classification of Diseased Heart

Harsha Vardhan Seelam

11/10/2020

```
## 'data.frame':    303 obs. of  14 variables:
##  $ ï..age  : int  63 37 41 56 57 57 56 44 52 57 ...
##  $ sex     : int  1 1 0 1 0 1 0 1 1 1 ...
##  $ cp      : int  3 2 1 1 0 0 1 1 2 2 ...
##  $ trestbps: int  145 130 130 120 120 140 140 120 172 150 ...
##  $ chol    : int  233 250 204 236 354 192 294 263 199 168 ...
##  $ fbs     : int  1 0 0 0 0 0 0 0 1 0 ...
##  $ restecg : int  0 1 0 1 1 1 0 1 1 1 ...
##  $ thalach : int  150 187 172 178 163 148 153 173 162 174 ...
##  $ exang   : int  0 0 0 0 1 0 0 0 0 0 ...
##  $ oldpeak : num  2.3 3.5 1.4 0.8 0.6 0.4 1.3 0 0.5 1.6 ...
##  $ slope   : int  0 0 2 2 2 1 1 2 2 2 ...
##  $ ca      : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ thal    : int  1 2 2 2 2 1 2 3 3 2 ...
##  $ target  : int  1 1 1 1 1 1 1 1 1 1 ...
```

The ultimate goal of this task which is classification, is to know the presence of heart disease in the patient. The dataset consists the data of 303 patients with 14 attributes in which the last column represents the whether the patient is suffering with heart disease or not. As you can see in the Vtable above, except the oldpeak attribute, remaining all other attributes are represented as integers.

According to the values in the vtable, except for the integer attributes like age, trestbps(resting blood pressure), chol(serum cholestoral), thalach (maximum heart rate achieved) and numeric attribute oldpeak, we need to convert the remaining attributes like sex, target, fasting blood pressure and etc., to factors. This has to be done due to their nature of representation, each factor in a single attribute might help our model to classify data precisely. A visualization of this can be seen in next section.

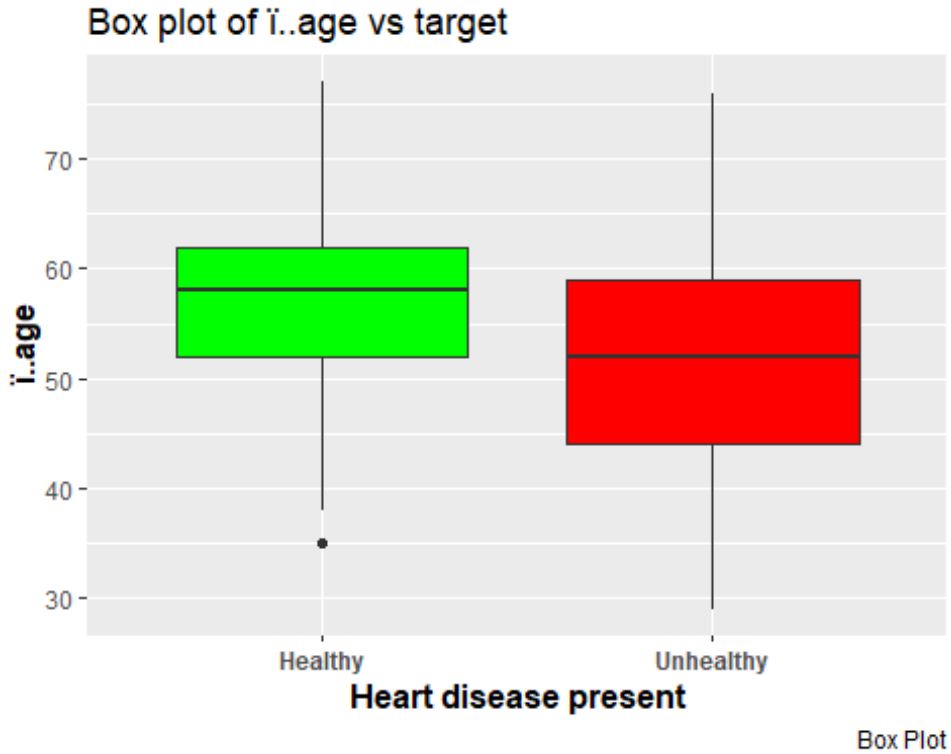
```
## 'data.frame':    303 obs. of  14 variables:
##  $ ï..age  : int  63 37 41 56 57 57 56 44 52 57 ...
##  $ sex     : Factor w/ 2 levels "0","1": 2 2 1 2 1 2 1 2 2 2 ...
##  $ cp      : Factor w/ 4 levels "0","1","2","3": 4 3 2 2 1 1 2 2 3 3 ...
##  $ trestbps: int  145 130 130 120 120 140 140 120 172 150 ...
##  $ chol    : int  233 250 204 236 354 192 294 263 199 168 ...
##  $ fbs     : Factor w/ 2 levels "0","1": 2 1 1 1 1 1 1 1 2 1 ...
##  $ restecg : Factor w/ 3 levels "0","1","2": 1 2 1 2 2 2 1 2 2 2 ...
##  $ thalach : int  150 187 172 178 163 148 153 173 162 174 ...
##  $ exang   : Factor w/ 2 levels "0","1": 1 1 1 1 2 1 1 1 1 1 ...
##  $ oldpeak : num  2.3 3.5 1.4 0.8 0.6 0.4 1.3 0 0.5 1.6 ...
##  $ slope   : Factor w/ 3 levels "0","1","2": 1 1 3 3 3 2 2 3 3 3 ...
```

```
##  $ ca      : Factor w/ 5 levels "0","1","2","3",..: 1 1 1 1 1 1 1 1 1 1
...
##  $ thal    : Factor w/ 4 levels "fixed_defect",..: 2 1 1 1 1 2 1 4 4 1 ...
##  $ target  : Factor w/ 2 levels "Healthy","Unhealthy": 2 2 2 2 2 2 2 2 2 2
...
```

visualization

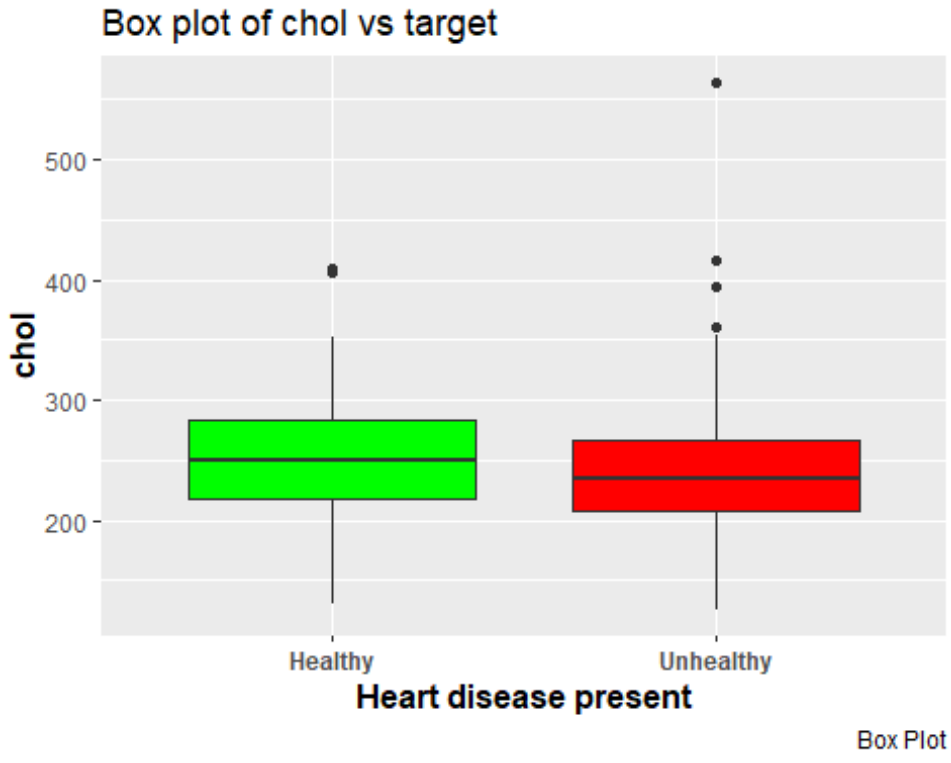
The primary goal of our visualization is to understand which features are applicable for separating our class into heart disease present or not.

Firstly, lets check the Numeric and integer features in our dataset using Box plot.I choose to use box plot because, they are really good at representing whether the distribution of data is skewed and highlight the outliers of the dataset.

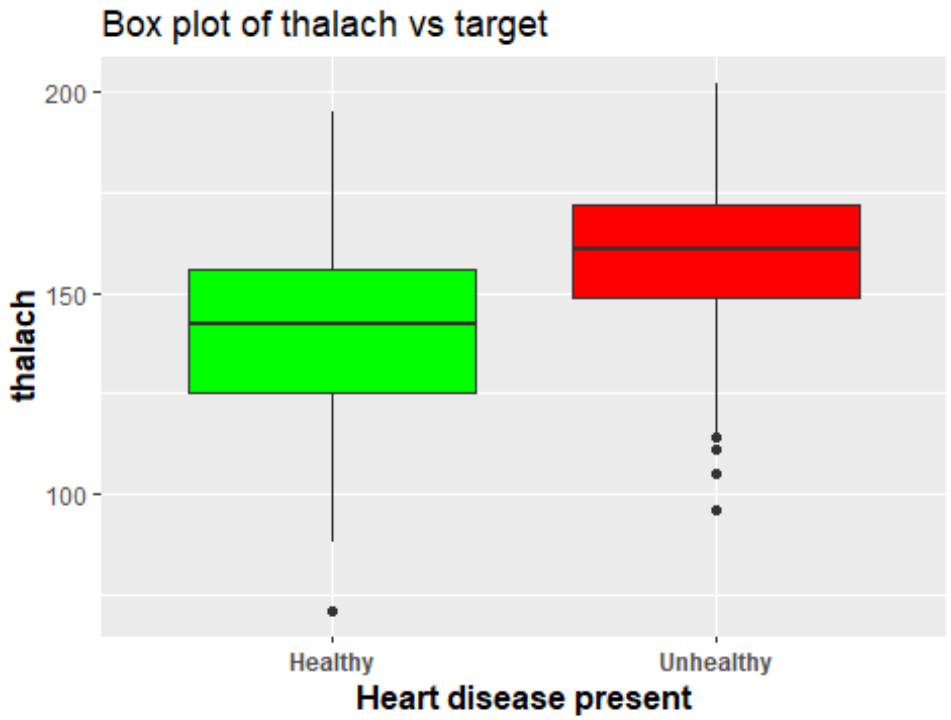


Box Plot

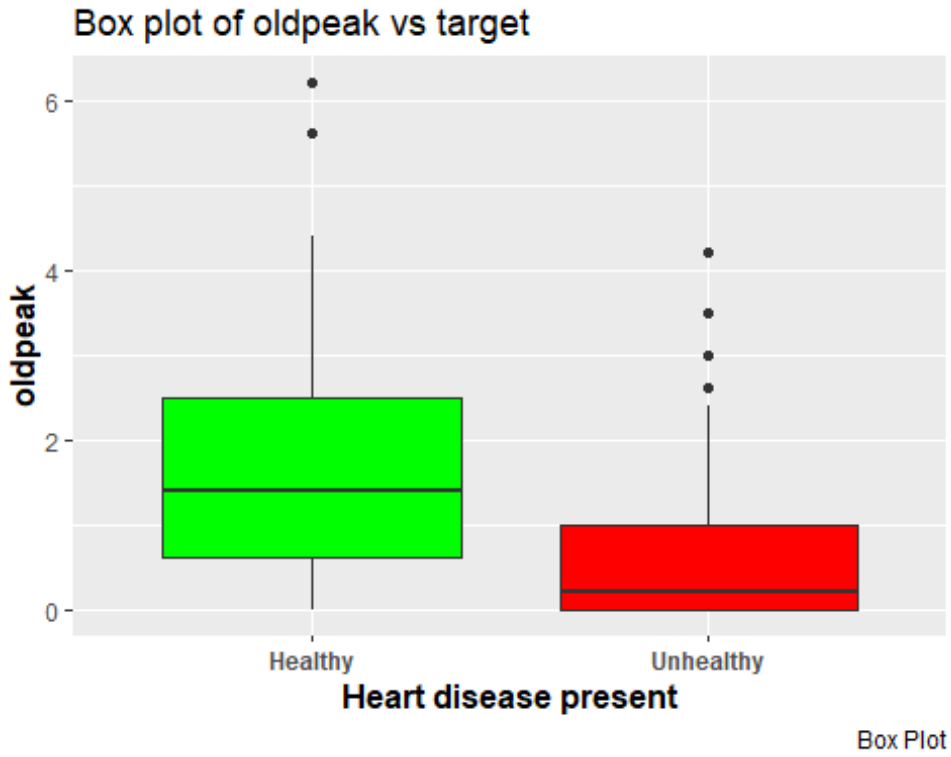
Box Plot



Box Plot



Box Plot



Box Plot

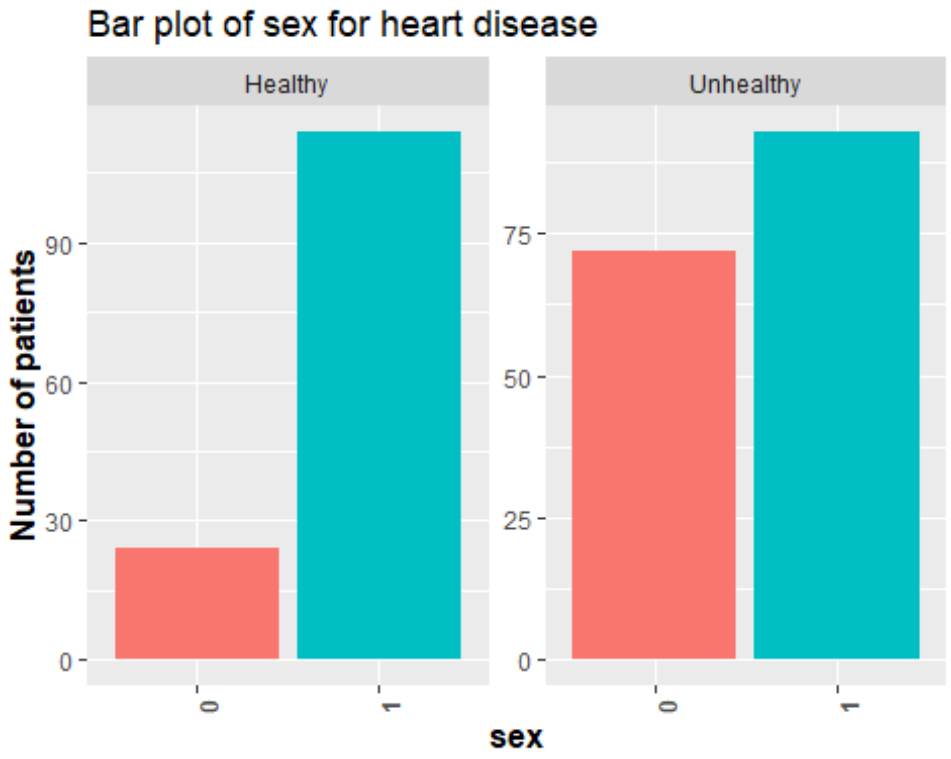
For each graph represented above for the attributes age, trestbps, chol, thalach, oldpeak., each graph green box represents healthy people and red box represents unhealthy people in that particular attribute. In addition, Y-axis represents the min to max values in that attribute. However, In each box plot there are three main things which represent the data. Firstly, the box represents 50% of the data where the median line which is in the middle separates the 50% data into 25% each. If the median line is squeeze towards the top or bottom of the box, it means that more no of data points are lying around in that area. Secondly, the skewer lines which is on the top and bottom of the box represents 25% of the data each. But, a small skewer lines usually represents that there are less number of data points lying around them, when compared to the box. Finally, the black dots which might be above or below the skewer lines represents the outliers in that particular attributes.

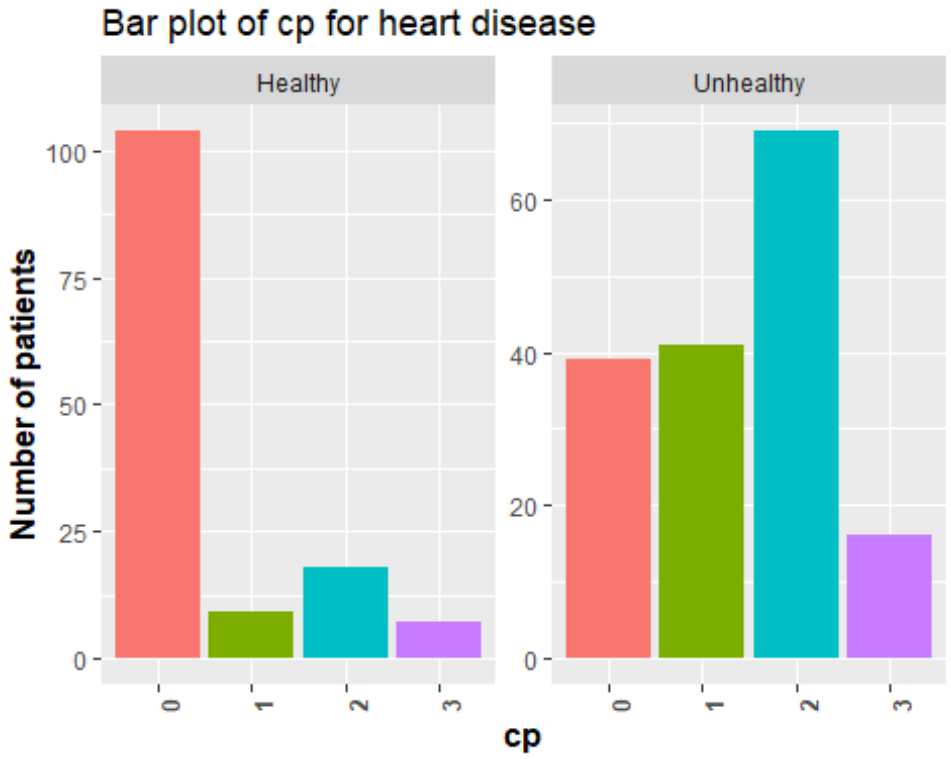
For example, the box plot of oldpeak vs target, there are so many outliers in unhealthy people category. And addition, the median line is squeezing down to the box, which means that, the data points the edge of the box and median line are more prone to have a heart disease. And also, there is no skewer line below the box so, it says that there no data points which represent that area.

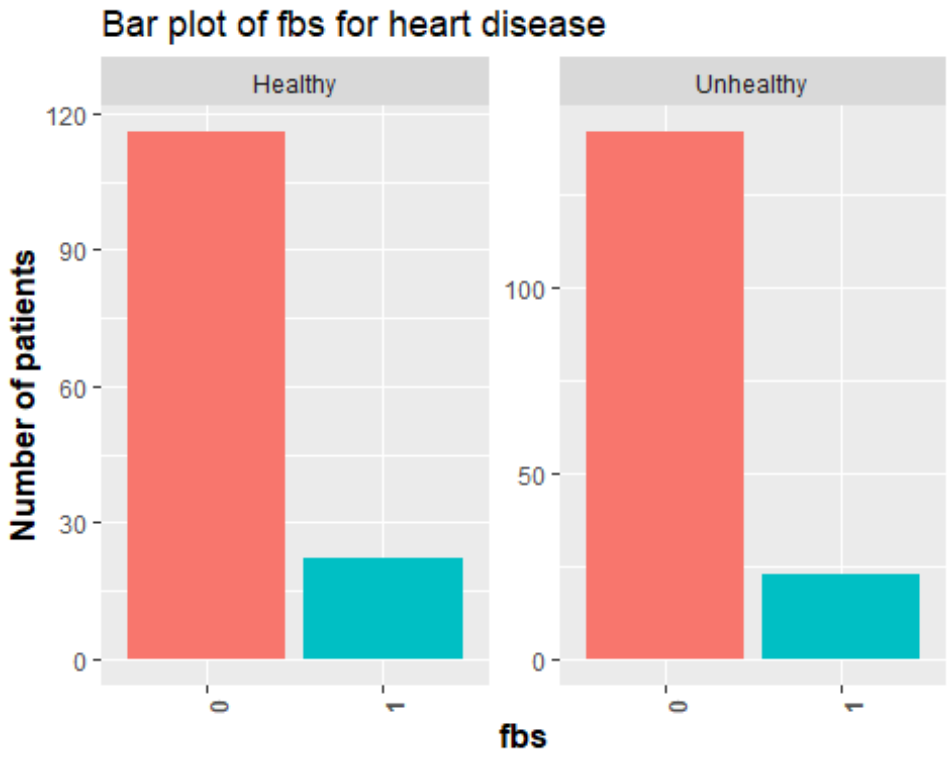
From looking at the above boxplots, we can say that, the data points represents a good distribution to seperate the class of healthy or unhealthy. So i came to a conclusion that, all the numeric and integer features are playing an important role in seperating the class between healthy and unhealthy.

visualization of categorical features.

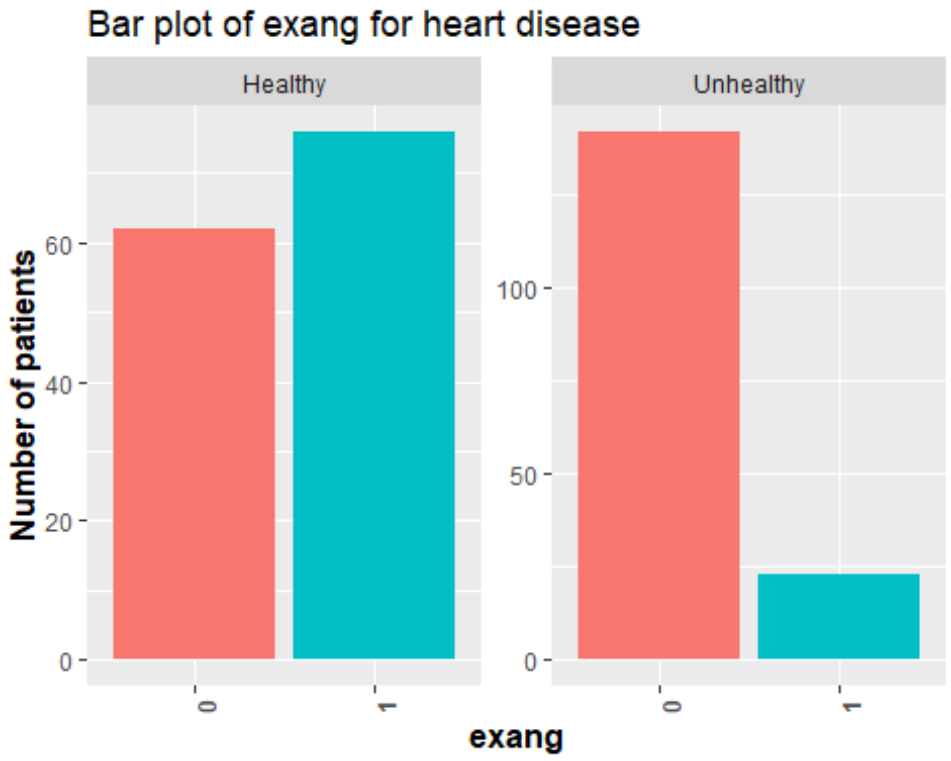
Features like sex, cp, fbs, restecg, exang, slope, ca and thal., are categorical variables where, within each feature, the values are limited and are based on a finite group which is represented in a one single feature. To deal with this kind of features, i choose to use bar plots. Bar plots can be used to represent label's proportion in an categorical feature. We can use this to plot what each label in categorical feature is contributing towards representing a class. We can plot both class cases and draw conclusions.


Bar plot of sex for heart disease
Healthy
Unhealthy
Number of patients
90
60
30
0
75
50
25
0
0
1
0
1
sex

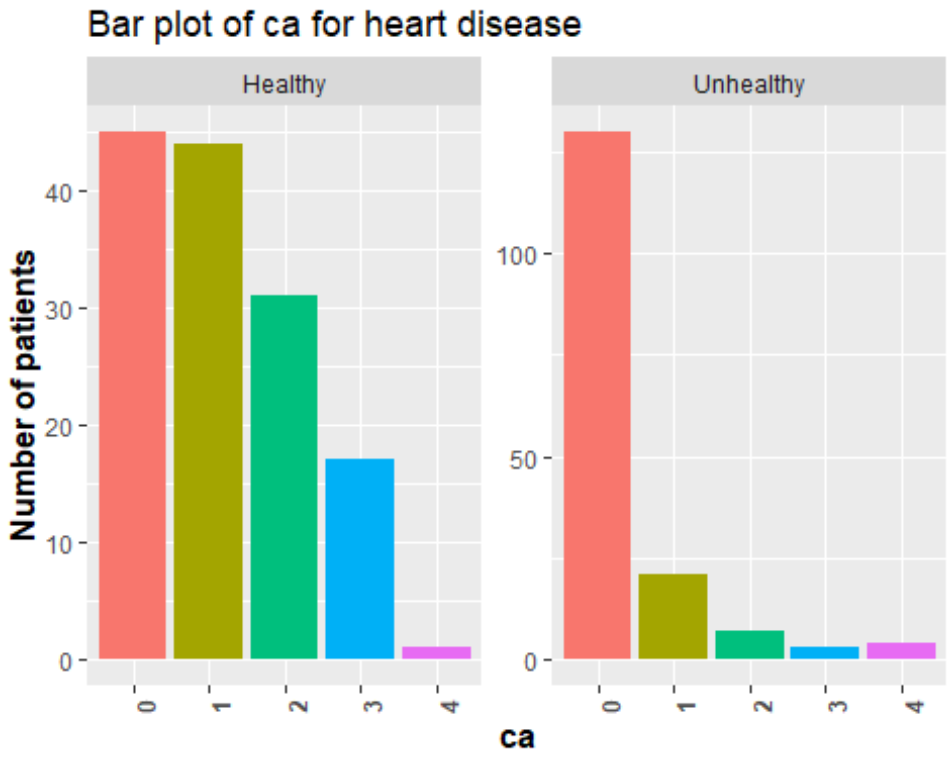

Bar plot of cp for heart disease
Healthy
Unhealthy
Number of patients
100
75
50
25
0
60
40
20
0
0
1
2
3
0
1
2
3
cp

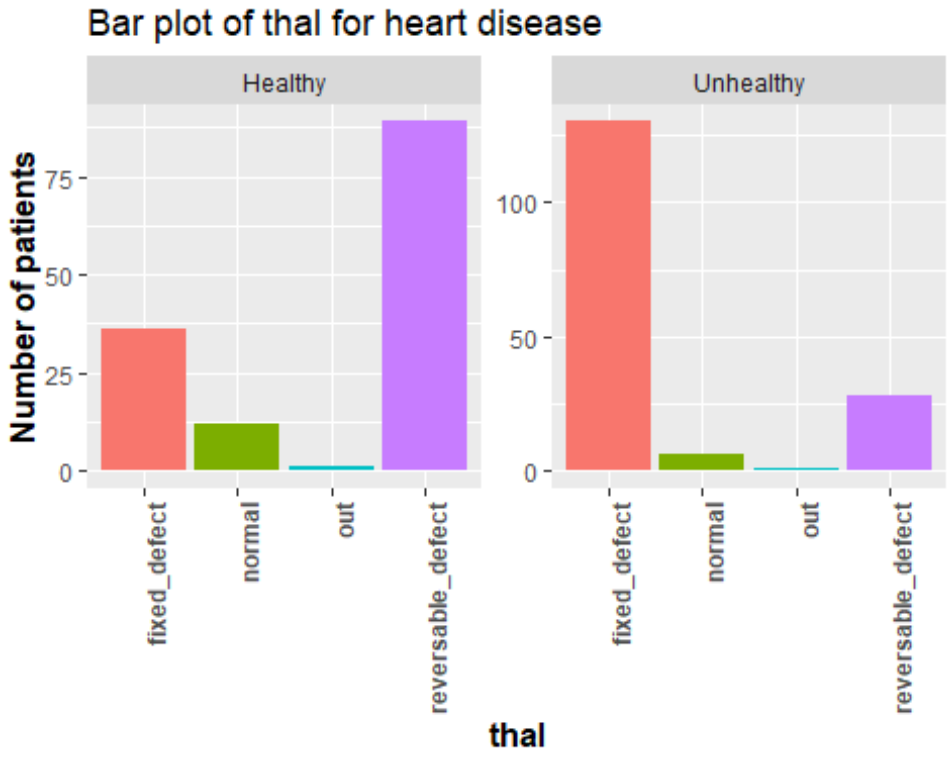

Bar plot of fbs for heart disease
Healthy
Unhealthy
Number of patients
120
90
60
30
0
100
50
0
0
1
fbs


Bar plot of restecg for heart disease
Healthy
Unhealthy
Number of patients
80
60
40
20
0
100
75
50
25
0
0
1
2
restecg


Bar plot of exang for heart disease
Healthy
Unhealthy
Number of patients
0
20
40
60
0
50
100
0
1
0
1
exang


Bar plot of slope for heart disease
Healthy
Unhealthy
Number of patients
0
25
50
75
0
30
60
90
0
1
2
0
1
2
slope





For each bar plots above we need to interpret the plots by comparing the proportions of the categories for each of the label values. if the proportions are distinctly different for each label, the feature is likely to be useful. If each proportion is similar, the feature is most likely that it will not be useful to classify the data.

There are few things to be noted from this plots: 1. some Features like sex, cp, exang, ca and thal have significantly different distribution of categories between the label categories. 2. Few other features like fbs show a small differences but these differences are unlikely to be significant. 3. Using these plots i've noticed that, In features thal and ca labels values with 0 in thal and 4 in ca are not a part of dataset and their contribution towards classifying the class is very low. So, i decided to remove them. 4.Also, some features like 1 in fbs, 2 in restecg, normal in thal are not much different towards classifying the data. So i decided to remove them as well. I've removed some features because only some of them can be used for separating the class.

logistic regression

There are two types of logistic regression, FOr our data-set we gonna use the Binary logistic regression classifier. Logistic regression predicts whether something is true or false. Unlike other machine learning models like linear regression, instead of fitting a line to the data, it fits an "S" shaped logistic function where the curve of S goes from "0" to "1". In logistic regression, these 0 to 1 is represented as y-axis and all the features in x-axis. This means that, the curve tells you the probability of something is True or false based on X-axis features.

Logistic regression belongs to a family, named Generalized Linear Model (GLM). The model is basically an ectenstion of linear regression model. As the algorithm is based on probability, to range the value between 0 and 1, we need to give a threshold value which will be used to decide the class of true or false.

As logistic regression fits the data into an "S" shaped curve, it is defined as "p = exp(y) / [1 + exp(y)]" Where:

```
1. y = b0+b1*x;
2. exp() is exponential.
3. given x, p is the probability of an event to occur.
```

but in order to fit the data into an S shaped curve, we need to do a bit of manipulation and do log-odds. Now it can represented as, p/(1-p) = exp(b0 + b1*x) where, b0 and b1 are coefficients. After applying logarithm on both sides, log[p/(1-p)] = b0 + b1*x where log(odds of data) = log(p/1-p)

From the statement above, we might think that logistic regression is similar to linear regression, but the major difference is that, logistic regression uses maximum likelihood to draw the "S" shaped curve to project the data points. We can't use the least squared of data points technique to find the best fitting line as the Y-axis which is either 0 or 1 is projected as +infinity to -infinity of Y-axis inside the computation. Now the initial slope is drawn from +infinity to -infinity of Y-axis and project them onto the candidate line. This gives a log(odds) value for each data point. Now we can transform the log(odds) values to probabilities of each data-points using "p = exp(y) / [1 + exp(y)]". Now we can calculate the overall likelihood of the given data which is projected on the slope. Now we just keep doing the log(odds) line again and again and projecting the data onto it and again calculating the overall-likelihood of the data until we find the best optimal fit of the given data.

Naive bayes

There two types of Naive bayes which can be used. For our data-set, Gaussian Naive Bayes is the approach used as a classifier. Bayes theorem is used to make predictions based on prior knowledge and current evidence. prior probability aka prior knowledge considers the most probable outcome without additional evidence. The current evidence is indicated as likelihood that reflects the probability of a predictor.

For each feature in the given data-set, given the probability of the outcome is true, we need to draw a gaussian aka normal distribution using the mean and standard deviation of each feature. For the same feature given the probability of the outcome is false, we need to draw a corresponding gaussian distribution of false case beside the true case. We will end-up with combination of true and false case gaussian distribution of each feature in the data-set. After representing the training data like this, we need to calculate the prior probability of true and false cases, using the training data-set. This value has to be saved to predict the future new data. This can be calculated as "P(True) = Total number of true cases/((Total number of true cases) + (Total number of false cases))"

Now as the new data arrives, We need to find the Y-axis coordinate(which is Likelihood) on the gaussian curve that corresponds to the x-axis coordinate. This has to be done on each feature's gaussian distribution. The likelihood of each feature can be calculated using "L(Distrubution of feature | True)" which can give us the Y-axis coordinates of the each feature.

After getting the likelihood of the each feature, we have to multiply the each feature's likelihood with probability of True case (aka prior probability). We have to repeat the process of finding the likelihood with false case and multiply with false case prior probability. Once we have both the scores for true and false cases, we can draw conclusion using which score is higher.

Decision Tree

With decision trees Shown below, it is really easy to interpret the data visually and predict further inputs' outcomes. But how do we know which features are more significant than others? This is done with calculating and comparing something called "information gain". Information gain is a powerful tool for creating an accurate decision tree and it is calculated using entropy and the probability of occurrence of a certain task, or to be more specific, Conditional Entropy. This lets us know which features are more important and have more effect in the final result, which is whether someone has heart disease or not in our particular data set. Using the decision tree we can predict whether a new patient with particular attributes is going to be sick or not, by simply walking the tree from the root node (which is the one with the maximum information gain) towards one of the leaves which is going to be marked with the result. In our case again it's whether our patient has a heart disease. If we end up on the leaf that says "1", we can tell, with reasonable accuracy, that a patient with these attributes has a heart disease.

##Justification

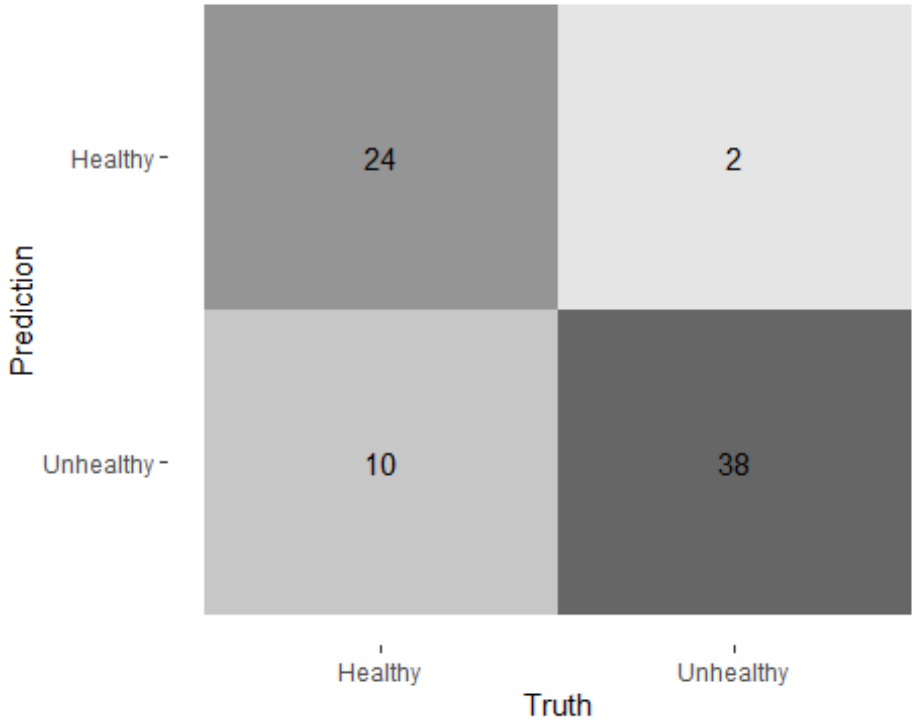
In my opinion, All three classifiers are good for handling this data-set. But, I think the Logistic regression and the naive bayes handles this data-set much better than the decision trees.

I give the above statement due to the features of handling big data-sets in logistic regression and naive bayes are handled in a efficient way while in decision tree, big data-sets are tend to over fit the tree resulting less accuracy while facing an unseen data.

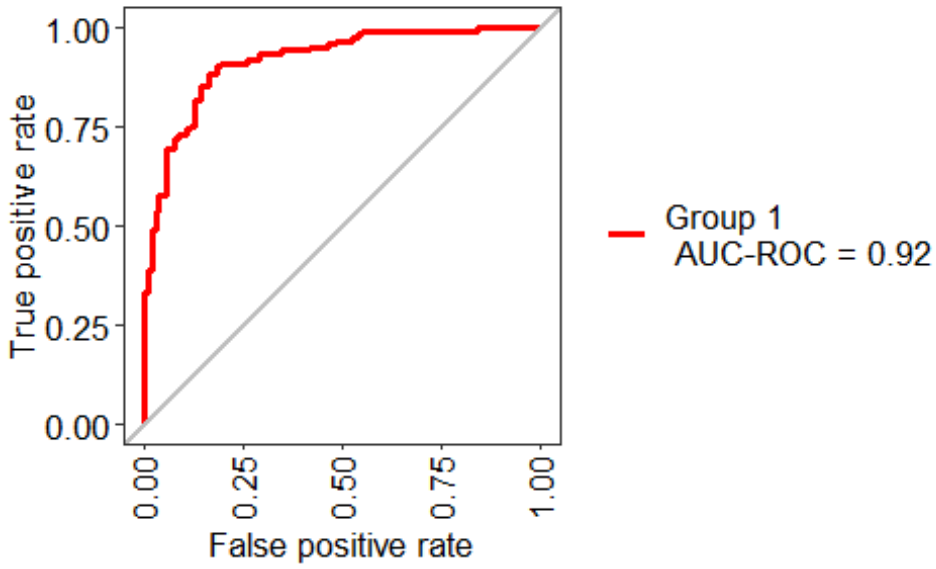
But, unlike logistic regression and naive bayes, Decision trees are easy to understand, visualize and less effort in data preparation for feature selection. However, Decision trees can be unstable because small variations in data might result in a completely different tree. In addition, there is a high chance that some class might dominate if we create a bias tree without a balanced data-set.

It's not like decision tree is not suitable for this data-set, even naive bayes also has flaws in it. For instance, Navie bayes mostly depends on the distribution of the features in training and compare it with likelihood when facing an unseen data. This has same problem of class domination where, when if one feature is more dominant than the remaining features, It has a tendency to misclassify the new unseen data accurately because the conclution is made using the greater score of true and false case.

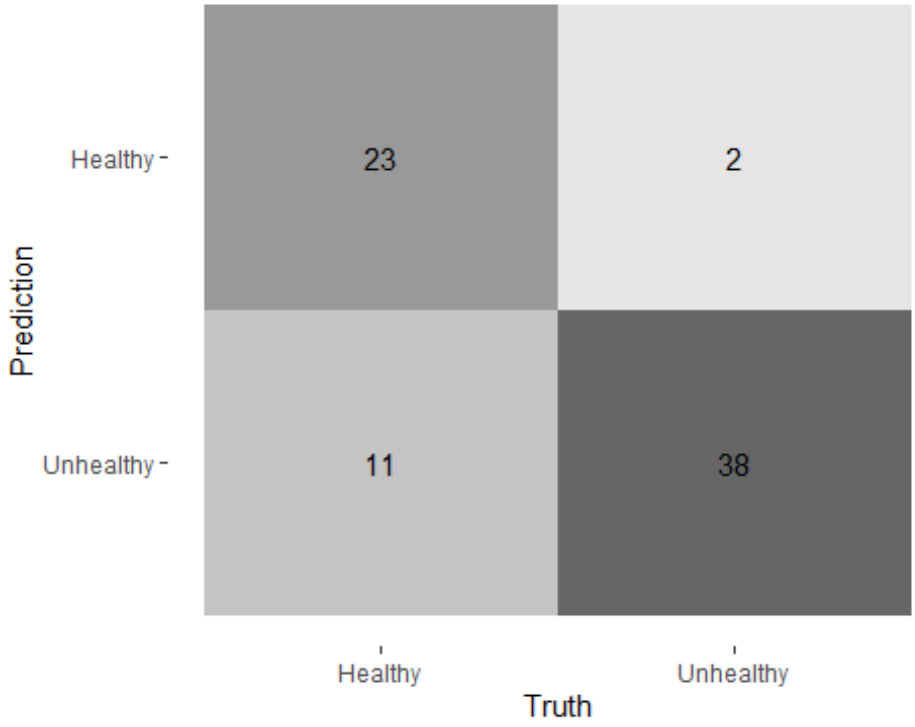
```
#Logistic regression using caret's train function.

LogisticModel <- train(target ~ .,
                       data = trainingHeartData,
                       preProc = c("center", "scale"),
                       trControl = trainControl(method = "cv", number = 10,
                                                classProbs = TRUE,
savePredictions = T,
                                                summaryFunction =
twoClassSummary),
                       metric = "ROC",
                       method = "glm",
                       family=binomial(link = "logit"))

##
##  ROC value for logistic Regression is:  0.9256119

##
##  Accuracy of for logistic Regression model is:  0.8378378
```



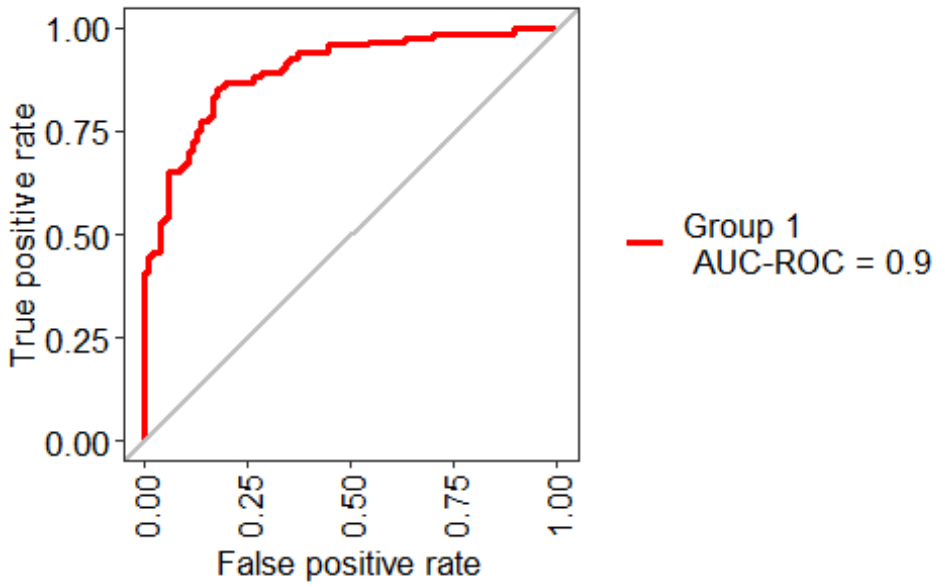

```
## ***MLeval: Machine Learning Model Evaluation***

## Group 1 AUC-ROC = 0.92
```



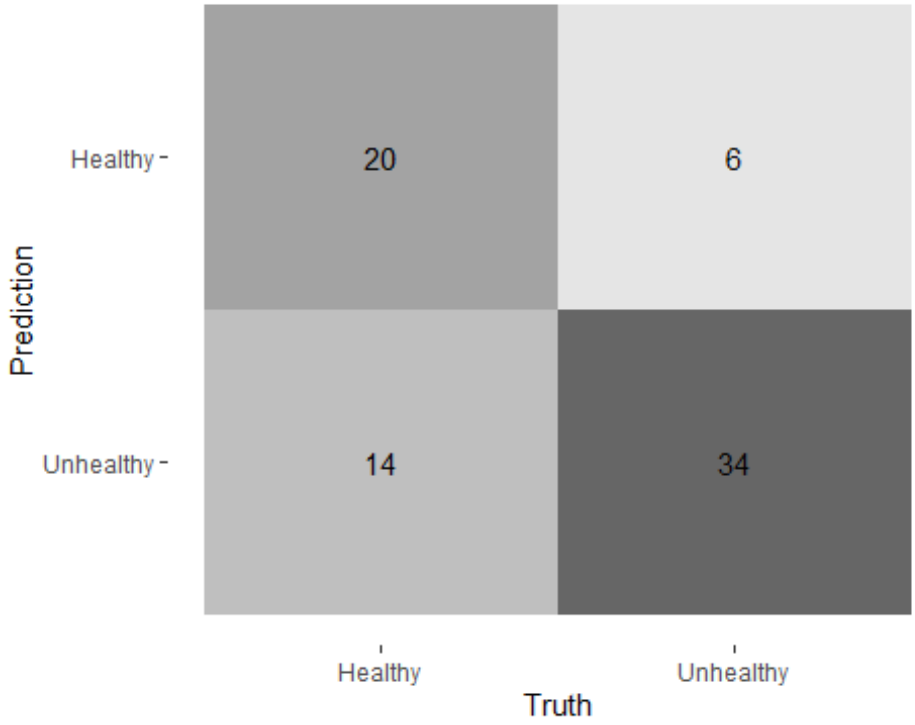
```r
#Navie Bayes using caret's train function.

NB_Model <- train(target ~ .,
                          data = trainingHeartData,
                          preProc = c("center", "scale"),
                          trControl = trainControl(method = "cv", number = 10,
                                                   classProbs = TRUE,
savePredictions = T,
                                                   summaryFunction =
twoClassSummary),
                          metric = "ROC",
                          method = "naive_bayes")

##
##  ROC value for Navie Bayes where probability
##  of True is:  0.8998019
##  and False is:  0.9086888

##
##  Accuracy of for Navie Bayes model is:  0.8243243
```



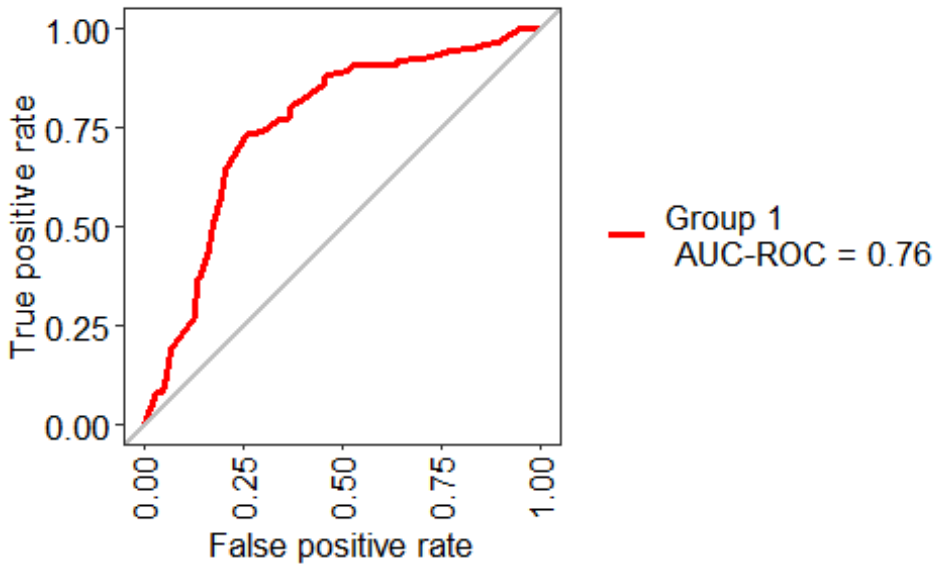

```r
## ***MLeval: Machine Learning Model Evaluation***

## Group 1 AUC-ROC = 0.9
```




```
#Decision Tree using caret's train function.

DT_Model <- train(target ~ .,
                          data = trainingHeartData,
                          preProc = c("center", "scale"),
                          trControl = trainControl(method = "cv", number = 10,
                                                   classProbs = TRUE,
savePredictions = T,
                                                   summaryFunction =
twoClassSummary),
                          metric = "ROC",
                          method = "rpart")

##
##  ROC value for Decision Tree for main feature Chest pain level is:
0.7631789 0.745338 0.6461713

##
##  Accuracy of for Decision Tree model is:  0.7297297
```




```
## ***MLeval: Machine Learning Model Evaluation***

## Group 1 AUC-ROC = 0.76
```



According to the accuracy rates of each model where logistic regression is 0.8266667, Naive Bayes is 0.7733333 and Decision Tree is 0.76., Logistic regression is performing much better when comparing to the accuracy rates of the other 2 models.

But While comparing the area under the ROC Curve(AUC), where logistic regression is 0.9, Naive Bayes is 0.9 and Decision Tree is 0.68., Both Logistic regression and Naive Bayes are performing at the same level while Decision Tree's performace is very less when compared to the Other 2 classifiers.

However, even though both Logistic regression and Naive Bayes are performing at the same level, by looking at the confusion matrix of both of the classifiers i would say that, logistic regression is best for the task.

I strongly support the above statement because in confusion matrix of Logistic regression, the true positives are much higher and False negatives are less when compared to the confusion matrix of naive bayes.

Logistic regression works well and much better than the naive bayes due to its functionality of Maximum Likelihood of the data. Where even though naive bayes uses the likelihood, it has the draw back of dominant feature where it can manipulate the probability of an true or false case. Logistic regression also have an advantage of having a "P" value which is the threshold for categorizing into true or false. ```