Assignment for PMIM402

Swansea University
Prifysgol Abertawe
Medical School
Ysgol Feddygaeth

Module number:	PMIM402
Module name:	Health Data Modelling
Title of assignment:	Machine Learning in Health Care
Student ID number:	445348
Word count:	
Declaration:	I understand the following conditions which apply throughout this course: 1. I confirm that I am the sole author of this work. 2. I understand that proof reading by a third party is discouraged, but if used, records should be available as per guidelines. 3. I understand the need for academic integrity and that all my submitted work will adhere to its principles. 4. I understand that the teaching team will take measures to deter, detect and report any academic misconduct. 5. I agree to my work being submitted to the TurnItIn academic database. 6. I understand the importance of assignment deadlines and the need to seek help in good time where personal circumstances interrupt my work.
Please copy and paste this declaration onto the front of the submission.	

Machine Learning in Health Care

Andy Gray
445348

Part 1: Clustering

1. Run K-means clustering on the above heart disease dataset and answer the following questions

1. **Why should the attribute "*class*" in *heart-c.csv* ("*num*") not be included for clustering?**

The class num is the intended cluster predictions, so in essence, these outputs are the labels. As K-Means is an unsupervised algorithm, these are not needed and are a way for us to check that the algorithm has plotted/predicted well.

2. **Run K-means algorithm by choosing different numbers of clusters, *numCluster* = 2, 3, 4,5, then observe the differences of clusters generated:**
    1. **How are the *Within Cluster Sum of Squared Errors*[1] changed for different numbers of clusters?**

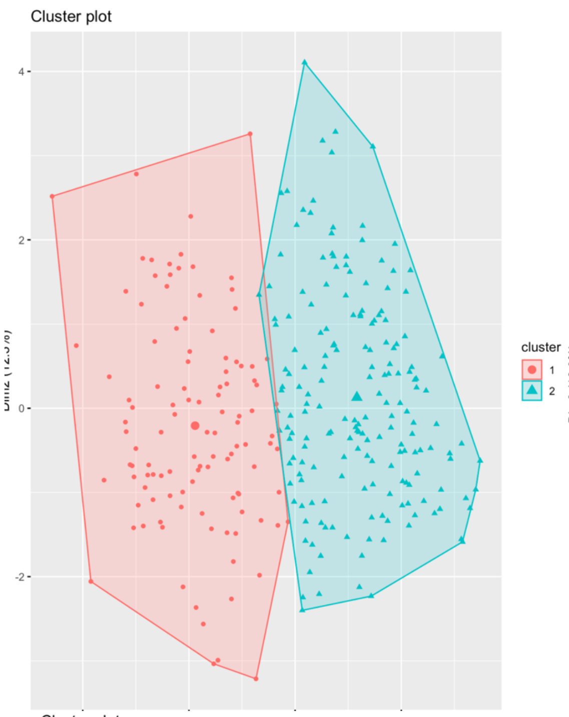
K Number	Centroid Values	between_SS/total_SS
2	1306.399, 1865.346	17.3%
3	1500.4056, 505.1190, 938.7798	23.2%
4	466.9881, 838.7374, 812.6521, 597.6057	29.2%
5	458.2797, 811.3692, 451.2931, 339.3949, 529.1168	32.5%

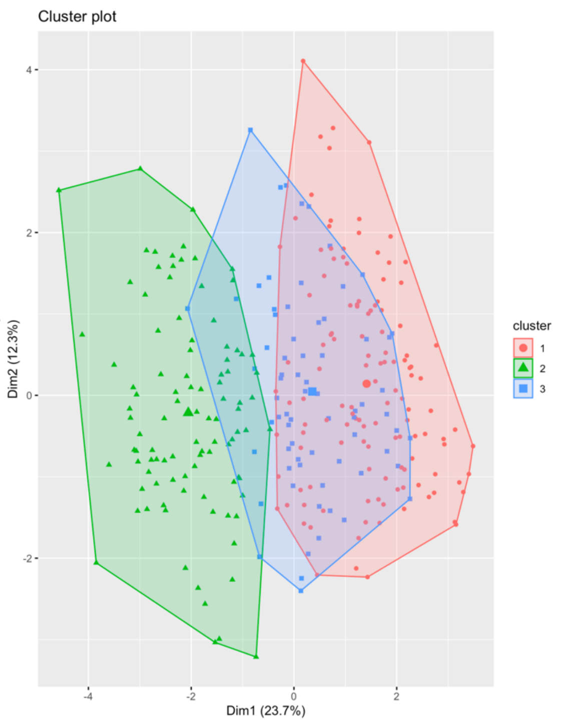
2. **What can you conclude?**

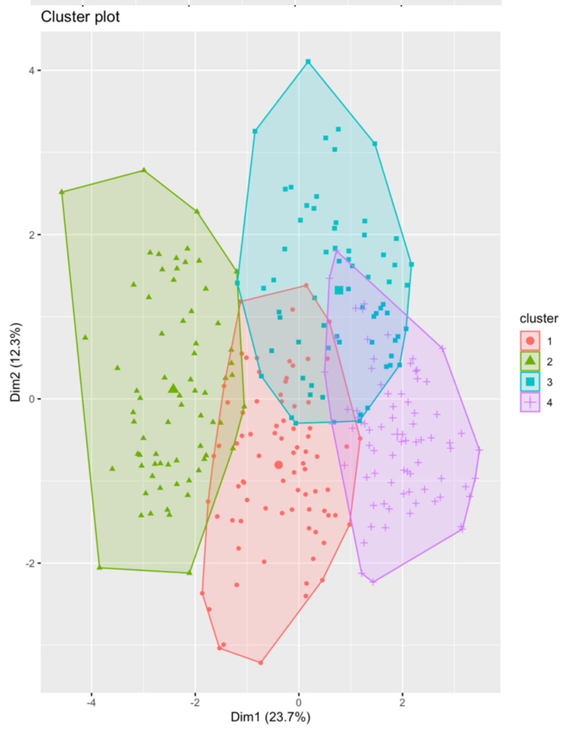
That as the cluster k number gets bigger, the distance between the data points or variance per cluster is getting bigger. Therefore, showing that the data points are pretty far apart from the centroids (centres of the clusters) shows not to be a very good k value to be using.

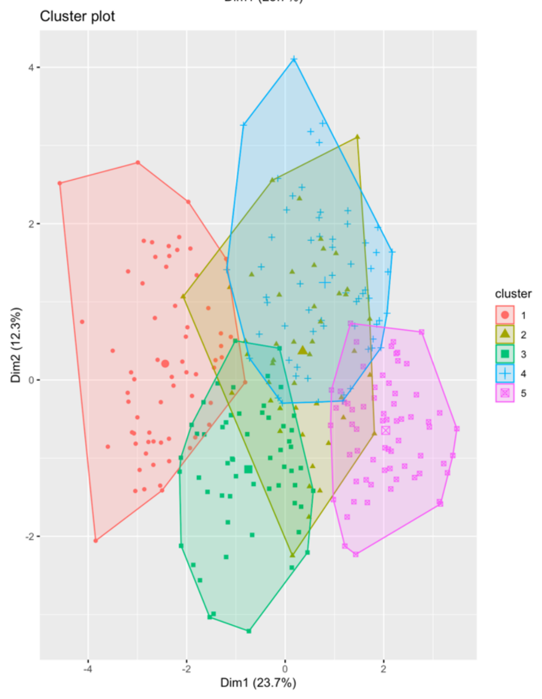
3. **How can you explain this conclusion from clustering analysis point of view?**

When we look at the data points visualisations, we can see the cluster boundaries' overlapping, which is expected due to the dataset's high dimensionality nature. However, the data does not seem to cluster well. For example, when we have 5 clusters compare to just the 2 clusters and with the within-cluster *sum of squared errors* value getting higher as the ks get bigger indicate that the k is going in the wrong way. Additionally, with the fact that we can only go to a minimum of k = 2, it shows that k = 2 is the ideal number for this dataset, which we know as the dataset has two labels within the "num" column, showing that there are two main clusters within the data.


Cluster plot
Dim2 (12.3%)
cluster
1
2
4
2
0
-2

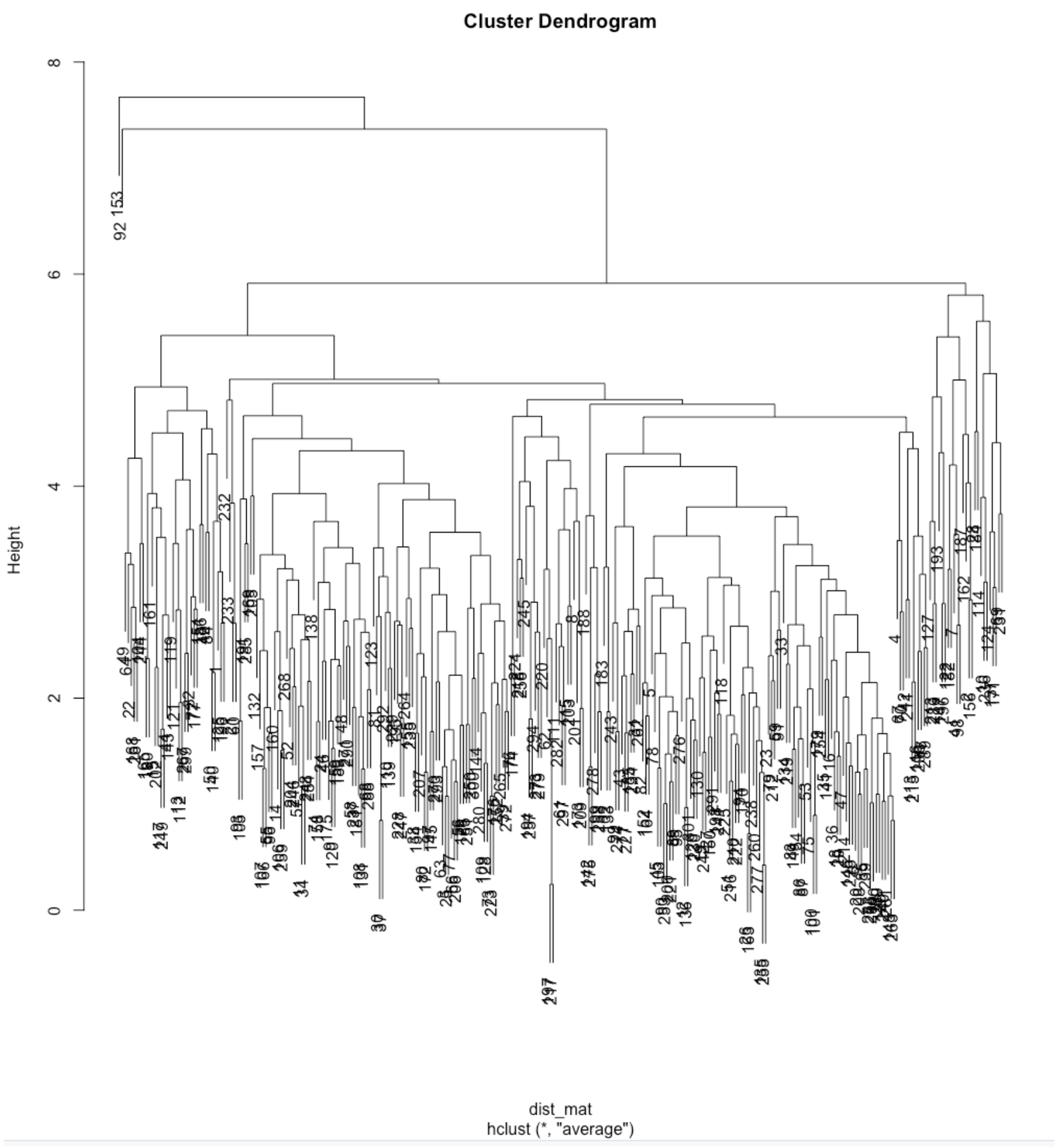

Cluster plot
Dim2 (12.3%)
Dim1 (23.7%)
cluster
1
2
3
4
2
0
-2
-4
-2
0
2


Cluster plot
Dim2 (12.3%)
Dim1 (23.7%)
cluster
1
2
3
4
4
2
0
-2
-4
-2
0
2


Cluster plot
Dim2 (12.3%)
Dim1 (23.7%)
cluster
1
2
3
4
5
4
2
0
-2
-4
-2
0
2

2. Run the hierarchical clustering on above heart disease dataset, and answer the following questions

1. Show the clustering results in tree structure



2. Describe the link method you used

The method used to configure the distribution is the Euclidean distance, while the method used for linking the hierarchical cluster is the average method. The average link method is an average of all the distance pairs. This method takes an average of the distributed Euclidean distance to create a hierarchy of the data clusters [1].

3. **What are the strengths and limitations of this link method in hierarchical clustering?**

It shows the flow and the link of the clustering well and can get visualised easily, making it easy to follow. Hierarchical clustering is also an excellent way to be able to detect outliers. The average link method also works as a compromise between the single link, a greedy method, and the complete linkage [1]. However, due to its being an average of the centroids' data values, much information can potentially be lost. Additionally, Hierarchical clustering does not scale well to large datasets; therefore, this type of model is only good with modest-sized datasets [1].

Part 2: Classification

Naïve Bayes:

1) **Did you undertake any prepossessing? If so, why?**

The data frame first had the variables X and Patient_ID removed due to these values not having any significant values. X seems to be the original column number, and Patient_ID is the auto-id number given to the patient when they first enter a database system.

The data frame then has any NA values removed, and the class variable gets converted to a factor value. The NA values get removed to ensure that all the variables had values and that any observations that had missing values would be removed entirely from the dataset. The class changed to a factor to allow the algorithms to know that the class is a category data type that is needed to classify the data predictions.

The data frame then split into a train and test set while a 75-25% split to the data. The split of the dataset got done to see how well the models would predict an unseen dataset. 75% of the data got used to train the model, while the 25% withheld was used to test the model's predictions.

2) **Run the classifier with default parameters.**
    a. **How accurately can the classifier predict those that develop heart disease? What is in the output that signifies this?**

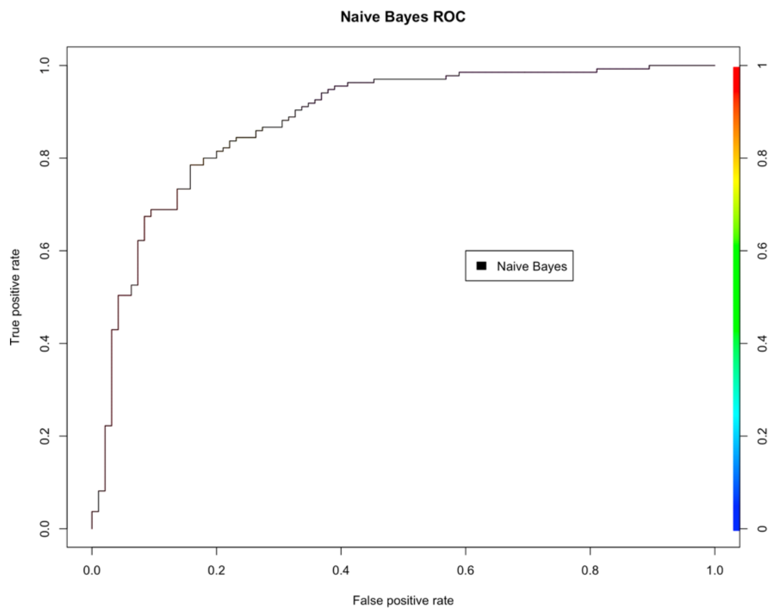
There are two ways we can determine how well this model can predict. We could use its accuracy percentage while training, which was 77.83%. We can also use its confusion matrix output to show how well it predicted correctly or incorrectly predicted.

   b. **How many people are misclassified as developing heart disease? Where is this answer found in the output?**

Twenty-one people have gotten misclassified as developing heart disease but did not have it. This output got discovered when predicting using the testing dataset, and the output was shown in the confusion matrix. A confusion matrix (CM) is a method used to see how well a model has predicted. A CM requires having labelled data and shows how the model predicts every data point into one of four categories. These four categories are True Positive – Top left, False Positive (Type 1 error) – top right, False Negative (Type 2 error) – bottom left, True Negative – bottom right [3].

```
          Reference
Prediction  0   1
         0 73  21
         1 22 114
```

3) **Plot and submit the ROC curve for the class that develops heart disease. What is another measure of accuracy commonly used? Please provide this.**



The Area Under Curve (AUC) is another accuracy predictor. The AUC figures out the area under the ROC curve. A perfect classifier would have a ROC AUC score of 1, where a classifier that is done to pure randomness would equal 0.5. Therefore, the higher the AUC score, the better the classifier. The AUC score is helpful when two ROC curves are close and hard to determine by the naked eye. The AUC for the naïve Bayes model was 0.8821053, as well as a confusion matrix can be used to determine accuracy.

Random Forest:

1) **Did you undertake any prepossessing? If so, why?**

he same dataset and train test split data were used as the naïve Bayes. So all the pre-processing was the same as before. No additional processing got done as both these models do not need any additional modifications to the dataset to work.

2) **Run the classifier with default parameters.**

a. **How accurately can the classifier predict those that develop heart disease? What is in the output that signifies this?**

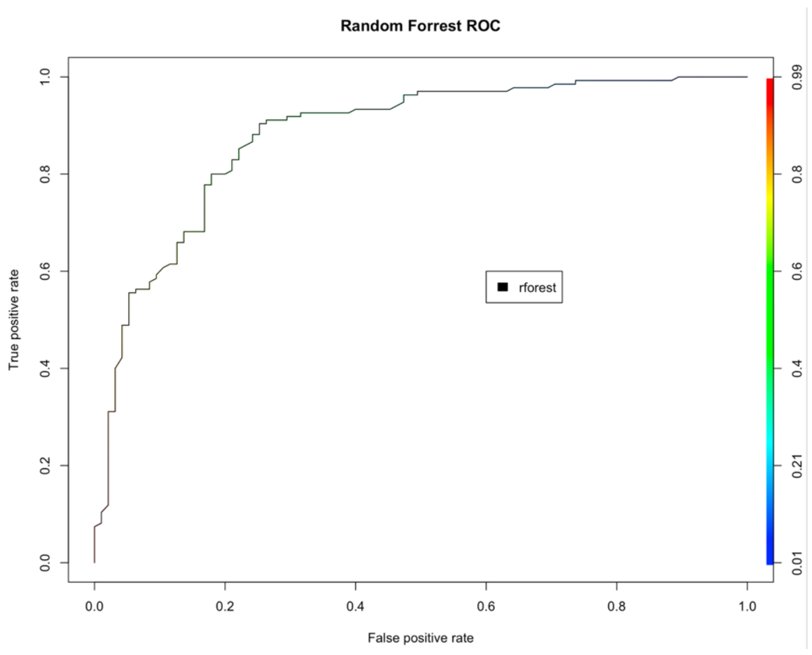
The random forest had an accuracy percentage of 84.35% with the training data.

b. **How many people are misclassified as developing heart disease? Where is this answer found in the output?**

14 people were misclassified as developing heart disease. This answer was found in the top right of the confusion matrix, or the false positive section of the matrix.

```
          Reference
Prediction   0   1
         0  82  14
         1  27 107
```

3) **Plot and submit the ROC curve for the class that develops heart disease. What is another measure of accuracy commonly used? Please provide this.**



Another metric that can be used to measure accuracy is a confusion matrix as well as AUC. A CM has been shown above and the AUC for the random forest was 0.8786355.

Random Forrest (Model) Optimised:

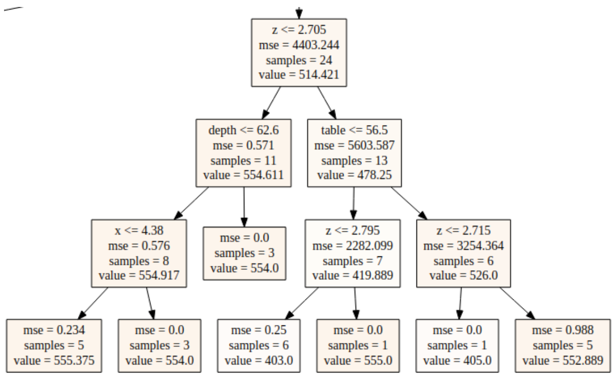
1. **Why did you choose this classifier over the other?**

The random forest performed better than the naïve Bayes model compared to on the ROC curve graph, AUC value and the accuracy percentage with training. The random forest was getting a higher true-positive rate to a lower false-positive value than the naïve Bayes model. Due to the random forest doing better than the naive Bayes, it got decided to try and improve this model by changing some of the parameters.

2. **Briefly explain how this classifier works from a theoretical point of view.**

A random forest combines multiple decision trees. The number is depending on the amount specified amount for training.

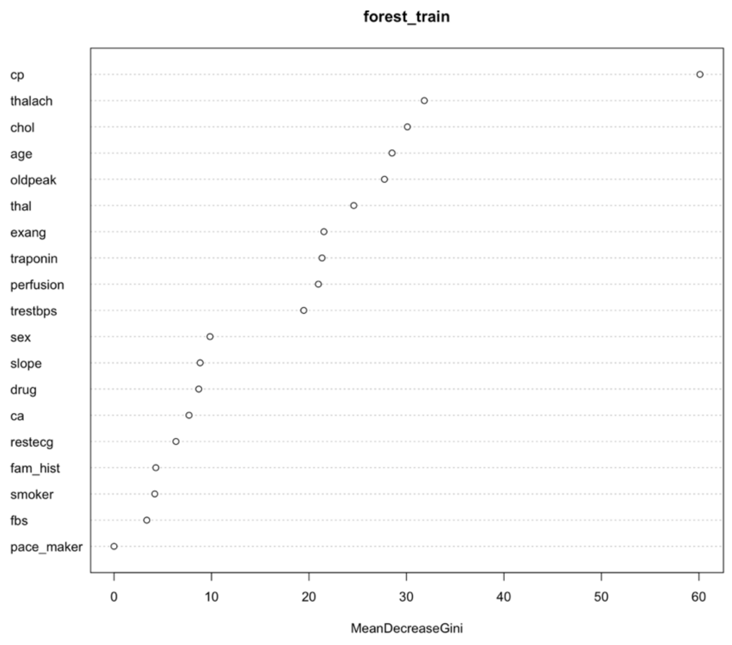
A decision tree is a versatile ML algorithm that can perform both classification and regression. Decision trees can produce multi-output tasks too. A decision tree will consist of a root node, a left child node and a right child node. If the nodes do not have any child nodes, they get referred to as the leaf node. The leaf node is where the model will provide its predicted output [2]. Apart from the leaf node, all the nodes will consist of an if/else style statement. Therefore depending on the outcome will depend on what direction down the tree the data moves (see image below).


z <= 2.705 mse = 4403.244 samples = 24 value = 514.421
depth <= 62.6 mse = 0.571 samples = 11 value = 554.611
table <= 56.5 mse = 5603.587 samples = 13 value = 478.25
x <= 4.38 mse = 0.576 samples = 8 value = 554.917
mse = 0.0 samples = 3 value = 554.0
z <= 2.795 mse = 2282.099 samples = 7 value = 419.889
z <= 2.715 mse = 3254.364 samples = 6 value = 526.0
mse = 0.234 samples = 5 value = 555.375
mse = 0.0 samples = 3 value = 554.0
mse = 0.25 samples = 6 value = 403.0
mse = 0.0 samples = 1 value = 555.0
mse = 0.0 samples = 1 value = 405.0
mse = 0.988 samples = 5 value = 552.889
[4]

The random forest then trains each of the decision trees on a slightly different set of observations, splitting nodes in each tree considering a limited number of the features. The random forest's final prediction is made by averaging the predictions of each of the individual trees [4].

3. **Try to optimize the classifier to achieve a higher accuracy (no matter how small) than first found. Remember that we have a particular focus on predicting those that develop heart disease.**
    a. **Were there any features that could be removed? Please print the output that helped you make this decision.**

The value pacemaker overall had no importance on the predicting factor of the output. Therefore, this feature could be dropped with not having any impact on the model's performance.



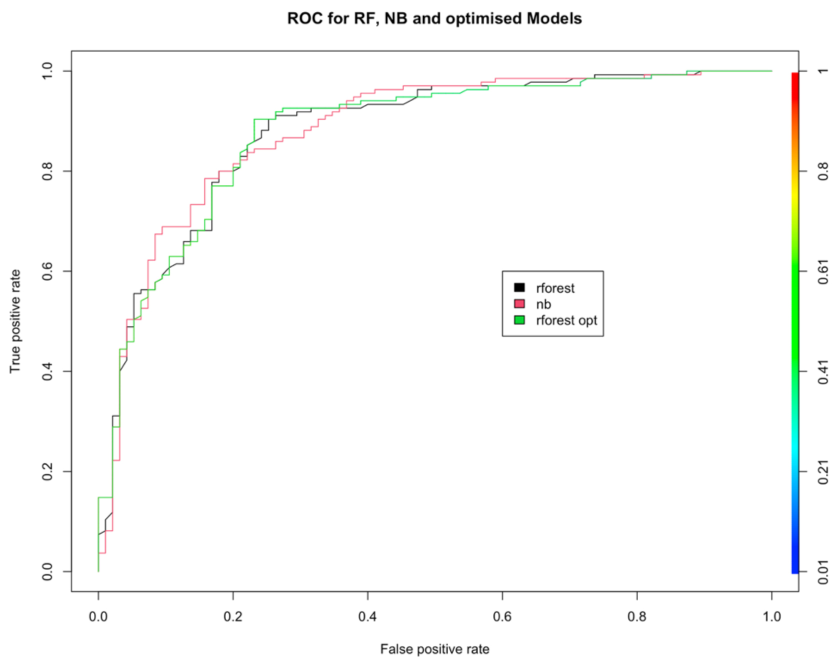
b. **Did changing the way data is sampled during training/testing affect the accuracy?**

Yes, the data can have an impact on the overall results. For example, the Random Forest produced an accuracy percentage of 82.17%. However, when the dataset's train and test data were reshuffled and reallocated, this created 84.35%, which is not a considerable difference of significance but a difference none the less. Even how to data gets split into a training and testing set can have a bit impact, as the split is random, it will put different values into different sets. These different values can have a significant impact. When training and testing a model, just from the accuracy percentage, as big as a 5% difference has been seen between the models, based on different splits between the data.

c. **What about some of the internal parameters specific to the classifier? Please explain how one of these parameters can affect accuracy.**

Changing the parameters would have either a negative or positive effect on the model's predictions, as these parameters change how the algorithms will work. For example, changing the number of ntree will change how many trees to grow in the forest, with the default being five trees.

In this case, when the random forest's parameters ntree, mtry and max nodes got changed to 800, 4 and 24, this created a slightly higher accuracy. The original random forest had an accuracy of 84.35%, an error rate of 16.96% and an AUC value of 0.8786355. In comparison, the optimised model had an accuracy of 84.78%, an error rate of 15.94% and an AUC value of 0.8781676.



4. **In general, a classifier is only as good as the data it is trained on. Please comment on what is needed from training data to train a good classifier. How can utilizing classifiers help feed back into healthcare settings with regards to data collection?**

Fundamentally, ML models rely on data to be able to train them. Therefore, to get good model prediction accuracy, the data used to be of high quality. Ultimately, no feature is more critical to ML than quality training data as this data has a massive effect on the model's development. If the data is not of high quality, then compromises might have to be made to make the data suitable to be used [5].

Having a big, clean dataset containing as much information as possible, with classification labels, will allow the ML models to search through the data to look for potential unseen patterners, which might unlock links between different conditions that have been unknown to medical staff before. However, this can only get done if the data getting collected is consistent and good.

References

[1] Bruce, P., Bruce, A., and Gedeck, P. *Practical Statistics for Data Scientists: 50+ Essential Concepts Using R and Python.* O'Reilly Media, 2020.

[2] Géron, A. *Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow: Concepts, tools, and techniques to build intelligent systems.* O'Reilly Media, 2019.

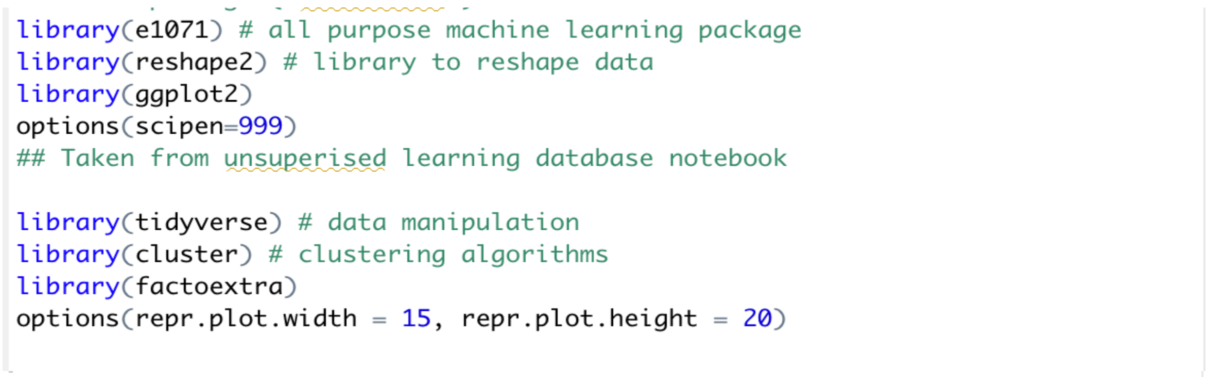
[3] Grus, J. *Data science from scratch: first principles with python.* O'Reilly Media, 2019.

[4] Koehrsen, W. An implementation and explanation of the random forest in python, 2018. Towards Data Science Medium, Online: https://towardsdatascience.com/an-implementation-and-explanation-of-the-random-forest-in-python-77bf308a9b76.

[5] Lander, J. P., and Beigelmacher, M. The essential guide to quality training data for machine learning, 2018. Cloud Factory, Online: https://www.cloudfactory.com/training-data-guide.

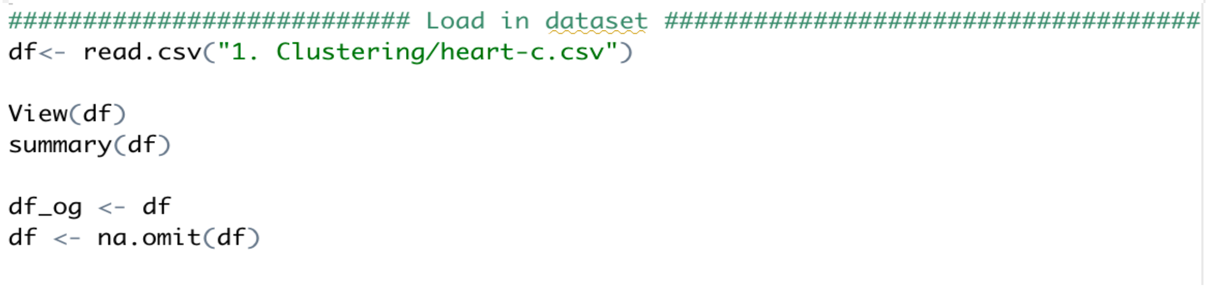
Appendix

Clustering Code



```
library(e1071) # all purpose machine learning package
library(reshape2) # library to reshape data
library(ggplot2)
options(scipen=999)
## Taken from unsuperised learning database notebook

library(tidyverse) # data manipulation
library(cluster) # clustering algorithms
library(factoextra)
options(repr.plot.width = 15, repr.plot.height = 20)
```



```
######################### Load in dataset ##################################
df<- read.csv("1. Clustering/heart-c.csv")

View(df)
summary(df)

df_og <- df
df <- na.omit(df)
```

```r
################# Converting Data to Numerical Values ##########################
for (i in 1:nrow(df['sex'])){
  if (df['sex'][i,1] == 'male') {
    df['sex'][i,1] = as.numeric(1)
  }
  else if (df['sex'][i,1] == 'female') {
    df['sex'][i,1] = as.numeric(0)
  }
}

for (i in 1:nrow(df['cp'])){
  if (df['cp'][i,1] == 'typ_angina') {
    df['cp'][i,1] = as.numeric(1)
  }
  else if (df['cp'][i,1] == 'atyp_angina') {
    df['cp'][i,1] = as.numeric(2)
  }
  else if (df['cp'][i,1] == 'non_anginal') {
    df['cp'][i,1] = as.numeric(3)
  }
  else if (df['cp'][i,1] == 'asympt') {
    df['cp'][i,1] = as.numeric(4)
  }
}


for (i in 1:nrow(df['fbs'])){
  if (df['fbs'][i,1] == 't') {
    df['fbs'][i,1] = as.numeric(1)
  }
  else if (df['fbs'][i,1] == 'f') {
    df['fbs'][i,1] = as.numeric(0)
  }
}

for (i in 1:nrow(df['restecg'])){
  if (df['restecg'][i,1] == 'normal') {
    df['restecg'][i,1] = as.numeric(0)
  }
  else if (df['restecg'][i,1] == 'st_t_wave_abnormality') {
    df['restecg'][i,1] = as.numeric(1)
  }
  else if (df['restecg'][i,1] == 'left_vent_hyper') {
    df['restecg'][i,1] = as.numeric(2)
  }
}
```

```r
for (i in 1:nrow(df['exang'])){
  if (df['exang'][i,1] == 'yes') {
    df['exang'][i,1] = as.numeric(1)
  }
  else if (df['exang'][i,1] == 'no') {
    df['exang'][i,1] = as.numeric(0)
  }
}

for (i in 1:nrow(df['slope'])){
  if (df['slope'][i,1] == 'up') {
    df['slope'][i,1] = as.numeric(1)
  }
  else if (df['slope'][i,1] == 'flat') {
    df['slope'][i,1] = as.numeric(2)
  }
  else if (df['slope'][i,1] == 'down') {
    df['slope'][i,1] = as.numeric(3)
  }
}
```

```r
for (i in 1:nrow(df['thal'])){
  if (df['thal'][i,1] == 'normal') {
    df['thal'][i,1] = as.numeric(3)
  }
  else if (df['thal'][i,1] == 'fixed_defect') {
    df['thal'][i,1] = as.numeric(6)
  }
  else if (df['thal'][i,1] == 'reversable_defect') {
    df['thal'][i,1] = as.numeric(7)
  }
  else if (df['thal'][i,1] == 'NA') {
    df['thal'][i,1] = as.numeric(0)
  }
}
```

```r
################ Removing non-required data features ##########################
df <- within(df, rm(num, X))

#################### Casting data to numeric type ############################
df$age <- as.numeric(df$age)
df$sex <- as.numeric(df$sex)
df$cp <- as.numeric(df$cp)
df$trestbps <- as.numeric(df$trestbps)
df$chol <- as.numeric(df$chol)
df$fbs <- as.numeric(df$fbs)
df$restecg <- as.numeric(df$restecg)
df$thalach <- as.numeric(df$thalach)
df$exang <- as.numeric(df$exang)
df$oldpeak <- as.numeric(df$oldpeak)
df$slope <- as.numeric(df$slope)
df$ca <- as.numeric(df$ca)
df$thal <- as.numeric(df$thal)
```

```
######################## Normalising the data ##################################
df_norm <- as.data.frame(scale(df))
head(df_norm)

######################### K-Means Clustering ####################################
df_K <- kmeans(df_norm, centers = 2)
df_K
fviz_cluster(df_K, geom = "point", data = df)

df_K3 <- kmeans(df_norm, centers = 3)
df_K3
fviz_cluster(df_K3, geom = "point", data = df)

df_K4 <- kmeans(df_norm, centers = 4)
df_K4
fviz_cluster(df_K4, geom = "point", data = df)

df_K5 <- kmeans(df_norm, centers = 5)
df_K5
fviz_cluster(df_K5, geom = "point", data = df)


df_K
df_K3
df_K4
df_K5

######################## Hierarchical Clustering ###############################
dist_mat <- dist(df_norm, method = 'euclidean')
hclust_avg <- hclust(dist_mat, method = 'average')
plot(hclust_avg)
```

Classification Code

```
############################ Load in dataset #####################################
df<- read.csv("2. Classification/heart_disease_modified.csv")
View(df)
summary(df)

################# Removing non-required data features ##########################
df <- within(df, rm(X, Patient_ID))
df <- na.omit(df)
df$class <- as.factor(df$class)

print(df_labels)
print(df)

############### Split data into training and testin ############################
bound <- floor((nrow(df)/4)*3)   # 75-25% training testing split
df <- df[sample(nrow(df)),]          # shuffles the data
df_train <- df[1:bound, ]              # get training set
df_test <- df[(bound+1):nrow(df), ]    # get test set
```

```r
####################### Naive Bayes Model ###################################
nb_model <- naiveBayes(class ~ ., data = df_train)
print(nb_model)

# predicting and plotting simple CM
nb_pred <- predict(nb_model, newdata = df_test)
table(predicted = nb_pred, observed = df_test$class)

print(nb_pred)

# ROC
test.nb = predict(nb_model, type = "raw", newdata = df_test)
nbpred = prediction(test_nb[,2], df_test$class)
nbperf = performance(nbpred, "tpr", "fpr")
plot(nbperf, main="Naive Bayes ROC", colorize=T)#
plot(nbperf, col=1, add=TRUE)
legend(0.6, 0.6, c('Naive Bayes'), 1:3)

# More detailed CM
confusionMatrix(nb_pred, df_test$class)

##### AUC values
auc_nb <- performance(nbpred, measure = "auc")
auc_nb <- auc_nb@y.values[[1]]

print(auc_nb)

##################### Random Forrest ########################################
forest_train <- randomForest(class ~ ., data = df_train)
print(forest_train)
plot(forest_train)

# Prediction and simple CM
testforest = predict(forest_train, newdata=df_test)
table(testforest, df_test$class)

# ROC
test.forest = predict(forest_train, type = "prob", newdata = df_test)
forestpred = prediction(test.forest[,2], df_test$class)
forestperf = performance(forestpred, "tpr", "fpr")
plot(forestperf, main="Random Forrest ROC", colorize=T)#
plot(forestperf, col=1, add=TRUE)
legend(0.6, 0.6, c('rforest'), 1:3)

# Feature Importance
varImpPlot(forest_train)

# More detailed CM
confusionMatrix(testforest, df_test$class)

# AUC
auc_rf <- performance(forestpred, measure = "auc")
auc_rf <- auc_rf@y.values[[1]]
print(auc_rf)
```

```r
################ Optimizing A selected Model ###################################
forest_train2 <- randomForest(class ~ ., data = df_train,
                              ntree = 800, mtry = 4,
                              maxnodes=24)

print(forest_train2)
plot(forest_train2)

# Predict and simple CM
testforest2 = predict(forest_train2, newdata=df_test)
table(testforest2, df_test$class)

# ROC compare against 3 Models
test.forest2 = predict(forest_train2, type = "prob", newdata = df_test)
forestpred2 = prediction(test.forest2[,2], df_test$class)
forestperf2 = performance(forestpred2, "tpr", "fpr")
plot(forestperf2, main="ROC for RF, NB and optimised Models", colorize=T)
plot(forestperf, col=1, add=TRUE)
plot(nbperf, col=2, add=TRUE)
plot(forestperf2, col=3, add=TRUE)
legend(0.6, 0.6, c('rforest', 'nb', 'rforest opt'), 1:3)

# AUC
auc_rf2 <- performance(forestpred2, measure = "auc")
auc_rf2 <- auc_rf2@y.values[[1]]
print(auc_rf2)
```

```r
#################### Comparing Models ##########################################
# Confusion Matrix
confusionMatrix(nb_pred, df_test$class)
confusionMatrix(testforest, df_test$class)
confusionMatrix(testforest2, df_test$class)

# AUC
auc_nb <- performance(nbpred, measure = "auc")
auc_nb <- auc_nb@y.values[[1]]
auc_rf <- performance(forestpred, measure = "auc")
auc_rf <- auc_rf@y.values[[1]]
auc_rf2 <- performance(forestpred2, measure = "auc")
auc_rf2 <- auc_rf2@y.values[[1]]
print(auc_nb)
print(auc_rf)
print(auc_rf2)

######################## EOF ###################################################
```