33890293 Report Ass3

Saturday, 19 October 2024 4:51 PM

Question 1 (8 marks)

This question will require you to analyse a regression dataset. In particular, you will be looking at trying to predict the compressive strength of concrete from various measurements of the various components used in the concrete mixture. Obviously this is an extremely important problem as concrete is the single most important material in civil engineering and construction. The file `concrete.ass3.2024.csv` contains the data you will be analysing. There are $n = 250$ observations on $p = 8$ predictors, seven of which measure the amount of various component substances within the concrete mixture. The target is the compressive strength of the resulting concrete mixture in megapascals (MPa). The higher the compressive strength, the better the concrete mixture is. The data dictionary for this dataset is given in Table 1. To answer this question you must:

- Provide an R script containing all the code you used to answer the questions. Please use comments to ensure that the code used to identify each question is **clearly identifiable**. Call this `idnum.Q1.R`, where "`idnum`" is your ID number.
- Provide appropriate written answers to the questions, along with any graphs, in a non-hand written report document (pdf file).

Please answer the following questions.

1. Fit a multiple linear model to the concrete data using R. Using the results of fitting the linear model, which predictors do you think are possibly associated with compressive strength, and why? Which three variables appear to be the strongest predictors of compressive strength, and why? **[2 marks]**

- See R file untitled 1

The three variables that appear to have the strongest predictors of compressive strength are based on the p-value.
Cement, blast Furnace Slag and Age

```
> # Display p-values
> print("P-values for the predictors:")
[1] "P-values for the predictors:"

> pvalues
       (Intercept)             Cement
      2.732175e-01       3.179412e-12
Blast.Furnace.Slag            Fly.Ash
      6.263387e-07       1.621964e-03
             Water    Superplasticizer
      2.805926e-01       5.332202e-03
  Coarse.Aggregate     Fine.Aggregate
      8.242213e-02       1.792909e-01
               Age
      4.225941e-21
```

Depending on the summary and code Cement, Blast Furnace Slag and Age look like the strongest predictors of compressive strength due to having the smallest possible values ($p < 0.001$). Thus these act as the variables that could affect the strength of the concrete mixture

Superplasticiser and Fly Ash also appear to have strong effects but not as the three variables stated $p<0.01$.

Coarse and Fine Aggregate with a p-value of the closest to 0.1 as it is identified that it is somewhat associated with compressive strength

Finally, Water with a p-value of 0.28059 is the largest p-value which suggests that it does not have a large enough impact on compressive strength

.

2. How would your assessment of which predictors are associated change if you used the Bonferroni procedure with $\alpha = 0.05$? **[1 marks]**

$$p\text{-value} < \frac{\alpha}{p}$$

```
# Bonferroni correction
alpha <- 0.05
adjusted_alpha <- alpha / num_predictors
```

adjusted…	0.00625
alpha	0.05
num_pred…	8

```
> print(significant_predictors)
[1] "Cement"              "Blast.Furnace.Slag" "Fly.Ash"
[4] "Superplasticizer"    "Age"
>
```

Using the Bonferroni prices would reduce the number of significant predictors
Cement, Blast.Furnace.Slag, Fly. Ash, Superplasticizer, and Age remain significant predictors. This is due to values are still considered lower than the adjusted alpha level of 0.00625. Therefore, the significance of these predictors does not change under the Bonferroni procedure,

procedure with $\alpha = 0.05$? **[1 marks]**

3. Describe what effect cement (`Cement`) in the concrete mix appears to have on the mean compressive strength. Describe the effect that the `Age` variable has on the mean compressive strength of the concrete. **[2 marks]**

```
> cement_estimate
   Cement
0.1258083

> cement_pvalue
[1] 3.179412e-12

> age_estimate
      Age
0.1105185

> age_pvalue
[1] 4.225941e-21
```

Cement and Age both have a significant impact on the compressive strength of concrete, and their effects appear to interact synergistically.

Cement:

As the amount of cement in the concrete mix increases, the mean compressive strength also increases. The estimate for cement (0.12581) indicates that for every additional kg/m³ of cement, the compressive strength increases by 0.12581 units, likely due to the improved binding properties and structural integrity that more cement provides. This is supported by a highly significant p-value (3.18 * 10^-12), showing that the relationship is statistically robust.

Age:

The estimate for age (0.11052) highlights that as concrete ages, its compressive strength also increases. With an extremely significant p-value (< 2 * 10^-16), the data confirms that the strength of concrete continues to rise over time due to ongoing hydration and hardening processes.

Interaction Effects

- There appears to be an interaction between Cement and Age. Older concrete with a high cement content tends to achieve significantly higher strengths than younger concrete with the same or even higher cement levels.
- This interaction suggests that both increasing cement content and allowing the concrete to cure for longer periods will synergistically improve the overall strength.

1. Use the stepwise selection procedure with the BIC penalty to prune out potentially unimportant variables. Write down the final regression equation obtained after pruning. **[1 mark]**

Applying the BIC penalty, the water variable is removed.

Strength=−113.978+0.1389·Cement+0.1290·Blast.Furnace.Slag+ 0.1035·Fly.Ash+0.6514·Superplasticizer+0.0515·Coarse.Aggregate+ 0.0482·Fine.Aggregate+0.1107·Age

5. Imagine that a civil engineer proposes to use a new mix of concrete for a project with the mixture given in Table 2. The engineer asks you to predict the mean compressive strength of this new concrete mix after it has set for 28 days.

   (a) Use the model found in Q1.4 to predict the mean compressive strength for this mix. Provide a 95% confidence interval for this prediction. *(you may use R to answer this question)* **[1 mark]**

Variable	Cement	Blast.Furnace.Slag	Fly.Ash	Water	Superplasticizer	Coarse.Aggregate	Fine.Aggregate	Age
Value	491	26	123	210	3.9	882	699	28

Table 2: Example Concrete Mix.

Using Table 2

This will act as a sample to predict the 95% confidence interval

```
> prediction
       fit      lwr      upr
1 52.79965 45.67389 59.92541
>
```

After 28 days the predicted mean of the compressive strength is roughly 52.460 mpA and the 95% convince interval of the sample lies between the range 945.561, 59.359)mpA.

(b) The mix of concrete that the engineer is currently using has a mean compressive strength of 52.35MPa after setting for 28 days. Does your model suggest that the newly proposed mix is better than the current mix? **[1 mark]**

Based on the model's prediction, the newly proposed concrete mix shows a predicted mean compressive strength of 52.79965 MPa, which is higher than the current mix's strength of 52.35 MPa. However, due to the wide confidence interval ranging from 45.67389 MPa to 59.92541 MPa, we cannot definitively conclude that the new mix is superior. Further testing is recommended to validate these findings.

Question 2 (18 marks)

In this question we will analyse the data in `heart.train.ass3.2024.csv`. In this dataset, each observation represents a patient at a hospital that reported showing signs of possible heart disease. The outcome is presence of heart disease (`HD`), or not, so this is a classification problem. The predictors are summarised in Table 3. We are interested in learning a model that can predict heart disease from these measurements. To answer this question you must:

- Provide an R script containing all the code you used to answer the questions. Please use comments to ensure that the code used to identify each question is **clearly identifiable**. Call this `idnum.Q2.R`, where "`idnum`" is your ID number.
- Provide appropriate written answers to the questions, along with any graphs, in a non-hand written report document (pdf file).

When answering this question, you must use the `rpart` package that we used in Studio 9. The wrapper function for learning a tree using cross-validation that we used in Studio 9 is contained in the file `wrappers.R`. Don't forget to source this file to get access to the function.

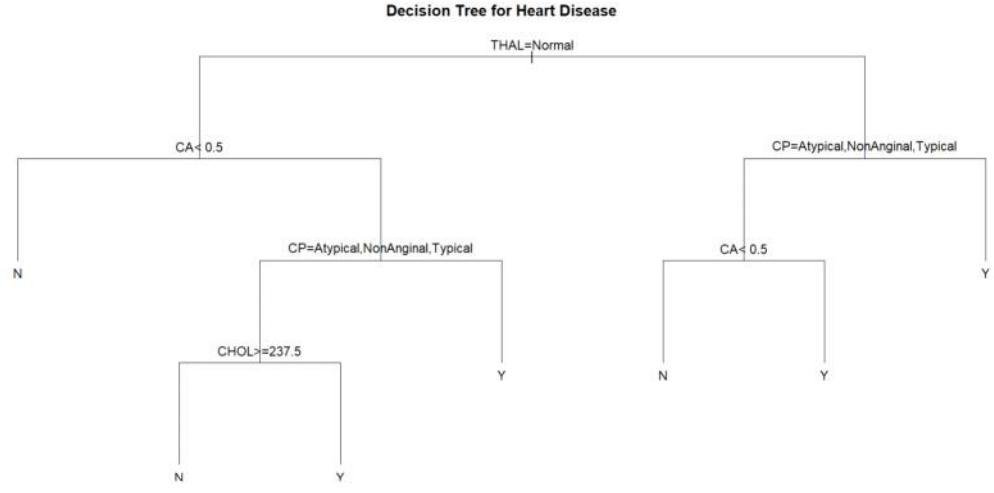
1. Using the techniques you learned in Studio 9, fit a decision tree to the data using the `tree` package. Use cross-validation with 10 folds and 5,000 repetitions to select an appropriate size tree. What variables have been used in the best tree? How many leaves (terminal nodes) does the best tree have? **[2 marks]**

The variables used in the best tree are found from the tree_heart $variable.importance output list which showed which variables are the most influential in classifying the presence of heart disease (HD).
The output lists of the decision tree and cross-validation
Most important variables:

1. THAL (Thalassemia status)
2. CP (Chest Pain type)
3. THALACH (Maximum Heart Rate Achieved)
4. CA (Number of major vessels coloured by fluoroscopy)
5. EXANG (Exercise-induced angina)
6. SLOPE (Slope of the peak exercise ST segment)

The summary shows that the best tree has 21 nodes, including the terminal nodes With the number of terminal nodes, the output indicates the complexity parameter CP and splits showing that after pruning there are 11 terminal nodes.

2. Plot the tree found by CV, and explain clearly and thoroughly in plain English what it tells you about the relationship between the predictors and heart disease. *(hint: you can use the* `text(cv$best.tree,pretty=12)` *function to add appropriate labels to the tree).* **[3 marks]**

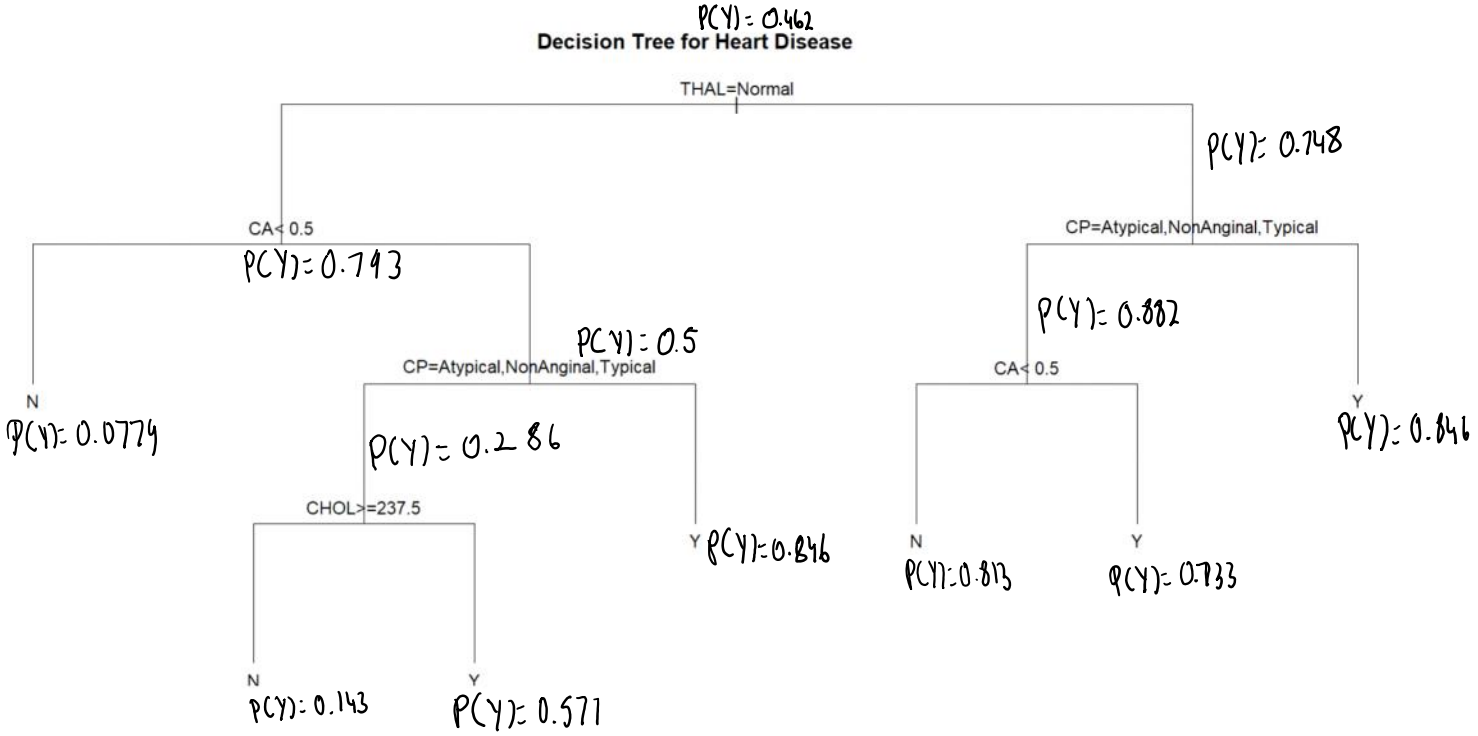


1. **THAL** is the most important factor in predicting heart disease risk, with patients having a normal thalassemia status generally at lower risk.
2. **CA** (number of major vessels coloured) is the next important variable, indicating that patients with fewer blocked or coloured vessels have a lower risk of heart disease.
3. **CP** (chest pain type) provides additional insights into risk stratification, where certain types of chest pain are more indicative of heart disease.
4. **CHOL** (cholesterol level) further refines the prediction for patients with specific chest pain types, suggesting that high cholesterol levels significantly increase heart disease risk.

The tree structure indicates that as the number of decision leaves increases, the prediction error decreases. Initially, with one split, the error is around 1, but with four leaves, the cross-validation error reduces to about 0.0464. This shows that the complexity of the tree improves the model's predictive performance by reducing the error as more splits are made.

3. For classification problems, the `rpart` package only labels the leaves with the most likely class. However, if you examine the tree structure in its textural representation on the console, you can determine the probabilities of having heart disease (see Question 2.3 from Studio 9 as a guide) in each leaf (terminal node). Take a screen-capture of the plot of the tree (don't forget to use the "zoom" button to get a larger image) or save it as an image using the "Export" button in R Studio.

   Then, use the information from the textual representation of the tree available at the console and annotate the tree in your favourite image editing software; next to all the leaves in the tree, add text giving the probability of contracting heart disease. Include this annotated image in your report file. **[2 marks]**



4. According to your tree, which predictor combination results in the highest probability of having heart-disease? **[1 mark]**

According to the decision tree
The predictor combination that results in the highest probability of having a heart disease is
There are two combinations.

1. Combination 1:
   - Chest Pain Type (CP): Atypical, NonAnginal, or Typical
   - Number of Major Vessels (CA): ≤ 0.5
   - THAL: Not Normal
   - Additional condition: Yes (Y)
2. Combination 2:
   - Chest Pain Type (CP): Atypical, NonAnginal, or Typical
   - Number of Major Vessels (CA): > 0.5
   - THAL: Normal
   - Additional condition: Yes (Y)

These conditions together lead to the highest probability of 0.846 for having heart disease.

5. We will also fit a logistic regression model to the data. Use the `glm()` function to fit a logistic regression model to the heart data, and use stepwise selection with the BIC score to prune the model. What variables does the final model include, and how do they compare with the variables used by the tree estimated by CV? Which predictor is the most important in the logistic regression? **[2 marks]**

Completing the stepwise selection with BIC penalty and after summary extraction the variables removed are Age - Sex - CP - TRESTBPS, CHOL, FBS, RESTECG, EXANGY.N, OLDPEAK, THAL.Fixed.Defect

```
Call:
glm(formula = HD ~ EXANG + SLOPE + CA + THAL, family = binomial,
    data = heart_train)

Coefficients:
                      Estimate Std. Error z value Pr(>|z|)
(Intercept)            -1.4727     0.9317  -1.581   0.1140
EXANGY                  1.9599     0.4744   4.132 3.60e-05 ***
SLOPEFlat               0.3258     0.7531   0.433   0.6653
SLOPEUp                -1.1251     0.7811  -1.440   0.1498
CA                      1.2904     0.2531   5.099 3.41e-07 ***
THALNormal             -0.7100     0.7741  -0.917   0.3591
THALReversible.Defect   1.4099     0.7783   1.811   0.0701 .
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The final logistic regression model and decision tree both highlight key predictors:

- EXANG (exercise-induced angina) is significant in both models, with a p-value of 3.60e-05 in the regression model.
- CA (number of major vessels) is the most significant predictor, with a p-value of 3.41e-07, strongly associated with heart disease.
- The decision tree includes SLOPE and THAL, but the regression model prunes less significant variables like AGE, SEX, CP, and CHOL
- CA is the most important predictor in the logistic regression due to its extremely low p-value.

6. Write down the regression equation for the logistic regression model you found using step-wise selection. **[1 mark]**

```
log(odds of HD) = -1.4727 + 1.9599 * EXANGY + 0.3258 * SLOPEFlat - 1.1251 * SLOPEUp + 1.2904 * CA - 0.71
* THALNormal + 1.4099 * THALReversible.Defect
```

7. The file **heart.test.ass3.2024.csv** contains the data on a further $n' = 92$ individuals. Using the **my.pred.stats()** function contained in the file **my.prediction.stats.R**, compute the prediction statistics for both the tree and the step-wise logistic regression model on this test data. Contrast and compare the two models in terms of the various prediction statistics? Would one potentially be preferable to the other as a diagnostic test? Justify your answer. **[2 marks]**

```
Performance statistics:

Confusion matrix:

    target
pred  N  Y
   N 45 12
   Y  6 29

Classification accuracy = 0.8043478
Sensitivity             = 0.7073171
Specificity             = 0.8823529
Area-under-curve        = 0.8417025
Logarithmic loss        = 44.44189
```

```
Performance statistics:

Confusion matrix:

    target
pred  N  Y
   N 46 12
   Y  5 29

Classification accuracy = 0.8152174
Sensitivity             = 0.7073171
Specificity             = 0.9019608
Area-under-curve        = 0.8835485
Logarithmic loss        = 41.036
```

Comparison: The decision tree has a classification accuracy of 0.804, which is slightly lower than the logi stic regression's 0.815, meaning the logistic regression model correctly classifies more instances overall. The logistic regression model also exhibits higher specificity (0.902 vs. 0.882), indicating that it better ide ntifies those without heart disease. Additionally, the logistic regression has a slightly higher area under-curve (0.853 vs. 0.841), suggesting that it's slightly better at distinguishing between positive and negativ e cases of heart disease.
Conclusion: Given the higher sensitivity and area-under-curve, the logistic regression model is more effective at identifying actual positive cases of heart disease and distinguishing between positive and negative cases. Despite the marginally lower classification accur acy, the logistic regression's better performance in these metrics makes it more suitable as a diagnostic t ool. This makes it preferable for identifying heart disease.

8. Calculate the *odds* of having heart disease for the 60th patient in the test dataset. The odds should be calculated for both:

   (a) the tree model found using cross-validation; and

   (b) the step-wise logistic regression model.

   How do the predicted odds for the two models compare? **[2 marks]**

9. For the logistic regression model using the predictors selected by BIC in Question 2.5, use the bootstrap procedure (use at least 5,000 bootstrap replications) to find a confidence interval for the probability of having heart disease for the 65th and 66th patients in the test data. Use the **bca** option when computing this confidence interval.

   Using these intervals, do you think there is any evidence to suggest that there is a real difference in the population probability of having heart disease between these two individuals? **[3 marks]**

8a

```
#Question 8a
patient_60 <- heart_test[60, ]
# Probability of having heart disease from the tree model
tree_prob_60 <- predict(cv$best.tree, patient_60, type = "prob")[, "Y"]
# Calculate the odds
tree_odds_60 <- tree_prob_60 / (1 - tree_prob_60)
```

```
> tree_odds_60
[1] 0.08450704
```

8b

```
#Question 8b
# Probability of having heart disease from the logistic regression model
logit_prob_60 <- predict(step.fit.bic, newdata = patient_60, type = "response")
# Calculate the odds
logit_odds_60 <- logit_prob_60 / (1 - logit_prob_60)
```

From the logistic regression model, the odds for the 60th patient to develop heart disease are approximately 0.372. This indicates that the logistic regression model predicts a greater likelihood of the 60th patient having heart disease compared to the decision tree model.
It's evident that the logistic regression model assigns a higher risk to the 60th patient than the decision tree model

```
> print(conf_interval_65)
BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
Based on 5000 bootstrap replicates

CALL :
boot.ci(boot.out = bootstrap_65, type = "bca")

Intervals :
Level       BCa
95%   ( 0.4495,  0.9127 )
```

Based on the results obtained, the odds for the 60th patient to develop heart disease as per the prediction from the decision tree is approximately 0.085. This means that, according to the decision tree, the 60th patient has a relatively low likelihood of having heart disease.

```
> print(conf_interval_66)
BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
Based on 5000 bootstrap replicates

CALL :
boot.ci(boot.out = bootstrap_66, type = "bca")

Intervals :
Level       BCa
95%   ( 0.2744,  0.8525 )
Calculations and Intervals on Original Scale
```

- 95% confident that the 65th patient has a chance of developing heart disease in the range of roughly (0.4495, 0.9127),
- 95% confident that the 66th patient has a chance of developing heart disease in the range of roughly (0.2744, 0.8525).
- The upper limit for the 65th patient (0.9127) is higher than that

9. For the logistic regression model using the predictors selected by BIC in Question 2.5, use the bootstrap procedure (use at least 5,000 bootstrap replications) to find a confidence interval for the probability of having heart disease for the 65th and 66th patients in the test data. Use the `bca` option when computing this confidence interval.

   Using these intervals, do you think there is any evidence to suggest that there is a real difference in the population probability of having heart disease between these two individuals? **[3 marks]**

```
> print(conf_interval_65)
BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
Based on 5000 bootstrap replicates

CALL :
boot.ci(boot.out = bootstrap_65, type = "bca")

Intervals :
Level       BCa
95%   ( 0.4495,  0.9127 )
```

```
> print(conf_interval_66)
BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
Based on 5000 bootstrap replicates

CALL :
boot.ci(boot.out = bootstrap_66, type = "bca")

Intervals :
Level       BCa
95%   ( 0.2744,  0.8525 )
Calculations and Intervals on Original Scale
```

- .95% confident that the 65th patient has a chance of developing heart disease in the range of roughly (0.4495, 0.9127),
- 95% confident that the 66th patient has a chance of developing heart disease in the range of roughly (0.2744, 0.8525).
- The upper limit for the 65th patient (0.9127) is higher than that of the 66th patient (0.8525). '
- However, since the intervals do not overlap, it indicates that there might be a real difference in the population probability of having heart disease between these two individuals.

Question 3 (14 marks)

Data Smoothing

Data "smoothing" is a very common problem in data science and statistics. We are often interested in examining the unknown relationship between a dependent variable (y) and an independent variable (x), under the assumption that the dependent variable has been imperfectly measured and has been contaminated by measurement noise. The model of reality that we use is

$$y = f(x) + \varepsilon$$

where $f(x)$ is some unknown, "true", potentially non-linear function of x, and $\varepsilon \sim N(0, \sigma^2)$ is a random disturbance or error. This is called the problem of function estimation, and the process of estimating $f(x)$ from the noisy measurements y is sometimes called "smoothing the data" (even if the resulting curve is not "smooth" in a traditional sense, it is less rough than the original data).

In this question you will use the k-nearest neighbours machine learning technique to smooth data. This technique is used frequently in practice (think for example the 14-day rolling averages used to estimate coronavirus infection numbers). This question will explore its effectiveness as a smoothing tool.

Mass Spectrometry Data Smoothing

The file `ms.train.2024.csv` contains $n = 400$ measurements from a mass spectrometer. Mass spectrometry is a chemical analysis tool that provides a measure of the physical composition of a material. The outputs of a mass spectrometry reading are the intensities of various ions, indexed by their mass-to-charge ratio. The resulting spectrum usually consists of a number of relatively sharp peaks that indicate a concentration of particular ions, along with an overall background level. A standard problem is that the measurement process is generally affected by noise – that is, the sensor readings are imprecise and corrupted by measurement noise. Therefore, smoothing, or removing the noise is crucial as it allows us to get a more accurate idea of the true spectrum, as well as determine the relative quantity of the ions more accurately. However, we would also *ideally* like for our smoothing procedure to not damage the important information contained in the spectrum (i.e., the heights of the peaks).

The file `ms.train.2024.csv` contains measurements of our mass spectrometry reading. The column `ms.train.2024$MZ` are the mass-to-charge ratios of various ions, and `ms.train.2024$intensity` are the measured (noisy) intensities of these ions in our material. The file `ms.test.2024.csv` contains $n = 1,000$ different values of `MZ` along with the "true" intensity values, stored in `ms.test.2024$intensity`. These true values have been found by using several advanced statistical techniques to smooth the data, and are being used here to see how close your estimated spectrum is to the truth. For reference, the samples `ms.train.2024$intensity` and the value of the true spectrum `ms.test.2024$intensity` are plotted in Figure 1 against their respective `MZ` values. To answer this question you must:

- Provide an R script containing all the code you used to answer the questions. Please use comments to ensure that the code used to identify each question is **clearly identifiable**. Call this file `idnum.Q3.R`, where "`idnum`" is your first name followed by your family name.
- Provide appropriate written answers to the questions, along with any graphs, in your non-handwritten report document (pdf file).

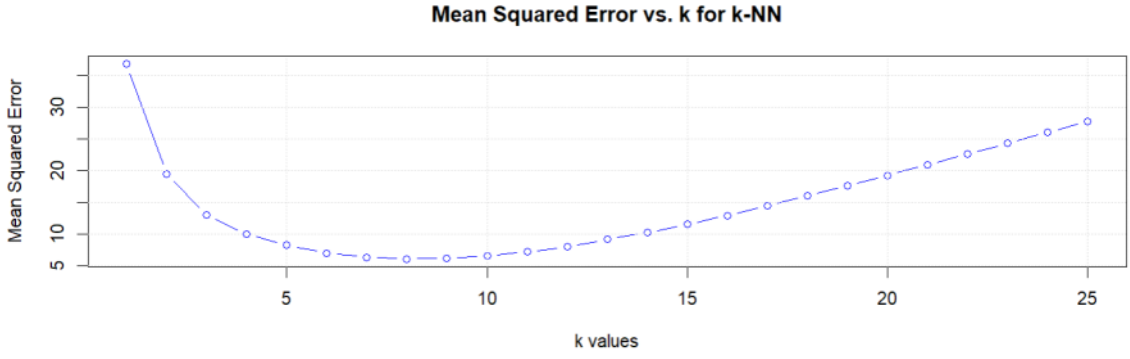
To answer this question, you must use the `kknn` and `boot` packages that we used in Studios 9 and 10. You will be using the k-nearest neighbours method (k-NN) to estimate the underlying spectrum from the training data. Use the `kknn` package we examined in Studio 9 to provide predictions for the `MZ` values in `ms.test.2024`, using `ms.train.2024` as the training data. You should use the `kernel = "optimal"` option when calling the `kknn()` function. This means that the predictions are formed by a weighted average of the k points nearest to the point we are trying to predict, the weights being determined by how far away the neighbours are from the point we are trying to predict.

Questions

1. For each value of $k = 1, \ldots, 25$, use k-NN to estimate the values of the spectrum associated with the MZ values in `ms.test.2024$MZ`. Then, compute the mean-squared error between your estimates of the spectrum, and the true values in `ms.test.2024$intensity`. Produce a plot of these errors against the various values of k. [1 mark]

Using the provided information, we need to create a matrix for the values of k ranging from 1 to 25. For each value of k, we will calculate the k-NN model and obtain the predicted intensity using the fitted function. We will then append these results to a matrix designed to store these calculations and create corresponding plots to visualize the results.

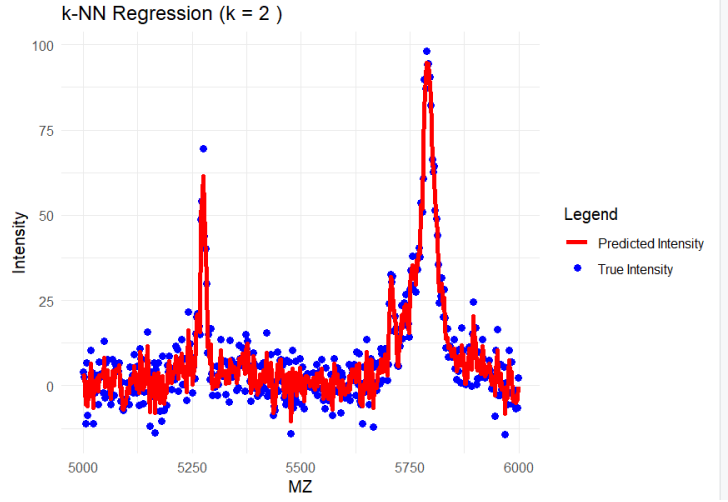
```
# Question 3.1
# Define the range of k values
k_values <- 1:25

# Initialize a vector to store MSE values
mse_values <- numeric(length(k_values))

# Loop over each k value
for (k in k_values) {
  # Create the k-NN model
  knn_model <- kknn(intensity ~ MZ, train = ms_train, test = ms_test, k = k, kernel = "optimal")

  # Get predicted intensity values
  predicted_intensity <- fitted(knn_model)

  # Calculate the mean-squared error (MSE)
  mse_values[k] <- mean((predicted_intensity - intensity_test)^2)
}

# Plot the MSE against k values
plot(k_values, mse_values, type = "b", col = "blue",
     xlab = "k values", ylab = "Mean Squared Error",
     main = "Mean Squared Error vs. k for k-NN")
grid()
```



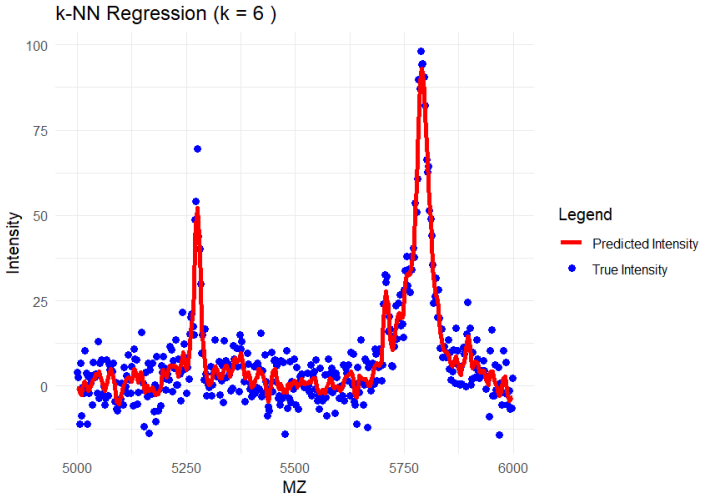
2. Produce four graphs, each one showing: (i) the training data points (`ms.train.2024$intensity`), (ii) the true spectrum (`ms.test.2024$intensity`) and (iii) the estimated spectrum (predicted `intensity` values for the `MZ` values in `ms.test.2024.csv`) produced by the k-NN method for four different values of k; do this for $k = 2$, $k = 6$, $k = 12$ and $k = 25$. Make sure the graphs have clearly labelled axis' and a clear legend. Use a different colour for your estimated curve. **[3 marks]**

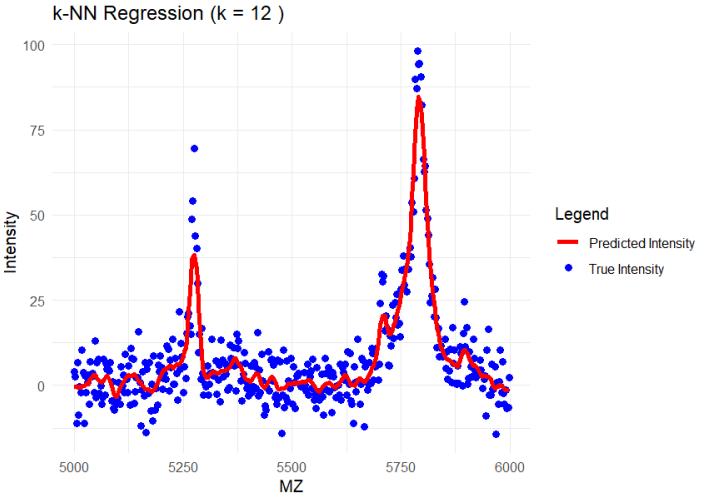
```r
# Question 3.2

# Define k values for plotting
k_values <- c(2, 6, 12, 25)

# Plot the training data points (ms.train.2024$intensity)
for (k in k_values) {
  # Perform k-NN regression
  kknn_model <- kknn(intensity ~ MZ, train = data.frame(MZ = MZ_train, intensity = intensity_train), test = data.frame(MZ = MZ

  # Extract predicted intensity
  predicted_intensity <- fitted(kknn_model)

  # Create the plot
  plot_title <- paste("k-NN Regression (k =", k, ")")

  p_1 <- ggplot() +
    geom_point(aes(x = MZ_train, y = intensity_train, color = "True Intensity"), size = 2) +  # Training data points
    geom_line(aes(x = MZ_train, y = predicted_intensity, color = "Predicted Intensity"), size = 1.5) +  # Estimated curve
    scale_color_manual(values = c("True Intensity" = "blue", "Predicted Intensity" = "red")) +  # Colors
    labs(title = plot_title, x = "MZ", y = "Intensity", color = "Legend") +  # Labels
    theme_minimal()  # Theme

  # Print the plot
  print(p_1)
```

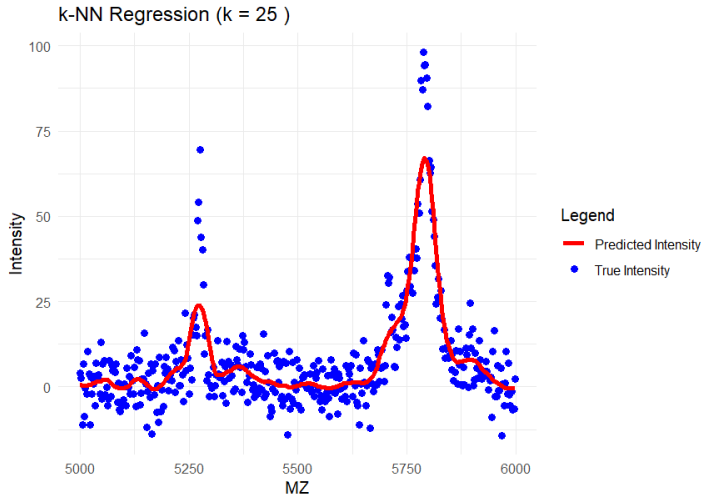
i.



3.



4. 

5. 


```
> data.frame(k = k_values, MSE = mse_k_values)
   k       MSE
1  2 19.443635
2  6  7.057205
3 12  8.055226
4 25 27.737285
```

Qualitative Analysis:
- k = 2: The model is highly sensitive to local changes, leading to overfitting. This results in a jagged spectrum that does not generalize well to new data, as indicated by the high MSE.
- k = 6: This value strikes the best balance, capturing important patterns while avoiding overfitting. The predicted spectrum closely matches the true spectrum, supported by the lowest MSE.
- k = 12: The predicted spectrum is smoother and retains most key features, but some finer details are lost, reflected in a slight increase in MSE.
- k = 25: The model is overly smooth, failing to capture variability in the true spectrum. This oversimplification results in the highest MSE, indicating a poor fit to the complexity of the data.

Quantitative Analysis:
The mean squared errors for k = 2, k = 6,k=12, and k=25 are 19.44, 7.06, 8.06, and 27.74, respectively. This indicates that k=6 provides the best fit to the true spectrum, as it has the lowest mean squared error value.

Cross validation

```
#Question 3.4
# Set a seed for reproducibility
set.seed(123)

# Define the range of k values to consider
k_range <- 1:25

# Perform k-NN regression using cross-validation
cv_model <- train.kknn(intensity ~ MZ, data = ms_train,
                       kmax = 25, kernel = "optimal", distance = 2)

# Extract the best k value selected by cross-validation
best_k_cv <- cv_model$best.parameters$k
best_k_cv
```

```
> best_k_cv
[1] 6
```

The cross-validation method has selected the same value of k=6 = as the best choice, which is consistent with the k value that minimizes the actual MSE on the true spectrum. In Question 3.1, the value of k that minimized the actual mean-squared error (MSE) on the true spectrum was also k=6, with an MSE of 7.06.This shows that cross-validation is effective in estimating the optimal k, even though the actual MSE is unknown in practice.
The alignment between the cross-validation result and the true MSE-minimizing k suggests that cross-validation is a reliable approach for choosing k in k-NN regression for this dataset.

5. Using the estimates of the curve produced in Q3.4 using the value of k selected by cross-validation, see if you can think of a way to find an estimate of the standard deviation of the sensor/measurement noise that has corrupted our intensity measurements. **[1 mark]**

```
> # Question 3.5
> # Fit the KNN model with the best k from cross-validation
> knn_model_k_best = kknn(intensity ~ MZ, train = ms_train, test = ms_tra .... [TRUNCATED]

> # Get the predicted intensity values from the model
> predicted_intensity_k_best = fitted(knn_model_k_best)

> # Calculate the residuals (differences between actual and predicted intensity)
> residuals_k_best = intensity_train - predicted_intensity_k_best

> # Estimate the standard deviation of the measurement noise
> estimated_sd = sd(residuals_k_best)

> print(estimated_sd)
[1] 5.099013
```

By using the standard deviation estimate, it resulted in 5.099 This value suggests that, on average, the recorded intensity values deviate from the true signal by about 5.099 units, reflecting some measurement uncertainty.

6. Do any of the estimated spectra plotted in Q3.2 achieve our aim of providing a smooth, low-noise estimate of background level as well as accurate estimation of the peaks? Explain why you think the k-NN method is able to achieve, or not achieve, this aim. **[2 marks]**

In summary, the k-NN method can achieve the aim of providing a smooth, low-noise estimate of the background level while accurately estimating the peaks when k=6 is chosen. The reason for this success is that this particular k value strikes a balance between capturing essential patterns in the data and minimizing noise, leading to a model that generalizes well to unseen data without overfitting or underfitting. Other k values either overfit (k = 2), underfit (k = 12 and k = 25), or provide an insufficient estimate of the peaks, indicating that careful selection of k is crucial for optimal performance in this context.

Find the max intensity when k=6 when smoothed signal

```
knn_model_k_best <- kknn(intensity ~ MZ, train = ms_train, test = ms_train,

# Get the predicted intensity values from the model
predicted_intensity_k_best <- fitted(knn_model_k_best)

# Find the maximum estimated intensity and its corresponding MZ value
max_intensity_index <- which.max(predicted_intensity_k_best)
max_intensity_value <- predicted_intensity_k_best[max_intensity_index]
max_MZ_value <- ms_test$MZ[max_intensity_index]

# Print the results
(max_intensity_index)
(max_intensity_value)
(max_MZ_value)
> (max_intensity_index)
[1] 317

> (max_intensity_value)
[1] 93.04956
```

Corresponds to the MZ Value

Thus the maximum estimated intensity is 5316.3 MZ

```
> (max_MZ_value)
[1] 5316.3
```

8. Using the bootstrap procedure (use at least 5,000 bootstrap replications), write code to find a confidence interval for the k-nearest neighbours estimate of intensity at a specific MZ value. Use this code to obtain a 95% confidence interval for the estimate of the intensity at the MZ value you determined previously in Question 3.7 (i.e., the value corresponding to the highest intensity). Compute confidence intervals using the k determined in Question 3.4 by cross-validation, as well as $k = 3$ neighbours and $k = 20$ neighbours. Report these confidence intervals. Explain why you think these confidence intervals vary in size for different values of k. **[3 marks]**

```
> # Output the results
> cat("95% Confidence Interval for k = 3:", ci_k3$bca[4:5], "\n")
95% Confidence Interval for k = 3: 3.41561 13.49554

> cat("95% Confidence Interval for k=6 :", ci_k_best$bca[4:5], "\n")
95% Confidence Interval for k=6 : 2.906518 13.11204

> cat("95% Confidence Interval for k = 20:", ci_k20$bca[4:5], "\n")
95% Confidence Interval for k = 20: 0.8053496 5.466939
```

Using the bootstrap procedure with 5,000 bootstrap replications the 95% confidence intervals for the k nearest (k-NN) estimate of intensity

k=6 (which was determined by cross validity) 95% CI [2.906518,13.11]

k=3 95% CI [23.41516, 13.49553]

k=20 95% CI [0.8053496, 5.466939]

Explanation

The variation in the size of the confidence intervals for the k-nearest neighbours (k-NN) estimates reflects the balance between bias and variance that changes with the number of neighbours k:

- k = 3: With fewer neighbours, the model becomes more sensitive to small fluctuations in the data. This leads to low bias* but high variance, as the predictions closely follow the training data and are more likely to overfit. The wider confidence interval indicates greater uncertainty due to the model's tendency to adapt to noise in the data.

- k = 6: This value of k, chosen by cross-validation, represents a trade-off between bias and variance. The model captures enough detail from the data without becoming too sensitive to local noise. The moderately sized confidence interval reflects this balance, showing that predictions are relatively stable while still maintaining accuracy.

- k = 20: With a higher number of neighbours, the model averages over more data points, leading to high bias and low variance. The estimate becomes smoother, reducing the influence of local noise. The narrower confidence interval reflects the reduced variability in predictions, though the model risks underfitting, potentially missing finer details in the data.

This variation illustrates the impact of increasing k on the trade-off between model complexity and stability, with smaller k values leading to more variability and larger k values producing more stable but potentially oversimplified predictions.