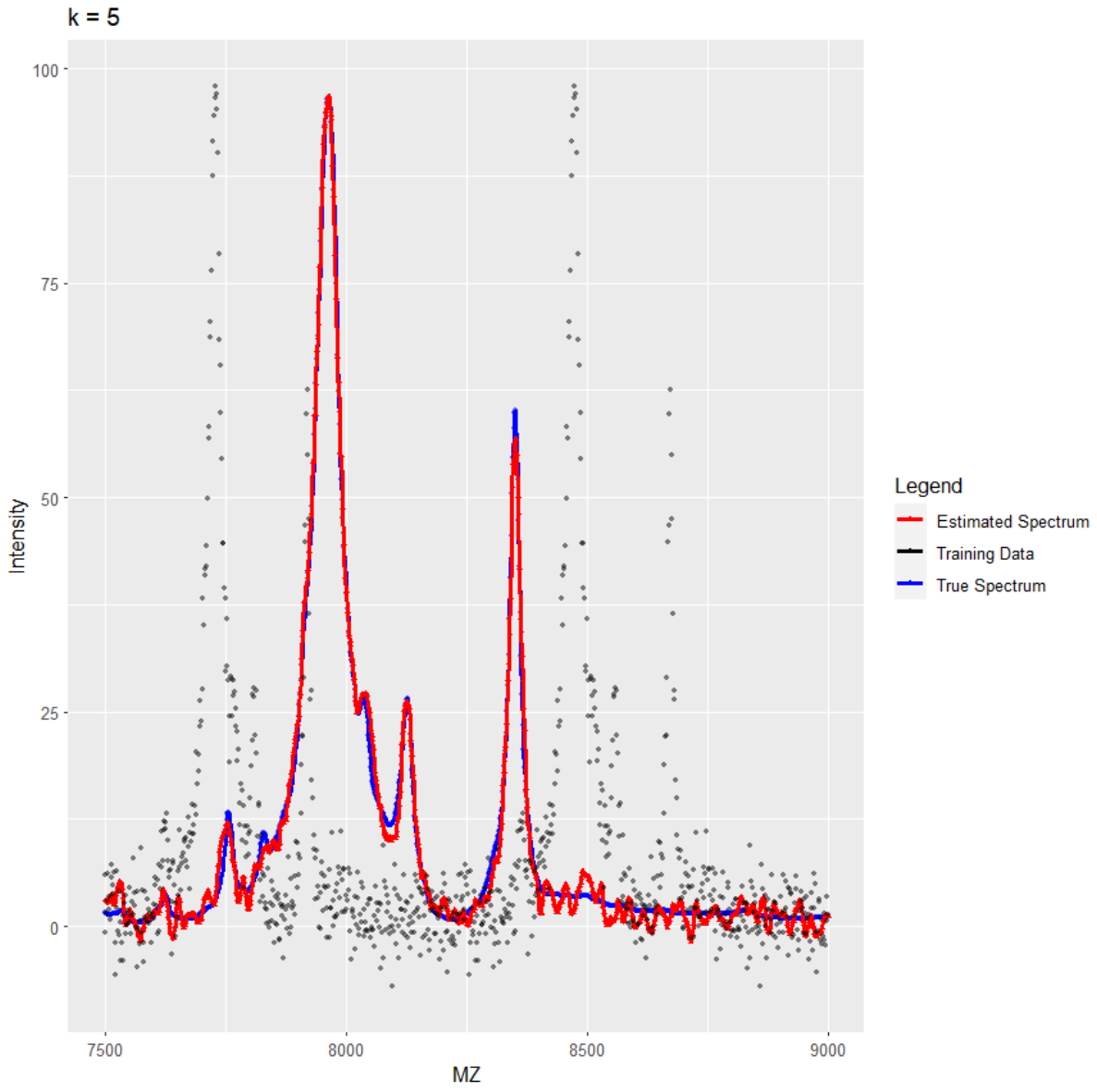
Question 3B

```
#Q3.2

spectra_plot <- function(k) {


  knn_model <- kknn(intensity ~ MZ, train = ms.measured, test = ms.truth, k = k,
kernel = "optimal")




  predictions <- predict(knn_model)




  plot_data <- data.frame(MZ = ms.truth$MZ, True_Intensity = ms.truth$intensity,
Predicted_Intensity = predictions, Measured_Intensity = ms.measured$intensity)




  ggplot(plot_data, aes(x = MZ)) +

    geom_line(aes(y = True_Intensity, color = "True Spectrum"), size = 1) +

    geom_point(aes(y = True_Intensity, color = "True Spectrum"), size = 1, alpha =
0.5, shape = 16) +

    geom_line(aes(y = Predicted_Intensity, color = "Estimated Spectrum"), size = 1) +

    geom_point(aes(y = Predicted_Intensity, color = "Estimated Spectrum"), size = 1,
shape = 17) +

    geom_point(aes(y = Measured_Intensity, color = "Training Data"), size = 1, alpha
= 0.5, shape = 16) +

    labs(title = paste("k =", k), x = "MZ", y = "Intensity", color = "Legend") +

    scale_color_manual(values = c("True Spectrum" = "blue", "Estimated Spectrum" =
"red", "Training Data" = "black"))

}


spectra_plot(2)

spectra_plot(5)

spectra_plot(10)

spectra_plot(25)
```

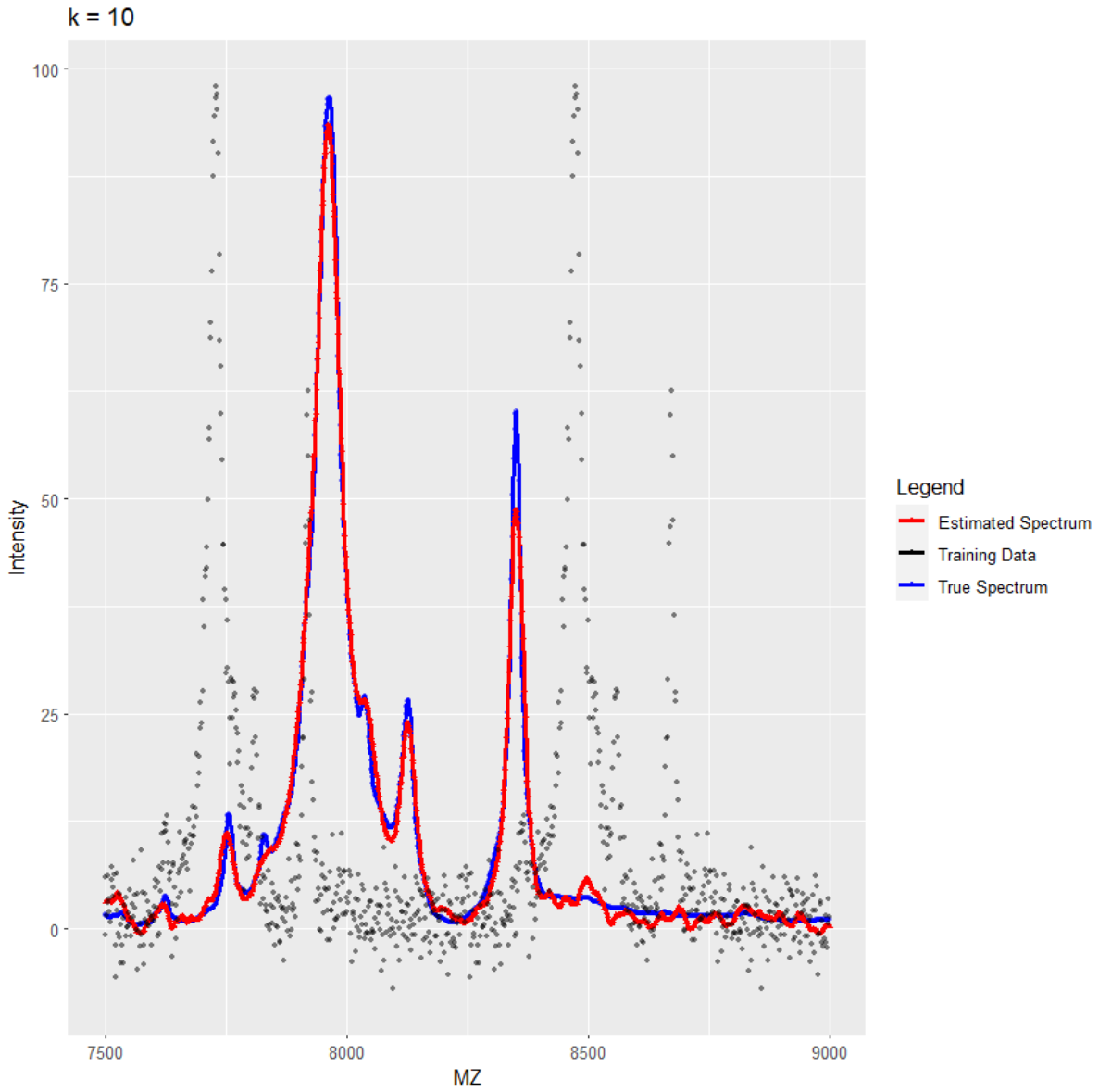
When k = 2


k = 2
100
75
50
25
0
Intensity
7500
8000
8500
9000
MZ
Legend
Estimated Spectrum
Training Data
True Spectrum

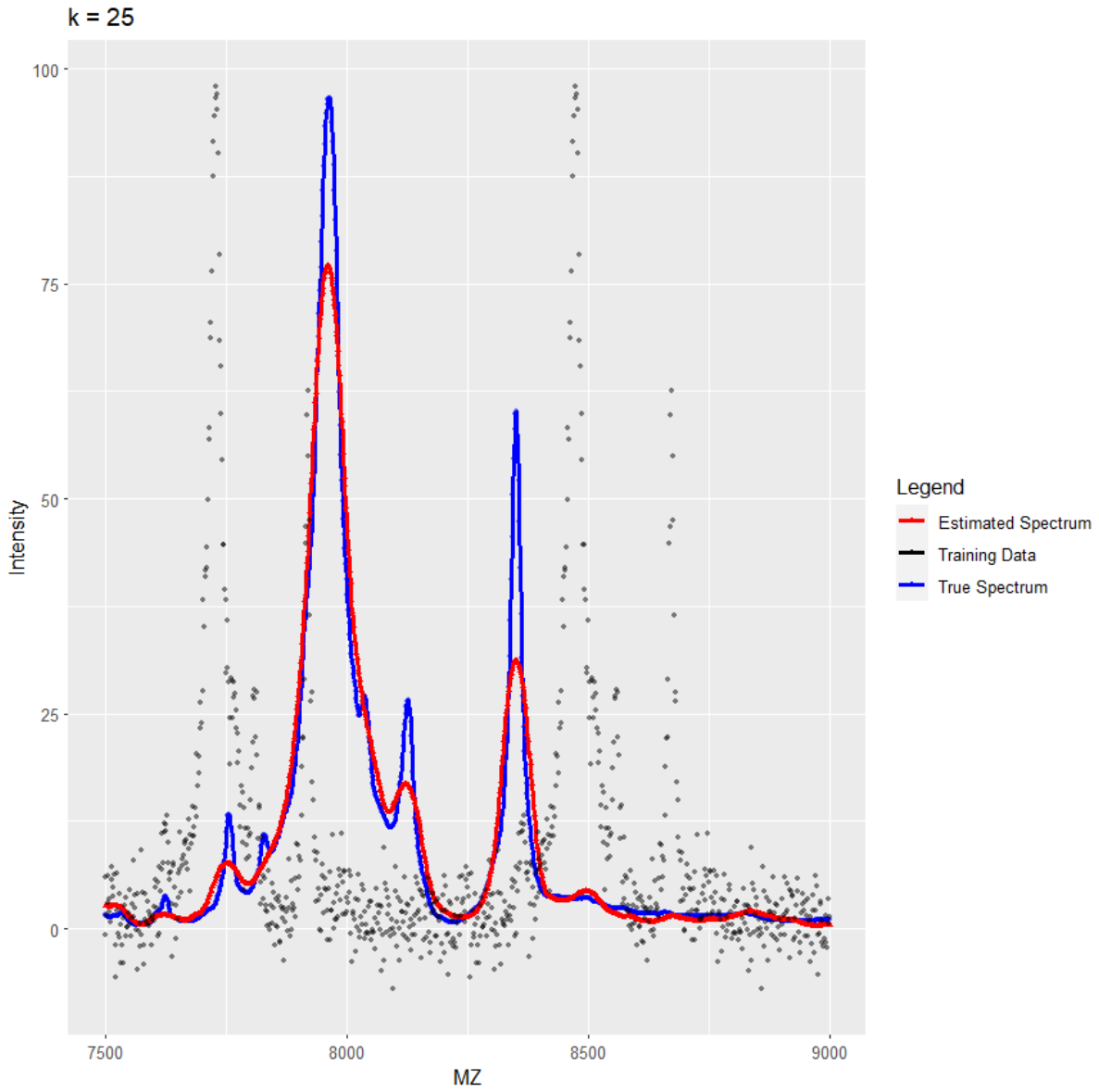
When k = 5


k = 5
100
75
50
25
0
Intensity
7500
8000
8500
9000
MZ
Legend
Estimated Spectrum
Training Data
True Spectrum

When k = 10


k = 10
100
75
50
25
0
Intensity
7500
8000
8500
9000
MZ
Legend
Estimated Spectrum
Training Data
True Spectrum

When k = 25



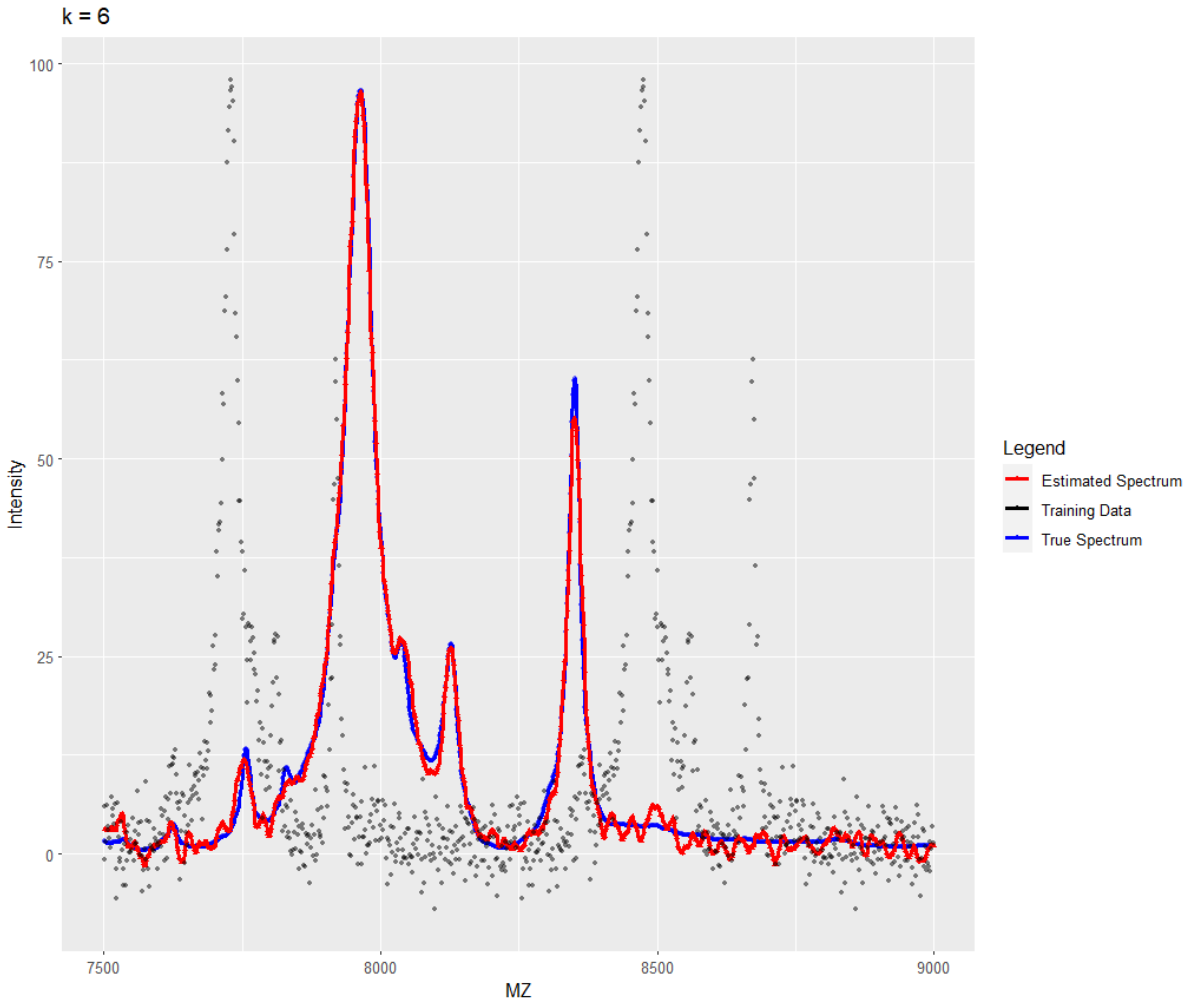
Question 3C

By combining the graph of question 3.1, we can see that among k= 2,5,10 and 25, the mean square error is the smallest when k=5, and the mean square error reaches the lowest value when it is between 5 and 10. Then k=10, k=2, k=25. Therefore, based on the graph, it can be concluded that in k=2, 5, 10, and 25, the estimate is most accurate when k=5. When k=10, the estimate is better than k=2 and k=25. When k=2, the estimate is better than k=25.

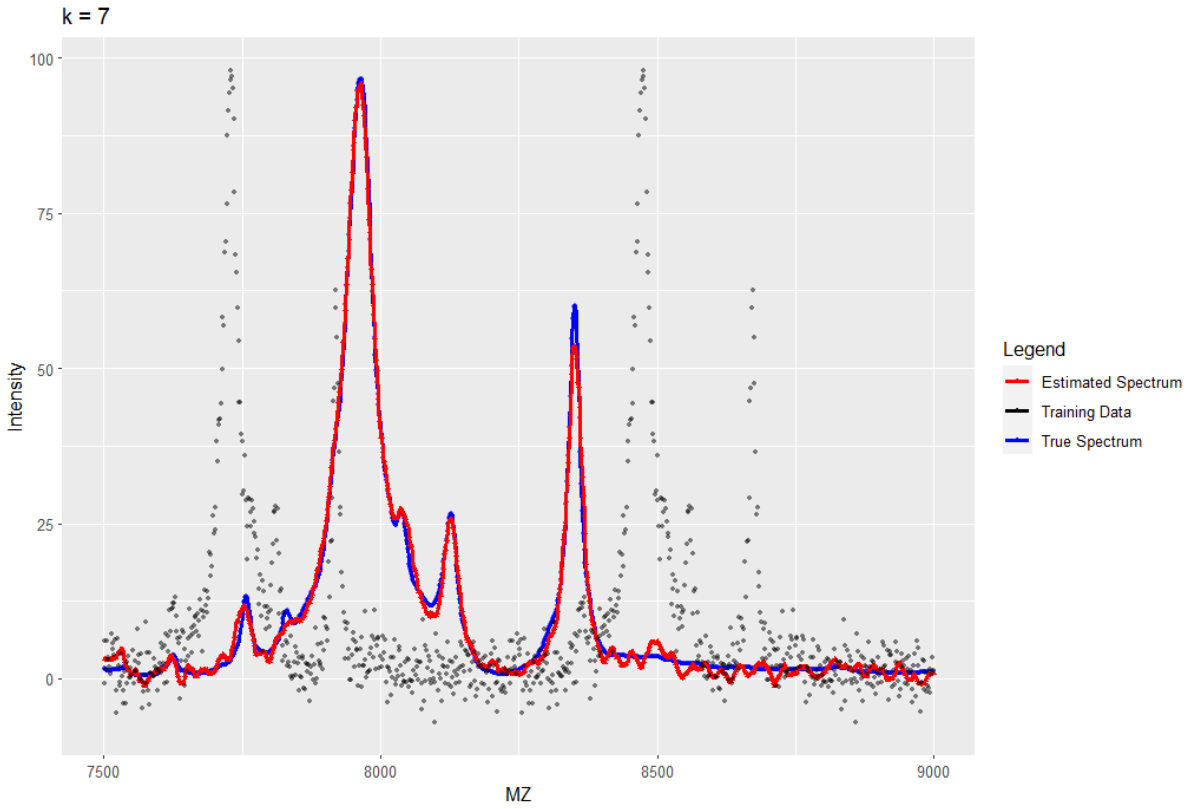
Question 3D

Yes. We can see when k=6, the red line provides a smoother and less noise estimate of the background level as well as an accurate peak estimate. And what's more important is that the double peaks are better covered when k=6 than when k=7. Especially for the second-highest peak, there's a little bit more white space at k=7.

The graph of k=6 is shown below.



The graph of k=7 is shown below.



Question 3E

```
#Q3.5

cv_results <- train.kknn(intensity ~ MZ, data = ms.measured, kmax=25, kernel =
"optimal")

best_k <- cv_results$best.parameters$k

best_k


mse_values <- numeric(25)


for (k in 1:25) {

  knn_model <- kknn(intensity ~ MZ, train = ms.measured, test = ms.truth, k = k,
kernel = "optimal")

  predictions <- predict(knn_model)

  mse <- mean((predictions - ms.truth$intensity)^2)

  mse_values[k] <- mse

}

mse_values

min_mse_k <- which.min(mse_values)

min_mse_k

> best_k

[1] 6

> min_mse_k

[1] 7

> mse_values

 [1]  8.704256  5.104779  3.410489  2.656165  2.262812  2.021296  2.004127  2.084660

 [9]  2.286621  2.608518  3.012139  3.553871  4.124015  4.838148  5.619558  6.482609

[17]  7.436011  8.422623  9.547819 10.733335 11.927679 13.234540 14.597129 15.985650

[25] 17.420855
```

This means that according to cross-validation, the best value of k is 6, which may provide a good balance between smoothing and capturing important features in the data. However, when evaluating the actual mean-squared error on the test data, k = 7 is the best.

Question 3F

```
ytest.hat = fitted(kknn(intensity ~ .,ms.measured, ms.truth, kernel = "optimal", k =
6))

sd(ms.measured$intensity - ytest.hat)

> sd(ms.measured$intensity - ytest.hat)

[1] 25.82931
```

The estimated standard deviation of the sensor/measurement noise is approximately 25.82931.

Question 3G

```
> ms.truth$MZ[which.max(ytest.hat)]

[1] 7963.3
```

The MZ value corresponding to the maximum estimated abundance is approximately 7963.3

Question 3H

```
#Q3.8

knn_estimate <- function(data, indices, mz_value, k) {

  d <- data[indices,]

  knn_model <- kknn(intensity ~ MZ, train = d, test = data.frame(MZ = mz_value), k =
k, kernel = "optimal")

  return(predict(knn_model))

}


bootstrap_ci <- function(k) {

  boot_results <- boot(ms.measured, knn_estimate, R = 5000, mz_value = 7963.3, k = k)

  ci <- boot.ci(boot_results, type = "basic", conf = 0.95)

  return(ci)

}


bootstrap_ci(3)

bootstrap_ci(6)

bootstrap_ci(20)


> bootstrap_ci(3)

BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS

Based on 5000 bootstrap replicates


CALL :

boot.ci(boot.out = boot_results, conf = 0.95, type = "basic")


Intervals :

Level      Basic

95%   ( 96.82, 102.10 )

Calculations and Intervals on Original Scale

> bootstrap_ci(6)

BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS

Based on 5000 bootstrap replicates


CALL :

boot.ci(boot.out = boot_results, conf = 0.95, type = "basic")


Intervals :
```

```
Level      Basic
95%   ( 95.06, 105.81 )
Calculations and Intervals on Original Scale
> bootstrap_ci(20)
BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
Based on 5000 bootstrap replicates

CALL :
boot.ci(boot.out = boot_results, conf = 0.95, type = "basic")

Intervals :
Level      Basic
95%   (73.04, 97.51 )
Calculations and Intervals on Original Scale
```

For k = 3: 95% CI is (96.82, 102.10)

- The method is sensitive to local changes in the data and it may capture more noise from the data. It has a relatively narrow confidence interval because it takes into account local changes and is more influenced by close neighbours.

(Selected by Cross-Validation) For k = 6: 95% CI is (95.06, 105.81)

- This value represents a balance between smoothing and sensitivity to local variations. The confidence interval is somewhat wider than k=3, as it smooths the signal a bit more, reducing the noise but still capturing some of the local variation.

For k = 20: 95% CI is (73.04, 97.51)

- It becomes less sensitive to local fluctuations and focuses more on capturing the overall trend in the data. As a result, the confidence interval is wider, providing a more stable estimate but with reduced sensitivity to small-scale variations.

In summary, the variation in confidence intervals arises from the different degrees of smoothing and local sensitivity associated with different values of k. Smaller k values lead to narrower confidence intervals but may be influenced by noise. Larger k values result in wider confidence intervals but provide more stable estimates by smoothing the data.