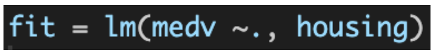
PART 3

Harshath Muruganantham

QUESTION 1

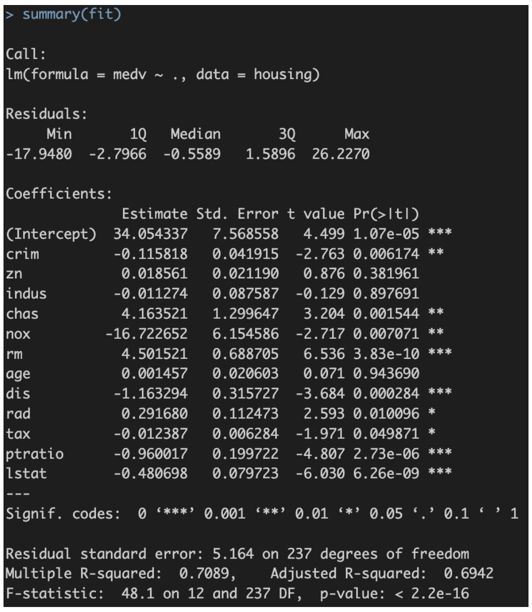
1.

On R Studio, we can use the command lm() to fit a multiple linear model to the housing data to predict *medv*:



```
fit = lm(medv ~., housing)
```

We can then use the summary(fit) function to find details of the fit.



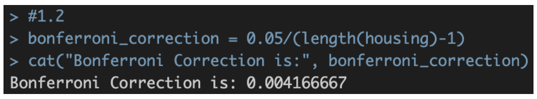
```
> summary(fit)

Call:
lm(formula = medv ~ ., 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
```

From the summary of the fitted multiple linear model, we can see that *crim, chas, nox, rm, dis, rad, tax, ptratio* and *lsat* seem to be the predictors which are associated with the median house value as their p-values are all less than 0.05. The p-value is the evidence against the null hypothesis that the coefficient of these predictors have a value of 0 at the population level (that it is associated with the median house value), so a smaller p-value will mean that there is a smaller chance of seeing an association by chance, allowing us to conclude that these predictors are associated with the median house value.

The three variables that seem to be the strongest predictors of housing price are *rm*, *lstat* and *ptratio* as they have the smallest p-values out of all the predictors.

2.

To find the Bonferroni Correction with $\alpha = 0.05$, we would have to divide 0.05 by the total number of predictors we have:



```
> #1.2
> bonferroni_correction = 0.05/(length(housing)-1)
> cat("Bonferroni Correction is:", bonferroni_correction)
Bonferroni Correction is: 0.004166667
```

Using this more conservative procedure, we can now only accept predictors that have a p-value smaller than the Bonferroni Threshold of 0.0042.

The only predictors we can now say are associated with the median house value are: *chas, rm, dis, ptratio* and *lstat*. We are now accepting less predictors (5) as opposed to before (9).

3.

In the above multiple linear model, the per-capita crime rate (*crim*) is associated with the median house price. We can see from the coefficient of *crim* in part 1, that as the per-capita crime rate increases, the median house price decreases. This happens at a rate of -0.1158 medv/crim as per the above multiple linear model.

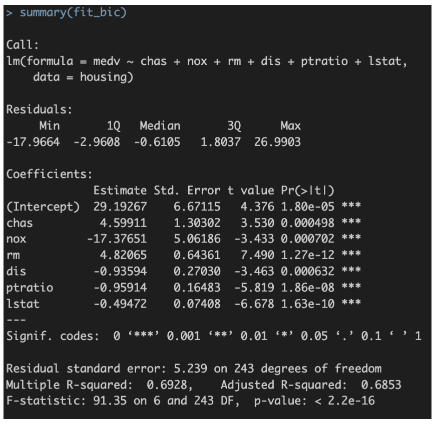
In the above multiple linear model, a suburb having frontage on the Charles river (*chas*) is associated with the median house price. We can see of the coefficient of *chas*, that if a suburb has frontage on the Charles River, then that suburb is estimated to have a higher median house price by around \$4163.52.

4.

We can use the stepwise selection with the BIC criterion to prune out potentially unimportant variables. This can be done in R through:

```
fit_bic = step(fit, k = log(length(housing$medv)), direction = "both")
```

Then we can use summary() to find the final regression equation obtained after pruning:



```
> summary(fit_bic)

Call:
lm(formula = medv ~ chas + nox + rm + dis + ptratio + lstat,
    data = housing)

Residuals:
     Min       1Q   Median       3Q      Max
-17.9664  -2.9608  -0.6105   1.8037  26.9903

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)  29.19267    6.67115   4.376 1.80e-05 ***
chas          4.59911    1.30302   3.530 0.000498 ***
nox         -17.37651    5.06186  -3.433 0.000702 ***
rm            4.82065    0.64361   7.490 1.27e-12 ***
dis          -0.93594    0.27030  -3.463 0.000632 ***
ptratio      -0.95914    0.16483  -5.819 1.86e-08 ***
lstat        -0.49472    0.07408  -6.678 1.63e-10 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 5.239 on 243 degrees of freedom
Multiple R-squared:  0.6928,    Adjusted R-squared:  0.6853
F-statistic: 91.35 on 6 and 243 DF,  p-value: < 2.2e-16
```

From here, we can see that the final regression equation will be:

$$\mathbb{E}[medv] = 4.5991 * chas - 17.3765 * nox + 4.8207 * rm - 0.9359 * dis - 0.9591 * ptratio - 0.4947 * lstat + 29.1927$$

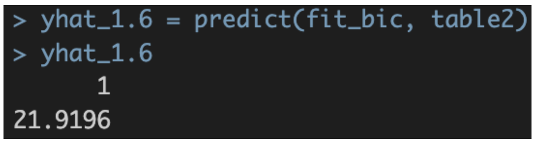
5.

According to the model in 1.4, if a council wanted to improve the median house value in their suburb, they would have to try:

- Adding frontage to the Charles River (maybe build higher for a view) (*chas*)
- Reducing Nitric oxides concentration (*nox*)
- Increasing average number of rooms per dwelling (*rm*)
- Decreasing the weighted distances to five employment centres (*dis*)
- Reducing the pupil-teacher ratio (*ptratio*)
- Decreasing the percentage of “lower status” of the population in the suburb (*lstat*)

6.

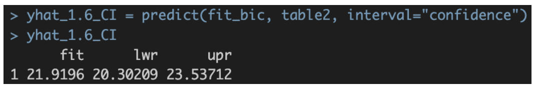
We can use the predict function in R to find a prediction for the median house price of the suburb in Table 2 using the model found in 1.4:



```
> yhat_1.6 = predict(fit_bic, table2)
> yhat_1.6
      1
21.9196
```

Using the model, we can predict that the median house price will be $21919.6 for the suburb in Table 2.

We can use the interval = "confidence" argument in the predict() function to get a 95% confidence interval for this prediction:

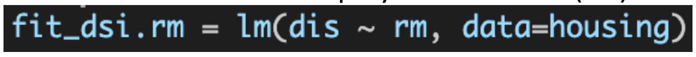


```
> yhat_1.6_CI = predict(fit_bic, table2, interval="confidence")
> yhat_1.6_CI
      fit      lwr      upr
1 21.9196 20.30209 23.53712
```

We are 95% confident that the predicted median house price made by our model for the suburb in Table 2 is between $20302.09 and $23537.12.

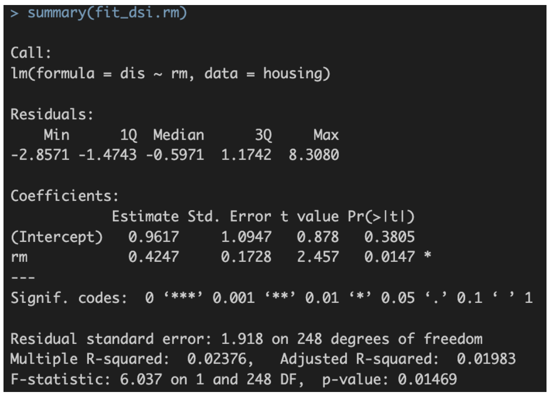
7.

We can use the lm function between the number of rooms a dwelling has (*rm*) and the distance to one of the employment centres (*dis*):



```
fit_dsi.rm = lm(dis ~ rm, data=housing)
```

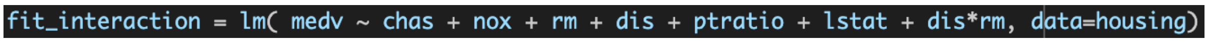
Now using the summary() function to analyse the fit:



```
> summary(fit_dsi.rm)

Call:
lm(formula = dis ~ rm, data = housing)

Residuals:
    Min      1Q  Median      3Q     Max 
-2.8571 -1.4743 -0.5971  1.1742  8.3080 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)  
(Intercept)   0.9617     1.0947   0.878   0.3805  
rm            0.4247     0.1728   2.457   0.0147 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 1.918 on 248 degrees of freedom
Multiple R-squared:  0.02376,	Adjusted R-squared:  0.01983 
F-statistic: 6.037 on 1 and 248 DF,  p-value: 0.01469
```

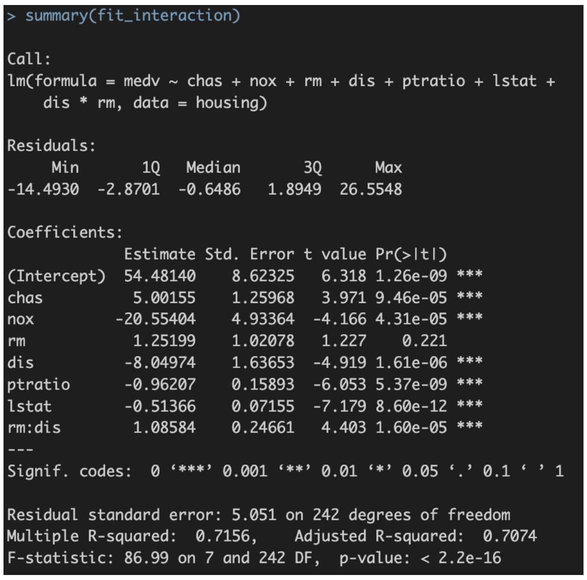
Since the p-value is less than 0.05 (at 0.0147), we are able to conclude that there is a strong possibility for an interaction between *dis* and *rm.*

As to what effect the interaction has with the model, we can fit a model using the predictors found in 1.4 and the interaction predictor:



```
fit_interaction = lm( medv ~ chas + nox + rm + dis + ptratio + lstat + dis*rm, data=housing)
```

Now using summary() to analyse the fit:



```
> summary(fit_interaction)

Call:
lm(formula = medv ~ chas + nox + rm + dis + ptratio + lstat +
    dis * rm, data = housing)

Residuals:
     Min       1Q   Median       3Q      Max
-14.4930  -2.8701  -0.6486   1.8949  26.5548

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)  54.48140    8.62325   6.318 1.26e-09 ***
chas          5.00155    1.25968   3.971 9.46e-05 ***
nox         -20.55404    4.93364  -4.166 4.31e-05 ***
rm            1.25199    1.02078   1.227    0.221
dis          -8.04974    1.63653  -4.919 1.61e-06 ***
ptratio      -0.96207    0.15893  -6.053 5.37e-09 ***
lstat        -0.51366    0.07155  -7.179 8.60e-12 ***
rm:dis        1.08584    0.24661   4.403 1.60e-05 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 5.051 on 242 degrees of freedom
Multiple R-squared:  0.7156,	Adjusted R-squared:  0.7074
F-statistic: 86.99 on 7 and 242 DF,  p-value: < 2.2e-16
```

There seems to be a strong association between the interaction and the model from 1.4, due to the interaction having a very small p-value (at $1.60 * 10^{-5}$), suggesting that it is unlikely for this association to have risen by chance.

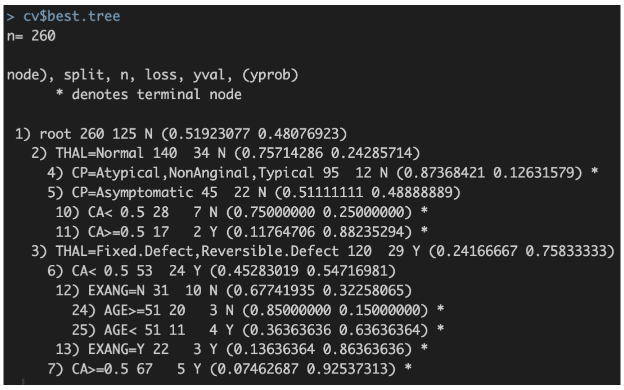
There is also a slight decrease in the residual standard error on this new model as opposed to the model in 1.4, suggesting that is probable that this interaction is associated with the median house price.

As to what effect this interaction has on the median house price, as the interaction increases, the median house price of the suburb also seems to increase.

QUESTION 2

1.

Using cross validation, we can find the best tree for the data on R studio:

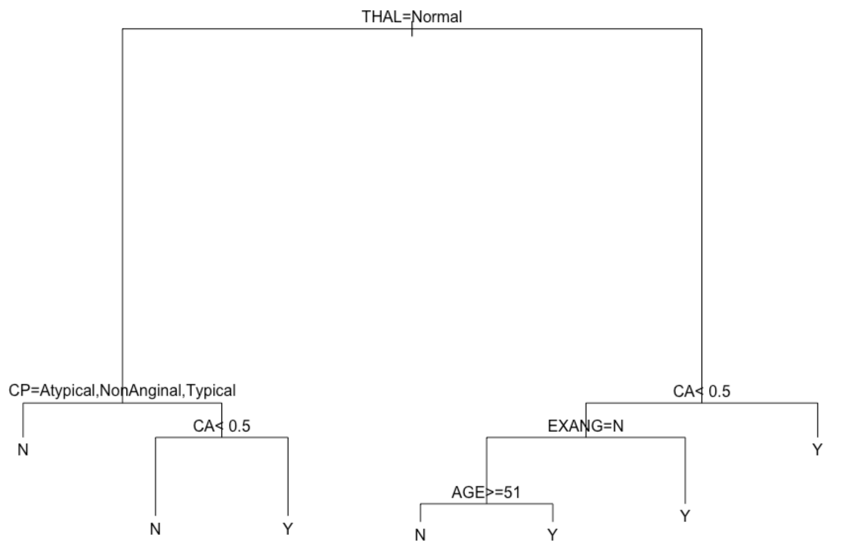


```
> cv$best.tree
n= 260

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) *
```

The variables used in the best tree are THAL, CP, CA, EXANG and AGE.

The best tree has 7 leaf (terminal) nodes.

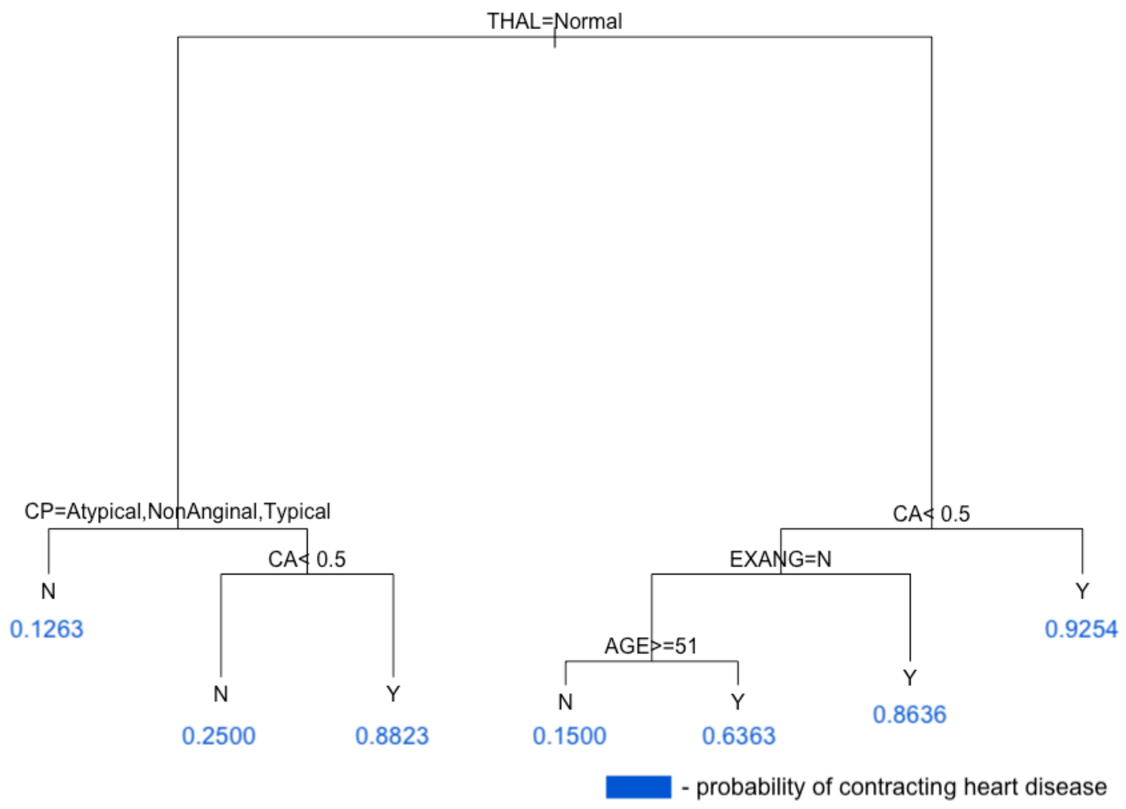
2. Here is the tree found by CV:



From this model, you can interpret that:

- If your Thallium scanning results are not normal, and the number of major vessels coloured by fluoroscopy is greater than or equal to 0.5, then you are likely to have heart disease.
- If your Thallium scanning results are not normal, and the number of major vessels coloured by fluoroscopy is less than 0.5, and you have exercise induced angina, then you are likely to have heart disease.
- If your Thallium scanning results are not normsal, and the number of major vessels coloured by fluoroscopy is less than 0.5, and you don't have exercise induced angina, and you are less than 51 years old, then you are likely to have heart disease.
- If your Thallium scanning results are not normal, and the number of major vessels coloured by fluoroscopy is less 0.5, and you don't have exercise induced angina, and you are greater than or equal to 51 years old, then you are likely to not have heart disease.
- If your thallium scanning results are normal, and you have Asymptomatic chest pain and the number of major vessels coloured by fluoroscopy is less than 0.5, then you are likely to not have heart disease.
- If your thallium scanning results are normal, and you have Asymptomatic chest pain and the number of major vessels coloured by fluoroscopy is more than or equal to 0.5, then you are likely to have heart disease.
- If your thallium scanning results are normal, and you have Typical, Atypical or Non Anginal chest pain, then you are likely to not have heart disease.

3.



4.

According to the above tree, the predictor combination that results in the highest probability of having heart-disease is:

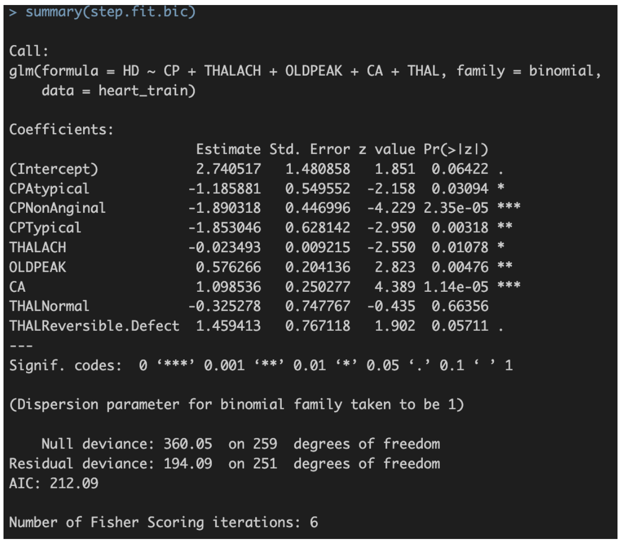
- Thallium Scanning Results is not Normal (THAL = Fixed.Defect or Thal = Reversible.Defect).
- Number of major vessels coloured by fluoroscopy is greater than or equal to 0.5 (CA ≥ 0.5).

5.

We can use the glm() function to fit a logistic regression model to the hard data, and then use stepwise selection with the BIC score to prune the model:

```
fullmod=glm(HD ~ ., data = heart_train, family = binomial)
step.fit.bic = step(fullmod, k = log(nrow(heart_train)), direction = "both")
```

Now we can use summary() to find details about this new model:



```
> summary(step.fit.bic)

Call:
glm(formula = 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
```

The variables included in this final model are CPAtypical, CPNonAnginal, CPTypical, THALACH, OLDPEAK, CA, THALNormal and ThalReversible.Defect.

Comparing this to the initial model using the CV tree, the CV tree contained the variables THAL, CP, CA, EXANG and AGE. So, this model contains THALACH and OLDPEAK which the CV best Tree does not, and the CV Best Tree uses EXANG and AGE which this model does not. Both models use 5 variables in total.

The most important predictor in the logistic regression seems to be the CA level as it is has the lowest p-value, suggesting that this predictor is the least likely of the others to have an association with heart disease that would have arisen by chance.

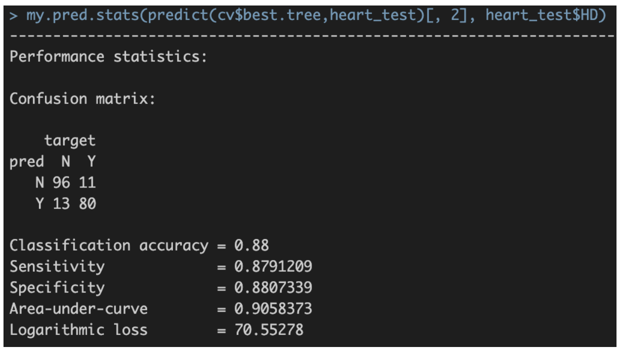
6.

The regression equation for the above model is:

$$E[\text{HD}] = 2.7405 - 1.1859 * \text{CPAtypical} - 1.8903 * \text{CPNonAnginal} - 1.8530 * \text{CPTypical} - 0.0235 * \text{THALACH} + 0.5763 * \text{OLDPEAK} + 1.0985 * \text{CA} - 0.3253 * \text{THALNormal} + 1.4594 * \text{THALRevisible.Defect}$$

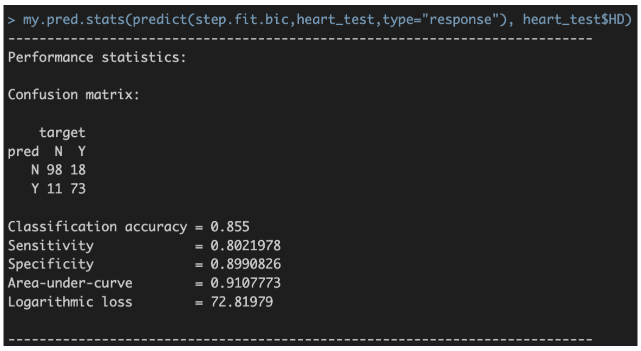
7.

The prediction statistics of the heart-test data for the tree model is:



```
> my.pred.stats(predict(cv$best.tree,heart_test)[, 2], heart_test$HD)
---------------------------------------------------------------------
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
```

The Prediction statistics of the heart-test data for the step-wise logistic regression model is:

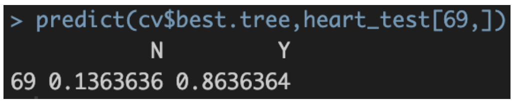


```
> my.pred.stats(predict(step.fit.bic,heart_test,type="response"), heart_test$HD)
---------------------------------------------------------------------
Performance statistics:

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

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

Looking at both models, I would say that the tree model is preferable as opposed to the step-wise logistic regression model. This is because the prediction made by the tree has a better classification accuracy at 0.88 compared to the step-wise logistic regression model at

0.855. The tree also has a much better sensitivity at 0.8791 comapred to the logistic model at 0.8022. This means that the tree is better at finding the true positive (i.e. can more accurately identify people with a heart disease correctly out of all subjects who actually have a heart disease). The specificity of the tree is slightly lower (which is expected because of the substantial improvement is specificity) at 0.8807 as opposed to 0.8991 in the other model. This is the ability of the model to properly classify people without heart disease out of all subjects who actually don't have heart disease. The tree also has a lower log loss meaning it is better in predicting our data.

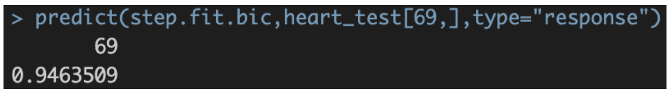
8.
   a)
      The odds of the 69th patient in the test dataset having heart disease according to the tree model found using cross-validation can be found through:



```
> predict(cv$best.tree,heart_test[69,])
          N         Y
69 0.1363636 0.8636364
```

The probability of the 69th patient in the test dataset **having** heart disease according to the tree model found using cross-validation is 0.8636. The odds are $\frac{0.8636}{0.1363} = 6.3333$.

   b) The odds of the 69th patient in the test dataset having heart disease according to the step-wise logistic regression model can be found through:

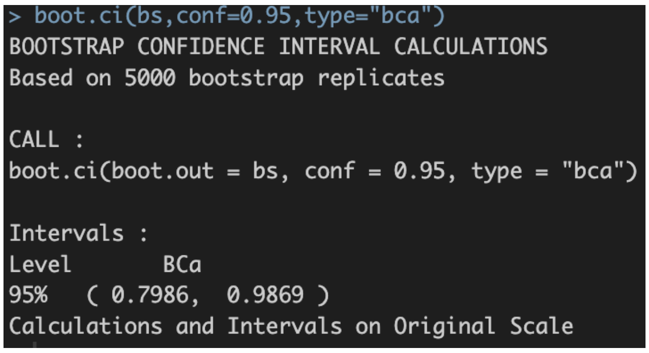


```
> predict(step.fit.bic,heart_test[69,],type="response")
       69
0.9463509
```

The probability of the 69th patient in the test dataset having heart disease according to the step-wise logistic regression model is 0.9464. The odds are $\frac{0.9464}{1-0.9464} = \frac{0.9464}{0.0536} = 17.6397$.

Looking at both the models, the predicted odds of the stepwise logistic regression model are much higher compared to the tree model, meaning that the logistic regression model is much more confident that the 69th person in the test dataset has a heart disease as opposed to the tree model. Looking at the test data manually, the 69th person does in-fact have heart disease.

9. The Confidence Interval found for the probability of the 69th person in the test data set having heart disease using the bootstrap procedure is ($0.7986, 0.9869$).

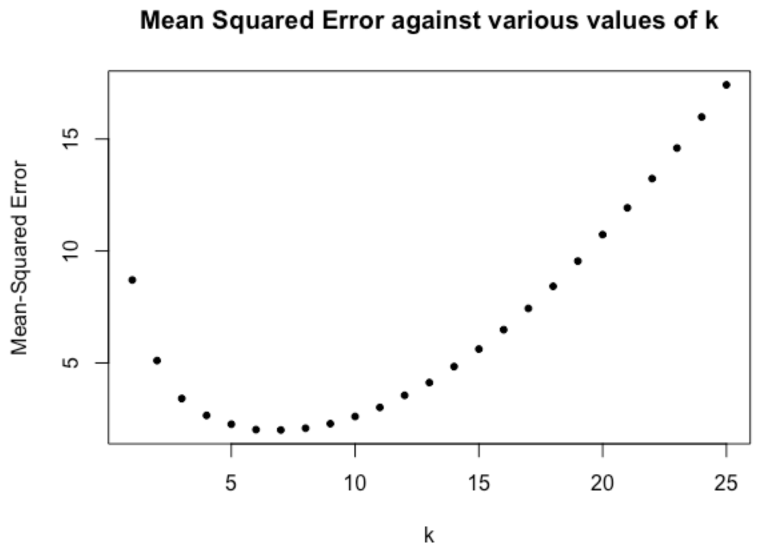


```
> boot.ci(bs,conf=0.95,type="bca")
BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
Based on 5000 bootstrap replicates

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

Intervals :
Level       BCa
95%   ( 0.7986,  0.9869 )
Calculations and Intervals on Original Scale
```

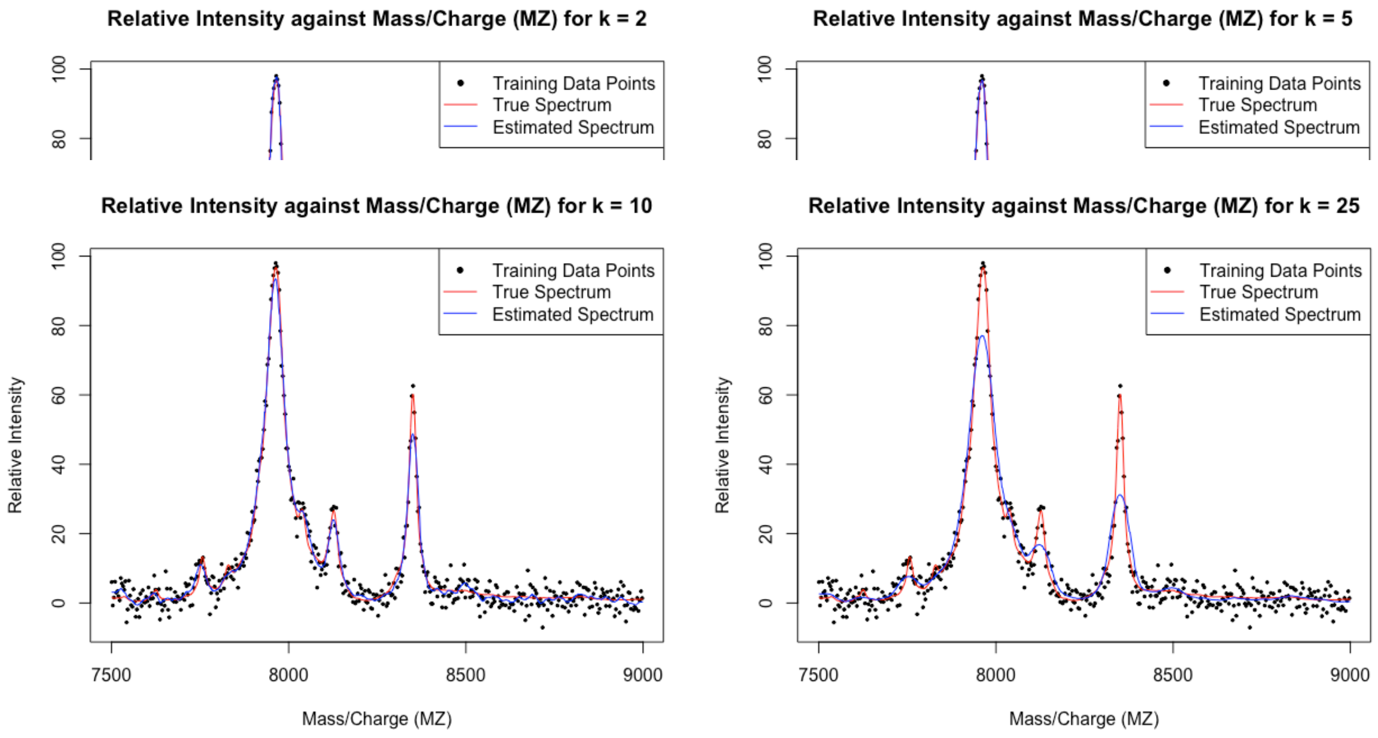
These values suggest that we are 95% confident that the odds of the 69th person in the test data set having heart disease is between 0.7986 and 0.9869. This is a slightly wide margin. The odds found using the tree model and the logistic regression model both sit within (around the middle of) the provided confidence interval.

QUESTION 3

1.



2.



3.

Qualitatively, as k increases, the estimated spectrum line becomes smootherwherease in a smaller k te estimated line is more jumpy. Another qualitative difference that is viewavel is that as k increases, the peaks of the estimated spectrum decrease from the tue peak.

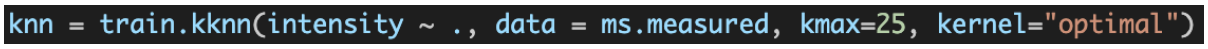
Quantitively, the MSE when $k = 2$ is 5.1048, MSE when $k = 5$ is 2.2628, the MSE when $k = 10$ is 2.6085 and the MSE when $k = 25$ is 17.4209. We can see that the MSE is lowest when $k = 5$, and increases after that as the k value increases. It is also observable that as the k value decreases from 5, the MSE increases as well.

4.

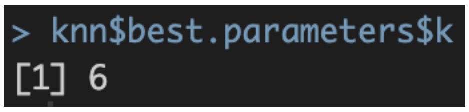
I think the estimated spectra when $k = 5$ as per Q3.2 and Q3.3 does seem to achieve our aim of providing a smooth, low noise estimate of background level as well as an accurate estimation of the peak. I believe the k-NN method is able to achieve this as the k-NN method uses the k nearest neighbours of the Intensity for a particular value form the training data set. The method then calculates the estimated spectrum as a weighted average of the k most similar spectra. This ensures that a layer of smoothening (thereby reducing noise) is achieved. By using a lower k-value (selecting a lower number of neighbours) up until a certain threshold, we are able to retain some of the original data accuracies such as height of peaks.

5.

Using R studio, we can use cross validation functionality in the kknn package to select an estimate of the best value of k:



```
knn = train.kknn(intensity ~ ., data = ms.measured, kmax=25, kernel="optimal")
```



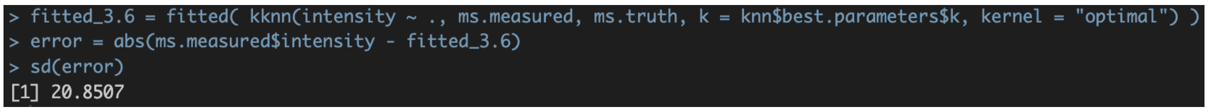
```
> knn$best.parameters$k
[1] 6
```

The value of k that this method selects is 6.

The value of k that would minimise the actual mean-squared error as computed in Q3.1 is 7. The value found be the cross-validation functionality is lower than the value found in Q3.1.

6.

After fitting a kknn model based on the results from 3.5, we can calculate the difference between the true intensity and our model to find the noise, then find the standard deviation of the difference.

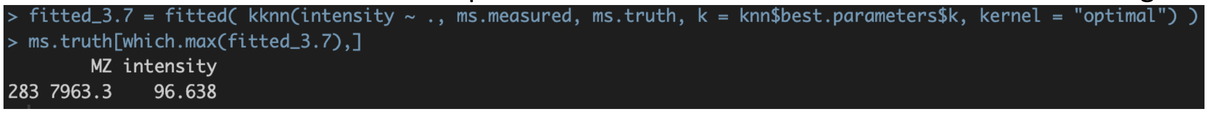


```
> fitted_3.6 = fitted( kknn(intensity ~ ., ms.measured, ms.truth, k = knn$best.parameters$k, kernel = "optimal") )
> error = abs(ms.measured$intensity - fitted_3.6)
> sd(error)
[1] 20.8507
```

The standard deviation of the sensor/measurement noise that has corrupted our intensity measurements is 20.8507.

7.

Using R Studio, from the smoothed signal produced using the value of k found in Q3.5, we can find the value of MZ that corresponds to the maximum estimated abundance through:

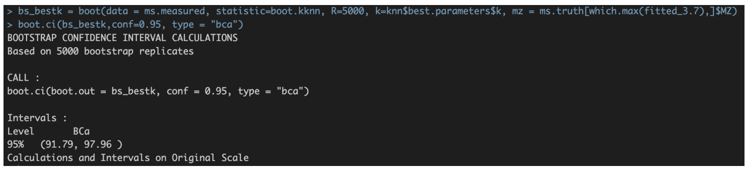


```
> fitted_3.7 = fitted( kknn(intensity ~ ., ms.measured, ms.truth, k = knn$best.parameters$k, kernel = "optimal") )
> ms.truth[which.max(fitted_3.7),]
        MZ intensity
283 7963.3    96.638
```

As per R Studio, the MZ value corresponding to the peak as per our smoothed signal is $MZ = 7963.3$.

8.

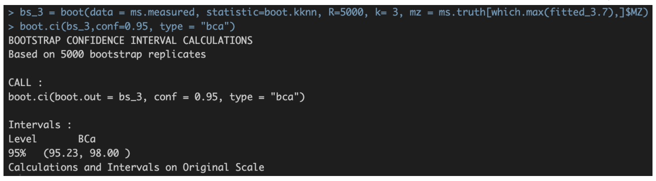
For the best k value, the bootstrapped 95% Confidence Interval for the estimate of relative abundance at the MZ value found in Q3.7 is:



```
> bs_bestk = boot(data = ms.measured, statistic=boot.kknn, R=5000, k=knn$best.parameters$k, mz = ms.truth[which.max(fitted_3.7),]$MZ)
> boot.ci(bs_bestk,conf=0.95, type = "bca")
BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
Based on 5000 bootstrap replicates

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

Intervals : 
Level       BCa          
95%   (91.79, 97.96 )  
Calculations and Intervals on Original Scale
```

We can see here that we are 95% confident that the estimate of relative abundance at the MZ value found in Q3.7 is between 91.79 and 97.96 for the best k value as per CV.

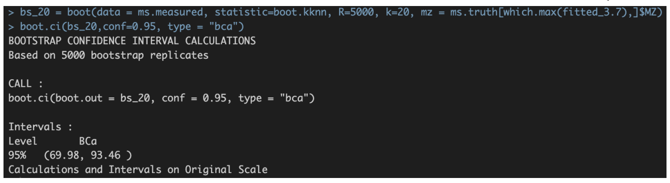
For the $k = 3$ neighbours, the bootstrapped 95% Confidence Interval for the estimate of relative abundance at the MZ value found in Q3.7 is:



```
> bs_3 = boot(data = ms.measured, statistic=boot.kknn, R=5000, k= 3, mz = ms.truth[which.max(fitted_3.7),]$MZ)
> 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%   (95.23, 98.00 )  
Calculations and Intervals on Original Scale
```

We can see here that we are 95% confident that the estimate of relative abundance at the MZ value found in Q3.7 is between 95.23 and 98.00 for $k = 3$.

For the $k = 20$ neighbours, the bootstrapped 95% Confidence Interval for the estimate of relative abundance at the MZ value found in Q3.7 is:



```
> bs_20 = boot(data = ms.measured, statistic=boot.kknn, R=5000, k=20, mz = ms.truth[which.max(fitted_3.7),]$MZ)
> 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%   (69.98, 93.46 )  
Calculations and Intervals on Original Scale
```

We can see here that we are 95% confident that the estimate of relative abundance at the MZ value found in Q3.7 is between 69.98 and 93.46 for $k = 20$.

These CIs vary in size for different k values because, as we saw from Q3.2, as the k values increased from the best value, the peaks became shorter, decreasing the intensity values, which explains the low CI for $k = 20$. As the k values become smaller, the distance between the intervals decreases as you retain more of the original peaks allowing you to produce more accurate intensities whilst bootstrapping leading to shorter intervals.