Predicting the Presence of Heart Disease Using 1998 Cleveland Hospital Data

By

Eric Cartaya

A Research Project Submitted in

Partial Fulfillment of the

Requirements for the Degree of

Master of Science in Statistics

Specialization in Data Science

At

Rutgers, the State University of New Jersey

December 2020

Original First Page Header:

Christopher Rabeony, Eric Cartaya, Matthew Dula

MSDS 534 Statistical Learning - Final Report

December 15, 2020

***Table of Contents***

1. Cover……………………………………...…………………………………………Page 1
2. Table of Contents…………………………………………………………………....Page 2
3. List of Tables………………………………………………………………...……...Page 2
4. List of Graphs………………………………………………………………………..Page 2
5. Abstract……………………………………………………………………………...Page 3
6. Introduction………………………………………………………………………….Page 3
7. Methods……………………………………………………………………………...Page 4
   1. Dataset……………………………………………………………………….Page 4
   2. Data Cleaning………………………………………………………………..Page 4
   3. Preliminary Analysis………………………………………………………...Page 4
   4. Model Building………………………………………………………………Page 5
8. Results……………………………………………………………………………….Page 5
   1. Variable Selection……………………………………………………………Page 5
   2. Model Baseline………………………………………………………………Page 5
   3. Variable Importance…………………………………………………………Page 6
9. Discussion…………………………………………………………………………...Page 6
10. Conclusion…………………………………………………………………………...Page 7
11. R-Markdown Code…………………………………………………………………..Page 9
12. Sources......................................................................................................................Page 34

***List of Tables***

1. Page 7: **Table 1.** Accuracy, sensitivity, precision, and F1 score of each model.
2. Page 8: **Table 2.** Figure 2 sub-table

***List of Graphs***

1. Page 8: **Figure 1.** ROC curves for logistic regression, decision tree, and random forest, k-nearest neighbors, gradient boosting trees, and support vector machines
2. Page 9: **Figure 2.** Plots of important predictor variables with relation to the presence of heart disease classification.

***Abstract***

In this study, we used the Heart Disease UCI dataset to test several machine learning techniques and identify an optimal predictive model. The dataset contains 303 observations of 14 predictors, which contain the characteristics of each individual’s cardiovascular health and whether they had signs of heart disease. Our response variable was “heart disease”, a binary variable that signals if a patient has suspected of having heart disease. Our data was balanced between individuals that had heart disease and individuals who didn't have any significant presence of heart disease. For this project we utilized several modeling techniques including logistic regression, classification tree, random forests, k-nearest neighbors, and gradient boosting and support vector machines. From our analysis we determined that logistic regression generated the best accuracy rate (86%) and sensitivity (89%) with support vector machines having the second best accuracy (83%) and third best sensitivity (83%), while gradient boosted machines had the second best accuracy (82%) and the third highest sensitivity (85%) of our models.

***I. Introduction***

Cardiovascular (heart) disease can describe a range of conditions that affect an individual’s heart. There are a variety of diseases that can be classified as heart disease. This includes blood vessel diseases, such as coronary artery disease, heart rhythm problems (arrhythmias) and heart defects that a person can be born with (congenital heart defects), among a variety of others. The World Health Organization (WHO) has reported that heart disease is the leading cause of death across the world, responsible for an estimated 17.9 million deaths each year3. 655,000 Americans die from heart disease each year, which translates to 1 in every 4 deaths4. Heart disease alone costs the United States about $219 billion each year from 2014 to 2016. This includes the cost of healthcare services, medicines, and lost productivity due to death. Prediction of cardiovascular disease can be regarded as one of the most critical subjects of interest in the section of clinical data analysis. The amount of data in the healthcare industry is large enough for machine learning to help produce some form of informed decisions and predictions. The goal of our research and analysis is to determine the most significant factors that lead to the threat of heart disease in the average person.

***II. Methods***

*Dataset.* For this project we used the Heart Disease dataset from the UCI Repository. The data was collected from the Cleveland Clinical Foundation. The dataset contains a 303 total observations with 14 attributes. Of these total variables 13 of these predictors variables were the characteristics of each patient. These predictor variables are *age*, *sex*, *chest pain*, *resting blood pressure*, *cholesterol*, *fasting blood sugar*, *resting ecg*, *max heart rate achieved*, *exercised induced angina, ST depression induced by exercise*, *peak exercise ST waves*, *major vessels identified*, and *presence of the blood disorder thalassemia.* Our response variable, *hd* (heart disease) displays whether the individual is suffering a form of heart disease. The dataset was formatted as a CSV file and loaded into R for analysis.

*Data Cleaning.* This heart disease dataset contains variables that are all considered numeric variables. For variables that utilize numbers as a way to categorically classify characteristics (*gender*, *chest pain*, *displays of thalassemia*) we transformed these numeric variables into factor variables. Most importantly, our response variable: heart disease (*hd)* was a variable that contained five different levels of coronary artery disease. We wanted to convert this categorical variable into a binary variable with only two options. A value of **0**, indicates an absence of heart disease where less than 50% of the artery’s diameter narrowed. A value of **1** indicates the presence of heart disease where more than 50% of the artery’s diameter narrowed.

As for any missing data, there were about six missing values.

*Preliminary Analysis*. Before we started our analysis using our current predictors and responses we wanted to determine if our predictor variables are heavily correlated with each other. This resulted in no signs of our variables being correlated. We also utilized some plots to

determine if our data was balanced between patients that had heart disease versus patients that did not have heart disease.

*Model Building.* Several machine learning techniques are employed to predict whether an individual is a risk for heart disease or not. These analytic techniques include logistic regression, classification decision trees, random forest, k-nearest neighbors, boosted trees, and support vector machines. Models were evaluated using accuracy, sensitivity, specificity, F1 score, and AUC. Variable importance was calculated from the models with the highest accuracy and sensitivity

***III. Results***

*Variable Selection*. In order to explore the possibility of a more parsimonious model, backward stepwise selection algorithms using AIC were performed. The minimal AIC was observed using 6 of the 13 features. These variables were: *sex*, *chest pain* (cp), *max heart rate* (thalach), *peak exercise ST waves* (slope), *major vessels identified* (ca), and *displays of thalassemia* (thal).

*Model Baseline.* Each classifier was trained to make probability predictions with a prediction threshold of 0.5 to test the performance. As seen in Table 1, logistic regression, support vector machines, and gradient boosting machines produced the highest test accuracies and F1 scores. The AUC values produced from the ROC in Figure 1 corroborate these findings. The precision, specificity, and F1 scores could be affected by the small number of observations in our testing set. For future consideration, it would be beneficial to have a larger training set and predict it on a respectively larger testing set. This would give us more accurate findings. We could also adjust our prediction threshold for each model in order to find the optimal performance of each model.

*Variable Importance.* After observing each of our models and seeing how they performed, the next step was testing the variable importance from our most optimal models. Earlier, we discussed how stepwise logistic regression gave us *sex*, *chest pain* (cp), *max heart rate* (thalach), *peak exercise ST waves* (slope), *major vessels identified* (ca), and *displays of thalassemia* (thal) as our featured variables. Now that we completed our model evaluation we decided to calculate variable importance from the optimal gradient boosting machines for boosted trees. We were able to determine that the most important factors that determine the presence of heart disease are whether or not there is a reversible defect in Thalassemia *(thal)*, followed by whether or not there is an occurrence of asymptomatic chest pain (*cp*). The third and fourth most important factor was if the maximum heart rate was achieved (*thalach*), and if there was significant ST depression after strenuous exercise (*oldpeak*).

***IV. Discussion***

Logistic regression, support vector machines, and gradient boosting machines were the top three models when it comes to accurately predicting heart disease. K-Nearest Neighbors performed adequately but not as well as our top three models due to the mix of categorical and continuous variables. Classification trees performed last, which can be explained by decision trees being the weakest performing models in terms of low predictive ability. Random forests performed the fourth best out of our models, due to it utilizing a large number of classification trees as weak classifiers and putting them together to form a random forest, with the diversity of classifiers within the ensemble to improve its performance.

The criterion we used to evaluate our models was Accuracy, Sensitivity, and Recall. The most straightforward one is Accuracy, which is the proportion of the total number of predictions that were correct. Where the closer these numbers are to 1, the better it is. Sensitivity or Recall is also known as true positive rate. A good way to understand it is, let’s say that our model classified 100 people as sick. A recall value of .9 means that there is a possibility that 90 people are correctly classified as sick and 10 people are not actually sick.

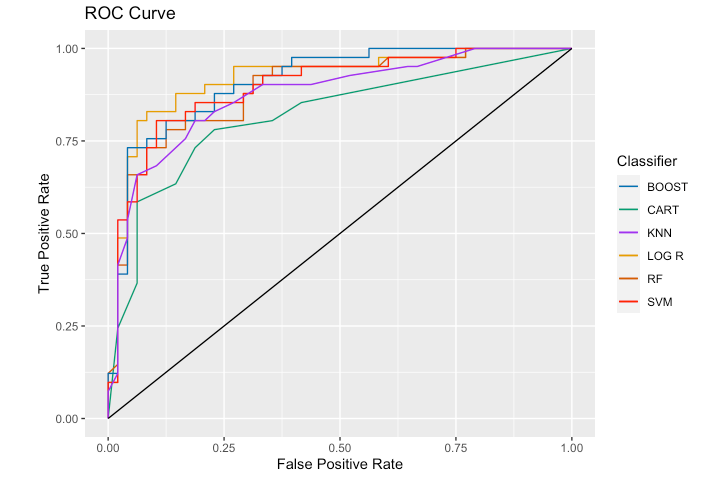
***V. Conclusion***

Heart disease is the leading cause of death currently in the world. Understanding the connection between different variables and cardiovascular disease itself can lead to a reduction in prevalence. If these methods are successful, then it’d be possible to reduce the number of deaths of victims of heart disease, and save millions of dollars. Based on our project, logistic regression, support vector machines, and gradient boosting machines performed the best in regards to predicting heart disease. According to our model, the odds of one patient having the disease is highly affected by the blood disorder, Thalassemia *(thal)*, and the occurrence of asymptomatic chest pain (*cp*). These variables make a lot of sense as chest pain and a blood disorder such as Thalassemia can be a sign of heart disease in people.

***Table 1.*** *Accuracy, sensitivity, precision, and F1 score of each model.*

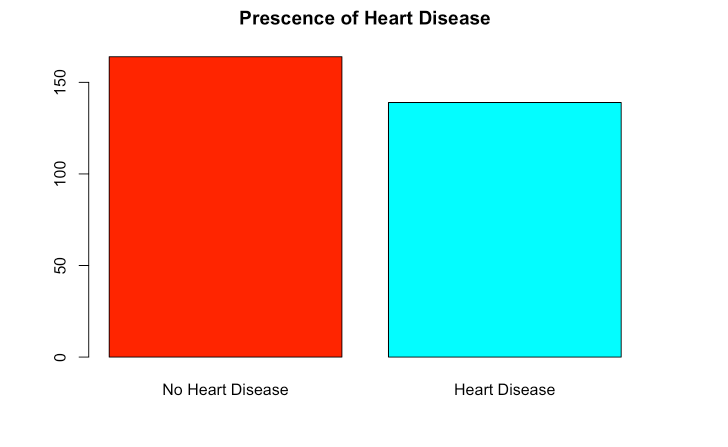
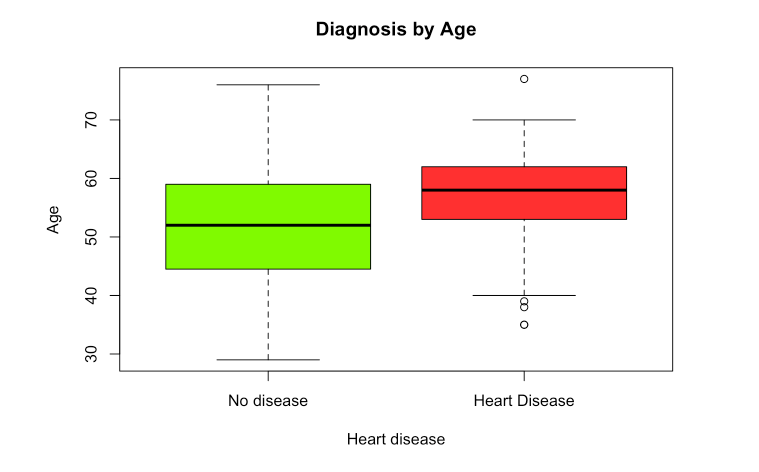
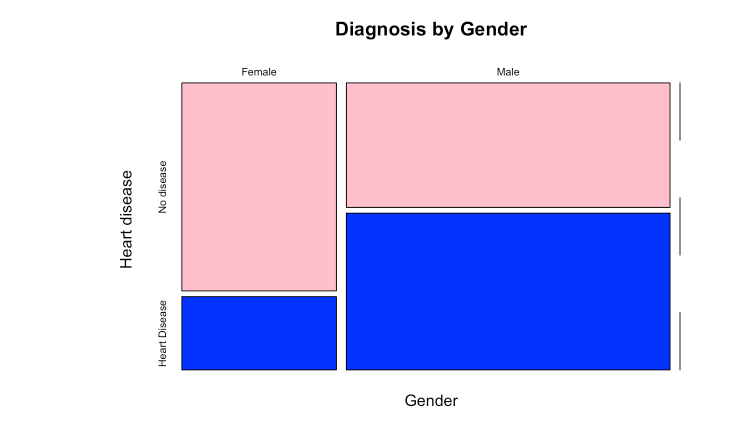
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **Accuracy** | **Sensitivity** | **Precision** | **F1 score** |
| ***Logistic Regression*** |  |  |  |  |
| Heart Disease | .865 | .896 | .860 | .878 |
| No Heart Disease | .865 | .829 | .872 | .850 |
| ***Decision Tree*** |  |  |  |  |
| Heart Disease | .753 | .854 | .732 | .788 |
| No Heart Disease | .753 | .634 | .788 | .703 |
| ***Random Forest*** |  |  |  |  |
| Heart Disease | .809 | .812 | .830 | .821 |
| No Heart Disease | .809 | .805 | .786 | .795 |
| ***K-Nearest Neighbors*** |  |  |  |  |
| Heart Disease | .798 | .833 | .800 | .816 |
| No Heart Disease | .798 | .756 | .795 | .775 |
| ***Gradient Boosting*** |  |  |  |  |
| Heart Disease | .820 | .854 | .820 | .837 |
| No Heart Disease | .820 | .780 | .821 | .800 |
| ***Support Vector Machines*** |  |  |  |  |
| Heart Disease | .832 | .833 | .851 | .842 |
| No Heart Disease | .832 | .830 | .810 | .819 |

***Figure 1.*** *ROC curves for logistic regression, decision tree, and random forest, k-nearest neighbors, gradient boosting trees, and support vector machines*

**

|  |  |
| --- | --- |
| **Method** | **AUC** |
| *Logistic Regression* | 0.916 |
| *Decision Tree* | 0.818 |
| *Random Forest* | 0.887 |
| *K-Nearest Neighbors* | 0.909 |
| *Gradient Boosting Machines* | 0.896 |
| *Support Vector Machines* | 0.875 |

***Figure 2.*** *Plots of important predictor variables with relation to the presence of heart disease classification.*



**R-Markdown Code:**

---

title: "MSDS Statistical Learning Project 12/14/2020"

author: "Christopher Rabeony, Eric Cartaya, Matthew Dula"

date: "12/14/2020"

output: html\_document

---

#### Import our Library

```{r, message = FALSE}

library(corrplot) # correlation detection

library(corrgram) # correlation plot

library(MASS) # stepwise function for model selection

library(caret) # confusion matrix

library(rpart) # desicion tree

library(rpart.plot) # descision tree plot

library(randomForest) # random forest

library(ROCR) # ploting ROC curve

library(pROC) # Function to calculate the ROC

library(e1071) # Performing Support Vector Machine

library(gbm) # Gradient Boosting

library(class) # K- Nearest Neighbors

```

## Heart Disease

Cardiovascular disease includes a number of conditions affecting the structures or function of the heart, including coronary artery disease and vascular (blood vessel) disease. Also known as Heart disease, it is by far the leading cause of death in the United States. Coronary artery disease is the narrowing of the arteries supplying blood to the heart. It has caused about one million heart attacks each year. Even more worrisome, hundreds of thousands people with heart attacks will die before even reaching the hospital.

Our goal is to investigate what factors lead to heart disease. We will use the dataset for this project which is available on UCI machine learning repository. This project, will run statistical testings and regression/classification models using data from the Cleveland heart disease dataset to asses what factors significantly influence the prescence of heart disease.

#### Description of our Variables

\* `age` - age in years. `CONTINUOUS`

\* `gender` - gender (0 = male; 1 = female) `BINARY`

\* `cp` - chest pain type `CATEGORICAL`

+ `1` = typical agina

+ `2` = atyptical angina

+ `3` = non-anginal pain

+ `4` = asymptomatic

\* `trestbps` - resting blood pressure (in mm HG) `CONTINUOUS`

\* `chol` - serum cholestoral in mg/dl `CONTINUOUS`

\* `fbs` - fasting blood sugar > 120 mg/dl (0 = false; 1 = true) `BINARY`

\* `restecg` - resting electrocardiographic results `CATEGORICAL`

+ `0` = normal

+ `1` = having ST-T Wave abnormal (T wave inversions and/or ST elevation or depression of > 005 mV)

+ `2` = showing probable or definite left vetricular hyptertropy

\* `thalach` - maximum heart rate achieved in beats per minute (bpm) `CONTINUOUS`

\* `exang` - exercise induced angina (0 = no; 1 = yes) `BINARY`

\* `oldpeak` - ST depression induced by exercise relative to rest `CONTINUOUS`

\* `slope` - the slope of the peak exercise ST segment `CATEGORICAL`

+ `1` - upsloping

+ `2` - flat

+ `3` - down-sloping

\* `ca` - number of major vessels (0-3) colored by fluoroscopy `CATEGORICAL`

\* `thal` - displays the thalassemia `CATEGORICAL`

+ `3` = normal

+ `6` = fixed defect

+ `7` = reversible defect

\* `hd` - predicted target variable. Diagnosis of heart disease (angiographic disease status) `BINARY`

+ `0` = < 50% diameter narrowing - No Prescense of heart disease

+ `1` = > 50% diameter narrowing - Prescense of heart disease

```{r setup, include=FALSE}

#Function needed to convert classes of predictor values

convert.magic <- function(obj,types){

for (i in 1:length(obj)){

FUN <- switch(types[i],character = as.character,

numeric = as.numeric,

factor = as.factor)

obj[,i] <- FUN(obj[,i])

}

obj

}

convert.names <- function(row){

row=gsub("sex1", "male", row)

row=gsub("thal7", "reversable defect thalassemia", row)

row=gsub("thal6", "fixed defect thalassemia", row)

row=gsub("cp4", "asymptomatic chest pain", row)

row=gsub("cp3", "non-anginal chest pain", row)

row=gsub("cp2", "atypical angina chest pain", row)

row=gsub("oldpeak", "ST depression from exercise", row)

row=gsub("thalach", "maximum heart rate achieved", row)

row=gsub("trestbps", "resting blood pressure", row)

row=gsub("ca2", "2 major vessels col/b fluoro., ca2", row)

row=gsub("ca1", "1 major vessel col/b fluoro., ca1", row)

row=gsub("slope2", "flat peak exercise ST segment", row)

row=gsub("slope1", "upsloping peak exercise ST segment", row)

row=gsub("slope3", "downsloping peak exercise ST segment", row)

row=gsub("chol", "serum cholestoral", row)

row=gsub("exang", "exercise induced angina", row)

row=gsub("restecg2", "restec: showing left ventricular hypertrophy", row)

row=gsub("restecg1", "restec: having ST-T wave abnormality", row)

row=gsub("fbs1", "fasting blood sugar > 120 mg/dl", row)

}

```

### Import and label our data

```{r}

# Read the data into a data frame

heart.data <- read.csv("/Users/chris/Documents/Statistical Learning MSDS 534/Project/processed.cleveland.data", header = F)

dim(heart.data)

```

```{r}

# Give our columns the following names.

names(heart.data) <- c("age", "sex", "cp", "trestbps", "chol", "fbs", "restecg", "thalach", "exang", "oldpeak", "slope", "ca", "thal",

"hd")

# Description of our data

str(heart.data)

```

#### Clean up our data before analysis

Our data has a mix of classification and continuous variables. We will format our data to simplify our classifications and make it easier to read.

```{r message=FALSE, warning=FALSE}

# HD column. Originally a continuous variable where clinicians had graded patients as either having no heart disease (value of 0) or displaying various degrees of heart disease (values 1 to 4). We chose to group the data into 2 categories of ‘no heart disease’ (value of 0) and ‘displaying heart disease’ (value of 1) so it became binary.

heart.data$hd[heart.data$hd > 0] <- 1

```

#### Quick EDA

```{r}

# Check if our data is balanced

prop.table(table(heart.data$hd))

table(heart.data$hd)

```

###### Visualize our data

```{r}

# Graph of our plot to determine how many of our patients potentially have heart disease vs. the one's who are healthy.

barplot(table(heart.data$hd), main = "Prescence of Heart Disease", names.arg=c("No Heart Disease","Heart Disease"),col=rainbow(2))

```

#### Clean up our Data

```{r, message=FALSE, warning=FALSE}

# change a few predictor variables from integer to factors (make dummies)

change\_var <-c("numeric","factor","factor","numeric","numeric","factor","factor","numeric","factor","numeric","factor","factor","factor","factor")

heart.data <- convert.magic(heart.data,change\_var)

```

###### Check for missing values - only 6 so just remove them.

```{r message=FALSE, warning=FALSE}

# ca and thal have missing values indicated by “?”. Put NA instead.

heart.data$ca <- as.character(heart.data$ca)

heart.data$ca[is.na(heart.data$ca)] <- "0.0"

heart.data$thal <- as.character(heart.data$thal)

heart.data$thal[is.na(heart.data$thal)] <- "3.0"

# ca and thal have missing values indicated by “?”. Put NA instead.

heart.data$ca[heart.data$ca == "?"] <- NA

heart.data$thal[heart.data$thal == "?"] <- NA

s <- sum(is.na(heart.data))

heart.data <- na.omit(heart.data)

#str(heart.data)

```

#### Quick EDA explore Boxplot

```{r}

# Create a new dataset that changes some of our variable values in order to plot it cleanly

heart <- heart.data #add labels only for plot

levels(heart$hd) = c("No disease","Heart Disease")

levels(heart$sex) = c("Female","Male","")

mosaicplot(heart$sex ~ heart$hd,

main="Diagnosis by Gender", shade=FALSE,col=c("pink", "blue"),

xlab="Gender", ylab="Heart disease")

boxplot(heart$age ~ heart$hd,

main="Diagnosis by Age", col = c("chartreuse1", "firebrick1"),

ylab="Age",xlab="Heart disease")

```

##### Exploring the Variable’s Correlation

```{r}

# Check for multicollinearity

heart.cor <- sapply( heart.data, as.numeric )

# Plot our results

corr <- cor(heart.cor[,1:13])

corrplot(corr)

```

Most machine learning algorithms assume that the predictor variables are independent from each others. This the reason why the multicollinearity will be removed to achieve a more robust analysis.

None of our variables seem to be highly correlated with any other. All correlations are less than 0.8.

#### Our "Cleaned Up" Heart Disease Dataset

```{r}

head(heart.data)

```

## Testing our 5 Different Methods.

#### Training and testing data for validation

Split the data into Training (70%) and Testing (30%) data. Percentage of heart disease or not must be same in training and testing.

```{r, warning = FALSE}

set.seed(10, sample.kind = "Rounding")

split <- createDataPartition(heart.data$hd,p=0.7,list=FALSE)

train <- heart.data[split,]

test <- heart.data[-split,]

nrow(train)/(nrow(test)+nrow(train)) #checking whether really 70% -> OK

```

### Create a function to caculate accuracy, precision, and f1 scores.

```{r}

# Function to calculate our accuracy, sensitivity, and f1 scores for our tables.

calculate\_stats <- function(table) {

n = sum(table) # number of instances

diag = diag(table) # number of correctly classified instances per class

rowsums = apply(table, 1, sum) # number of instances per class

colsums = apply(table, 2, sum) # number of predictions per class

accuracy = sum(diag) / n

sensitivity = diag / colsums

precision = diag / rowsums

f1 = 2 \* sensitivity \* precision / (sensitivity + precision)

return(data.frame(accuracy, sensitivity, precision, f1))

}

```

### Predict with 6 different methods with different tuning parameters and compare best model of each method

Results are going to be stored in variable AUC. AUC is the area under the ROC which represents the proportion of positive data points that are correctly considered as positive and the proportion of negative data points that are mistakenly considered as positive. We also store Accuracy which is true positive and true negative divided by all results.

```{r}

# List to hold our results

AUC = list()

Accuracy = list()

```

### Selecting a model for Logistic Regression

#### Model Selection

##### Forward/Backward Selection

```{r, message = FALSE, warning = FALSE}

LogReg\_model.all <- glm(hd ~ ., data=train, family = 'binomial')

summary(LogReg\_model.all)

```

Features for the logistic regression is selected using the stepwise regression. As the basis for feature selection, we will use LogReg\_model.all, which is a model that uses all the features. The direction of features selection is backwards, which means features are selected from the available features and uses the AIC criterion.

```{r}

# Stepwise Regression

LogReg\_model.step <- step(LogReg\_model.all, direction = 'backward', trace = 0)

variables <- attr(LogReg\_model.step$terms, 'term.labels')

summary(LogReg\_model.step)

```

Out of the 13 features available, 6 features : sex, cp, thalach, slope, ca, thal were chosen

##### We will primarily use the full model with all features.

Because detecting Heart Disease can be a complex diagnosis we also want to analyze with our full set of features. However, this new model can be helpful to predicting heart disease.

### Logistic Regrssion (Using the Full Model)

Our response variable (hd) is binary, so we use logistic regression model to train and predict the dataset. Initially we include all the variables to train and test the model.

```{r, warning = FALSE}

set.seed(10, sample.kind = "Rounding")

# Fit a logistic regression model to our data

LogReg\_model <- train(hd ~ ., data=train, method = 'glm', family = 'binomial')

summary(LogReg\_model)

```

```{r message=FALSE, warning=FALSE}

# Fit a prediction and find a confusion matrix

LogReg\_pred <- predict(LogReg\_model, test)

LogReg\_pred.prob <- predict(LogReg\_model, test, type='prob')[2]

LogReg\_ConfMat <- confusionMatrix(LogReg\_pred, test[,"hd"])

LogReg\_ConfMat

```

```{r}

# Calculate our The performance of our Modified Model

LogReg\_tab <- table(LogReg\_pred, test$hd)

calculate\_stats(LogReg\_tab)

```

```{r message = FALSE, include = FALSE}

#ROC Curve

AUC$logReg <- roc(as.numeric(test$hd),as.numeric(as.matrix((LogReg\_pred.prob))))$auc

Accuracy$logReg <- LogReg\_ConfMat$overall['Accuracy']

```

#### Logistic Regrssion (Using the Stepwise Model)

Let's test the predictive power of our Stepwise Model

```{r}

LogReg\_model.step <- glm(hd ~ sex + cp + thalach + slope + ca + thal, family = "binomial", data = train)

summary(LogReg\_model.step)

```

```{r}

# Make a prediction on our training set

LogReg\_step.prob <- predict(LogReg\_model.step, test, type = 'response')

LogReg\_step.pred <- ifelse(predict(LogReg\_model.step, test, type = 'response') > 0.5, "Yes", "No")

LogReg\_step.table <- table(predicted = LogReg\_step.pred, actual = test$hd)

LogReg\_step.table

```

```{r}

# Calculate the performance of our Modified Model

stepLogReg.tab <- table(LogReg\_step.pred, test$hd)

calculate\_stats(stepLogReg.tab)

```

Hence, this new stepwise model has lower accuaracy and is not better at predicting the presence or absence of heart disease than the first model.

### Decision Trees

Decision tree algorithms use the training data to segment the predictor space into non-overlapping regions, the nodes of the tree. Each node is described by a set of rules which are then used to predict new responses. The predicted value for each node is the most common response in the node (classification), or mean response in the node (regression).

```{r, warning = FALSE}

set.seed(10, sample.kind = "Rounding")

# Implementing Decision Tree

Tree\_model <- rpart(hd~.,data = train, method = "class" )

# Visual representation of our Tree

rpart.plot(Tree\_model)

```

Based off our tree we can see that the most significant variables are Thalassemia the inherited blood disorder that affects the body’s ability to produce hemoglobin and red blood cells (thal), chest pain type (cp), Number of major vessels identified (ca), ST depression induced by exercise (oldpeak), and cholestoral (chol)

```{r}

# Predict on training set

Tree\_pred <- predict(Tree\_model, test, type = "class")

Tree\_pred.prob <- predict(Tree\_model, test, type = "prob")

Tree\_ConfMat <- confusionMatrix(Tree\_pred, test$hd)

Tree\_ConfMat

```

```{r message = FALSE, include = FALSE}

# ROC Curve for Tree

AUC$tree <- roc(as.numeric(test$hd),as.numeric(as.matrix((Tree\_pred))))$auc

Accuracy$tree <- Tree\_ConfMat$overall['Accuracy']

```

```{r}

# Calculate the performance of our Decision Tree Model

calculate\_stats(table(Tree\_pred, test$hd))

```

### Random Forest

Random forests are about having multiple trees, a forest of trees. Those trees can all be of the same type or algorithm or the forest can be made up of a mixture of tree types (algorithms).

```{r, warning = FALSE}

# Generate optimal model

set.seed(10, sample.kind = "Rounding")

RF\_model <- randomForest(hd ~ ., data = train, importance = TRUE)

RF\_model

```

On average, about two thirds of of each data set is sampled each time a bootstrap sample is taken. With one third of observations remaining, we utilize this subset for testing each newly created tree, creating out-of-bag (OOB) error, with which we can gague the accuracy of each tree.

```{r}

# Plot for the number of trees

plot(RF\_model, main = "Error rate of Random Forest")

```

As the number of trees grow the error rate slightly decreases.

```{r}

# Fit a prediction and find a confusion matrix

RF\_pred <- predict(RF\_model, test)

RF\_pred.prob = predict(RF\_model,test,type="prob")[, 2]

RF\_ConfMat <- confusionMatrix(RF\_pred, test[,"hd"])

RF\_ConfMat

```

```{r message = FALSE, include = FALSE}

# ROCR Curve

AUC$RF <- roc(as.numeric(test$hd),as.numeric(as.matrix((RF\_pred.prob))))$auc

Accuracy$RF <- RF\_ConfMat$overall['Accuracy']

```

```{r}

# Calculate the performance of our Random Forests Model

calculate\_stats(table(RF\_pred, test$hd))

```

### Boosted Trees

Boosted tree model (gbm) with adjusting learning rate and and trees. Boosting is a class of ensamble learning techniques for regression and classification problems. Boosting aims to build a set of weak learners (i.e. predictive models that are only slightly better than random chance) to create one ‘strong’ learner (i.e. a predictive model that predicts the response variable with a high degree of accuracy).

```{r message = FALSE, warning = FALSE}

set.seed(10, sample.kind = "Rounding")

# Control parameters for model building

objControl <- trainControl(method='cv', number=10,repeats=10, returnResamp = "all")

# Multiplying n.tree by 5 from 10 to 500 for demo purposes

gbmGrid <- expand.grid(

interaction.depth = 1:2,

shrinkage = .1, n.trees = c(10, 50, 100, 500, 1000), n.minobsinnode = 10

)

```

```{r message = FALSE, warning = FALSE}

# Fit a Boosted Model

Boost\_model <- train(hd ~ .,data=train, method='gbm', trControl=objControl, tuneGrid = gbmGrid, verbose=F)

Boost\_model

```

```{r}

# Summary of the model results with the importance plot of the predictors

summary(Boost\_model)

```

We can see that whether or not there is a reversable defect in Thalassemia (thalassma), asymptomatic chest pain (cp), and maximum heart rate achieved in beats per minute (thalach)

```{r}

# Check how much the number of trees affect the accuracy of our boosted model.

plot(Boost\_model)

```

We can see the optimal parameters are about n.trees = 50, with interaction.depth = 1.

```{r}

# Predict the Boosted Tree Model

Boost\_pred <- predict(Boost\_model, test)

Boost\_pred.prob <- predict(Boost\_model, test, type='prob')[2]

Boost\_ConfMat <- confusionMatrix(Boost\_pred, test[,"hd"])

Boost\_ConfMat

```

```{r messages = FALSE, include = FALSE}

# ROCR

AUC$Boost <- roc(as.numeric(test$hd),as.numeric(as.matrix((Boost\_pred.prob))))$auc

Accuracy$Boost <- Boost\_ConfMat$overall['Accuracy']

```

```{r}

# Calculate the performance of our Gradient Boosted Model

calculate\_stats(table(Boost\_pred, test$hd))

```

Because our Gradient Boosted Trees has a high Accuracy and Sensitivity we can use this predictive ability to find the importance of each variable in our model.

```{r}

boost\_Imp =varImp(Boost\_model, scale = FALSE)

row = rownames(varImp(Boost\_model, scale = FALSE)$importance)

row = convert.names(row)

rownames(boost\_Imp$importance)=row

plot(boost\_Imp,main = 'Variable importance for heart failure prediction with Boosted Tree')

```

### Support Vector Machine

Another approach we used to to predict the presense of the heart disease is support vector machines (SVM). SVM divides the dataset into classes with the use of hyperplanes. Below, we used SVM to create a model for our dataset. Additionally, we did k-fold cross validation with 10 folds. We found the best SVM with the tune function.

```{r, warning = FALSE, message = FALSE}

set.seed(10, sample.kind = "Rounding")

# Reformat our dataframe because SVM requires data to be strictly classification variables

feature.names=names(heart.data)

for (f in feature.names) {

if (class(heart.data[[f]])=="factor") {

levels <- unique(c(heart.data[[f]]))

heart.data[[f]] <- factor(heart.data[[f]],

labels=make.names(levels))

}

}

# Split our data into training sets (70%) and testing sets (30%)

split2 <- createDataPartition(heart.data$hd,p=0.7,list=FALSE)

train2 <- heart.data[split2,]

test2 <- heart.data[-split2,]

```

When the cost argument is small, then the margins will be wide and many support vectors will be on the margin or will violate the margin. When the cost argument is large, then the margins will be narrow and there will be few support vectors on the margin or violating the margin.

```{r message = FALSE, warning = FALSE}

# Control parameters for model building

fitControl <- trainControl(method = "cv", number = 10,

repeats = 3, # number is number of iteration, repeat the cross validation

# Estimate class probabilities

classProbs = TRUE,

returnResamp = "all",

# Evaluate performance using the following function

summaryFunction = twoClassSummary)

# Fit the Support Vector Machine model

SVM\_model <- train(hd ~ ., data = train2, method = "svmLinear2", trControl = fitControl, preProcess = c("center", "scale"),

tuneLength = 10, tuneGrid = data.frame(cost = c(.1, .25, .5, .75, 1, 2)), metric = "ROC")

SVM\_model

```

```{r}

summary(SVM\_model)

```

80 support vectors with 39 in the first class, and 41 in the second class.

```{r}

# Plot the SVM model

plot(SVM\_model)

```

We can see that the optimal cost for our SVM model is C = 0.25

```{r}

# Predict the Support Vector Machine model

SVM\_pred <- predict(SVM\_model, test2)

SVM\_pred.prob <- predict(SVM\_model, test2, type='prob')[2]

SVM\_ConfMat <- confusionMatrix(SVM\_pred, test2[,"hd"])

SVM\_ConfMat

```

```{r message = FALSE, include = FALSE}

#ROC Curve

AUC$svm <- roc(as.numeric(test2$hd),as.numeric(as.matrix((SVM\_pred.prob))))$auc

Accuracy$svm <- SVM\_ConfMat$overall['Accuracy']

```

```{r}

# Calculate the performance of our Support Vector Machine Model

calculate\_stats(table(SVM\_pred, test2$hd))

```

### K-Nearest Neighbor (KNN)

K Nearest Neighbors is a simple algorithm but works incredibly in practice that stores all the available cases and classifies the new data or case based on a similarity measure. It suggests that if the new point added to the sample is similar to the neighbor points, that point will belong to the particular class of the neighbor points.

```{r message = FALSE, warning = FALSE}

set.seed(10, sample.kind = "Rounding")

# Control parameters for model building

x = trainControl(method = "cv",

number = 10,

returnResamp = "all",

repeats = 3,

classProbs = TRUE,

summaryFunction = twoClassSummary)

# Fit a K- Nearest Neighbors Model

KNN\_model = train(hd~. , data = train2, method = "knn",

preProcess = c("center","scale"),

trControl = x,

metric = "ROC",

tuneLength = 10)

# print model results

print(KNN\_model)

```

In general, KNN algorithm uses in search applications where people looking for similar items. K in the KNN algorithm denotes the number of nearest neighbors of the new point which needed to be predicted.

```{r}

# Plot how the Number of Neighbors affect the ROC

plot(KNN\_model)

```

We can see that as the number of neighbors k = 23 the ROCR increases.

```{r}

# Predict the KNN model

KNN\_pred <- predict(KNN\_model, test2)

KNN\_pred.prob <- predict(KNN\_model, test2, type = "prob")[,2]

KNN\_ConfMat <- confusionMatrix(KNN\_pred, test2[,"hd"])

KNN\_ConfMat

```

```{r}

# Calculate the performance of our K-Nearest Neighbors Model

calculate\_stats(table(KNN\_pred, test2$hd))

```

```{r message = FALSE, include = FALSE}

#ROC Curve

AUC$knn <- roc(as.numeric(test2$hd),as.numeric(as.matrix((KNN\_pred.prob))))$auc

Accuracy$knn <- KNN\_ConfMat$overall['Accuracy']

```

#### Plot the ROC curves for our various models

```{r}

# create a prediction object

pr <- ROCR::prediction(LogReg\_pred.prob, test$hd)

prf <- performance(pr, measure = "tpr", x.measure = "fpr")

# create a data frame for TP and FP rates

dd1 <- data.frame(FP = prf@x.values[[1]], TP = prf@y.values[[1]])

# CART

pr2 <- ROCR::prediction(Tree\_pred.prob[,2], test$hd)

prf2 <- performance(pr2, measure = "tpr", x.measure = "fpr")

dd2 <- data.frame(FP = prf2@x.values[[1]], TP = prf2@y.values[[1]])

# RF

pr3 <- ROCR::prediction(RF\_pred.prob, test$hd)

prf3 <- performance(pr3, measure = "tpr", x.measure = "fpr")

dd3 <- data.frame(FP = prf3@x.values[[1]], TP = prf3@y.values[[1]])

# BOOST

pr4 <- ROCR::prediction(Boost\_pred.prob, test$hd)

prf4 <- performance(pr4, measure = "tpr", x.measure = "fpr")

dd4 <- data.frame(FP = prf4@x.values[[1]], TP = prf4@y.values[[1]])

# SVM

pr5 <- ROCR::prediction(SVM\_pred.prob, test2$hd)

prf5 <- performance(pr5, measure = "tpr", x.measure = "fpr")

dd5 <- data.frame(FP = prf5@x.values[[1]], TP = prf5@y.values[[1]])

# KNN

pr6 <- ROCR::prediction(KNN\_pred.prob, test2$hd)

prf6 <- performance(pr6, measure = "tpr", x.measure = "fpr")

dd6 <- data.frame(FP = prf6@x.values[[1]], TP = prf6@y.values[[1]])

# plot ROC curve for logistic regression

g <- ggplot() +

geom\_line(data = dd1, aes(x = FP, y = TP, color = 'LOG R')) +

geom\_line(data = dd2, aes(x = FP, y = TP, color = 'CART')) +

geom\_line(data = dd3, aes(x = FP, y = TP, color = 'RF')) +

geom\_line(data = dd4, aes(x = FP, y = TP, color = 'BOOST')) +

geom\_line(data = dd5, aes(x = FP, y = TP, color = 'SVM')) +

geom\_line(data = dd6, aes(x = FP, y = TP, color = 'KNN')) +

geom\_segment(aes(x = 0, xend = 1, y = 0, yend = 1)) +

scale\_y\_continuous(limits=c(0,1)) +

scale\_x\_continuous(limits=c(0,1)) +

ggtitle('ROC Curve') +

labs(x = 'False Positive Rate', y = 'True Positive Rate')

g + scale\_colour\_manual(name = 'Classifier', values = c('LOG R'='#E69F00', 'CART'='#009E73', 'RF'='#D55E00', 'BOOST'='#0072B2', 'SVM' = 'red' , 'KNN' = 'purple' ))

```

```{r}

# The AUC for each of our models using the ROC curve.

auc <- rbind(performance(pr, measure = 'auc')@y.values[[1]],

performance(pr2, measure = 'auc')@y.values[[1]],

performance(pr3, measure = 'auc')@y.values[[1]],

performance(pr4, measure = 'auc')@y.values[[1]],

performance(pr5, measure = 'auc')@y.values[[1]],

performance(pr6, measure = 'auc')@y.values[[1]])

rownames(auc) <- (c('LOG R', 'CART', 'RF', "BOOST", "SVM", "KNN"))

colnames(auc) <- 'Area Under ROC Curve'

round(auc, 4)

```

## Conclusion

We can see that Logistic Regression and Boosted Trees perform the best when it comes to predicting Heart Disease. While Decision Tree performs the worst due to the fact that it's a weaker classification model.

Using these models we can see that the variables that best predict the presecence of Heart Disease in patients are: Thalassemia the inherited blood disorder that affects the body’s ability to produce hemoglobin and red blood cells (thal), chest pain type (cp).

End of Code.

**Sources:**

1. Heart Disease Dataset -   
   [**https://archive.ics.uci.edu/ml/datasets/Heart+Disease**](https://archive.ics.uci.edu/ml/datasets/Heart+Disease)
2. Constructing a Confusion Matrix - [**https://www.dataschool.io/simple-guide-to-confusion-matrix-terminology/**](https://www.dataschool.io/simple-guide-to-confusion-matrix-terminology/)
3. World Health Organization Cardiovascular Disease Information - [**https://www.who.int/health-topics/cardiovascular-diseases/**](https://www.who.int/health-topics/cardiovascular-diseases/)
4. Center for Disease Control Heart Disease Information -  
   [**https://www.who.int/health-topics/cardiovascular-diseases/#tab=tab\_1**](https://www.who.int/health-topics/cardiovascular-diseases/#tab=tab_1)