2018

Enterprise Database Technologies

CA1

CONOR GRIFFIN X00111602 LEE HEALY X00120179

INSTITUTE OF TECHNOLOGY TALLAGHT

Contents

Part 1: Getting to know the Dataset using R.

1. Information on each predictor variable

a. Attribute Type

Age: numeric Sex: binary Cp: nominal Trestbps: numeric Cholesterol: numeric

Fasting blood sugar < 120: binary Restecg: nominal Diastpexerc: numeric

Thalach: numeric Exang: binary Oldpeak: numeric Slope: nominal Ca: ordinal

Thal: nominal Class: binary

b. % of missing data

Age: 0 Sex: 0 Cp: 0 Trestbps: 0 Cholesterol: 0.64%

Fasting blood sugar < 120: 0 Restecg: 0.97% Diastpexerc: 0

Thalach: 0 Exang: 0 Oldpeak: 0 Slope: 0

Ca: 0 Thal: 0 Class: 0.64%

c. Max

Age: 87 Sex: Male Cp: NoTang Trestbps: 200 Cholesterol: 564

Fasting blood sugar < 120: 1 Restecg: Normal Diastpexerc: 128

Thalach: 202 Exang: 1 Oldpeak: 6.2 Slope: Up

Ca: 3 Thal: Rev Class: Sick

c. Min

Age: 18 Sex: Female Cp: Asymptomatic Trestbps: 94 Cholesterol: 126

Fasting blood sugar < 120: 0 Restecg: Abnormal Diastpexerc: 60.16

Thalach: 71 Exang: 0 Oldpeak: 0 Slope: Down

Ca: 0 Thal: Fix Class: Healthy

c. Mean

Age: 54.23701 Sex: NA Cp: NA Trestbps: 131.4318 Cholesterol: 246.2712

Fasting blood sugar < 120: 0.1461039 Restecg: NA Diastpexerc: 84.81338

Thalach: 149.7208 Exang: 0.324675 Oldpeak: 1.049351 Slope: NA

Ca: 0.6688312 Thal: NA Class: NA

c. Mode

Age: 58 Sex: Male Cp: NoTang Trestbps: 120 Cholesterol: 204

Fasting blood sugar < 120: FALSE Restecg: Normal Diastpexerc: 76.8

Thalach: 162 Exang: FALSE Oldpeak: 0 Slope: Up

Ca: 0 Thal: Normal Class: Healthy

c. Median

Age: 55 Sex: NA Cp: NA Trestbps: 130 Cholesterol: 240

Fasting blood sugar < 120: 0 Restecg: Normal Diastpexerc: 83.2

Thalach: 153 Exang: 0 Oldpeak: 0.8 Slope: NA

Ca: 0 Thal: NA Class: NA

c. Standard Deviation

Age: 9.775942 Sex: NA Cp: NA Trestbps: 17.49002 Cholesterol: NA

Fasting blood sugar < 120: 0.3537851 Restecg: NA Diastpexerc: 10.71816

Thalach: 22.76805 Exang: 0.469015 Oldpeak: 1.166037 Slope: NA

Ca: 0.9312189 Thal: NA Class: NA

d. The type of distribution (e.g normal)

Age: p-value = 0.0007914 -> Deviates from Normality

Trestbps: p-value = 8.431e-07 -> Normal Distribution

Cholesterol: p-value = 4.548e-09 -> Normal Distribution

Diastpexerc: p-value = 3.104e-07 -> Normal Distribution

Thalach: p-value = 5.608e-05 -> Normal Distribution

Oldpeak: p-value < 2.2e-16 -> Deviates From Normal Distribution

Ca: p-value < 2.2e-16 -> Deviates From Normality

e. Is the numeric data skewed? Type

Age: -0.1612079, Negatively

Trestbps: 0.7219654, Positively

Cholesterol: 1.13237, Positively

Diastpexerc: 0.7654076, Positively

Thalach: -0.5391399, Negatively

Oldpeak: 1.229654, Positively

Ca: 1.186547, Positively

f. The level of correlation (Highest)

Age: 0.2385335 diastbpexerc

Trestbps: 0.9505095 diastbpexerc

Cholesterol: NA

Diastpexerc: 0.9505095 trestbps

Thalach: -0.04608416 trestbps

Oldpeak: 0.2164572 age

Ca: 0.3120721 age

Overall findings/observations

After completing the data we found that 3 variables had missing data. Cholesterol was missing 0.64% of its data, restecg was missing 0.97% of its data and class was missing 0.64% of its data. The small amount of missing data in Cholesterol and Resting ECG really surprised us as these two variables greatly contribute to a patients wellbeing and help determine whether a patient is sick or not.

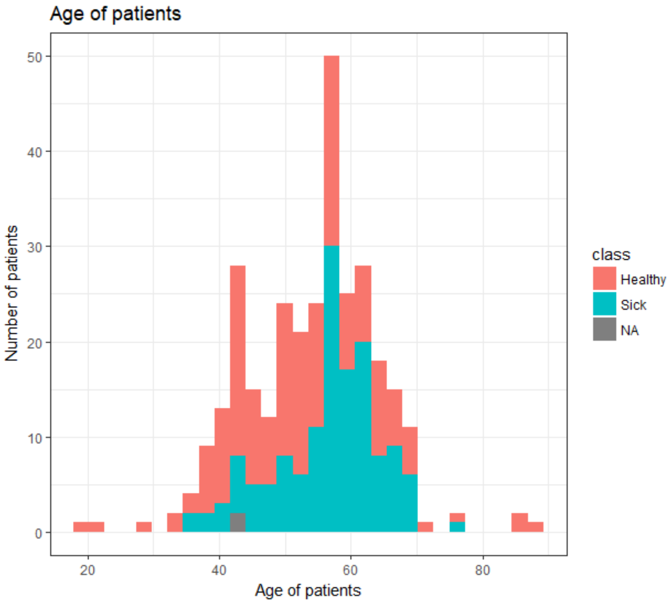
We also found that the mode for variable sex was Male. This surprised us also as we thought that the data should have been well balanced for both sexes to get a better understanding of sickness, instead this means that results gathered from the dataset will relate more to males than females and this doesn't give an accurate balanced reading.

We also established that the mode for the variable class was healthy. This meaning most of the patients in the dataset are healthy. In backing up this discovery we found that the majority (mode) of patients had normal Thalassemia instead of fixed or reversible, which is a blood disorder. This justifies our discovery of the majority of patients being healthy. In contradiction to the discovery we found that a cholesterol reading of 204 came up the most, we then done some research and found that any reading below 200 is desirable while any reading above 200 is borderline high. This tells us that if a reading of 204 comes up the most, then most people have borderline high cholesterol which isn't so healthy.

In regards to the skewness of numeric data we found that the majority of the variables were positively skewed. Which meant that the graph for these variables would peak on the left side instead of the right side which would be negatively skewed.

2. Construct a histogram for each numerical variable, with an overlay of the target variable.

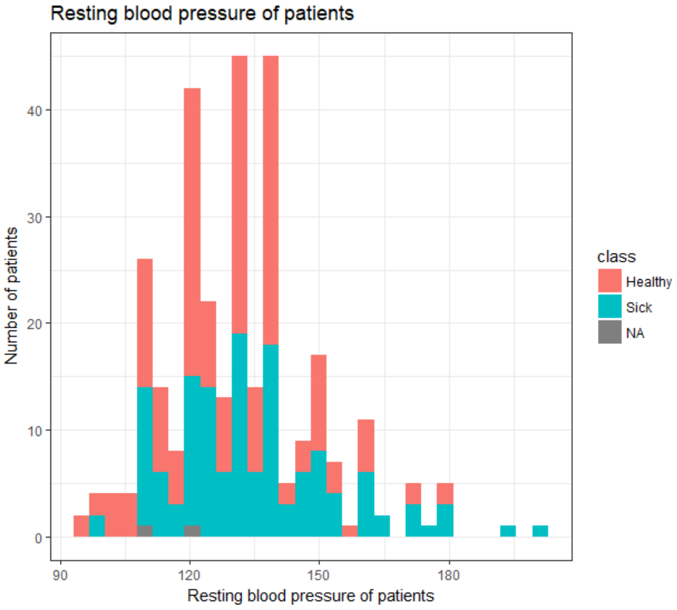
Age Histogram:



We could see from this histogram that the majority of sick patients are aged 35 to 70 years old. We could also deduct from the graph that thirty out of the fifty 55-57 year olds were sick. You can see from the histogram that the older the patients are (the further you go down the right of the X axis), the more sick the patients are. Also, if there were more older patients in the range 70-80 you could say from looking at the overlay of the class variable that they would also be sick. There seems to be more younger patients that are healthy than there are older patients.

From reading the age attribute histogram and also looking at the skewness of the attribute you could say that it would make an appearance in a machine learning algorithm. Due to the negative skewness and the occurrence of outliers using either IQR or (question 4) you could assume that the variable would make a significant appearance in a machine learning classification model. Using logistic regression where p is the probability of the outcome being a sick or healthy patient I think age would definitely be a variable to use in this type of scenario, using it over class would output the probability of patient being sick or healthy.

Trestbps Histogram:



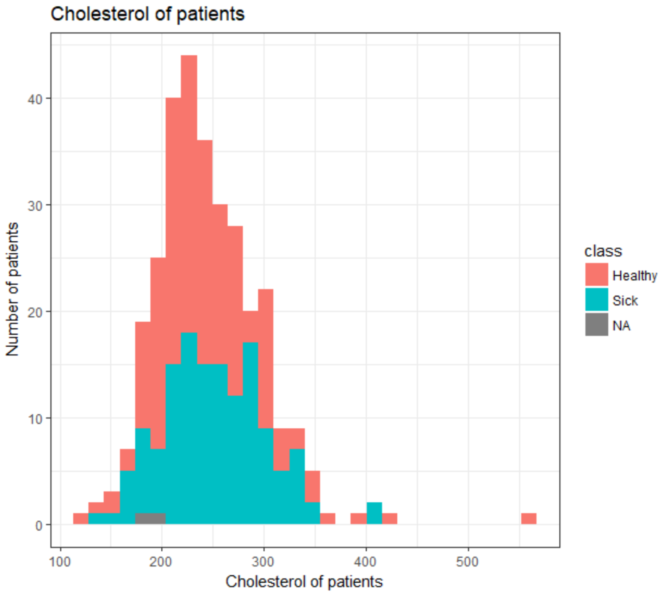
We can see that the majority of sick patients have a resting blood pressure of approximately 100-155. That being said it is clear that there are much more healthy patients within the same resting blood pressure bracket than there is sick patients. Nearly all patients with an RBP of over 160 are sick.

In terms of making an appearance in a machine learning classification model, the attributes distribution is normal distribution and positively skewed, skewness in a machine learning model is not ideal in regression modelling. Also the range of values in the data are close with min = 94 max = 200, putting the data into a logistic regression test it would be hard to say whether they would make a significant appearance in a machine learning model.

Cholesterol Histogram:

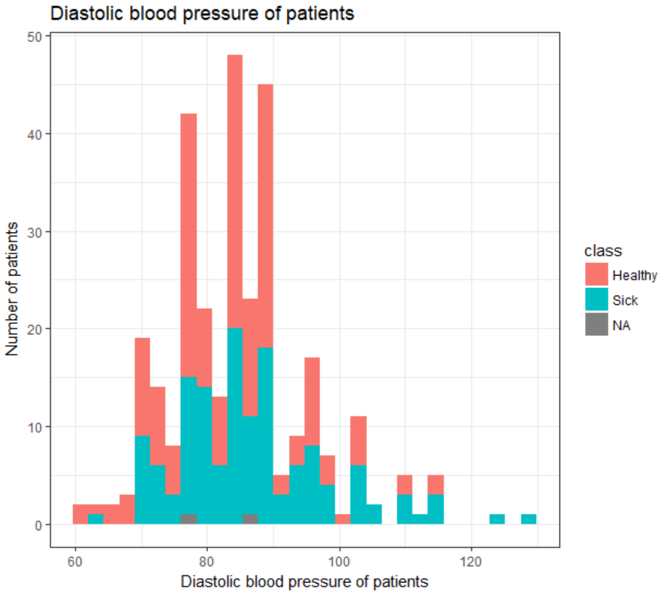
In this histogram the healthy patients greatly outweigh the sick patients. There are no more than 20 sick patients per each cholesterol reading, whereas the majority of the cholesterol readings have well over 20 healthy patients. The bulk of sick patients are in the 160-340 cholesterol reading range.

Because of the skewness and the look of normal distribution between the cholesterol of patients with the target class attribute I would say that this variable would make a significant appearance in a machine learning model. Although with some of the data missing in cholesterol it would be hard to perform to perform logistic regression testing on the attribute and to get the outcome if a patient is sick or healthy.



Diastbpexerc Histogram:

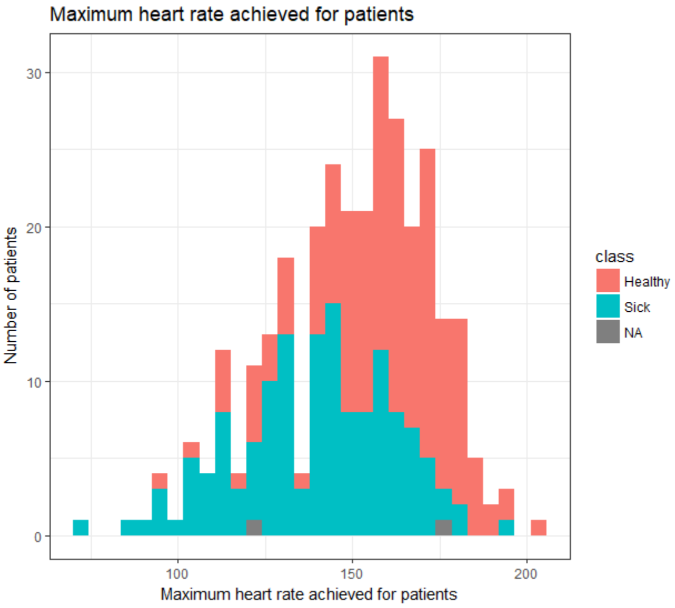
After viewing this histogram we established that it was extremely similar to the trestbps (Resting blood pressure) histogram. The attribute information tells us that trestbps is resting blood pressure and the diastbpexerc is also pressure in the arteries when the heart rests between beats. This would explain why the two graphs are so similar with a similar number of sick and healthy patients.



Most doctors define blood pressure too low if its below 90, causes can be strokes, heart attacks, and kidney failure, we would class anybody with these symptoms as 'Sick.' Looking at the histogram you can see that most of the sick patients are <= 90, taking this conclusion and analysis from the graph you could say that the Diastbpexerc attribute could be used in a machine learning classification model as the info the graph is giving us allows us to conclude it would give us an accurate probability of a patient being sick or healthy.

Thalach Histogram:

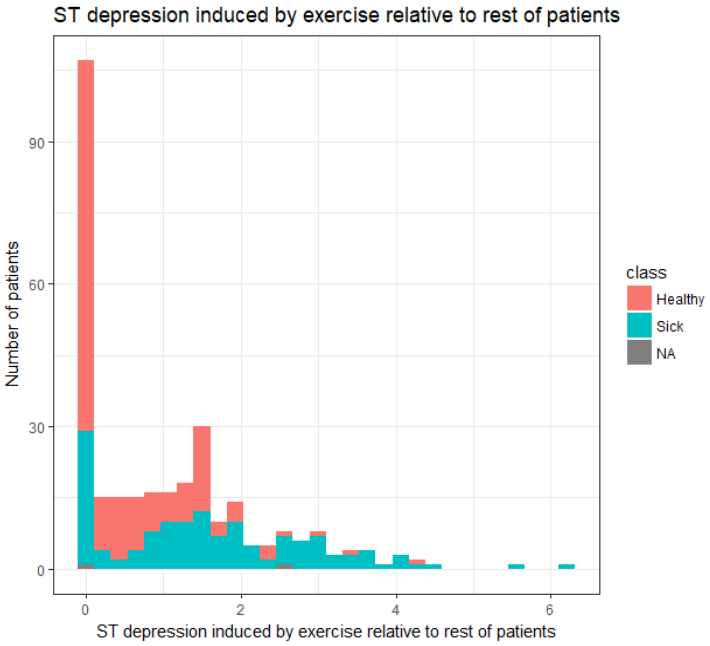
This histogram was the clearest example of a negatively distributed histogram. The majority of healthy patients had a high max heart rate while the majority of sick patients had a lower max heart rate. This might be because the healthier patients were more active and fit than the sick patients. This is a clear indication that keeping active and exercising means keeping healthy.



This attribute could be used in a machine learning model, taking the fact that a lower heart rate more or less means patient is sick and high heart rate means healthy you would expect this variable to make a significant appearance in a machine learning classification model.

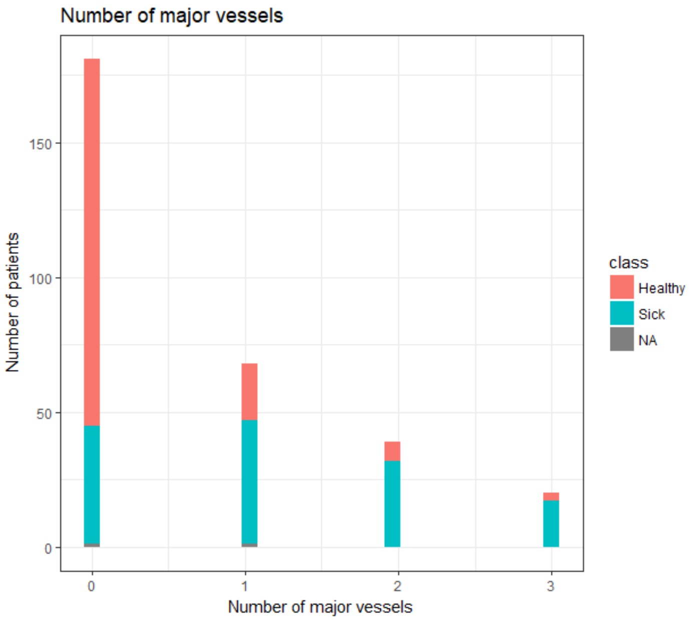
Oldpeak Histogram:

In this histogram the majority of patients are healthy with a large spike at 0 telling us that a lot of patients had 0 ST depression findings on the electrocardiogram. 30 of these patients however are still sick. Most patients with at least 2 ST depression readings where still healthy while most patients with over 3 readings where sick. This is a clear indication that the higher the ST depression finding then the more likely you are to be sick.



Ca Histogram:

This histogram shows us that if you have 1 or more major vessels you are nearly most certainly sick. If you have 0 then you are more than likely healthy with more than half of the patients with 0 readings being healthy. The readings for 1, 2 and 3 major vessels are nearly full of sick patients. Approx. 18 out of 20 patients with 3 major vessels are sick.

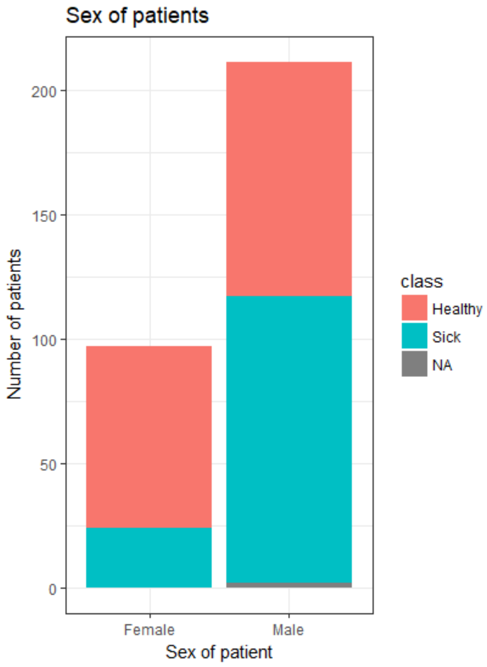


3. Construct a bar chart for each categorical variable, with an overlay of the target variable.

Sex Bar Chart:

This bar chart showed us that approximately more than half of males were sick. This was not the same for females as approximately only one quarter of females were sick. We found from this bar chart that there were more sick males than there were females altogether in the data set.

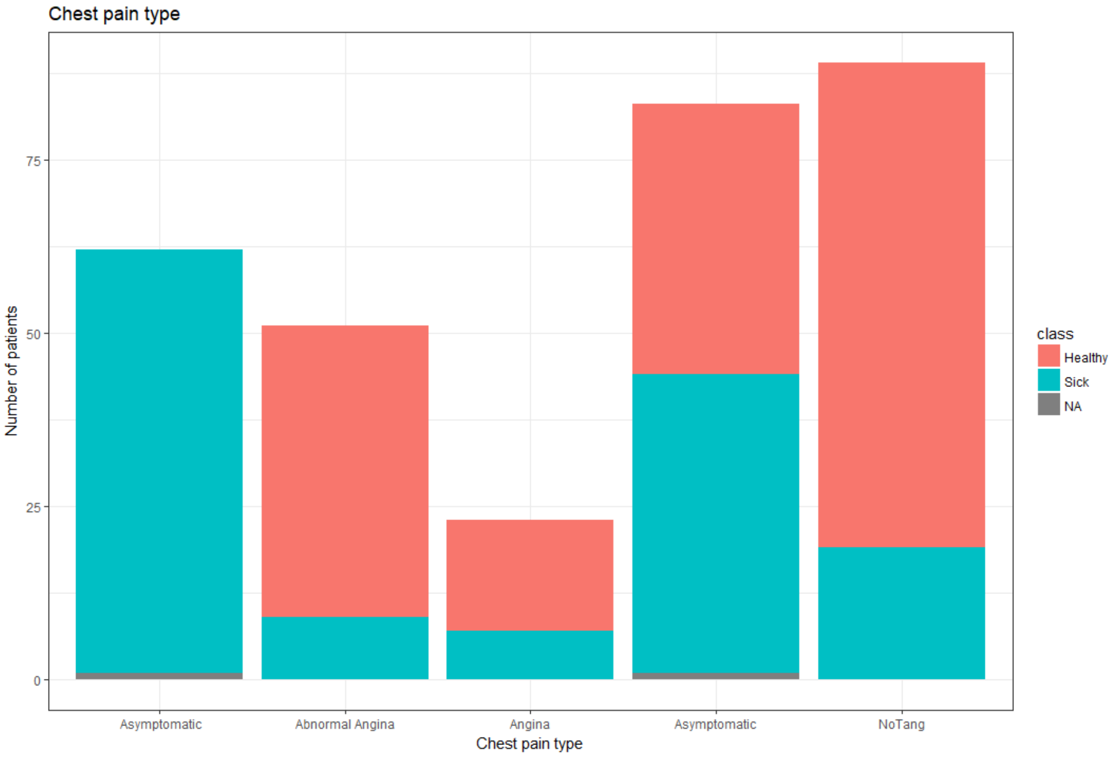
You would expect to see the Sex attribute used in a machine learning algorithm because it defines whether a patient is male or female, you can see from the histogram that there is an even correlation between the male sick and healthy patients which is useful when choosing attributes to use in a machine learning model.



Chest Pain Bar Chart:

From this bar chart we could see that all patients with Asymptomatic chest pains were sick. This means Asymptomatic is the most deadly chest pain of all. NoTang is the healthiest chest pain to have when it comes to scale as the most patients have this chest pain and it holds the healthiest amount of patients regarding chest pain.

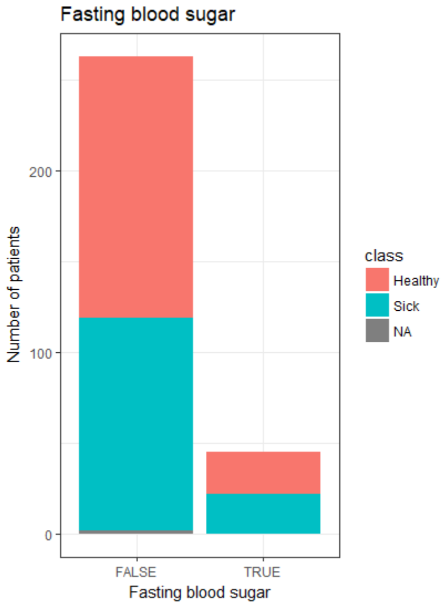
This attribute would be used in a machine learning model because you can conclude that patients in the Asymptomatic category are sick and most in NoTang are healthy, taking these conclusions would be useful in a machine learning model. In logistic regression this attribute would be helpful in defining the probability of a patient being sick or healthy.



Fasting Blood Sugar Bar Chart:

More than half of the patients that are not fasting are considered healthy. For patients that are fasting half are healthy and half are sick. If fasting is causing the sickness then going by this bar chart the healthier option would be to not fast.

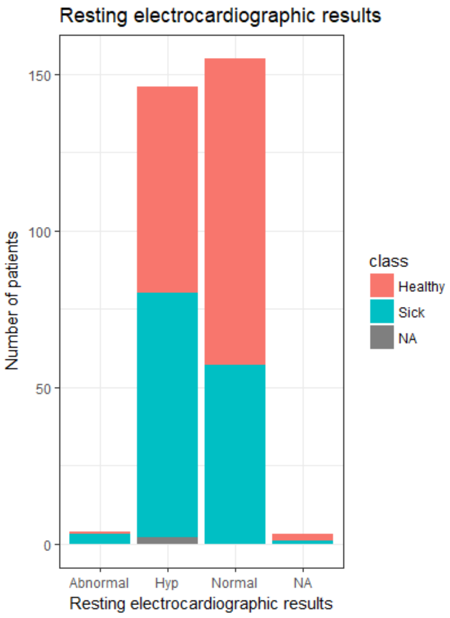
With half the patients being sick or healthy in the False category it would be difficult for a machine learning model to decide whether a patient is sick or healthy based off Fasting blood sugar category.



Resting Electrocardiographic Bar Chart:

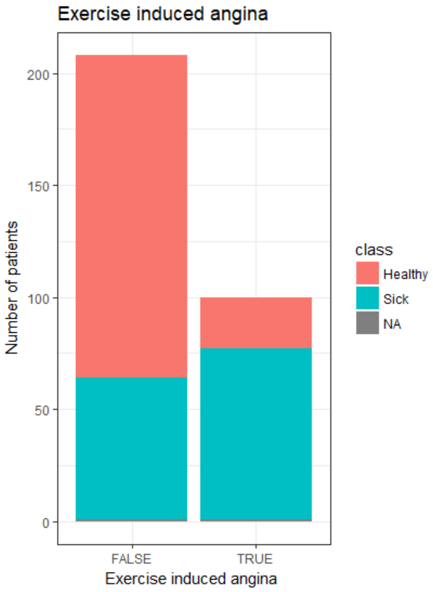
Most Abnormal results meant that the patients were sick. Most Hypertrophy results also meant that the patients were sick. More than half of patients with Normal results were considered healthy and were patients results were Non-Applicable there were slightly more healthier patients than sick.

Because the normal category has half sick and half healthy patients it would be difficult for a machine learning model to class whether a patient is sick or healthy based off this predictor attribute, therefore I would not expect to see this variable in a machine learning model.



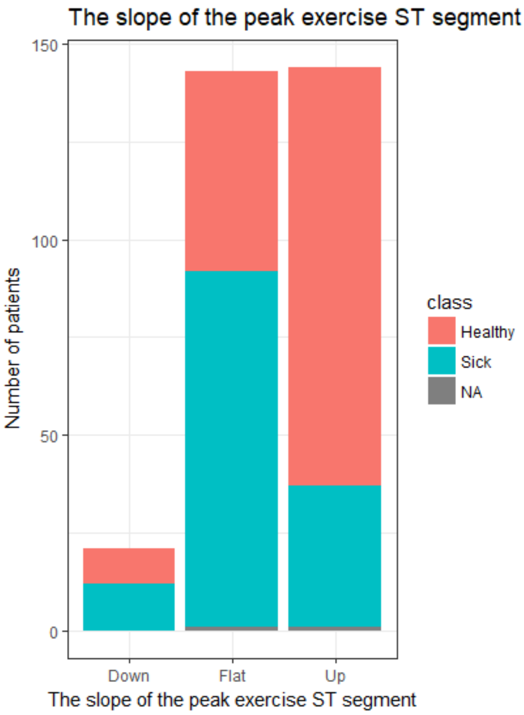
Exercise Induced Angina Bar Chart:

The majority of patients that did experience angina after exercise were considered sick. Patients that did not experience angina (200+ patients) were considered healthy with only 60 patients or so being considered sick without experiencing angina.



Slope Bar Chart:

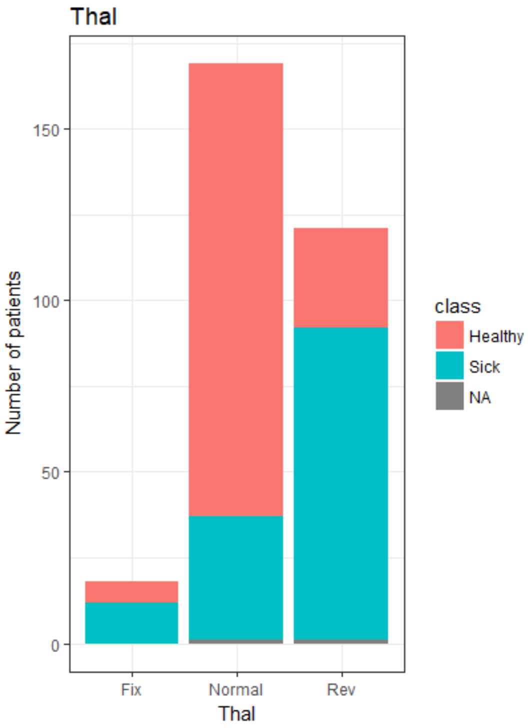
There was slightly more sick patients than healthy where the slope of the peak exercise ST segment was down. There was much more healthy patients when the peak was up and when the peak was flat there was about two thirds more sick patients than healthy patients.



Thalassemia Bar Chart:

From looking at the bar chart it is clear to see that the majority of patients with a fixed defect or a reversible defect are considered sick. Patients with normal Thalassemia were mostly healthy with about one quarter of those being considered sick.

From this you could say that you would expect to see this attribute in a machine learning model because it almost definitively says that patients in Fix or Rev category are healthy and those with normal are sick, this data would be useful in a machine learning model, performing logistic regression on this attribute you would be able to calculate the probability of a patient being sick or healthy based off their thal category.



4. Graphically and statistically detect outliers.

Graphical method to find outliers.

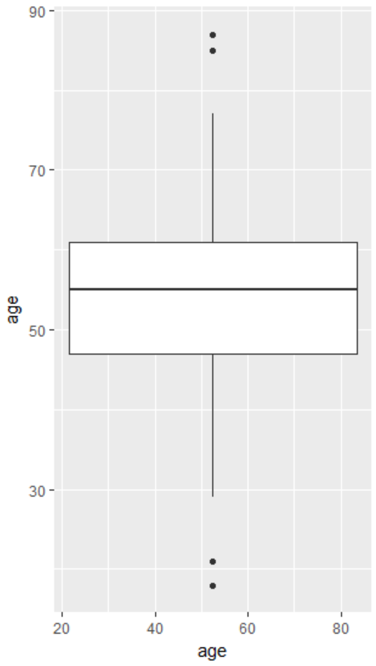
We used the boxplot as the graphical way to find outliers in our data variables. The box in the graph represented the interquartile range (top of the box is the third quartile and the bottom of the box is the first quartile) and the dark line the middle of the box represents the median of the data. The lines represented the max and minimum values excluding outliers, while the dots represented the actual outliers themselves. (Note: Repeating outliers will only have one dot. E.G. Age has 3 outliers but two dots as 85 appears twice.)

Statistical method to find outliers.

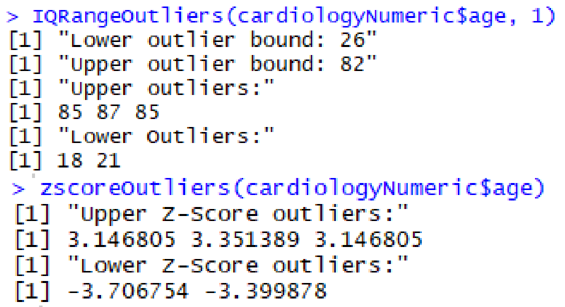
To statistically confirm our assertions from the box plot we made two functions in RStudio. One to generate outliers using the interquartile range and another function to generate outliers using the z score of the data.

Age Box Plot:

The age boxplot had outliers above and below the interquartile range. These outliers lie within the 80's age and the teens to 20's age group. The median of the age data is about 55/56. From looking at the box plot we seen that the maximum value excluding outliers was around 77/78 while the minimum value excluding outliers was approximately 30.



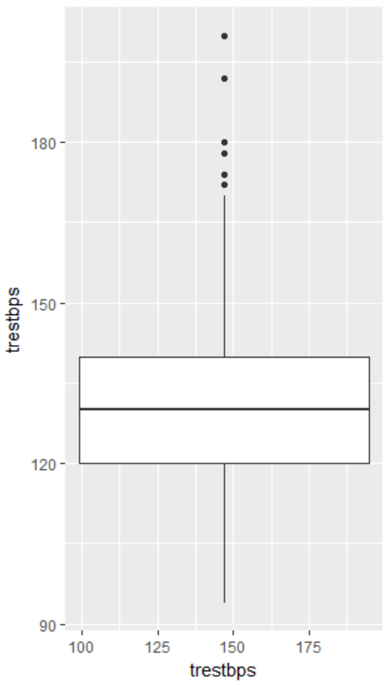
Our statistical functions confirmed our box plot assertions as you can see below. Two upper outliers where the same and two upper z-score outliers were also the same. Both functions also returned two lower outliers which also matches the box plot.

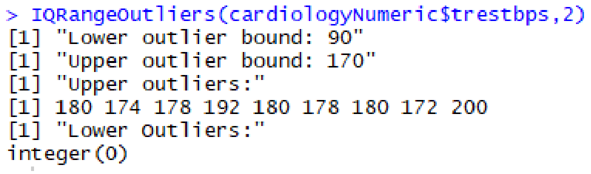

```
> IQRangeOutliers(cardiologyNumeric$age, 1)
[1] "Lower outlier bound: 26"
[1] "Upper outlier bound: 82"
[1] "Upper outliers:"
[1] 85 87 85
[1] "Lower Outliers:"
[1] 18 21
 > zscoreOutliers(cardiologyNumeric$age)
 [1] "Upper Z-Score outliers:"
 [1] 3.146805 3.351389 3.146805
 [1] "Lower Z-Score outliers:"
 [1] -3.706754 -3.399878
```

Trestbps Box Plot:

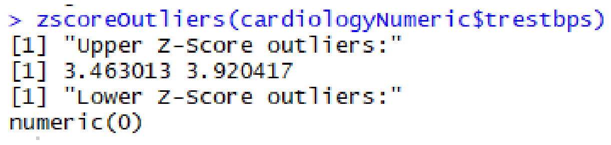
In the trestbps data there are 6 dots above the box and none below it. This tells us that there are only outliers in the upper half of the data. This also tells us that it doesn't have any outliers in the lower half of the data. The median of the trestbps data is also around 127/128. According to the boxplot the maximum value excluding outliers lies around 170. The box plot showed no lower value outliers.

Our box plot assertions were also confirmed here with 6 dots matching 6 upper outliers in our IQR function. Our IQR function and z-score function both also returned 0 lower outliers which matched the box plot. The z-score only returned 2 outliers but we think this may be because of the z-score range differing from the IQR range which means the z-score will only sometimes match the IQR. This can be seen with the other variables throughout.





```
> IQRangeOutliers(cardiologyNumeric$trestbps,2)
[1] "Lower outlier bound: 90"
[1] "Upper outlier bound: 170"
[1] "Upper outliers:"
[1] 180 174 178 192 180 178 180 172 200
[1] "Lower Outliers:"
integer(0)
```

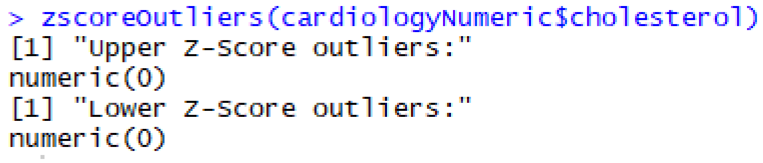


```
> zscoreOutliers(cardiologyNumeric$trestbps)
[1] "Upper Z-Score outliers:"
[1] 3.463013 3.920417
[1] "Lower Z-Score outliers:"
numeric(0)
```

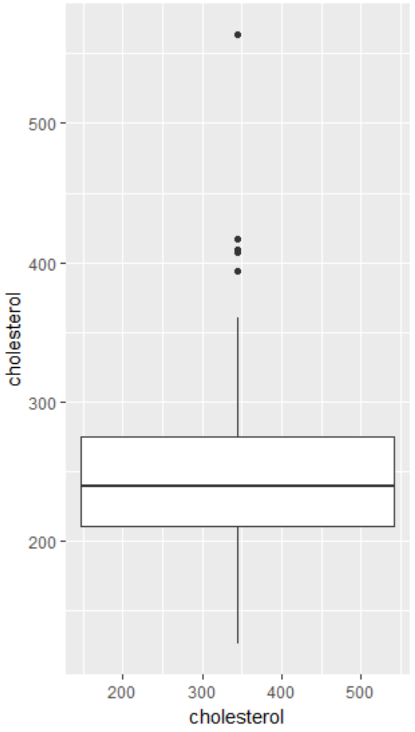
Cholesterol Box Plot:

The Cholesterol boxplot consisted of two groups of outliers. The maximum value excluding the outliers was approximately 355. Any values above this maximum where considered outliers. We could see from the box plot that there was two groups of outliers, one group around the 400 range and the other around the 550 range. The median of the cholesterol data was just below 250. The boxplot showed no lower value outliers.

Our IQR function didn't work for cholesterol due to some of the data inside the attribute and our z-score returned 0 for both upper and lower valued outliers. This contradicted our assertions but due to the high confirmation of our assumptions based on other graphs we can assume this was a once off.



```
> zscoreOutliers(cardiologyNumeric$cholesterol)
[1] "Upper Z-Score outliers:"
numeric(0)
[1] "Lower Z-Score outliers:"
numeric(0)
```

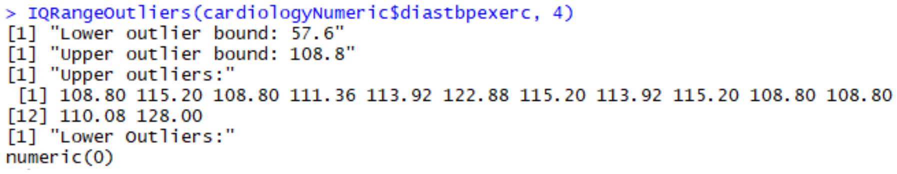


Diastbpexerc Box Plot:

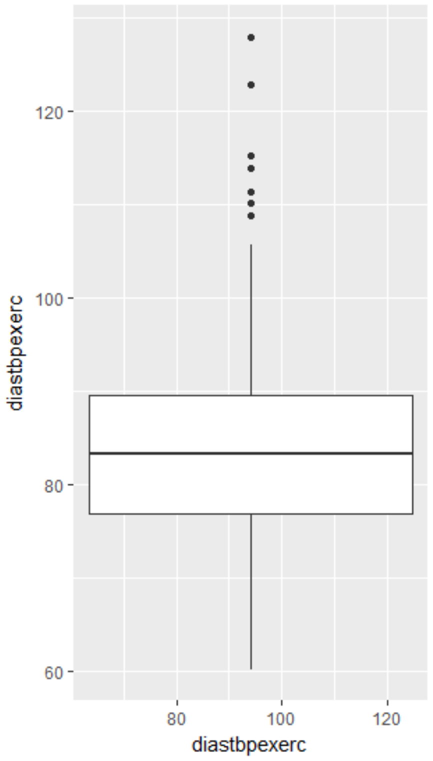
The boxplot displayed no lower valued outliers but many upper valued outliers. The maximum diastolic blood pressure value excluding outliers is approximately 107. The median diastolic blood pressure value is approximately 84. According to the box plot, the maximum outlier value is around 128, with many other outliers in the 109-115 blood pressure range and even further outliers with blood pressures of 122-128.

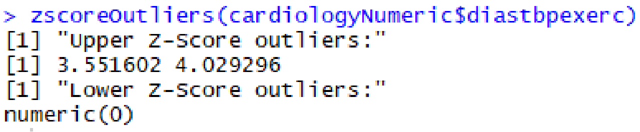
Here, once again our box plot outliers matched our IQR function and confirmed our assertions made based on the graph. The clusters of outliers in the graph match the closely related numbers returned in the IQR function. With no outliers in the graph both functions also returned 0 for lower value outliers

.



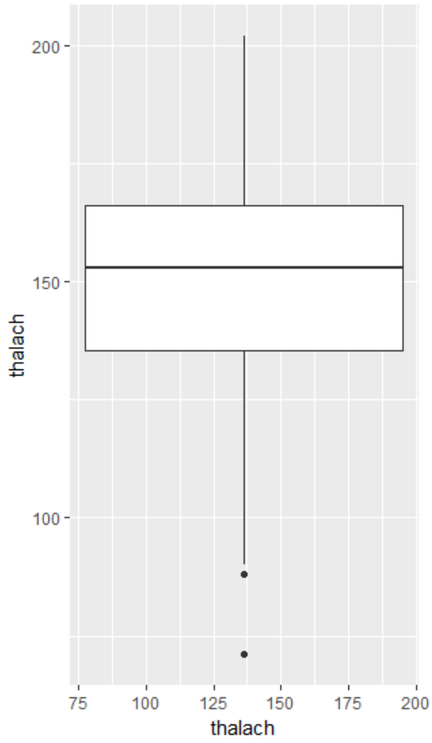
```
> IQRangeOutliers(cardiologyNumeric$diastbpexerc, 4)
[1] "Lower outlier bound: 57.6"
[1] "Upper outlier bound: 108.8"
[1] "Upper outliers:"
 [1] 108.80 115.20 108.80 111.36 113.92 122.88 115.20 113.92 115.20 108.80 108.80
[12] 110.08 128.00
[1] "Lower Outliers:"
numeric(0)
```



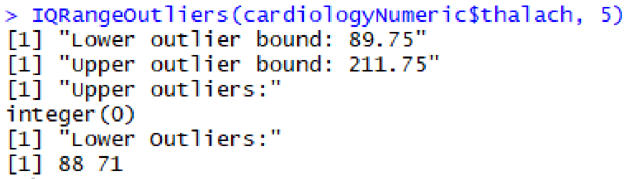


```
> zscoreOutliers(cardiologyNumeric$diastbpexerc)
[1] "Upper Z-Score outliers:"
[1] 3.551602 4.029296
[1] "Lower Z-Score outliers:"
numeric(0)
```

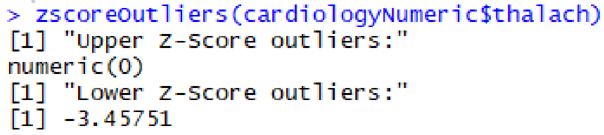
Thalach Box Plot:
This box plot displayed no upper valued outliers. The maximum value was just above 200 while the minimum value excluding outliers was around 88. The median value was also around 153. There are two dots in this boxplot with the furthest outlier sitting at around the 70 mark. This is the first box plot we have seen where the interquartile range box is in the upper half of the graph.



Our assertions where confirmed by our two functions here as both functions returned 0 for upper valued outliers which matches the graph. The graph had two dots for lower outliers and the IQR function also returned two values for lower outliers.

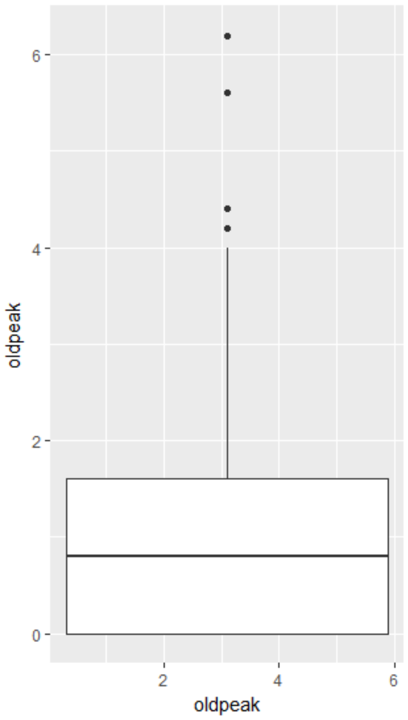


```
> IQRangeOutliers(cardiologyNumeric$thalach, 5)
[1] "Lower outlier bound: 89.75"
[1] "Upper outlier bound: 211.75"
[1] "Upper outliers:"
integer(0)
[1] "Lower Outliers:"
[1] 88 71
```

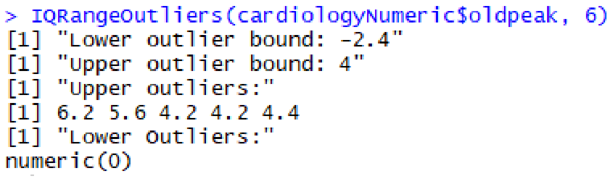


```
> zscoreOutliers(cardiologyNumeric$thalach)
[1] "Upper Z-Score outliers:"
numeric(0)
[1] "Lower Z-Score outliers:"
[1] -3.45751
```

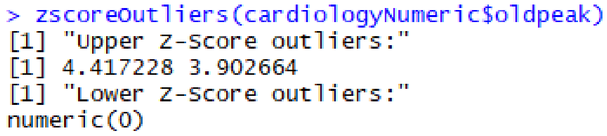
Old Peak Box Plot:
This is the first boxplot where the interquartile range box sits exactly at the bottom of the graph. This confirms that there are no lower valued outliers. It does however have upper valued outliers. The maximum value excluding outliers is 4. The outliers all start after 4 with the furthest outlier appearing just after the value 6. The first outlier is about 4.2 while the second is about 4.5. The median of the oldpeak data lies just under 1 with a value of approximately 0.8.



Here we can see that the IQR function returned 4 outliers with the graph also showing that we had 4 outliers, confirming our assertions once again. Both functions also returned 0 for lower valued outliers which also matched our assumptions based on the graph.

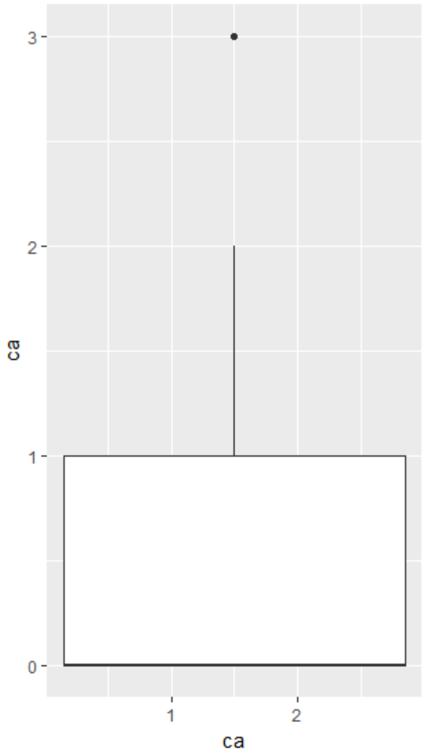


```
> IQRangeOutliers(cardiologyNumeric$oldpeak, 6)
[1] "Lower outlier bound: -2.4"
[1] "Upper outlier bound: 4"
[1] "Upper outliers:"
[1] 6.2 5.6 4.2 4.2 4.4
[1] "Lower Outliers:"
numeric(0)
```

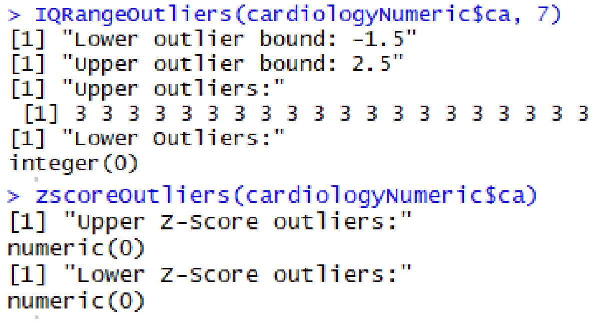


```
> zscoreOutliers(cardiologyNumeric$oldpeak)
[1] "Upper Z-Score outliers:"
[1] 4.417228 3.902664
[1] "Lower Z-Score outliers:"
numeric(0)
```

CA Box Plot:
This graph looks similar to the oldpeak graph as the interquartile range box sits exactly at the bottom. We also noticed that the median of the ca value also sits at the very bottom of the graph. The third quartile value is 1 and the maximum value excluding outliers is 2. This box plot only displayed one outlier value which we could see from the plot was 3.



The graph only has one dot but the IQR function returned twenty 3's. One dot on the graph doesn't mean one outlier. The graph doesn't compensate for outlier repetition which is evident here. Both functions also returned 0 lower valued outliers which matched our assertions.



```
> IQRangeOutliers(cardiologyNumeric$ca, 7)
[1] "Lower outlier bound: -1.5"
[1] "Upper outlier bound: 2.5"
[1] "Upper outliers:"
 [1] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
[1] "Lower Outliers:"
integer(0)
> zscoreOutliers(cardiologyNumeric$ca)
[1] "Upper Z-Score outliers:"
numeric(0)
[1] "Lower Z-Score outliers:"
numeric(0)
```

Merits:
Box Plot:

Very easy to see the outliers, the interquartile range, the median, and the min and max values excluding outliers.

IQR Function:

The function we created not only displayed the value of the outliers but also showed the boundary of which after a value becomes an outlier whether it's lower or upper.

Z-score Function:

Simple function that displays the z score once its outside the range of normal. Anything over 3 and anything below -3 will be displayed to the user, letting the user know that it is an outlier.

Demerits:
Box Plot:

Reading the graph is sometimes difficult when the axis numbers are on a large scale. When this happens it's hard to match the reading on the graph with the value on the axis. Tried to get labels working for outliers to save us guessing the value but to no success.

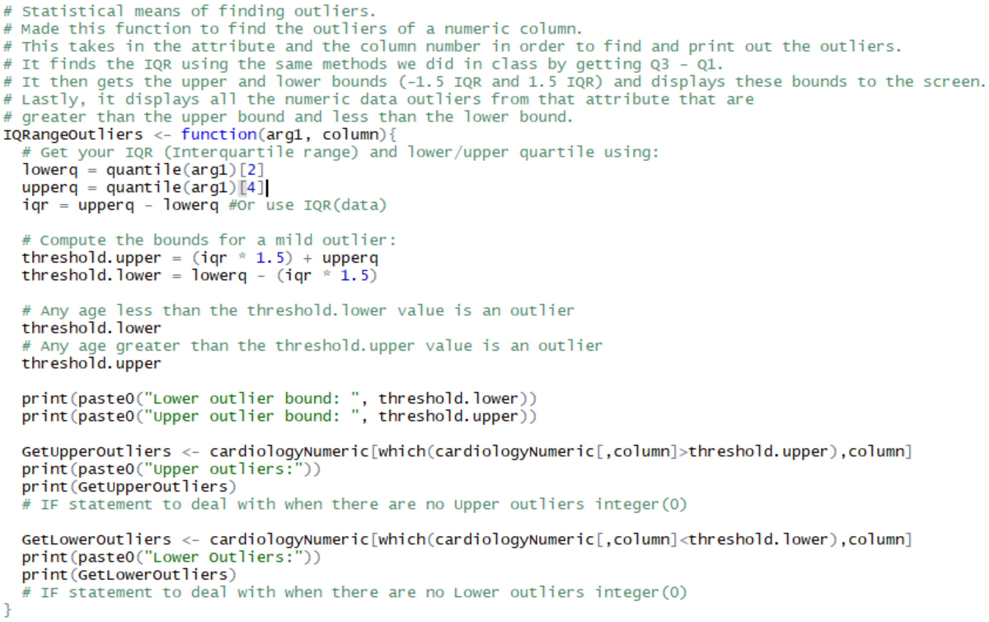
IQR Function:

At first we struggled with data that included NA's but eventually we got it working for this. We then couldn't get IF statements working inside the function to stop it displaying numeric(0) instead of just 0. We eventually decided that this wasn't a big deal and moved on.

Z-score Function:

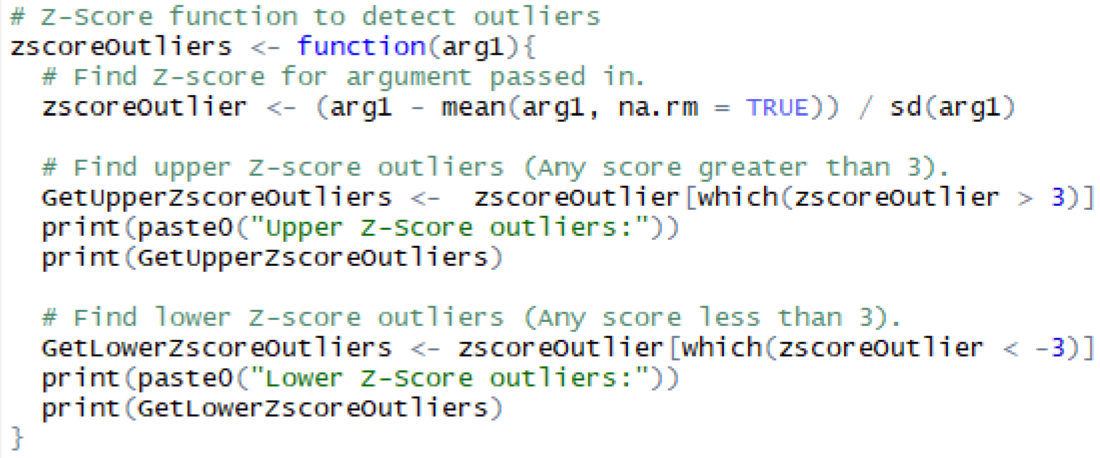
When running this function on certain variables, the results didn't match our boxplot assumptions and our IQR function. It worked perfectly on some variables like age but then on others it didn't match and we didn't know or have time to fix this. We think this may be because the range in z-score is different to IQR which means it will not return as many values back as the other functions because of this.

IQR Function:



```
# Statistical means of finding outliers.
# Made this function to find the outliers of a numeric column.
# This takes in the attribute and the column number in order to find and print out the outliers.
# It finds the IQR using the same methods we did in class by getting Q3 - Q1.
# It then gets the upper and lower bounds (-1.5 IQR and 1.5 IQR) and displays these bounds to the screen.
# Lastly, it displays all the numeric data outliers from that attribute that are
# greater than the upper bound and less than the lower bound.
IQRangeOutliers <- function(arg1, column){
  # Get your IQR (Interquartile range) and lower/upper quartile using:
  lowerq = quantile(arg1)[2]
  upperq = quantile(arg1)[4]
  iqr = upperq - lowerq #Or use IQR(data)

  # Compute the bounds for a mild outlier:
  threshold.upper = (iqr * 1.5) + upperq
  threshold.lower = lowerq - (iqr * 1.5)

  # Any age less than the threshold.lower value is an outlier
  threshold.lower
  # Any age greater than the threshold.upper value is an outlier
  threshold.upper

  print(paste0("Lower outlier bound: ", threshold.lower))
  print(paste0("Upper outlier bound: ", threshold.upper))

  GetUpperOutliers <- cardiologyNumeric[which(cardiologyNumeric[,column]>threshold.upper),column]
  print(paste0("Upper outliers:"))
  print(GetUpperOutliers)
  # IF statement to deal with when there are no Upper outliers integer(0)

  GetLowerOutliers <- cardiologyNumeric[which(cardiologyNumeric[,column]<threshold.lower),column]
  print(paste0("Lower Outliers:"))
  print(GetLowerOutliers)
  # IF statement to deal with when there are no Lower outliers integer(0)
}
```

Z-score function:



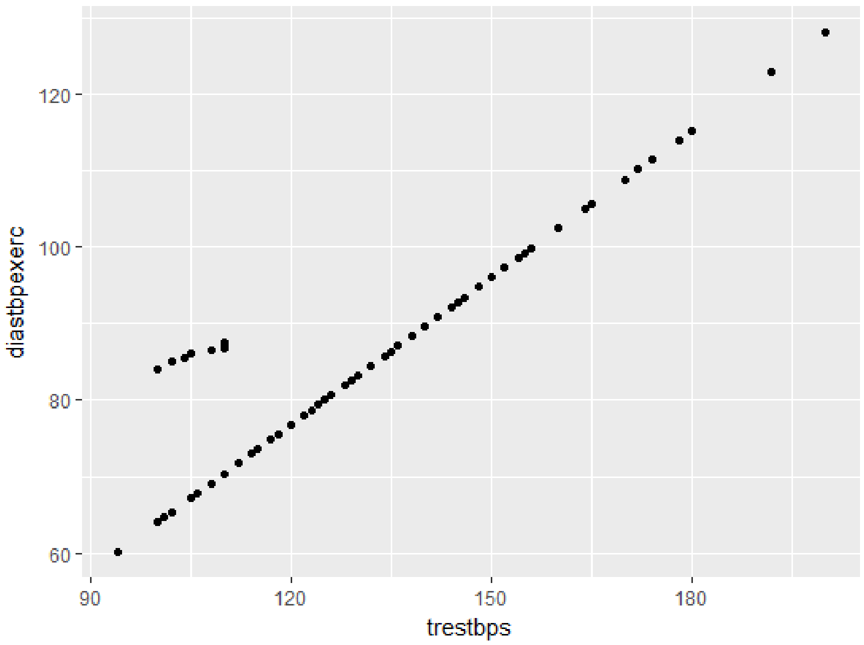
```
# Z-Score function to detect outliers
zscoreOutliers <- function(arg1){
  # Find Z-score for argument passed in.
  zscoreOutlier <- (arg1 - mean(arg1, na.rm = TRUE)) / sd(arg1)

  # Find upper Z-score outliers (Any score greater than 3).
  GetUpperZscoreOutliers <-  zscoreOutlier[which(zscoreOutlier > 3)]
  print(paste0("Upper Z-Score outliers:"))
  print(GetUpperZscoreOutliers)

  # Find lower Z-score outliers (Any score less than 3).
  GetLowerZscoreOutliers <- zscoreOutlier[which(zscoreOutlier < -3)]
  print(paste0("Lower Z-Score outliers:"))
  print(GetLowerZscoreOutliers)
}
```

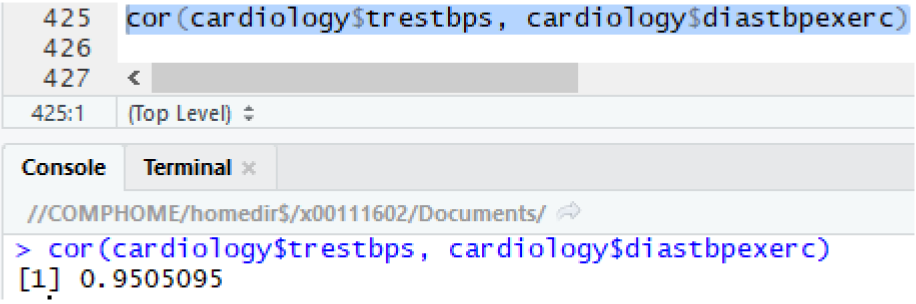
5. Investigate whether there are any correlated variables

Using R to create scatter plots we were able to identify if any of the numeric attributes are correlated. We were also able to verify the correlation by running a cor function in R to give us a decimal output of the correlation between numeric variables.

Most of the attributes have a low correlation with each other. To prove this we created scatter plots for each attribute and verified the low correlation using the cor function, this can be seen in our script.

The trestbps attribute has a high correlation with diastpexerc.

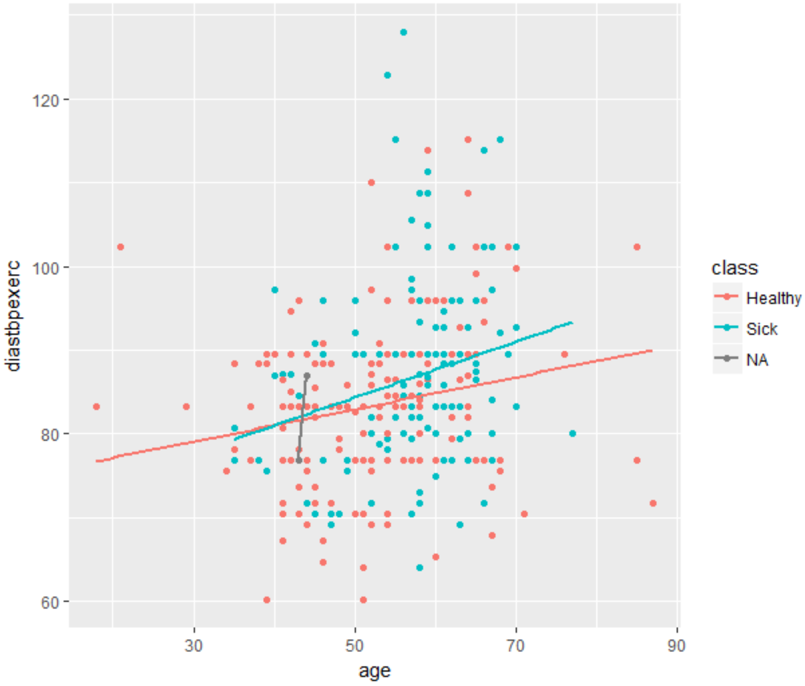




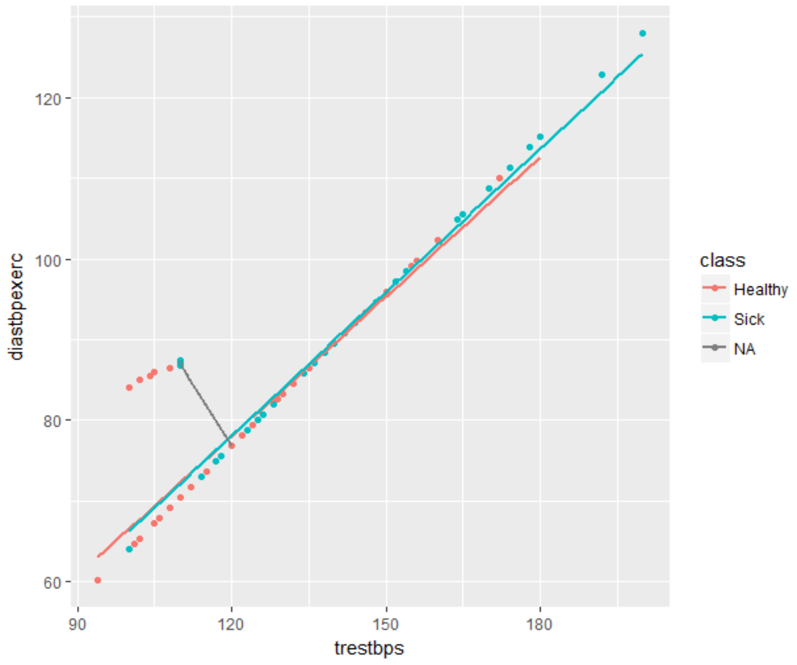
```
cor(cardiology$trestbps, cardiology$diastbpexerc)
```

```
> cor(cardiology$trestbps, cardiology$diastbpexerc)
[1] 0.9505095
```

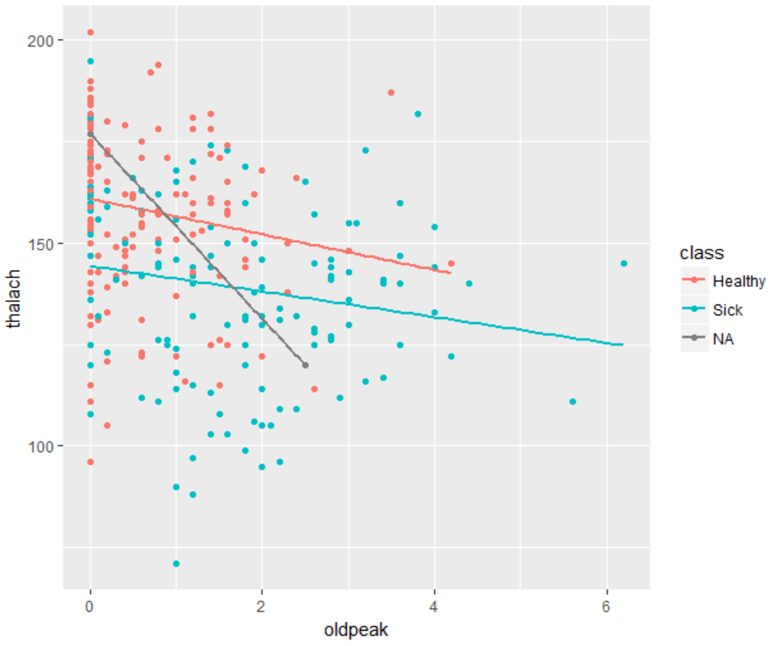
In regards to which attributes seem to be most linked to the target variable we found from creating scatter plots with an overlay of the target variable that the age attribute which has a high correlation with diastpexerc also are linked to the target class variable. Excluding the outliers in each attribute you can see that most of the patients have a correlation with the target variable.



Also there is a high correlation between diastpexerc and the trestbps attribute, each patient seems to be linked to the target variable. From looking at the graph you can confidently say that both of these attributes are linked to the target attribute.



Oldpeak vs Thalach has a low correlation and also doesn't seem to be linked to the target variable, from this you can see most of the patients aren't close to the target variable so it would be hard for any machine learning algorithm to use these attributes in relation to the target variable.



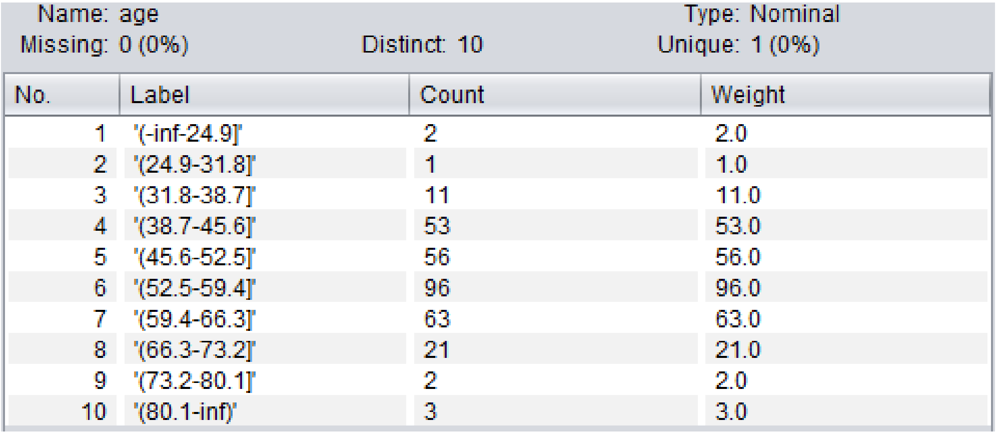
Part 2: Cleaning and Transforming the Dataset (using Weka)

6. Using Weka

We chose age as our numeric variable to bin (discretise) the data using “equal width binning”. After loading the dataset into WEKA as an arff file we then began the cleaning and transforming steps. To perform bin (discretize) the data we set the number of bins to 10 and ran the algorithm. The algorithm output put the age variable into 10 bins with a range < 24.9 to > 80.1. It produced a bin interval of 6.9, nominal data with 0% missing and 1 unique value. The highest count with 96 values came from bin 6, the age range 52.5 – 59.4.

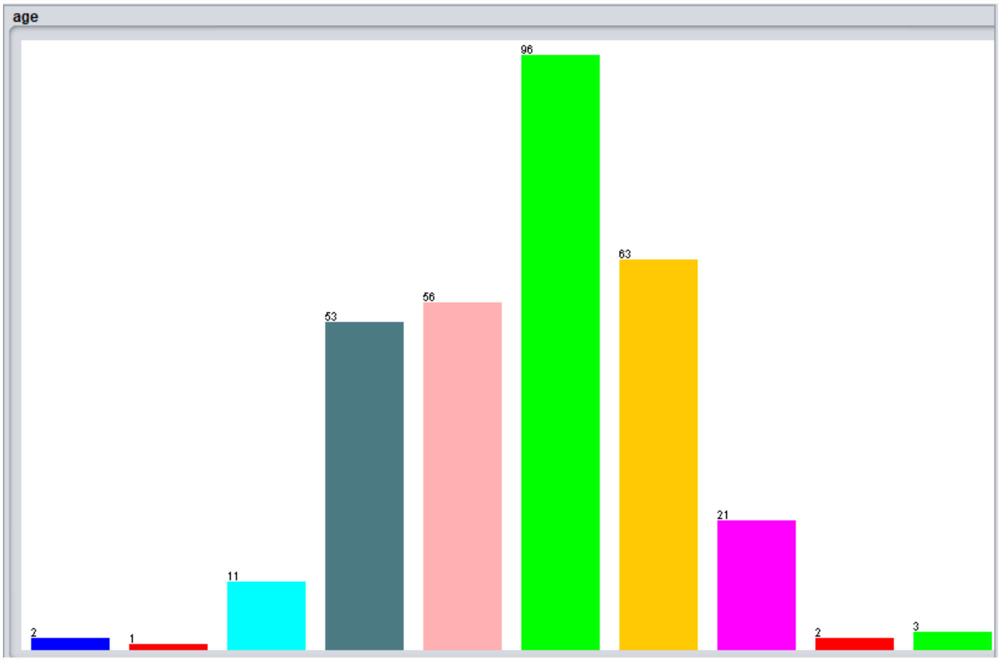
From looking at the histogram for age vs class you can see that most of the sick patients are also within this range.

Performing unsupervised discretization it broke the data into tasks that found the number of discrete values, the boundaries of the intervals, the range of the numeric age attribute.

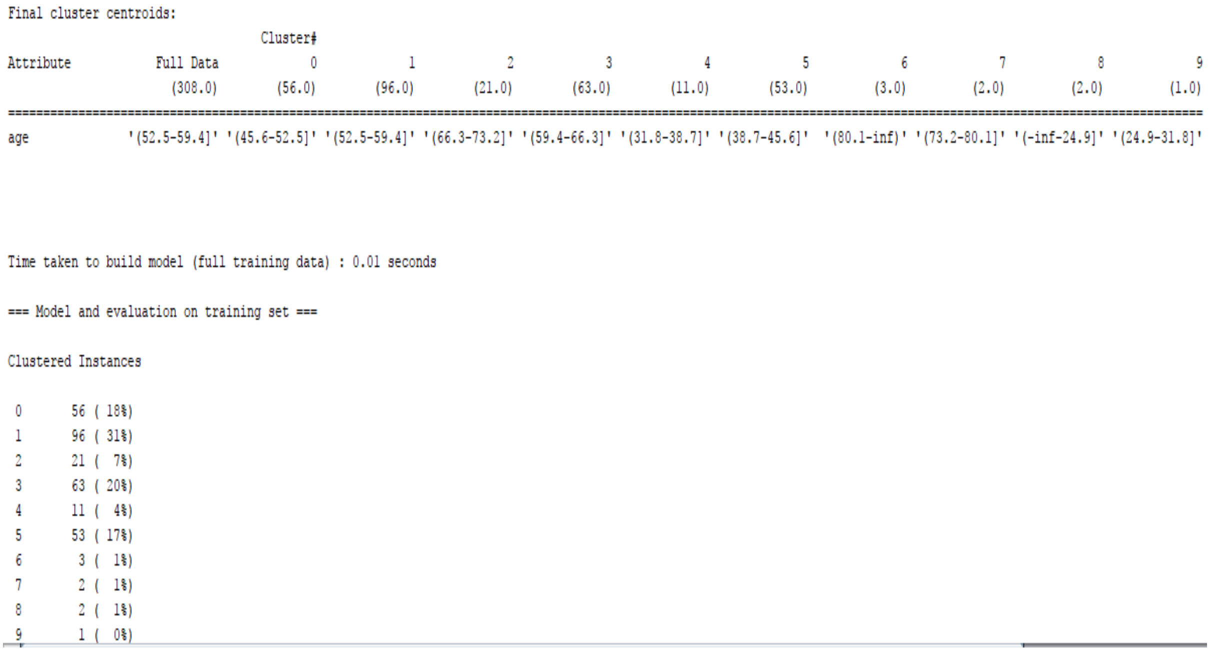


Name: age Type: Nominal
Missing: 0 (0%) Distinct: 10 Unique: 1 (0%)

No.	Label	Count	Weight
1	'(-inf-24.9]'	2	2.0
2	'(24.9-31.8]'	1	1.0
3	'(31.8-38.7]'	11	11.0
4	'(38.7-45.6]'	53	53.0
5	'(45.6-52.5]'	56	56.0
6	'(52.5-59.4]'	96	96.0
7	'(59.4-66.3]'	63	63.0
8	'(66.3-73.2]'	21	21.0
9	'(73.2-80.1]'	2	2.0
10	'(80.1-inf)'	3	3.0



Using K Means clustering algorithm on the numeric attribute we ran the simple k means function in WEKA. This produced a Clusterer output sheet with the run information, number of iterations: 2, number of clusters: 10, interval of 6.9. We chose 10 clusters to create a clustering algorithm similar to “equal width binning” algorithm. The output was similar to bin discretization but it grouped the clusters and produced a percentage of the count of patients in each cluster, the biggest cluster being the age group 52.5 – 59.4, 31% of the data and the smallest cluster being the age group 24.9 – 31.8, 0%.



```
Final cluster centroids:
                                   Cluster#
Attribute            Full Data            0              1              2              3              4              5              6              7              8              9
                       (308.0)       (56.0)         (96.0)         (21.0)         (63.0)         (11.0)         (53.0)          (3.0)          (2.0)          (2.0)          (1.0)
===================================================================================================================================================================================
age              '(52.5-59.4]' '(45.6-52.5]' '(52.5-59.4]' '(66.3-73.2]' '(59.4-66.3]' '(31.8-38.7]' '(38.7-45.6]'  '(80.1-inf)' '(73.2-80.1]' '(-inf-24.9]' '(24.9-31.8]'




Time taken to build model (full training data) : 0.01 seconds

=== Model and evaluation on training set ===

Clustered Instances

 0       56 ( 18%)
 1       96 ( 31%)
 2       21 (  7%)
 3       63 ( 20%)
 4       11 (  4%)
 5       53 ( 17%)
 6        3 (  1%)
 7        2 (  1%)
 8        2 (  1%)
 9        1 (  0%)
```

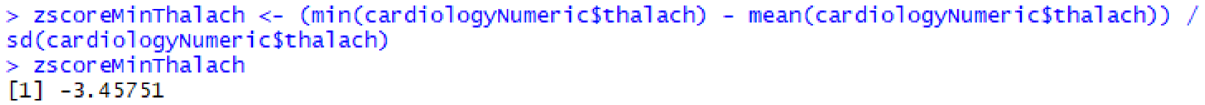
For an optimal solution I would choose k-means clustering because the output was easier to read and understand. From looking at equal width binning output it was harder to understand and difficult to read the output. K means clustering partitions the observations into clusters and each of the parameters can easily be modified to allow us to partition data into more or less clusters. Since a lot of the data is numeric k means algorithm is useful for finding groups in numerical data.

7. Transforming skewed data

Using one numeric variable which is skewed (Thalach (Negatively Skewed)), transform the data using the following methods to achieve normality:

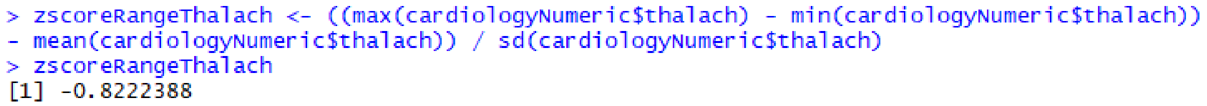
To normalise the data using transformation we used the variables min, max, and range (max-min) on each of the 4 common transformations.

A. Z-Score Standardisation



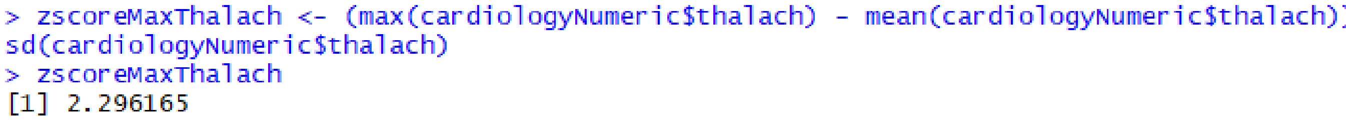
```
> zscoreMinThalach <- (min(cardiologyNumeric$thalach) - mean(cardiologyNumeric$thalach)) /
sd(cardiologyNumeric$thalach)
> zscoreMinThalach
[1] -3.45751
```

Data values below the mean will have a negative z-score standardisation.



```
> zscoreRangeThalach <- ((max(cardiologyNumeric$thalach) - min(cardiologyNumeric$thalach))
- mean(cardiologyNumeric$thalach)) / sd(cardiologyNumeric$thalach)
> zscoreRangeThalach
[1] -0.8222388
```

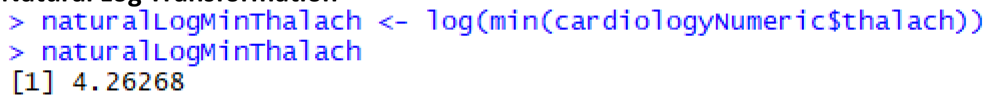
Data values falling on the mean will have zero z-score standardisation. (Negative 0 in this case)



```
> zscoreMaxThalach <- (max(cardiologyNumeric$thalach) - mean(cardiologyNumeric$thalach))
sd(cardiologyNumeric$thalach)
> zscoreMaxThalach
[1] 2.296165
```

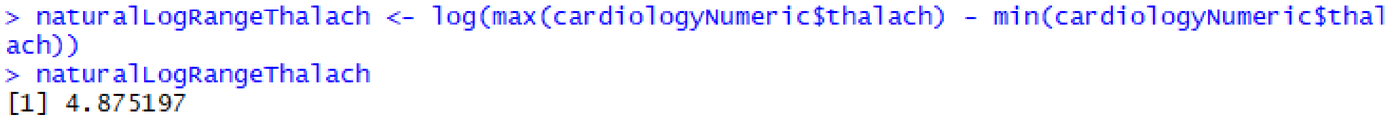
Data values above the mean will have a positive z-score standardisation.

B. Natural Log Transformation



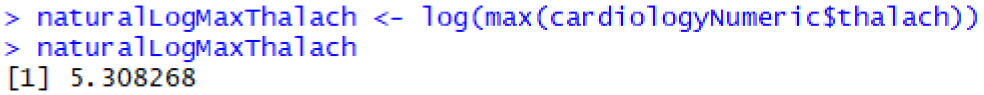
```
> naturalLogMinThalach <- log(min(cardiologyNumeric$thalach))
> naturalLogMinThalach
[1] 4.26268
```

Data values below the mean will have a positive natural log transformation.



```
> naturalLogRangeThalach <- log(max(cardiologyNumeric$thalach) - min(cardiologyNumeric$thal
ach))
> naturalLogRangeThalach
[1] 4.875197
```

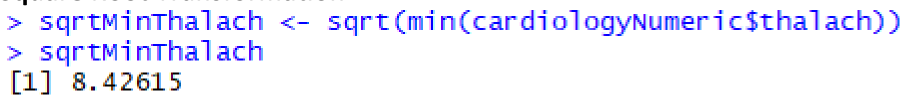
Data values falling on the mean will have a positive natural log transformation.



```
> naturalLogMaxThalach <- log(max(cardiologyNumeric$thalach))
> naturalLogMaxThalach
[1] 5.308268
```

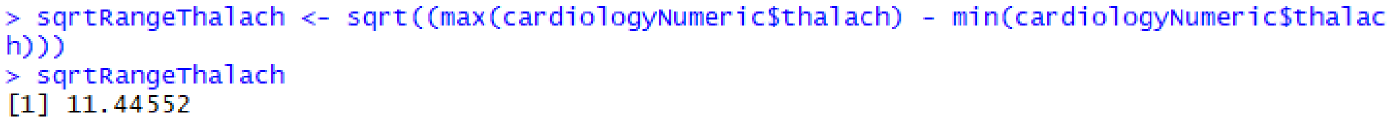
Data values above the mean will have a positive natural log transformation.

C. Square Root Transformation



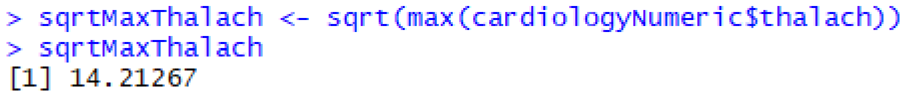
```
> sqrtMinThalach <- sqrt(min(cardiologyNumeric$thalach))
> sqrtMinThalach
[1] 8.42615
```

Data values below the mean will have a positive square root transformation.



```
> sqrtRangeThalach <- sqrt((max(cardiologyNumeric$thalach) - min(cardiologyNumeric$thalac
h)))
> sqrtRangeThalach
[1] 11.44552
```

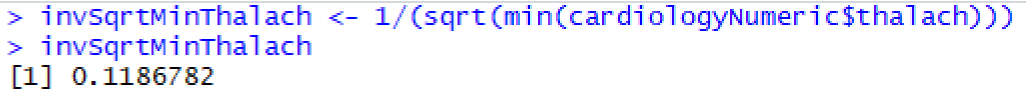
Data values below the mean will have a positive square root transformation.



```
> sqrtMaxThalach <- sqrt(max(cardiologyNumeric$thalach))
> sqrtMaxThalach
[1] 14.21267
```

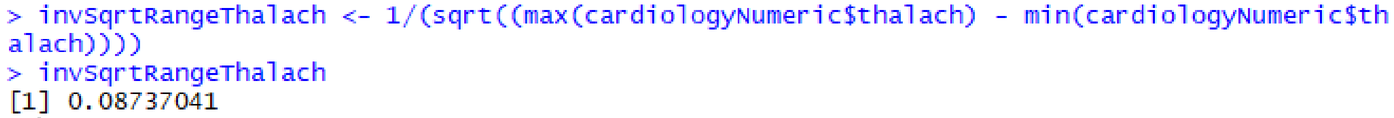
Data values below the mean will have a positive square root transformation.

D. **Inverse Square Root Transformation**



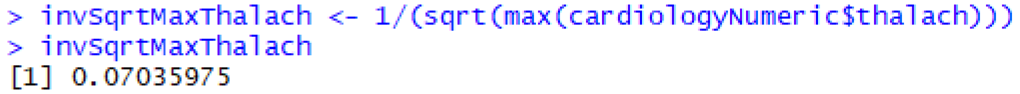
```
> invSqrtMinThalach <- 1/(sqrt(min(cardiologyNumeric$thalach)))
> invSqrtMinThalach
[1] 0.1186782
```

Data values below the mean will have a zero inverse square root transformation.



```
> invSqrtRangeThalach <- 1/(sqrt((max(cardiologyNumeric$thalach) - min(cardiologyNumeric$th
alach))))
> invSqrtRangeThalach
[1] 0.08737041
```

Data values below the mean will have a zero inverse square root transformation.



```
> invSqrtMaxThalach <- 1/(sqrt(max(cardiologyNumeric$thalach)))
> invSqrtMaxThalach
[1] 0.07035975
```

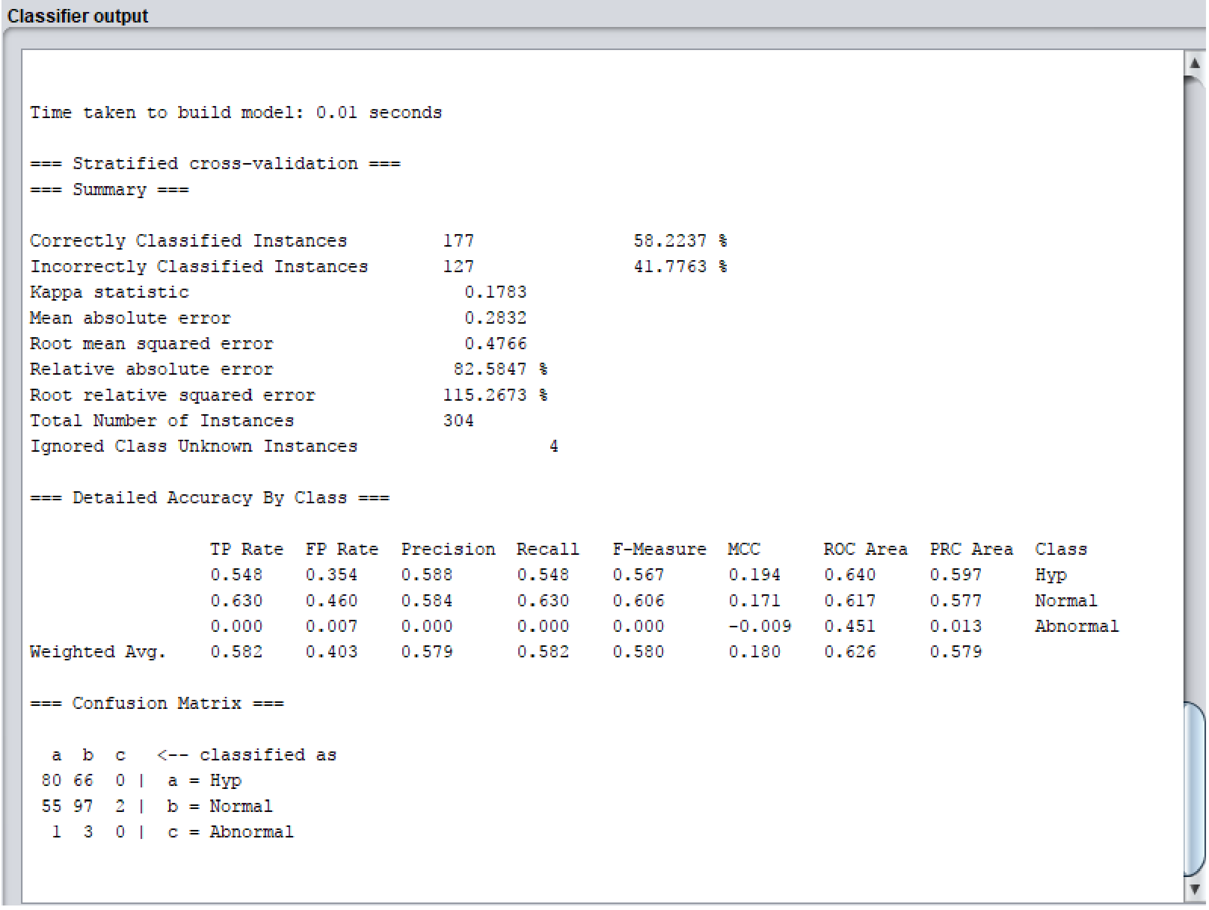
Data values below the mean will have a zero inverse square root transformation.

Each transformation will balance the skewness out differently. I found that the natural log and square root transformation results were a lot closer together than the z-score and inverse square root results. This would result in a more normalised bell-shape graph with reduced skewness throughout.

8. Impute missing values

For this question we took the categorical variable restecg (resting electrocardiographic results) to develop an appropriate machine learning model to impute missing values for that attribute. There are a number of 'NA' rows in the dataset so we began by replacing these values in the csv file with '?' so that these values can be read in a .arff file, this also stops weka reading in columns at strings when they should be nominal or numeric. We used weka to perform the necessary actions. We started by making the restecg variable our target variable and deleting the current class target variable. The next step we took was to load the new arff file into weka explorer.

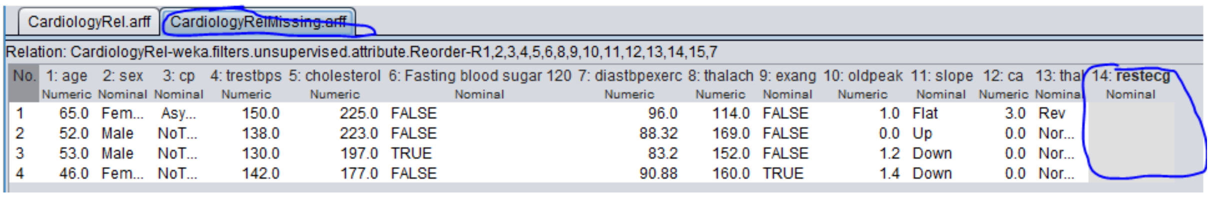
The next step was to go into classify in weka, then run the J48 package on our dataset. This gave a Classifier output giving us the build details.



Classifier output

```
Time taken to build model: 0.01 seconds

=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances         177               58.2237 %
Incorrectly Classified Instances       127               41.7763 %
Kappa statistic                          0.1783
Mean absolute error                      0.2832
Root mean squared error                  0.4766
Relative absolute error                 82.5847 %
Root relative squared error            115.2673 %
Total Number of Instances              304
Ignored Class Unknown Instances                   4

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.548    0.354    0.588      0.548    0.567      0.194    0.640     0.597     Hyp
                 0.630    0.460    0.584      0.630    0.606      0.171    0.617     0.577     Normal
                 0.000    0.007    0.000      0.000    0.000      -0.009   0.451     0.013     Abnormal
Weighted Avg.    0.582    0.403    0.579      0.582    0.580      0.180    0.626     0.579

=== Confusion Matrix ===

  a  b  c   <-- classified as
 80 66  0 |  a = Hyp
 55 97  2 |  b = Normal
  1  3  0 |  c = Abnormal
```

One thing to notice was the "Ignored class unknown instances = 4" this is the missing data. Then created another missing arff file with only the missing instances inside the file. With the new file it is now possible to create a test set in weka to run against the first classifier output.



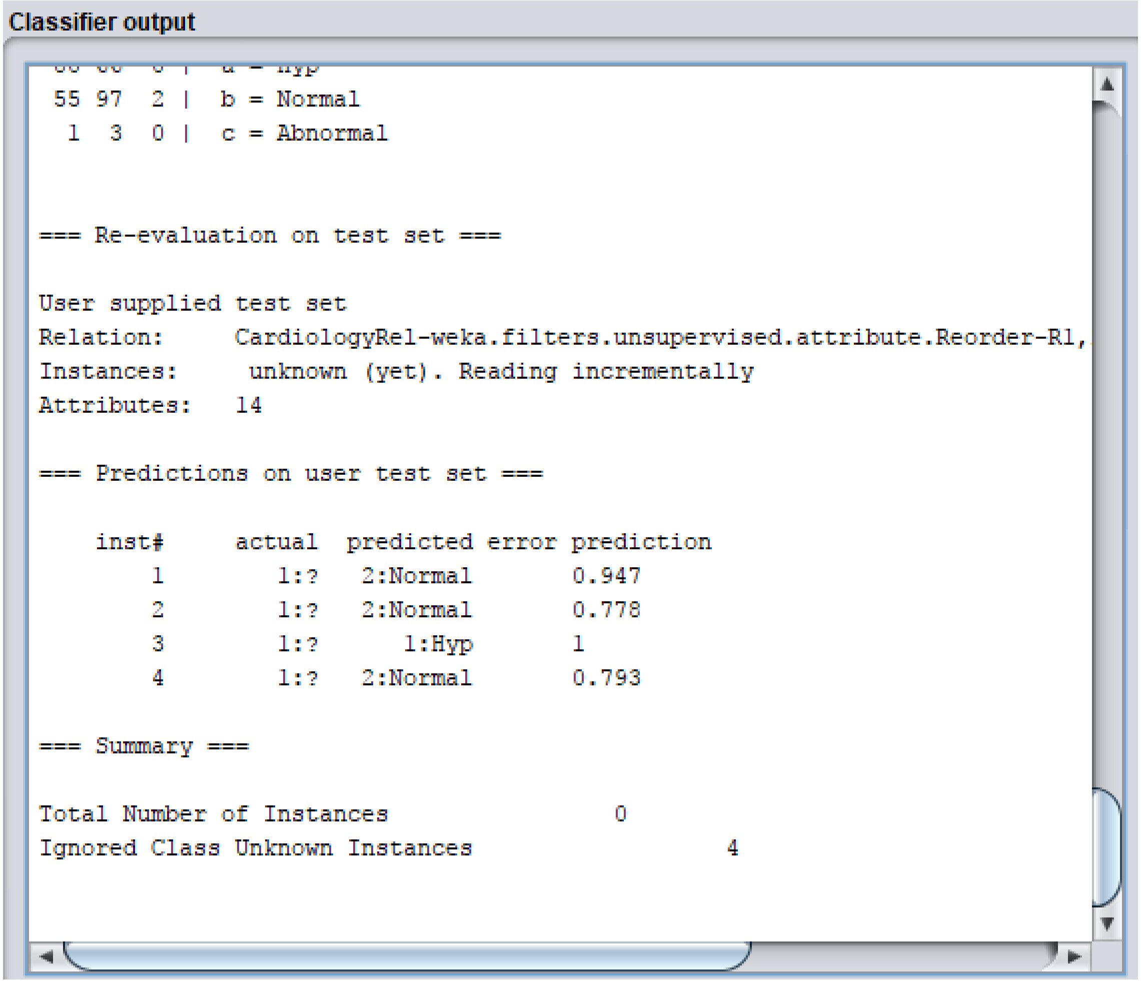
Relation: CardiologyRel-weka.filters.unsupervised.attribute.Reorder-R1,2,3,4,5,6,8,9,10,11,12,13,14,15,7

No.	1: age Numeric	2: sex Nominal	3: cp Nominal	4: trestbps Numeric	5: cholesterol Numeric	6: Fasting blood sugar 120 Nominal	7: diastbpexerc Numeric	8: thalach Numeric	9: exang Nominal	10: oldpeak Numeric	11: slope Nominal	12: ca Numeric	13: thal Nomina	14: restecg Nominal
1	65.0	Fem...	Asy...	150.0	225.0	FALSE	96.0	114.0	FALSE	1.0	Flat	3.0	Rev	
2	52.0	Male	NoT...	138.0	223.0	FALSE	88.32	169.0	FALSE	0.0	Up	0.0	Nor...	
3	53.0	Male	NoT...	130.0	197.0	TRUE	83.2	152.0	FALSE	1.2	Down	0.0	Nor...	
4	46.0	Fem...	NoT...	142.0	177.0	FALSE	90.88	160.0	TRUE	1.4	Down	0.0	Nor...	

The predictions in plain text gave us an output and a prediction of the restecg results. The output of each instance says that the first predicted instance is Normal restecg results with a 0.94 prediction, second is Normal with a 0.77 prediction, third is Hyp with a 1 prediction, and forth instance is Normal with a 0.79 prediction.

Each of the predictions are strong and from the results you could assume that they are correct. The kappa statistic is 0.1783 which is greater than 0 meaning that the classification is doing better than chance. Also each of the instances TP Rate, FP Rate, Recall, F-Measure and MMC is greater than 0 except for instance 3 which has a value of 0 meaning the predictor is random. The confusion matrix categorised the variables into Hyp, Normal and Abnormal. Since the precision of most of the instances is 58% this is probably not a good model.

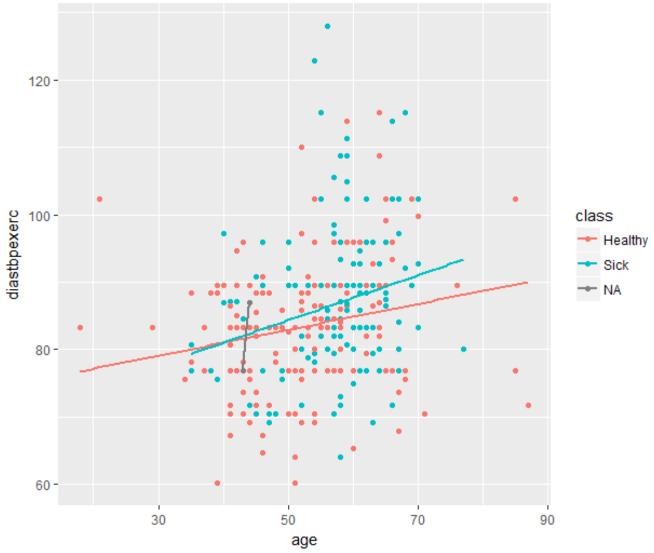
For the rest of the categorical and numeric data we replace the NA values with the mode and median.



```
 55 97  2 |  b = Normal
  1  3  0 |  c = Abnormal



=== Re-evaluation on test set ===

User supplied test set
Relation:     CardiologyRel-weka.filters.unsupervised.attribute.Reorder-R1,
Instances:     unknown (yet). Reading incrementally
Attributes:   14

=== Predictions on user test set ===

    inst#     actual  predicted error prediction
        1        1:?   2:Normal       0.947
        2        1:?   2:Normal       0.778
        3        1:?      1:Hyp       1
        4        1:?   2:Normal       0.793

=== Summary ===

Total Number of Instances                0
Ignored Class Unknown Instances                  4
```

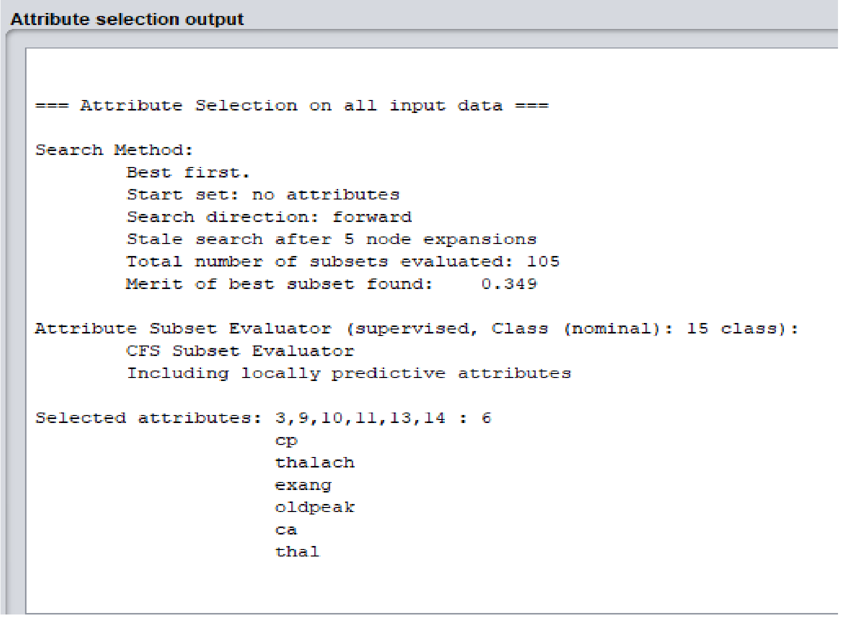
9. Are there any variables which can be eliminated

To eliminate variables from the dataset one technique we used was in question 5 where we had to identify the attributes which are most linked to the target variable, the more a predictor variable is linked to a target variable then the more useful it is to include in the dataset for Machine learning algorithms. Age, diastpexerc, and trestbps all have a high correlation with each other and are linked to the class attribute so it would be important to keep these in the dataset.



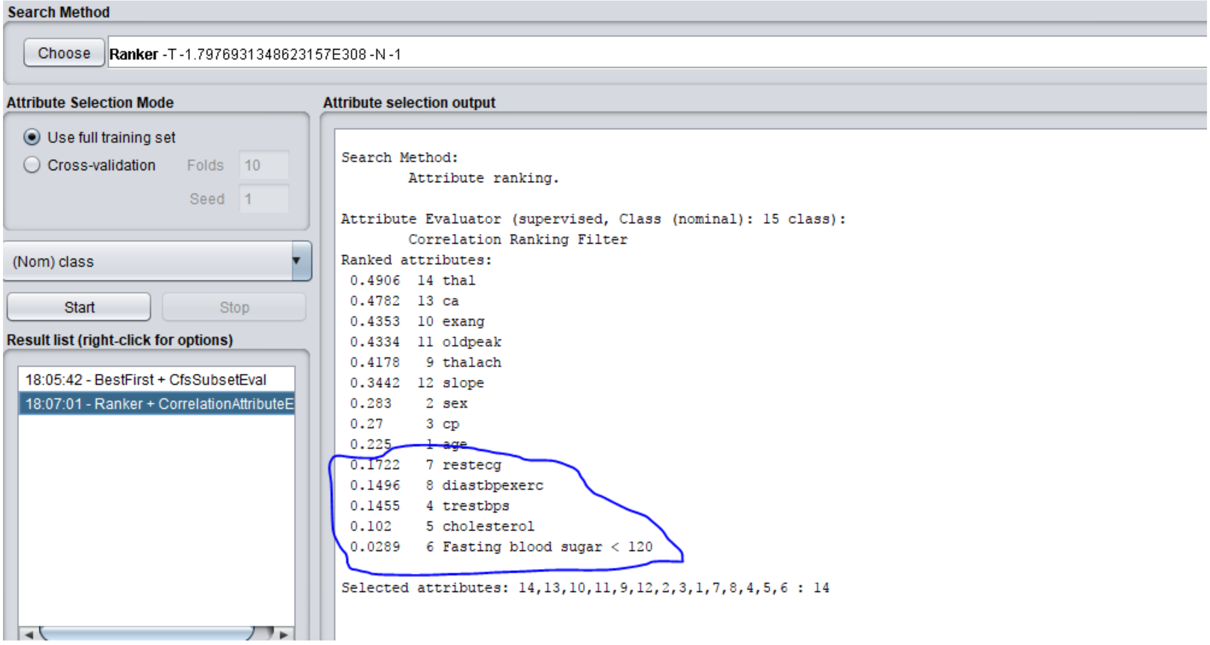
Another technique was using weka to impute missing data, we used Restecg which could also be kept in the dataset as using this categorical data was used to impute missing values and was useful for the machine learning algorithm in Weka to give a prediction on what category the missing data belongs to.

Using Weka techniques under the Select attributes tab we used two other techniques to find which values we could eliminate. The first was the best first algorithm which selected cp, thalach, exang, oldpeak, ca and thal so these attributes could be kept.



```
=== Attribute Selection on all input data ===

Search Method:
        Best first.
        Start set: no attributes
        Search direction: forward
        Stale search after 5 node expansions
        Total number of subsets evaluated: 105
        Merit of best subset found:    0.349

Attribute Subset Evaluator (supervised, Class (nominal): 15 class):
        CFS Subset Evaluator
        Including locally predictive attributes

Selected attributes: 3,9,10,11,13,14 : 6
                     cp
                     thalach
                     exang
                     oldpeak
                     ca
                     thal
```

Also we ran the Ranker + correlationAttribute algorithm which tells the rank of each attribute. If we set the cut-off point at 0.2 then we could remove restecg, diastpexerc, trestbps, cholesterol, fasting blood sugar < 120.


Search Method
Choose
Attribute Selection Mode
Attribute selection output
Start
Stop
Result list (right-click for options)

Ranker -T -1.7976931348623157E308 -N -1

● Use full training set
○ Cross-validation Folds 10 Seed 1

(Nom) class

18:05:42 - BestFirst + CfsSubsetEval
18:07:01 - Ranker + CorrelationAttributeE

```
Search Method:
        Attribute ranking.

Attribute Evaluator (supervised, Class (nominal): 15 class):
        Correlation Ranking Filter
Ranked attributes:
 0.4906  14 thal
 0.4782  13 ca
 0.4353  10 exang
 0.4334  11 oldpeak
 0.4178   9 thalach
 0.3442  12 slope
 0.283    2 sex
 0.27     3 cp
 0.225    1 age
 0.1722   7 restecg
 0.1496   8 diastbpexerc
 0.1455   4 trestbps
 0.102    5 cholesterol
 0.0289   6 Fasting blood sugar < 120

Selected attributes: 14,13,10,11,9,12,2,3,1,7,8,4,5,6 : 14
```

Appendices:

Appendix A: Script in R

```
cardiology <- read.table(file="C:/RCA1/CardiologyRel.csv", stringsAsFactors=FALSE, sep=",",
header=TRUE)

# show the first 10 records
head(cardiology)
cardiology[1:10,]

# show the first 12 records and first 3 columns
cardiology[1:12,1:3]

#Create a table of counts for each discrete value of outlook
A <- cardiology$sex
ZZ <- table(A)
ZZ

# Let's show the data as a piechart - not the best visualisation to use!
# Good for getting a rough idea of proportions
pie(ZZ, labels=names(ZZ), edges=200, col=c("yellow","red","navy","green"), radius=
    0.9)

# Part 1 Question 1 B
# % of missing values
sum(is.na(cardiology$age))
sum(is.na(cardiology$sex))
sum(is.na(cardiology$cp))
sum(is.na(cardiology$trestbps))
sum(is.na(cardiology$cholesterol))
sum(is.na(cardiology$Fasting.blood.sugar...120))
sum(is.na(cardiology$restecg))
sum(is.na(cardiology$diastbpexerc))
sum(is.na(cardiology$thalach))
sum(is.na(cardiology$exang))
sum(is.na(cardiology$oldpeak))
sum(is.na(cardiology$slope))
sum(is.na(cardiology$ca))
sum(is.na(cardiology$thal))
sum(is.na(cardiology$class))

# Part 1 Question 1 C
# Finding MAX
max(cardiology$age, na.rm=TRUE)
max(cardiology$sex, na.rm=TRUE)
max(cardiology$cp, na.rm=TRUE)
max(cardiology$trestbps, na.rm=TRUE)
max(cardiology$cholesterol, na.rm = TRUE)
max(cardiology$Fasting.blood.sugar...120, na.rm=TRUE)
max(cardiology$restecg, na.rm = TRUE)
max(cardiology$diastbpexerc, na.rm = TRUE)
max(cardiology$thalach, na.rm = TRUE)
max(cardiology$exang, na.rm = TRUE)
max(cardiology$oldpeak, na.rm=TRUE)
max(cardiology$slope, na.rm = TRUE)
max(cardiology$ca, na.rm = TRUE)
max(cardiology$thal, na.rm = TRUE)
max(cardiology$class, na.rm = TRUE)

# Part 1 Question 1 C
# Finding MIN
min(cardiology$age, na.rm=TRUE)
min(cardiology$sex, na.rm=TRUE)
min(cardiology$cp, na.rm=TRUE)
min(cardiology$trestbps, na.rm=TRUE)
min(cardiology$cholesterol, na.rm = TRUE)
min(cardiology$Fasting.blood.sugar...120, na.rm=TRUE)
min(cardiology$restecg, na.rm=TRUE)
min(cardiology$diastbpexerc, na.rm = TRUE)
min(cardiology$thalach, na.rm = TRUE)
min(cardiology$exang, na.rm=TRUE)
min(cardiology$oldpeak, na.rm=TRUE)
min(cardiology$slope, na.rm=TRUE)
min(cardiology$ca, na.rm=TRUE)
min(cardiology$thal, na.rm=TRUE)
min(cardiology$class, na.rm=TRUE)

# Part 1 Question 1 C
# Finding MEAN
mean(cardiology$age, na.rm = TRUE)
mean(cardiology$sex, na.rm = TRUE)
mean(cardiology$cp, na.rm = TRUE)
mean(cardiology$trestbps, na.rm = TRUE)
mean(cardiology$cholesterol, na.rm = TRUE)
mean(cardiology$Fasting.blood.sugar...120, na.rm = TRUE)
mean(cardiology$restecg, na.rm = TRUE)
mean(cardiology$diastbpexerc, na.rm = TRUE)
mean(cardiology$thalach, na.rm = TRUE)
mean(cardiology$exang, na.rm = TRUE)
mean(cardiology$oldpeak, na.rm = TRUE)
mean(cardiology$slope, na.rm = TRUE)
mean(cardiology$ca, na.rm = TRUE)
mean(cardiology$thal, na.rm = TRUE)
mean(cardiology$class, na.rm = TRUE)

# Part 1 Question 1 C
# Finding Mode Function
Mode <- function(cardiology){
  ux <- unique(cardiology)
  return (ux[which.max(tabulate(match(cardiology, ux)))])
}

Mode(cardiology$age)
Mode(cardiology$sex)
Mode(cardiology$cp)
Mode(cardiology$trestbps)
Mode(cardiology$cholesterol)
Mode(cardiology$Fasting.blood.sugar...120)
Mode(cardiology$restecg)
Mode(cardiology$diastbpexerc)
Mode(cardiology$thalach)
Mode(cardiology$exang)
Mode(cardiology$oldpeak)
Mode(cardiology$slope)
Mode(cardiology$ca)
Mode(cardiology$thal)
Mode(cardiology$class)

# Part 1 Question 1 C
# Finding Median
median(cardiology$age, na.rm=TRUE)
median(cardiology$sex, na.rm=TRUE)
median(cardiology$cp, na.rm=TRUE)
median(cardiology$trestbps, na.rm=TRUE)
median(cardiology$cholesterol, na.rm = TRUE)
median(cardiology$restecg, na.rm=TRUE)
median(cardiology$diastbpexerc, na.rm = TRUE)
median(cardiology$thalach, na.rm = TRUE)
median(cardiology$exang, na.rm=TRUE)
median(cardiology$oldpeak, na.rm=TRUE)
median(cardiology$slope, na.rm=TRUE)
median(cardiology$ca, na.rm=TRUE)
median(cardiology$thal, na.rm=TRUE)
median(cardiology$class, na.rm=TRUE)

# Part 1 Question 1 C
#Finding Standard Deviation
sd(cardiology$age)
sd(cardiology$sex)
sd(cardiology$cp)
sd(cardiology$trestbps)
sd(cardiology$cholesterol)
sd(cardiology$Fasting.blood.sugar...120)
sd(cardiology$restecg)
sd(cardiology$diastbpexerc)
sd(cardiology$thalach)
sd(cardiology$exang)
sd(cardiology$oldpeak)
sd(cardiology$slope)
sd(cardiology$ca)
sd(cardiology$thal)
sd(cardiology$class)

# Part 1 Question 1 D
# Shapiro-Wilks test for Normality
shapiro.test(cardiology$age)           # p-value = 0.0007914 -> Deviates From Normality
shapiro.test(cardiology$trestbps)      # p-value = 8.431e-07 -> Normal Distribution
shapiro.test(cardiology$cholesterol)   # p-value = 4.548e-09 -> Normal Distribution
shapiro.test(cardiology$diastbpexerc)  # p-value = 3.104e-07 -> Normal Distribution
shapiro.test(cardiology$thalach)       # p-value = 5.608e-05 -> Normal Distribution
# https://stats.stackexchange.com/questions/173893/interpreting-p-value-2-2e-16-in-r
# oldpeak is < 2.2e-16 means 0.00000000000000022. It is (very much) less than 0.05
shapiro.test(cardiology$oldpeak)       # p-value < 2.2e-16 -> Deviates From Normal Distribution
shapiro.test(cardiology$ca)            # p-value < 2.2e-16 -> Deviates From Normality

# Part 1 Question 1 E
# Skewness and type
skewness(cardiology$age)
skewness(cardiology$trestbps)
skewness(cardiology$cholesterol, na.rm = TRUE)
skewness(cardiology$diastbpexerc)
skewness(cardiology$thalach)
skewness(cardiology$oldpeak)
skewness(cardiology$ca)

# Part 1 Question 1 F
# 1 indicates a strong positive relationship.
# -1 indicates a strong negative relationship.
# A result of zero indicates no relationship at all.
# level of correlation for age with other predictor variables
cor(cardiology$age, cardiology$trestbps)
cor(cardiology$age, cardiology$cholesterol, use = "complete.obs")
cor(cardiology$age, cardiology$thalach)
cor(cardiology$age, cardiology$oldpeak)
cor(cardiology$age, cardiology$diastbpexerc)

# level of correlation for trestbps with other predictor variables
cor(cardiology$trestbps, cardiology$age)
cor(cardiology$trestbps, cardiology$cholesterol, use = "complete.obs")
cor(cardiology$trestbps, cardiology$thalach)
cor(cardiology$trestbps, cardiology$oldpeak)
cor(cardiology$trestbps, cardiology$diastbpexerc)

# level of correlation for diastbpexerc with other predictor variables
cor(cardiology$diastbpexerc, cardiology$age)
cor(cardiology$diastbpexerc, cardiology$trestbps)
cor(cardiology$diastbpexerc, cardiology$cholesterol, use = "complete.obs")
cor(cardiology$diastbpexerc, cardiology$thalach)
cor(cardiology$diastbpexerc, cardiology$oldpeak)

# level of correlation for thalach with other predictor variables
cor(cardiology$thalach, cardiology$age)
cor(cardiology$thalach, cardiology$trestbps)
cor(cardiology$thalach, cardiology$cholesterol, use = "complete.obs")
cor(cardiology$thalach, cardiology$diastbpexerc)
cor(cardiology$thalach, cardiology$oldpeak)

# level of correlation for oldpeak with other predictor variables
cor(cardiology$oldpeak, cardiology$age)
cor(cardiology$oldpeak, cardiology$trestbps)
cor(cardiology$oldpeak, cardiology$cholesterol, use = "complete.obs")
cor(cardiology$oldpeak, cardiology$diastbpexerc)
cor(cardiology$oldpeak, cardiology$thalach)

# level of correlation  for cholesterol with other predictor variables
cor(cardiology$cholesterol, cardiology$age, use = "complete.obs")
cor(cardiology$cholesterol, cardiology$trestbps, use = "complete.obs")
cor(cardiology$cholesterol, cardiology$thalach, use = "complete.obs")
cor(cardiology$cholesterol, cardiology$oldpeak, use = "complete.obs")
cor(cardiology$cholesterol, cardiology$diastbpexerc, use = "complete.obs")

# level of correlation for ca with other predictor variables
cor(cardiology$ca, cardiology$age)
cor(cardiology$ca, cardiology$trestbps)
cor(cardiology$ca, cardiology$cholesterol)
cor(cardiology$ca, cardiology$cholesterol, use = "complete.obs")
cor(cardiology$ca, cardiology$diastbpexerc)
cor(cardiology$ca, cardiology$thalach)

# Part 1 Question 2
# Histogram for each numerical variable, with an overlay of the target variable
# Histogram for Age
ggplot(cardiology, aes(x = age, fill = class)) + geom_histogram() + ggtitle("Age of patients") + labs(x =
"Age of patients", y = "Number of patients") + theme_bw()

# Histogram for Trestbps (Resting blood pressure of patients)
ggplot(cardiology, aes(x = trestbps, fill = class)) + geom_histogram() + ggtitle("Resting blood pressure
of patients") + labs(x = "Resting blood pressure of patients", y = "Number of patients") + theme_bw()

# Histogram for Cholesterol
ggplot(cardiology, aes(x = cholesterol, fill = class)) + geom_histogram() + ggtitle("Cholesterol of
patients") + labs(x = "Cholesterol of patients", y = "Number of patients") + theme_bw()

# Histogram for Diastbpexerc (Diastolic blood pressure of patients)
ggplot(cardiology, aes(x = diastbpexerc, fill = class)) + geom_histogram() + ggtitle("Diastolic blood
pressure of patients") + labs(x = "Diastolic blood pressure of patients", y = "Number of patients") +
theme_bw()

# Histogram for Thalach (Maximum heart rate achieved for patients)
ggplot(cardiology, aes(x = thalach, fill = class)) + geom_histogram() + ggtitle("Maximum heart rate
achieved for patients") + labs(x = "Maximum heart rate achieved for patients", y = "Number of
patients") + theme_bw()

# Histogram for Oldpeak (ST depression induced by exercise relative to rest of patients)
```

```
ggplot(cardiology, aes(x = oldpeak, fill = class)) + geom_histogram() + ggtitle("ST depression induced by exercise relative to rest of patients") + labs(x = "ST depression induced by exercise relative to rest of patients", y = "Number of patients") + theme_bw()

# Histogram for ca (Number of major vessels)
ggplot(cardiology, aes(x = ca, fill = class)) + geom_histogram() + ggtitle("Number of major vessels") + labs(x = "ST depression induced by exercise relative to rest of patients", y = "Number of patients") + theme_bw()

# Part 1 Question 3
# BarChart for each categorical variable,with an overlay of the target variable
# Bar chart for each sex
ggplot(cardiology, aes(x = sex, fill = class)) + geom_bar() + ggtitle("Sex of patients") + labs(x = "Sex of patient", y = "Number of patients") + theme_bw()

# Bar chart for each cp
ggplot(cardiology, aes(x = cp, fill = class)) + geom_bar() + ggtitle("Chest pain type") + labs(x = "Chest pain type", y = "Number of patients") + theme_bw()

# Bar chart for each Fasting blood sugar
ggplot(cardiology, aes(x = Fasting.blood.sugar...120, fill = class)) + geom_bar() + ggtitle("Fasting blood sugar") + labs(x = "Fasting blood sugar", y = "Number of patients") + theme_bw()

# Bar chart for each restecg
ggplot(cardiology, aes(x = restecg, fill = class)) + geom_bar() + ggtitle("Resting electrocardiographic results") + labs(x = "Resting electrocardiographic results", y = "Number of patients") + theme_bw()

# Bar chart for each exang
ggplot(cardiology, aes(x = exang, fill = class)) + geom_bar() + ggtitle("Exercise induced angina") + labs(x = "Exercise induced angina ", y = "Number of patients") + theme_bw()

# Bar chart for each slope
ggplot(cardiology, aes(x = slope, fill = class)) + geom_bar() + ggtitle("The slope of the peak exercise ST segment") + labs(x = "The slope of the peak exercise ST segment", y = "Number of patients") + theme_bw()

# Bar chart for each thal
ggplot(cardiology, aes(x = thal, fill = class)) + geom_bar() + ggtitle("Thal") + labs(x = "Thal", y = "Number of patients") + theme_bw()

# Part 1, Question 4:
# Detect outliers in numerical variables
cardiologyNumeric <- read.table(file="C:/RCA1/CardiologyNumeric.csv", stringsAsFactors=FALSE, sep=",", header=TRUE)

# Graphical means of finding outliers.
# Box Plot of Age vs. Age.
ggplot(cardiology, aes(x = age, y = classNumeric)) + geom_boxplot()

# Box Plot of Trestbps vs. Trestbps.
ggplot(cardiologyNumeric, aes(x = trestbps, y = trestbps)) + geom_boxplot()
```

```
# Box Plot of Cholesterol vs. Cholesterol.
ggplot(cardiologyNumeric, aes(x = cholesterol, y = cholesterol)) + geom_boxplot()

# Box Plot of Diastbpexercvs. Diastbpexerc.
ggplot(cardiologyNumeric, aes(x = diastbpexerc, y = diastbpexerc)) + geom_boxplot()

# Box Plot of Thalach vs. Thalach.
ggplot(cardiologyNumeric, aes(x = thalach, y = thalach)) + geom_boxplot()

# Box Plot of Oldpeak vs. Oldpeak.
ggplot(cardiologyNumeric, aes(x = oldpeak, y = oldpeak)) + geom_boxplot()

# Box Plot of Ca vs. Ca.
ggplot(cardiologyNumeric, aes(x = ca, y = ca)) + geom_boxplot()

# Statistical means of finding outliers.
# Made this function to find the outliers of a numeric column.
# This takes in the attribute and the column number in order to find and print out the outliers.
# It finds the IQR using the same methods we did in class by getting Q3 - Q1.
# It then gets the upper and lower bounds (-1.5 IQR and 1.5 IQR) and displays these bounds to the
screen.
# Lastly, it displays all the numeric data outliers from that attribute that are
# greater than the upper bound and less than the lower bound.
IQRangeOutliers <- function(arg1, column){
  # Get your IQR (Interquartile range) and lower/upper quartile using:
  lowerq = quantile(arg1)[2]
  upperq = quantile(arg1)[4]
  iqr = upperq - lowerq #Or use IQR(data)

  # Compute the bounds for a mild outlier:
  threshold.upper = (iqr * 1.5) + upperq
  threshold.lower = lowerq - (iqr * 1.5)

  # Any age less than the threshold.lower value is an outlier
  threshold.lower
  # Any age greater than the threshold.upper value is an outlier
  threshold.upper

  print(paste0("Lower outlier bound: ", threshold.lower))
  print(paste0("Upper outlier bound: ", threshold.upper))

  GetUpperOutliers <-
cardiologyNumeric[which(cardiologyNumeric[,column]>threshold.upper),column]
  print(paste0("Upper outliers:"))
  print(GetUpperOutliers)
  # IF statement to deal with when there are no Upper outliers integer(0)

  GetLowerOutliers <-
cardiologyNumeric[which(cardiologyNumeric[,column]<threshold.lower),column]
  print(paste0("Lower Outliers:"))
  print(GetLowerOutliers)
  # IF statement to deal with when there are no Lower outliers integer(0)
}

# IQR outliers for all numeric attributes.
IQRangeOutliers(cardiologyNumeric$age, 1)
IQRangeOutliers(cardiologyNumeric$trestbps,2)
IQRangeOutliers(cardiologyNumeric$cholesterol,3)
IQRangeOutliers(cardiologyNumeric$diastbpexerc, 4)
IQRangeOutliers(cardiologyNumeric$thalach, 5)
IQRangeOutliers(cardiologyNumeric$oldpeak, 6)
IQRangeOutliers(cardiologyNumeric$ca, 7)

# Z-Score function to detect outliers
zscoreOutliers <- function(arg1){
  # Find Z-score for argument passed in.
  zscoreOutlier <- (arg1 - mean(arg1, na.rm = TRUE)) / sd(arg1)

  # Find upper Z-score outliers (Any score greater than 3).
  GetUpperZscoreOutliers <-  zscoreOutlier[which(zscoreOutlier > 3)]
  print(paste0("Upper Z-Score outliers:"))
  print(GetUpperZscoreOutliers)

  # Find lower Z-score outliers (Any score less than 3).
  GetLowerZscoreOutliers <- zscoreOutlier[which(zscoreOutlier < -3)]
  print(paste0("Lower Z-Score outliers:"))
  print(GetLowerZscoreOutliers)
}

# Generate z-score outliers for all numeric attributes.
zscoreOutliers(cardiologyNumeric$age)
zscoreOutliers(cardiologyNumeric$trestbps)
zscoreOutliers(cardiologyNumeric$cholesterol)
zscoreOutliers(cardiologyNumeric$diastbpexerc)
zscoreOutliers(cardiologyNumeric$thalach)
zscoreOutliers(cardiologyNumeric$oldpeak)
zscoreOutliers(cardiologyNumeric$ca)

# Part 1 Question 5 A
# Investigate whether there are any correlated variables. Using Scatter Plots
# Age plots
ggplot(cardiology, aes(x=age, y=trestbps)) + geom_point()
ggplot(cardiology, aes(x=age, y=cholesterol)) + geom_point()
ggplot(cardiology, aes(x=age, y=thalach)) + geom_point()
ggplot(cardiology, aes(x=age, y=oldpeak)) + geom_point()
ggplot(cardiology, aes(x=age, y=diastbpexerc)) + geom_point()

# trestbps plots
ggplot(cardiology, aes(x=trestbps, y=age)) + geom_point()
ggplot(cardiology, aes(x=trestbps, y=cholesterol)) + geom_point()
ggplot(cardiology, aes(x=trestbps, y=thalach)) + geom_point()
ggplot(cardiology, aes(x=trestbps, y=oldpeak)) + geom_point()
ggplot(cardiology, aes(x=trestbps, y=diastbpexerc)) + geom_point()

ggplot(cardiology, aes(x=ca, y=age)) + geom_point()

# diastbpexerc plots
ggplot(cardiology, aes(x=diastbpexerc, y=age)) + geom_point()
ggplot(cardiology, aes(x=diastbpexerc, y=trestbps)) + geom_point()
ggplot(cardiology, aes(x=diastbpexerc, y=cholesterol)) + geom_point()
ggplot(cardiology, aes(x=diastbpexerc, y=thalach)) + geom_point()
ggplot(cardiology, aes(x=diastbpexerc, y=oldpeak)) + geom_point()

# thalach plots
ggplot(cardiology, aes(x=thalach, y=age)) + geom_point()
ggplot(cardiology, aes(x=thalach, y=trestbps)) + geom_point()
ggplot(cardiology, aes(x=thalach, y=cholesterol)) + geom_point()
ggplot(cardiology, aes(x=thalach, y=diastbpexerc)) + geom_point()
ggplot(cardiology, aes(x=thalach, y=oldpeak)) + geom_point()

# oldpeak plots
ggplot(cardiology, aes(x=oldpeak, y=age)) + geom_point()
ggplot(cardiology, aes(x=oldpeak, y=trestbps)) + geom_point()
ggplot(cardiology, aes(x=oldpeak, y=cholesterol)) + geom_point()
ggplot(cardiology, aes(x=oldpeak, y=diastbpexerc)) + geom_point()
ggplot(cardiology, aes(x=oldpeak, y=thalach)) + geom_point()

# cholesterol plots
ggplot(cardiology, aes(x=cholesterol, y=age)) + geom_point()
ggplot(cardiology, aes(x=cholesterol, y=trestbps)) + geom_point()
ggplot(cardiology, aes(x=cholesterol, y=oldpeak)) + geom_point()
ggplot(cardiology, aes(x=cholesterol, y=diastbpexerc)) + geom_point()
ggplot(cardiology, aes(x=cholesterol, y=thalach)) + geom_point()

# Part 1 Question 5 B Verifying assertions
# level of correlation for age with other predictor variables
cor(cardiology$age, cardiology$trestbps)
cor(cardiology$age, cardiology$cholesterol, use = "complete.obs")
cor(cardiology$age, cardiology$thalach)
cor(cardiology$age, cardiology$oldpeak)
cor(cardiology$age, cardiology$diastbpexerc)

# level of correlation for trestbps with other predictor variables
cor(cardiology$trestbps, cardiology$age)
cor(cardiology$trestbps, cardiology$cholesterol, use = "complete.obs")
cor(cardiology$trestbps, cardiology$thalach)
cor(cardiology$trestbps, cardiology$oldpeak)
cor(cardiology$trestbps, cardiology$diastbpexerc)

# level of correlation for diastbpexerc with other predictor variables
cor(cardiology$diastbpexerc, cardiology$age)
cor(cardiology$diastbpexerc, cardiology$trestbps)
cor(cardiology$diastbpexerc, cardiology$cholesterol, use = "complete.obs")
cor(cardiology$diastbpexerc, cardiology$thalach)
cor(cardiology$diastbpexerc, cardiology$oldpeak)

# level of correlation for thalach with other predictor variables
cor(cardiology$thalach, cardiology$age)
cor(cardiology$thalach, cardiology$trestbps)
cor(cardiology$thalach, cardiology$cholesterol, use = "complete.obs")
cor(cardiology$thalach, cardiology$diastbpexerc)
cor(cardiology$thalach, cardiology$oldpeak)

# level of correlation for oldpeak with other predictor variables
cor(cardiology$oldpeak, cardiology$age)
cor(cardiology$oldpeak, cardiology$trestbps)
cor(cardiology$oldpeak, cardiology$cholesterol, use = "complete.obs")
cor(cardiology$oldpeak, cardiology$diastbpexerc)
cor(cardiology$oldpeak, cardiology$thalach)

# level of correlation for cholesterol with other predictor variables
cor(cardiology$cholesterol, cardiology$age, use = "complete.obs")
cor(cardiology$cholesterol, cardiology$trestbps, use = "complete.obs")
cor(cardiology$cholesterol, cardiology$thalach, use = "complete.obs")
cor(cardiology$cholesterol, cardiology$oldpeak, use = "complete.obs")
cor(cardiology$cholesterol, cardiology$diastbpexerc, use = "complete.obs")

# Part 1 Question 5 C,
# Age plots
ggplot(cardiology, aes(x=age, y=trestbps, color = class)) + geom_point() + geom_smooth(method = "lm", se = FALSE)
ggplot(cardiology, aes(x=age, y=cholesterol, color = class)) + geom_point() + geom_smooth(method = "lm", se = FALSE)
ggplot(cardiology, aes(x=age, y=thalach, color = class)) + geom_point() + geom_smooth(method = "lm", se = FALSE)
ggplot(cardiology, aes(x=age, y=oldpeak, color = class)) + geom_point() + geom_smooth(method = "lm", se = FALSE)
ggplot(cardiology, aes(x=age, y=diastbpexerc, color = class)) + geom_point() + geom_smooth(method = "lm", se = FALSE)

# trestbps plots
ggplot(cardiology, aes(x=trestbps, y=age, color = class)) + geom_point() + geom_smooth(method = "lm", se = FALSE)
ggplot(cardiology, aes(x=trestbps, y=cholesterol, color = class)) + geom_point() + geom_smooth(method = "lm", se = FALSE)
ggplot(cardiology, aes(x=trestbps, y=thalach, color = class)) + geom_point() + geom_smooth(method = "lm", se = FALSE)
ggplot(cardiology, aes(x=trestbps, y=oldpeak, color = class)) + geom_point() + geom_smooth(method = "lm", se = FALSE)
ggplot(cardiology, aes(x=trestbps, y=diastbpexerc, color = class)) + geom_point() + geom_smooth(method = "lm", se = FALSE)

# diastbpexerc plots
```

```
ggplot(cardiology, aes(x=diastbpexerc, y=age, color = class)) + geom_point() +
geom_smooth(method = "lm", se = FALSE)
ggplot(cardiology, aes(x=diastbpexerc, y=trestbps, color = class)) + geom_point() +
geom_smooth(method = "lm", se = FALSE)
ggplot(cardiology, aes(x=diastbpexerc, y=cholesterol, color = class)) + geom_point() +
geom_smooth(method = "lm", se = FALSE)
ggplot(cardiology, aes(x=diastbpexerc, y=thalach, color = class)) + geom_point() +
geom_smooth(method = "lm", se = FALSE)
ggplot(cardiology, aes(x=diastbpexerc, y=oldpeak, color = class)) + geom_point() +
geom_smooth(method = "lm", se = FALSE)

# thalach plots
ggplot(cardiology, aes(x=thalach, y=age, color = class)) + geom_point() + geom_smooth(method =
"lm", se = FALSE)
ggplot(cardiology, aes(x=thalach, y=trestbps, color = class)) + geom_point() + geom_smooth(method
= "lm", se = FALSE)
ggplot(cardiology, aes(x=thalach, y=cholesterol, color = class)) + geom_point() +
geom_smooth(method = "lm", se = FALSE)
ggplot(cardiology, aes(x=thalach, y=diastbpexerc, color = class)) + geom_point() +
geom_smooth(method = "lm", se = FALSE)
ggplot(cardiology, aes(x=thalach, y=oldpeak, color = class)) + geom_point() + geom_smooth(method
= "lm", se = FALSE)

# oldpeak plots
ggplot(cardiology, aes(x=oldpeak, y=age, color = class)) + geom_point() + geom_smooth(method =
"lm", se = FALSE)
ggplot(cardiology, aes(x=oldpeak, y=trestbps, color = class)) + geom_point() + geom_smooth(method
= "lm", se = FALSE)
ggplot(cardiology, aes(x=oldpeak, y=cholesterol, color = class)) + geom_point() +
geom_smooth(method = "lm", se = FALSE)
ggplot(cardiology, aes(x=oldpeak, y=diastbpexerc, color = class)) + geom_point() +
geom_smooth(method = "lm", se = FALSE)
ggplot(cardiology, aes(x=oldpeak, y=thalach, color = class)) + geom_point() + geom_smooth(method
= "lm", se = FALSE)

# cholesterol plots
ggplot(cardiology, aes(x=cholesterol, y=age, color = class)) + geom_point() + geom_smooth(method
= "lm", se = FALSE)
ggplot(cardiology, aes(x=cholesterol, y=trestbps, color = class)) + geom_point() +
geom_smooth(method = "lm", se = FALSE)
ggplot(cardiology, aes(x=cholesterol, y=oldpeak, color = class)) + geom_point() +
geom_smooth(method = "lm", se = FALSE)
ggplot(cardiology, aes(x=cholesterol, y=diastbpexerc, color = class)) + geom_point() +
geom_smooth(method = "lm", se = FALSE)
ggplot(cardiology, aes(x=cholesterol, y=thalach, color = class)) + geom_point() +
geom_smooth(method = "lm", se = FALSE)

# Part 2 Question 7.
# Using one numeric variable which is skewed (Thalach (Negatively Skewed)),
# transform the data using the following methods to achieve normailty:
#   A.) Z-Score Standardisation.
#   B.) Natural Log Transformation.
#   C.) Square Root Transformation.
#   D.) Inverse Square Roor Transformation.

# A.) Z Score Standardisation.
zscoreMinThalach <- (min(cardiologyNumeric$thalach) - mean(cardiologyNumeric$thalach)) /
sd(cardiologyNumeric$thalach)
zscoreMinThalach
# Data values below the mean will have a negative z-score standardisation.

zscoreRangeThalach <- ((max(cardiologyNumeric$thalach) - min(cardiologyNumeric$thalach)) -
mean(cardiologyNumeric$thalach)) / sd(cardiologyNumeric$thalach)
zscoreRangeThalach
# Values falling on the mean will have zero (0) z score. (negative in this case)

zscoreMaxThalach <- (max(cardiologyNumeric$thalach) - mean(cardiologyNumeric$thalach)) /
sd(cardiologyNumeric$thalach)
zscoreMaxThalach
# Data values above the mean will have a positive Z-score standardisation.



# B.) Natural Log Transformation.
naturalLogMinThalach <- log(min(cardiologyNumeric$thalach))
naturalLogMinThalach
# (4.26268) Data values below the mean will have a

naturalLogRangeThalach <- log(max(cardiologyNumeric$thalach) - min(cardiologyNumeric$thalach))
naturalLogRangeThalach
# (4.875197) Values falling on the mean will have a

naturalLogMaxThalach <- log(max(cardiologyNumeric$thalach))
naturalLogMaxThalach
# (5.308268) Data values above the mean will have a



# C.) Square Root Transformation.
sqrtMinThalach <- sqrt(min(cardiologyNumeric$thalach))
sqrtMinThalach
# (8.42615) Data values below the mean will have a

sqrtRangeThalach <- sqrt((max(cardiologyNumeric$thalach) - min(cardiologyNumeric$thalach)))
sqrtRangeThalach
# (11.44552) Values falling on the mean will have

sqrtMaxThalach <- sqrt(max(cardiologyNumeric$thalach))
sqrtMaxThalach
# (14.21267) Data values above the mean will have a
# D.) Inverse Square Root Transformation.
invSqrtMinThalach <- 1/(sqrt(min(cardiologyNumeric$thalach)))
invSqrtMinThalach
# (0.1186782) Data values below the mean will have a

invSqrtRangeThalach <- 1/(sqrt((max(cardiologyNumeric$thalach) -
min(cardiologyNumeric$thalach))))
invSqrtRangeThalach
# (0.08737041) Values falling on the mean will have

invSqrtMaxThalach <- 1/(sqrt(max(cardiologyNumeric$thalach)))
invSqrtMaxThalach
# (0.07035975) Data values above the mean will have a

# Part 2 Question 8
# converting string to numeric
as.numeric(cardiology$cholesterol)


# Converting class to classNumeric 1 || 0
cardiology$classNumeric[cardiology$class=="Sick"]<-"0"
cardiology$classNumeric[cardiology$class=="Healthy"]<-"1"
```