Final project output

Hongkai Wang

2022-11-30

Introduction

Heart disease is the leading cause of death for all demographic groups in the United States. On average, one person would die from some form of cardiovascular conditions every 34 seconds, and heart diseases account for one out of every 5 deaths in the US in 2020. Given the severity and prevalence of heart diseases in the United States, it is important to develop accurate diagnostic approaches and provide strong guidance in the prevention, treatment, and recovery of heart diseases. One of the possible way of improving the diagnostic process and providing accurate prevention measures is by utilizing machine learning algorithms to predict the development of heart diseases given relevant clinical parameters. In this project, I will attempt to create multiple machine learning models that could predict the presence of heart diseases based on several common clinical measurements.

A data set was sourced from a page named Heart Failure Prediction Dataset. The data set contains 11 common clinical features related to cardiovascular health on a total of 918 patients. The patient information were compiled from a collection of heart disease data sets posted on the UCI Machine Learning Repository. This data set contains information sourced from Cleveland, Hungarian, Switzerland, Long Beach VA, and Stalog, and could be considered as a comprehensive description of global heart diseases with a fairly small data set. Attempts were made to increase the size of the available data, however, there is limited publications with detailed patients data. The current data set used for the project was the largest data set I was able to acquire.

The data set is composed of 11 clinical parameters and 1 target feature indicating the presence of heart diseases in a particular patient sample. The 11 clinical features include: age, sex, type of chest pain experience (includes Typical Angina, Atypical Angina, Non-Angina pain, no chest pain), resting blood pressure, level of serum cholesterol measured in millimeters of Mercury, level of fasting blood suger (denoted 1 if fasting blood suger >= 120 mg/dl, 0 otherwise), resting electrocardiogram results (categorized by Normal, ST-T wave abnormal, Left Ventricular Hypertrophy), maximum heart rate achieved, presence of exercise-induced angina (reported as yes or no), numerical value of ST value (denoted as old peak), slope of peak exercise ST segment (defined as upsloping, flat, or downsloping). There is a total of 6 categorical predictor covariates, and 5 continous numerical predictor covariates. Here is a name of each column defined in the data set, in the order shown above.

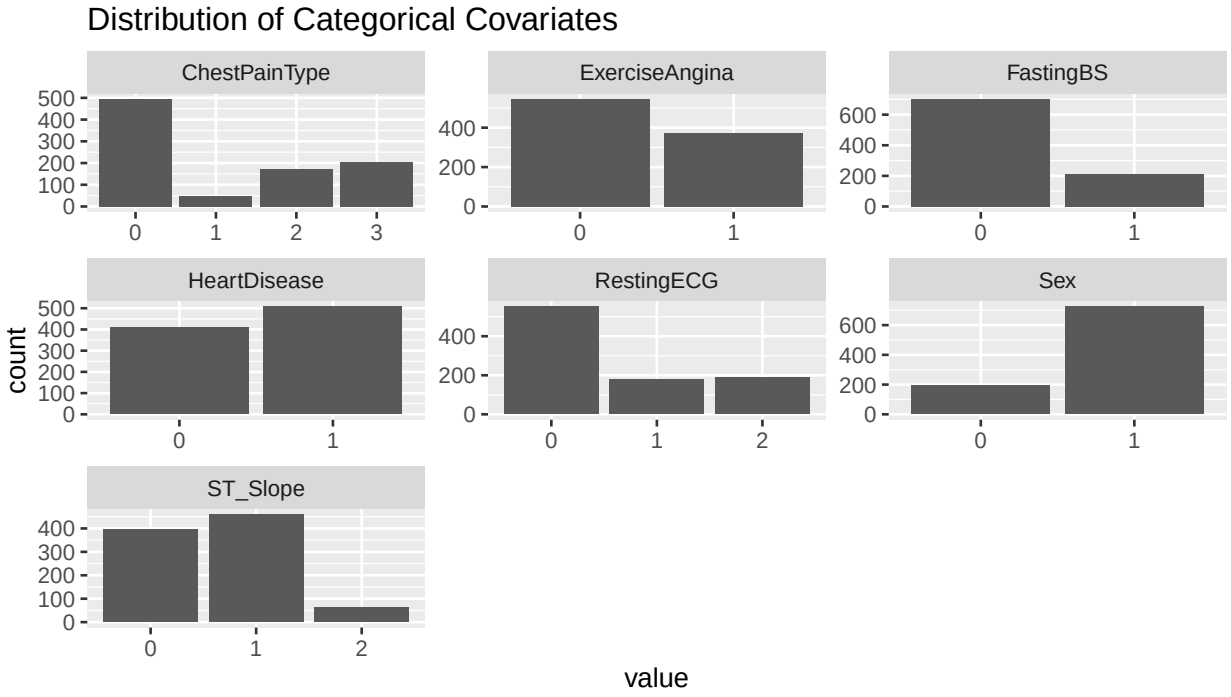
```
##  [1] "Age"            "Sex"            "ChestPainType"  "RestingBP"
##  [5] "Cholesterol"    "FastingBS"      "RestingECG"     "MaxHR"
##  [9] "ExerciseAngina" "Oldpeak"        "ST_Slope"       "HeartDisease"
```

Data transformation was conducted on all categorical covariates into the following numerical coding.

1. sex: female = 0, male = 1
2. ChestPainType: Asymptomatic = 0,
3. RestingECG: Normal = 0, ST = 1, LVH = 2

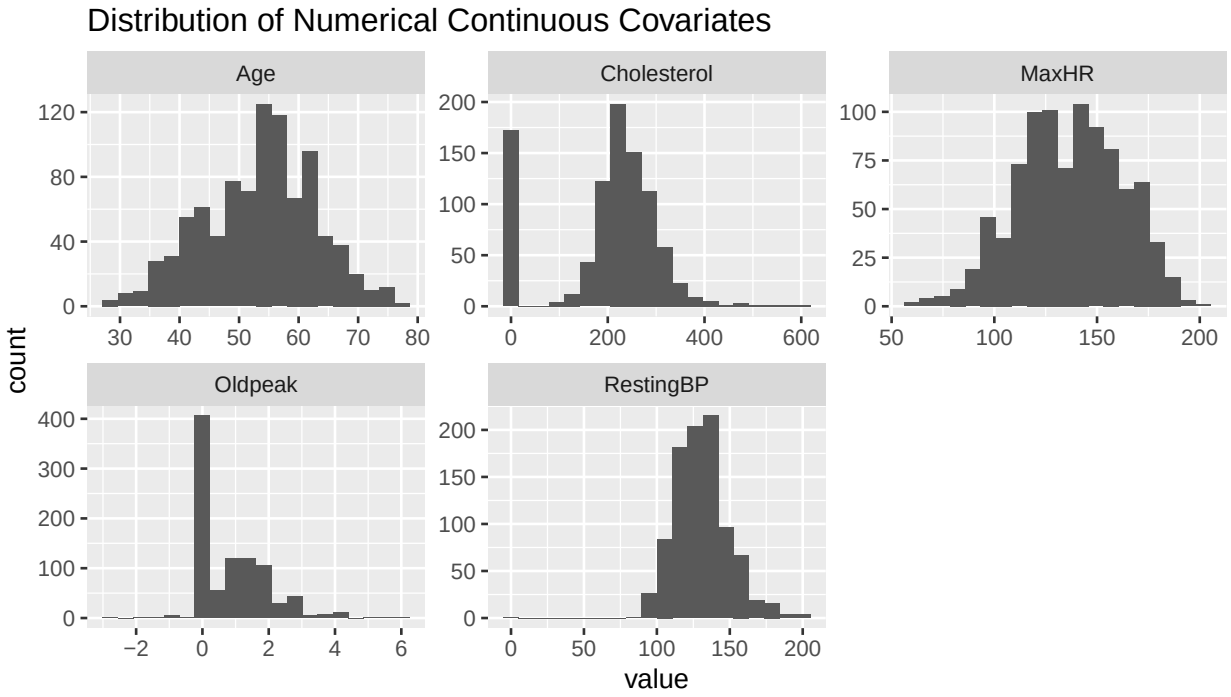
4. ExerciseAngina: N = 0, Y = 1
5. ST_Slope: up = 0, flat = 1, down = 2

Bar graphs were constructed to reveal the distribution of the categorical predictor covariates. The results shows that the binary outcome of heart diseases is balanced, and there is not need for re sampling to balance the data set. There are two categories that is somewhat concerning, we see that ChestPainType and ST slope both have categories that has less than 100 samples in their categories. However, since they are not the outcome classification, I will hold off on changing any of the existing data set.

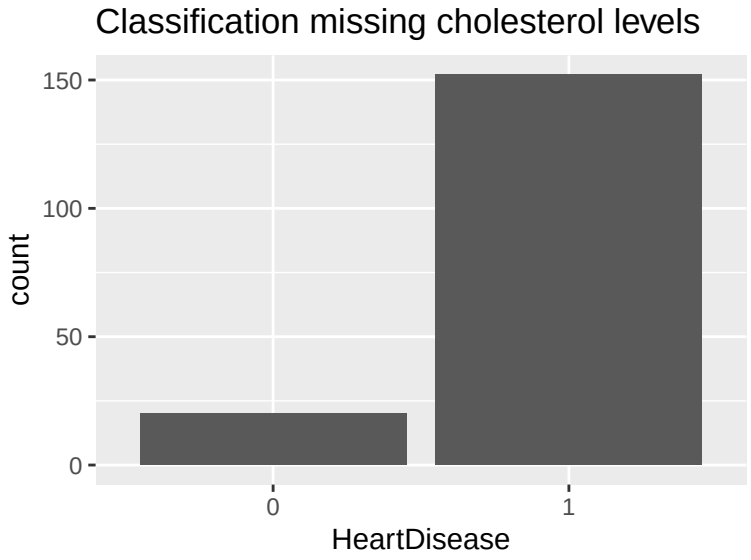


Histograms were constructed to reveal the distribution of the continuous numerical variables. Age, MaxHR, and restingBP all appear to be normally distributed with only a few outlier. There is a significant amount of 0 values in the cholesterol categories, and that was deemed to be missing data for some samples. These missing data will be removed from the machine learning analysis due to a lack of information on possible ways to fill these in. A possible approach is to fill the data with some form of regression analysis. However, that could affect our classification of heart disease, as there is no clear explanation on underlying reason for missing data.

At the same time, we see that the old peak covariates is mostly 0, probably due to the fact that it is a data not collected in all of the source studies. Thus, I decided to exclude old peak from further analysis.



Further analysis of the data shows that most of the missing data in cholesterol levels are classified as heart disease patients, and data filling techniques could result in bias in our classification model. Thus, data samples with missing cholesterol levels are removed from the data set.



Another round of EDA was conducted after removing the missing values, and no significant changes ind data distribution was observed. Normalization was then conducted on the continuous variables to achieve better machine learning out comes. The plots for the final distribution of the data is included in the appendix.

Methods

This project will utilize several machine learning classification methods to conduct predictions on the presence of heart disease based on the clinical parameters. Cross validations will be conducted before testing of the machine learning algorithms to create test and train data sets. Cross validation is essential in effective machine learning as it can significantly reduce overfitting and gives us a proper approach of evaluating our fitted models.

There are several machine learning algorithms that could be useful for this project. A logistic regression model provides a foundation and baseline for our project. Logistic regression is chosen as it is one of the tried and true methods for classifying binary outcomes. The excellent explainability of the model provides important insight into the factors that has a strong influence on heart diseases. K-Nearest Neighbor is another helpful algorithm in data classification through data clustering. This method is used due to ease of fitting a model, and easy hyper parameter tuning with the value of k. Random forest is another helpful ML algorithm for this project. Random forests utilizes bagging in regression trees to reduce the usual issue of high variance in decision trees.

In summary, the three methods that I have chosen for this project, logistic regression, KNN, and random forests, cover three distinct type of classification methods, which are regression analysis, clustering, and decision tree based classification. It is a goal of this project to explore the advantage and disadvantage of utilizing different types of machine learning algorithms for medical data.

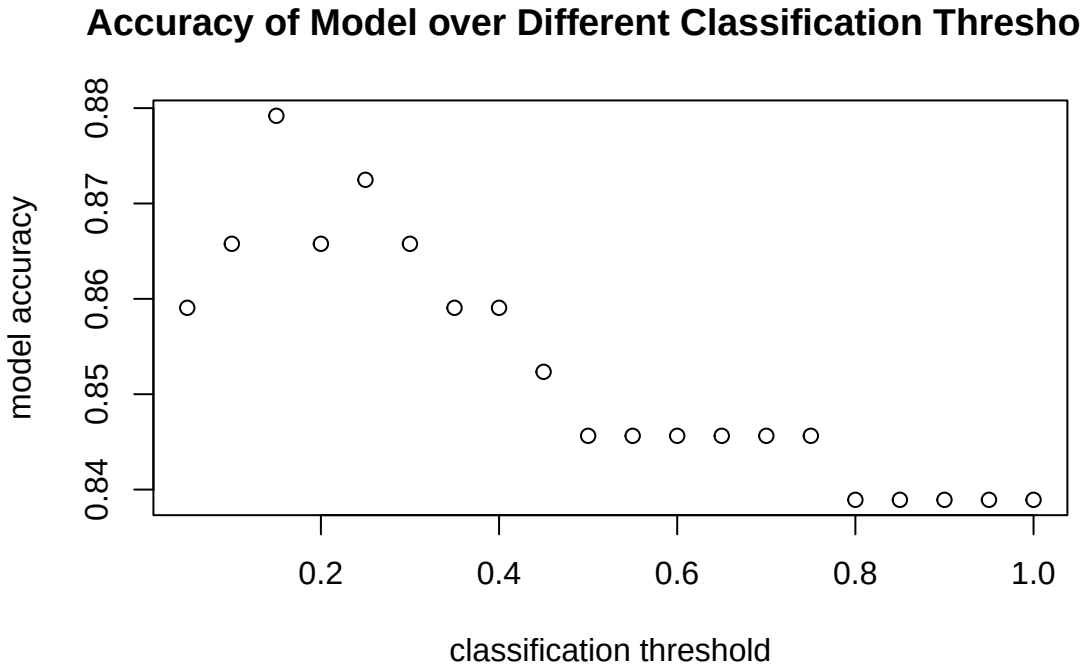
Results

A training-test 8:2 data split was conducted before any model fitting processes. There was no significant deviation from the distribution of the original data set. Visual inspection of the data distribution is included in the code section.

logisitc regression

A logistic regression model was first fitted onto the training data. Different combination of the covariates were tested for a best fitted model, and several covariates didn't produce a statistically significant fit for our model. Several classification threshold for the probability output is also tested. Interestingly, cholesteral was proven to be a insignificant predictor within the full model, but it is a significant predictor in a reduced model consisted only of itself. This possibility due to a stronger predictive power from other covariates that contains similar information to the cholesterol level. As a result, cholesterol was not included in the final model.

We also need to determine a well-suited classification threshold for the output probability. Below shows the model accuracy at different classification thresholds.



Based on the information of the plot, we can see that the model accuracy is the highest when classification threshold is set to be 0.15. We will now use this classification threshold to explore the predictability of the model by looking at the confusion matrix comparing the prediction outcomes of the test data set.

```
##           Reference
## Prediction  0  1
##          0 69  9
##          1  9 62
```

```
##       Accuracy          Kappa  AccuracyLower  AccuracyUpper   AccuracyNull
##   8.791946e-01   7.578548e-01   8.158081e-01   9.268075e-01   5.234899e-01
## AccuracyPValue  McnemarPValue
##   1.988143e-20   1.000000e+00
```

From the confusion matrix output, we see that logistic regression model achieves an respectable accuracy of 87.9%. From a clinical point of view, this is not very helpful as false negative will do significant damage to the diagnostic prediction process. On the other hand, false positive is not as bad, as further diagnosis by a physician could clear up confusion and perhaps serve as precautionary tale for the patients. Our current logistic regression model is equally likely to have a false positive results and a false negative result.

One way to reduce the false negative rate is to reduce the classification threshold of the model. I will set the threshold to be .05 to see if there is any significant differences between the recall of the models.

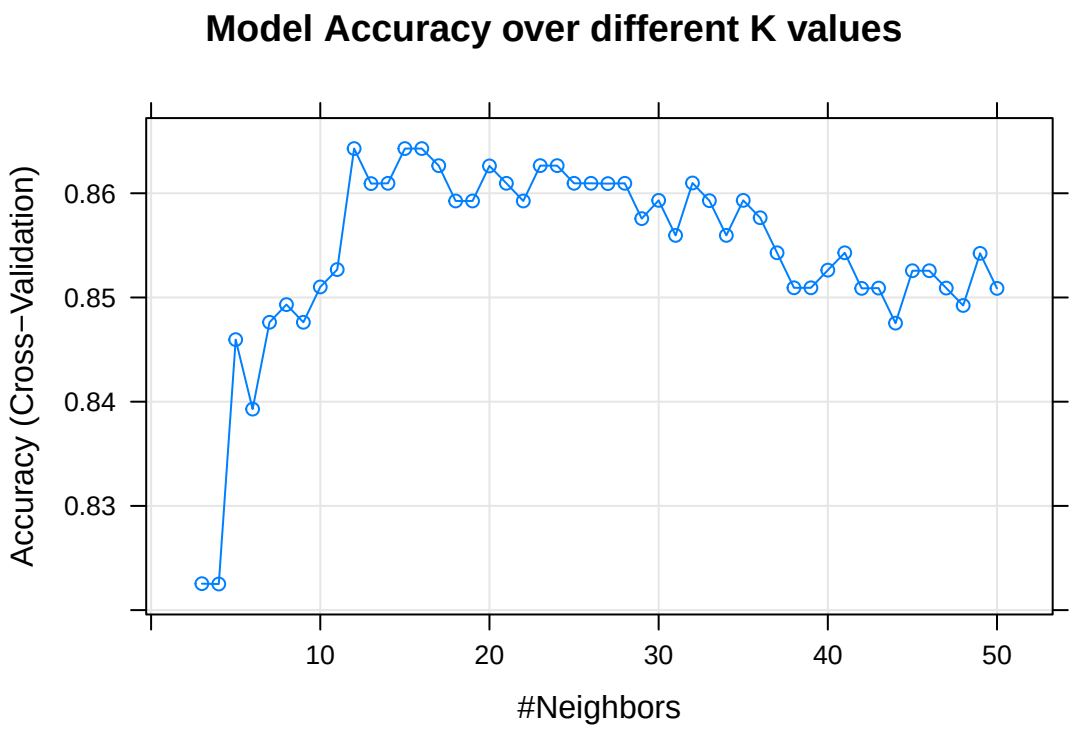
```
##           Reference
## Prediction  0  1
##          0 66  9
##          1 12 62
```

```
##       Accuracy          Kappa  AccuracyLower  AccuracyUpper   AccuracyNull
##   8.590604e-01   7.180319e-01   7.926613e-01   9.105905e-01   5.234899e-01
## AccuracyPValue  McnemarPValue
##   4.271252e-18   6.625206e-01
```

Based on the confusion table, there is no difference in the false negative predictions. We did lose out on the accuracy of the model as we have more false positives. Thus, we can conclude that the logistic regression model has the best performance at a classification threshold of 0.05.

K-Nearest Neighbors

Next, we are going to see if clustering methods provide a better result when compared to the logistic regression model. The K-Nearest neighbor model was established with an internal cross validation process and the test/train ratio was set to 1:9. All 10 remaining covariates were fed into the model in the hopes of finding new correlations between the clinical parameters and heart diseases. However, comparison of different input parameters shows that it is better to stick with the covariates that was significant in the previous logistic regression model and we can achieve better model accuracy. K values ranging from 3 to 50 were used to create different KNN models.



We can see that the highest accuracy of the model is obtained at K = 11, and any further clustering would only increase the noise in the model and reduce accuracy. Now let's evaluate the model outcome with the confusion matrix again.

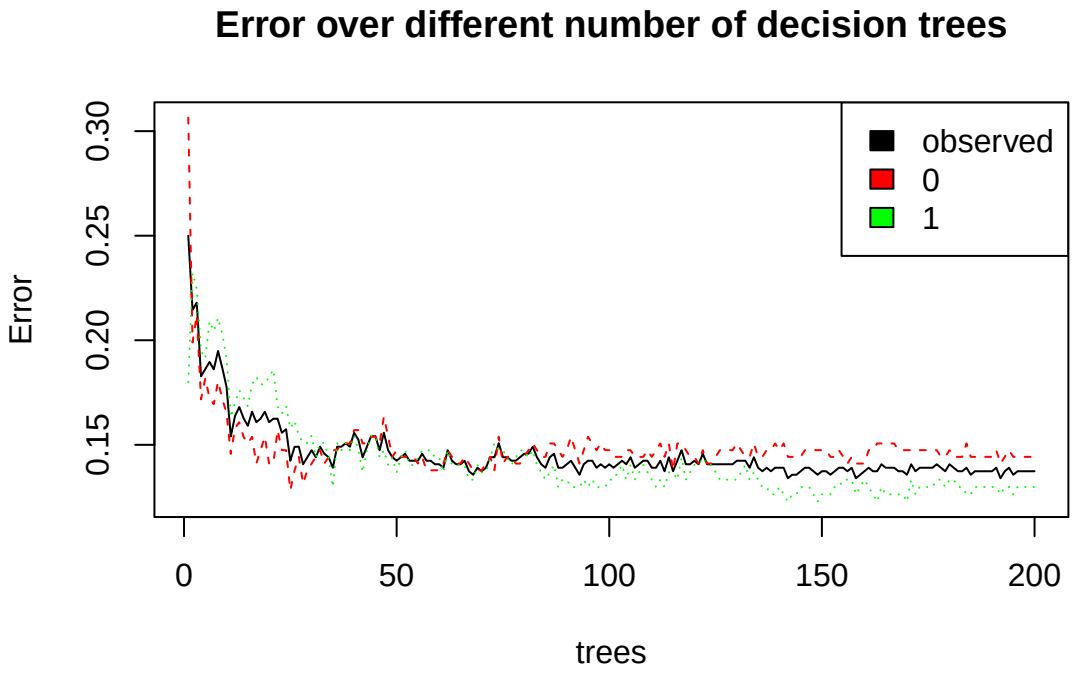
```
##            Reference
## Prediction  0  1
##          0 65  8
##          1 13 63
```

```
##       Accuracy          Kappa  AccuracyLower  AccuracyUpper   AccuracyNull
##   8.590604e-01   7.183872e-01   7.926613e-01   9.105905e-01   5.234899e-01
## AccuracyPValue  McnemarPValue
##   4.271252e-18   3.827331e-01
```

We can see that the KNN method has essentially the same recall on the test set when compared to the logistic regression model. However, the KNN model does lose out on accuracy as more false positives were made during the prediction process. The KNN model doesn't allow for hyperparameter tuning, aside from the selection of the K value. In our case, we can see that the optimal range for K is between 10 - 15. Judging from the confusion table result, there isn't too much improvement to be made to the KNN model either. Let's now move onto the final model for this project, random forest.

Random Forest

We saw that the KNN method produced a very similar model to logistic regression in terms of model performance. Remember, the evaluation of our model should be focused on having as little false negative as possible, as a false positive could be corrected by following diagnosis, but a false negative would result in worse outcomes. Let's now switch to a different types of classification method. Random forests depends on creating many different decision trees, and produce prediction outcomes based on the average of all decision trees.



After fitting a random forest model on our heart disease data, we can first evaluate the performance of the model by looking at the loss function of the model given different number of decision trees. The graph shows the magnitude of errors created during the cross validation process for the observed values, observed positive, and observed negatives. We see that the model reaches minimum error after about 100 trees were added to the model. Thus, we are fairly confident that the random forest model has reached the lowest error term possible for the model, and we don't have to increase the number of trees created for the model. The model

also predicts a positive classification (presence of heart disease) better than the a negative classification, which is exactly what we want out of our model.

We shall now take a look at the model performance with our test training set.

```
##           Reference
## Prediction  0  1
##          0 66 10
##          1 12 61
```

```
##       Accuracy          Kappa  AccuracyLower  AccuracyUpper   AccuracyNull
##   8.523490e-01   7.044184e-01   7.850301e-01   9.050956e-01   5.234899e-01
## AccuracyPValue  McnemarPValue
##   2.289340e-17   8.311704e-01
```

Based on the confusion matrix output, we can see that the random forest method didn't improve upon the results of the KNN method. We achieved similar accuracy results and slightly lower recall. Given the relatively small training data set, there might be very little difference between choosing the KNN and the random forest methods in this specific case.

Conclusion

In this project, I produced three different machine learning models using cross validation for the classification of heart diseases base on 11 common clinical parameters. The three models, logistic regressions, K Nearest Neighbors, Random Forest, produced very similar results with the logistic regression models having a small advantage margin over the other two models.

The performance of the three models after tuning is shown in the table below.

Model	Accuracy	Recall
Logistic Regression	87.9%	0.873
KNN	85.9%	0.887
Random Forest	85.2%	0.859

The comparison between the three models shows essentially no advantage for any methodologies. Each of the model achieved a similar accuracy and recall given the same training and testing set. Overall, I would consider this project to be successful in illustrating the usefulness of machine learning algorithms in creating heart disease conditions predictions.

At the same time, there are several improvements that could be made to this project.

1. We have a very limited data set. The full data set size of around 700 patient sample is simply not enough for most modern machine learning applications. The accuracy of the model could potentially increase about 90% if we have a larger data set. However, well defined clinical data is very hard to come by on the internet, and it is vital to have a reputable data source. One potential solution to this issue is through collaboration with nearby Boston health care infrastructures.
2. The data set have some issue with the definition of the data feature. A better understanding of cardiovascular disease and symptom will benefit the model prediction significantly. However, due to the time restraints of the model, no outside consult was made during the development of the project.
3. Other machine learning algorithms could be explored given a more extended time. One of the useful model that I can use is XGBoost. XGBoost is a boosting decision tree method that handles missing data very well. I would love to explore the performance of XGBoost in the future on the same data set, and see if a higher performing model could be made.

Reference

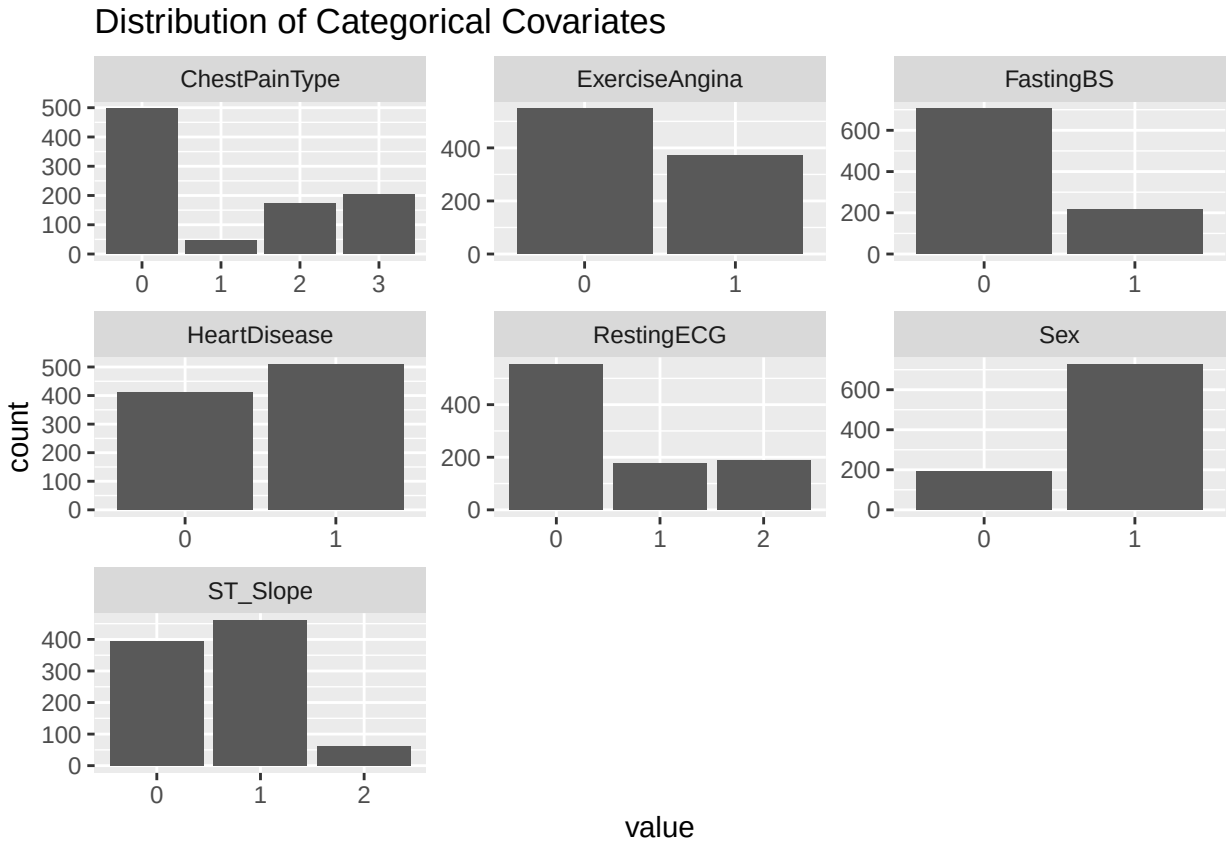
1. Kaggle data source: https://www.kaggle.com/datasets/fedesoriano/heart-failure-prediction
2. CDC report on heart attack stats. **click link here**

Appendix

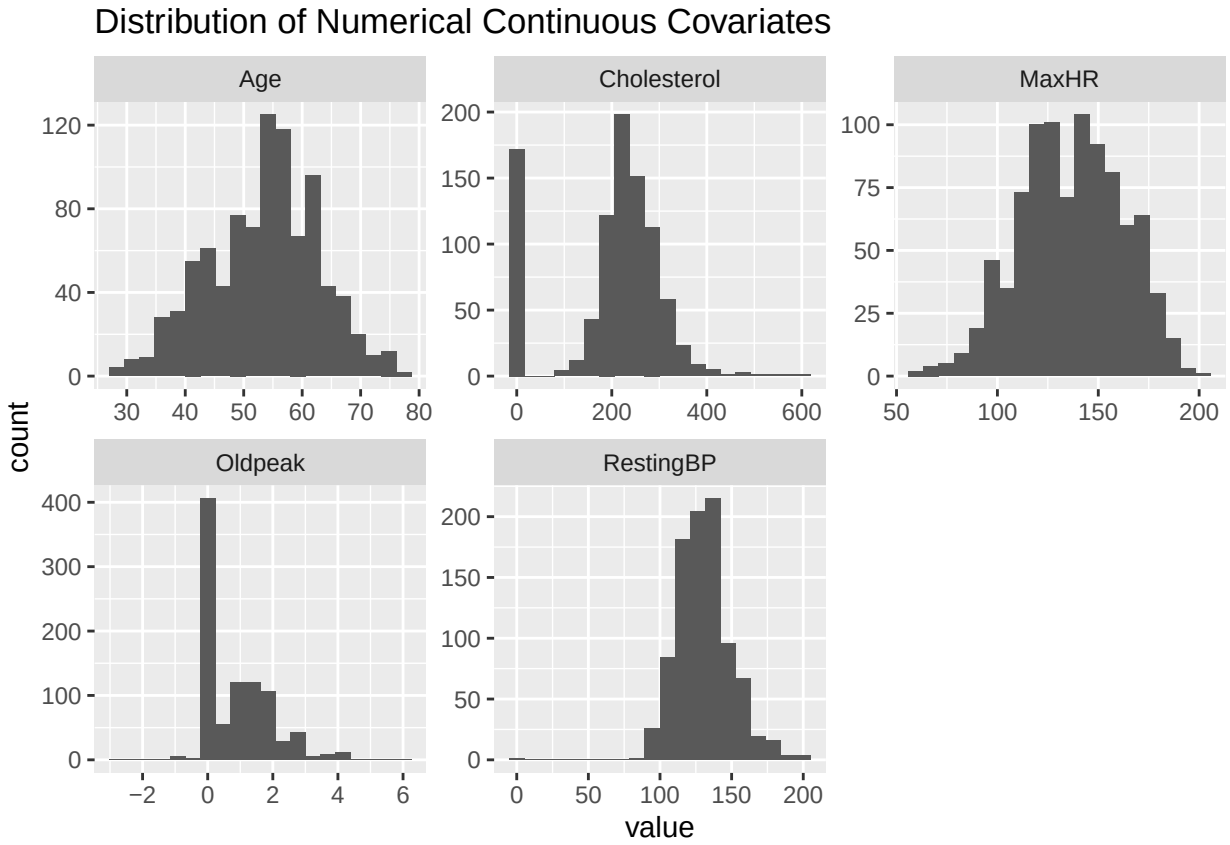
```
knitr::opts_chunk$set(echo = TRUE)
# loading necessary libraries
library(tidyverse)
library(caret)
library(ggplot2)
library(pastecs)
library(lmtest)
library(Hmisc)

heart = read.csv("heart.csv")
colnames(heart)
```

```
##  [1] "Age"            "Sex"            "ChestPainType"  "RestingBP"
##  [5] "Cholesterol"    "FastingBS"      "RestingECG"     "MaxHR"
##  [9] "ExerciseAngina" "Oldpeak"        "ST_Slope"       "HeartDisease"
```

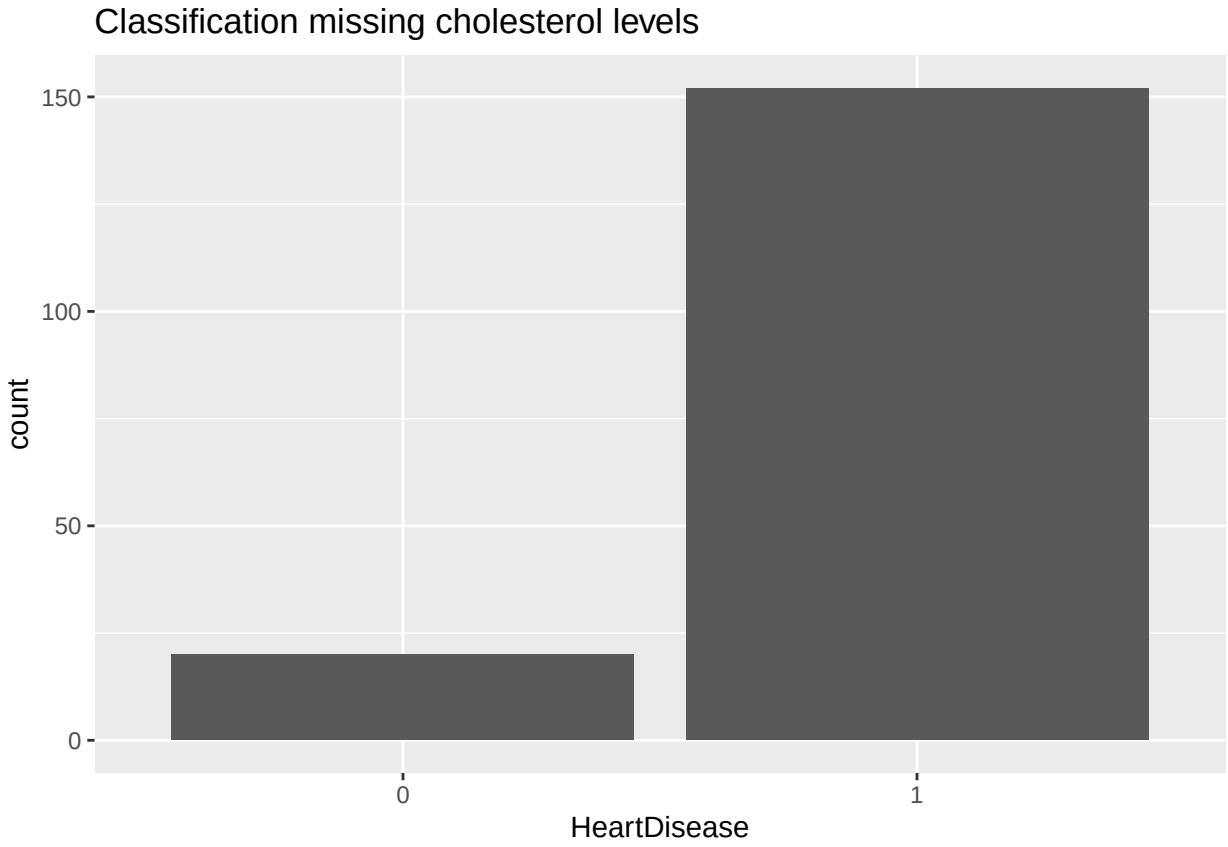
```
# number coding categorical predictors
heart = heart%>% mutate(Sex = case_when(Sex == "F" ~ 0, T ~ 1)) %>% mutate(ChestPainType = case_when(Ch
# factoring the categorical values for easy
heart$Sex = factor(heart$Sex)
heart$ChestPainType = factor(heart$ChestPainType)
heart$FastingBS = factor(heart$FastingBS)
heart$RestingECG = factor(heart$RestingECG)
heart$ExerciseAngina = factor(heart$ExerciseAngina)
heart$ST_Slope = factor(heart$ST_Slope)
heart$HeartDisease = factor(heart$HeartDisease)
cat = heart %>% select(Sex, ChestPainType, FastingBS, RestingECG, ExerciseAngina, ST_Slope, HeartDiseas
cat_long = cat %>% pivot_longer(colnames(cat)) %>% as.data.frame()

cat_long %>% ggplot(aes(value)) + geom_bar() + facet_wrap(~ name, scales = "free") + ggtitle("Distributi
```



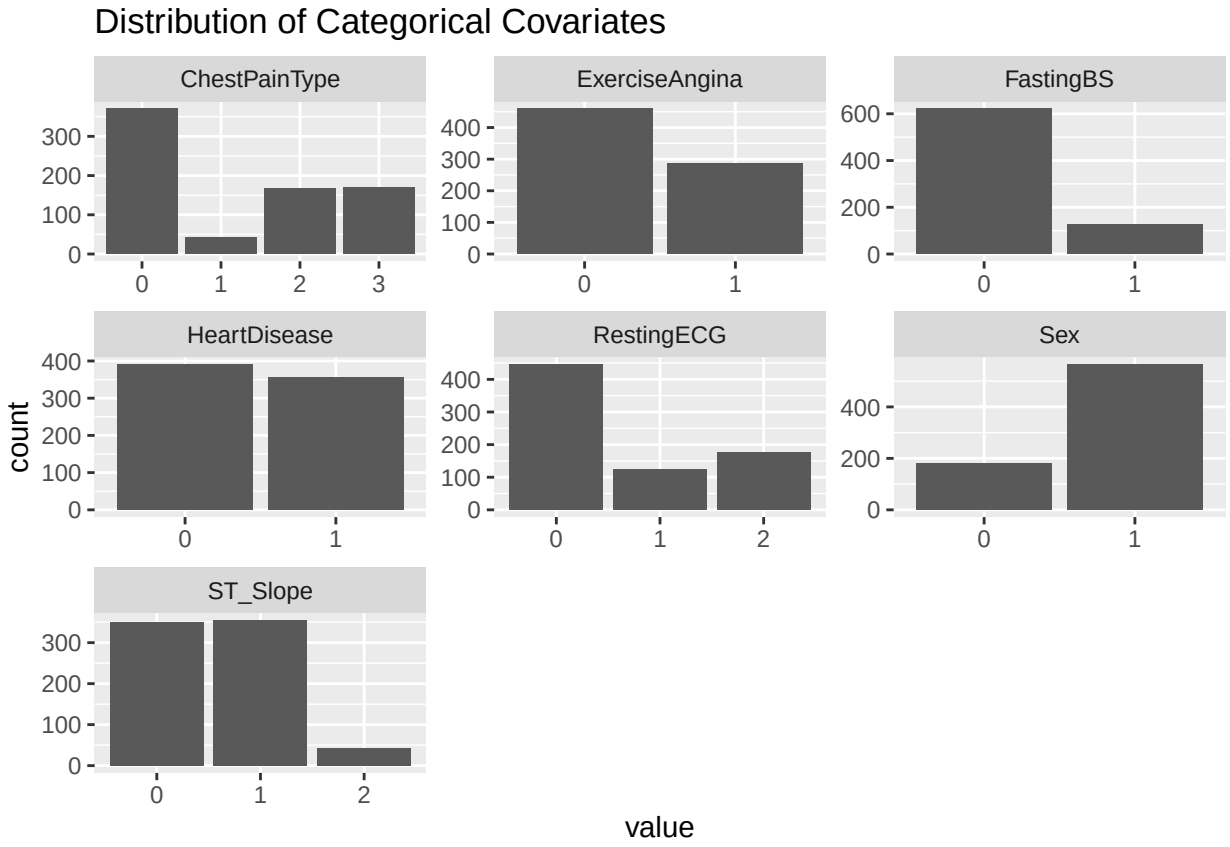

```
num = heart %>% select(Age, RestingBP, Cholesterol, MaxHR, Oldpeak)
num_long = num %>% pivot_longer(colnames(num)) %>% as.data.frame()

num_long %>% ggplot(aes(value)) + geom_histogram(bins = 20) + facet_wrap(~ name, scales = "free") + ggt
```



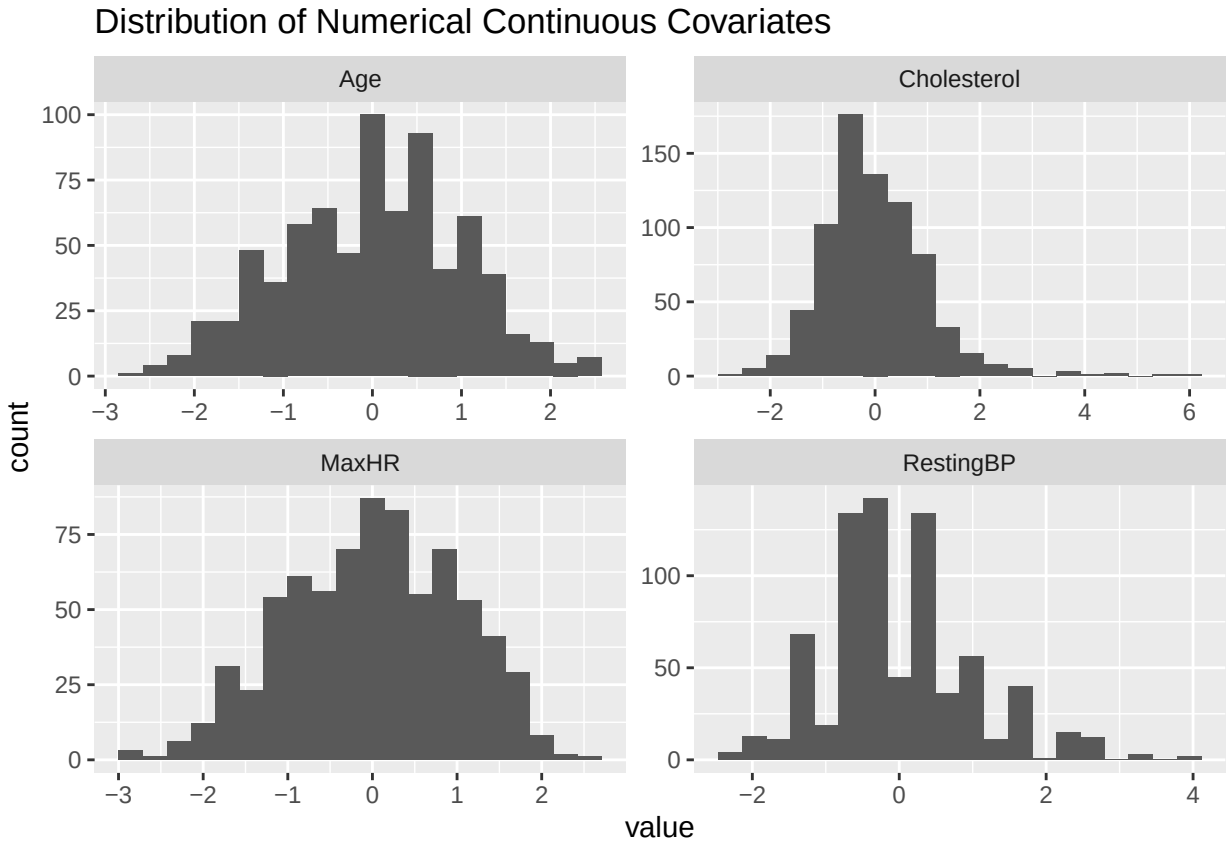

```
heart %>% filter(Cholesterol == 0) %>% select(HeartDisease) %>% ggplot(aes(HeartDisease)) + geom_bar()+
```



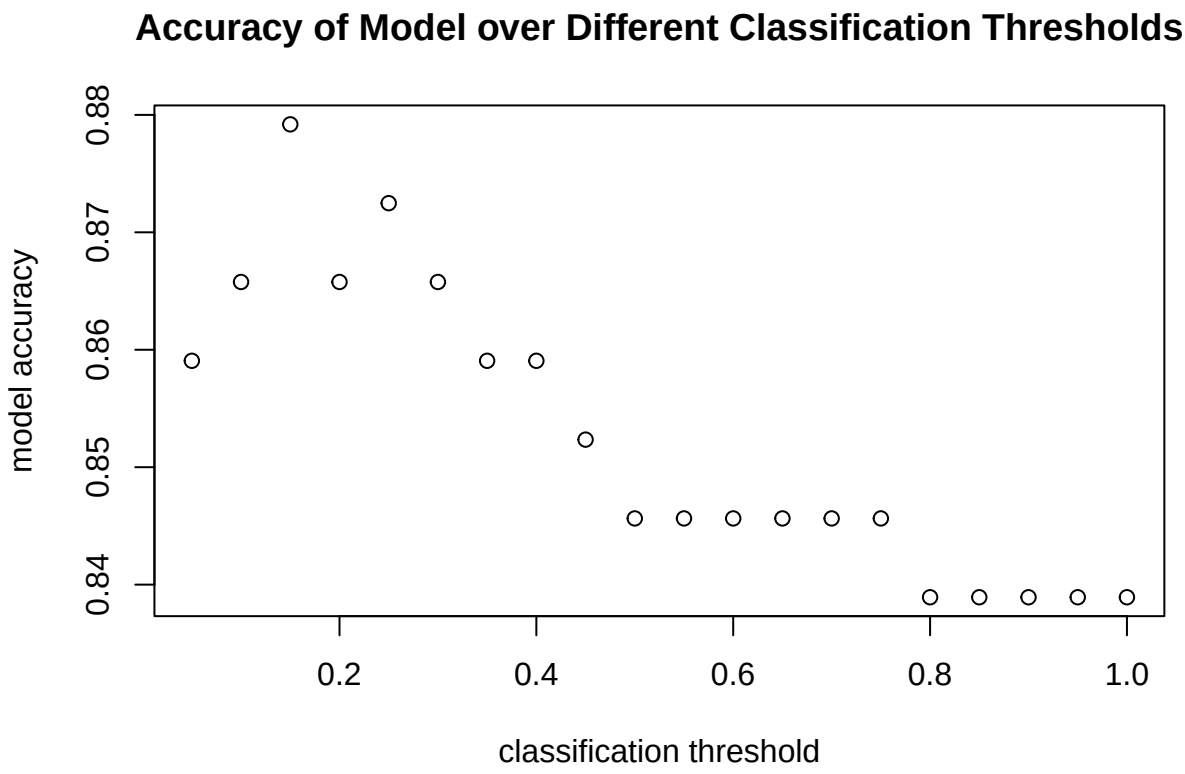

```
cat = dat %>% select(Sex, ChestPainType, FastingBS, RestingECG, ExerciseAngina, ST_Slope, HeartDisease)
cat_long = cat %>% pivot_longer(colnames(cat)) %>% as.data.frame()

cat_long %>% ggplot(aes(value)) + geom_bar() + facet_wrap(~ name, scales = "free") + ggtitle("Distributi
```




```
num = dat %>% select(Age, RestingBP, Cholesterol, MaxHR)
num_long = num %>% pivot_longer(colnames(num)) %>% as.data.frame()

num_long %>% ggplot(aes(value)) + geom_histogram(bins = 20) + facet_wrap(~ name, scales = "free") + ggt
```




```
dat = heart %>% filter(Cholesterol != 0)
dat = dat %>% mutate(Age = scale(Age), RestingBP = scale(RestingBP), Cholesterol = scale(Cholesterol), M

set.seed(110)
sample = createDataPartition(dat$HeartDisease, p = 0.8, list = F)

training_set = dat[sample,]
test_set = dat[-sample,]

mod2.1 = glm(HeartDisease ~ Age + Sex + ChestPainType + ExerciseAngina +ST_Slope, data = training_set, f

pred = predict(mod2.1, test_set)

thresh = seq(0.05, 1, 0.05)
accu = vector()
for (i in thresh){
  predicted = case_when(pred >= i ~ 1, T ~ 0)
  tmp = confusionMatrix(as.factor(predicted),as.factor(test_set$HeartDisease), mode = "everything")
  accu = rbind(accu, tmp$overall[1])
}

plot(thresh, accu, main = "Accuracy of Model over Different Classification Thresholds", xlab = "classifi
```




```
predicted = case_when(pred >= 0.15 ~ 1, T ~ 0)
confuse = confusionMatrix(as.factor(predicted),as.factor(test_set$HeartDisease), mode = "everything")

confuse$table
```

```
##           Reference
## Prediction  0  1
##          0 69  9
##          1  9 62
```

```
confuse$overall
```

```
##       Accuracy          Kappa  AccuracyLower  AccuracyUpper   AccuracyNull
##   8.791946e-01   7.578548e-01   8.158081e-01   9.268075e-01   5.234899e-01
## AccuracyPValue  McnemarPValue
##   1.988143e-20   1.000000e+00
```

```
predicted = case_when(pred >= 0.05 ~ 1, T ~ 0)
confuse = confusionMatrix(as.factor(predicted),as.factor(test_set$HeartDisease), mode = "everything")

confuse$table
```

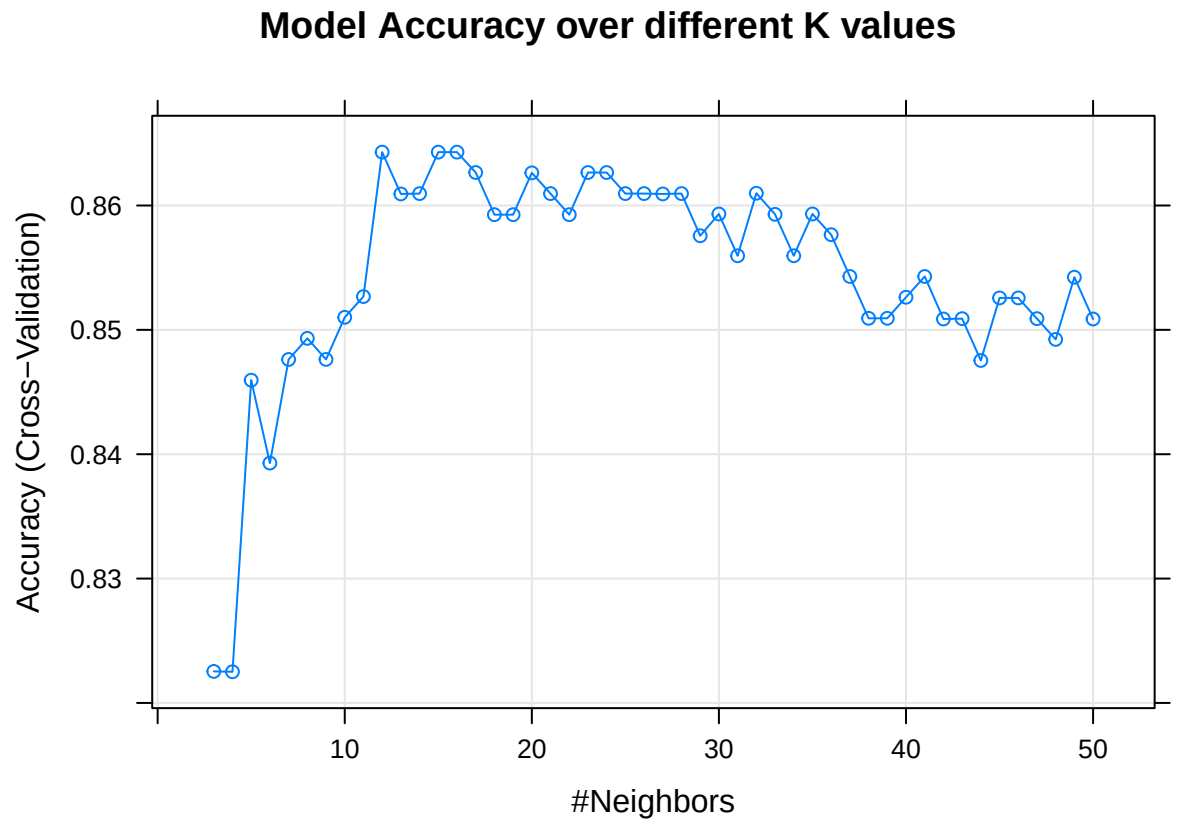
```
##           Reference
## Prediction  0  1
```

```
##          0 66  9
##          1 12 62
```

```
confuse$overall
```

```
##       Accuracy          Kappa  AccuracyLower  AccuracyUpper   AccuracyNull
##   8.590604e-01   7.180319e-01   7.926613e-01   9.105905e-01   5.234899e-01
## AccuracyPValue  McnemarPValue
##   4.271252e-18   6.625206e-01
```

```
set.seed(100)
control <- trainControl(method = "cv", number = 10, p = .9)
train_knn <- train(HeartDisease ~ Age + Sex + ChestPainType + ExerciseAngina +ST_Slope, data = training
                   method = "knn",
                   tuneGrid = data.frame(k = seq(3,50,1)),
                   trControl = control)
plot(train_knn, main = "Model Accuracy over different K values")
```




```
pred = predict(train_knn, test_set)
confuse = confusionMatrix(as.factor(pred),as.factor(test_set$HeartDisease), mode = "everything")

confuse$table
```

```
##          Reference
```

```
## Prediction  0  1
##          0 65  8
##          1 13 63
```

```
confuse$overall
```

```
##       Accuracy          Kappa  AccuracyLower  AccuracyUpper   AccuracyNull
##   8.590604e-01   7.183872e-01   7.926613e-01   9.105905e-01   5.234899e-01
## AccuracyPValue  McnemarPValue
##   4.271252e-18   3.827331e-01
```

```
set.seed(100)
control <- trainControl(method="cv", number = 5)
grid <- data.frame(mtry = c(1, 3, 5, 7))


train_rf <- train(HeartDisease ~ ., data = training_set,
                  method = "rf",
                 ntree = 200,
                  tuneGrid = grid,
                  trControl = control)
plot(train_rf$finalModel, main  = "Error over different number of decision trees", col = c("black","red
legend(x = "topright", legend = c("observed", "0", "1"), col = c("black","red", "green"), fill= c("blac
```

```r
pred = predict(train_rf, test_set)
confuse = confusionMatrix(as.factor(pred),as.factor(test_set$HeartDisease), mode = "everything")

confuse$table
```

```
##           Reference
## Prediction  0  1
##          0 66 10
##          1 12 61
```

```r
confuse$overall
```

```
##       Accuracy          Kappa  AccuracyLower  AccuracyUpper   AccuracyNull
##   8.523490e-01   7.044184e-01   7.850301e-01   9.050956e-01   5.234899e-01
## AccuracyPValue  McnemarPValue
##   2.289340e-17   8.311704e-01
```