Machine Learning Engineer Nanodegree

Capstone Project

Bavly Abdelmasih
December 31st, 2050

I. Definition

Project Overview

Heart disease is the leading cause of death for people of most ethnicities in the United States, including African Americans, Hispanics, and whites. For American Indians or Alaska Natives and Asians or Pacific Islanders, heart disease is second only to cancer. In the United States, the Centers for Disease Control and Prevention are a good resource for information about heart disease. According to their website About 610,000 people die of heart disease in the United States every year–that's 1 in every 4 deaths.

So predicting heart disease is an critical solution for this problem trying to save the life of many people by tracking the health state of people trying to predict if they are likely to be suffer from heart disease in the early future we will use kaggle dataset that collected from 300 person about their historical health state and who has suffer from heart disease to design our model

Problem Statement

The goal is to design a model to predict whether patients have heart disease or not. Data used is the UCI Heart Disease dataset obtained from Kaggle (https://www.kaggle.com/ronitf/heart-disease-uci). The tasks that involved are the following:

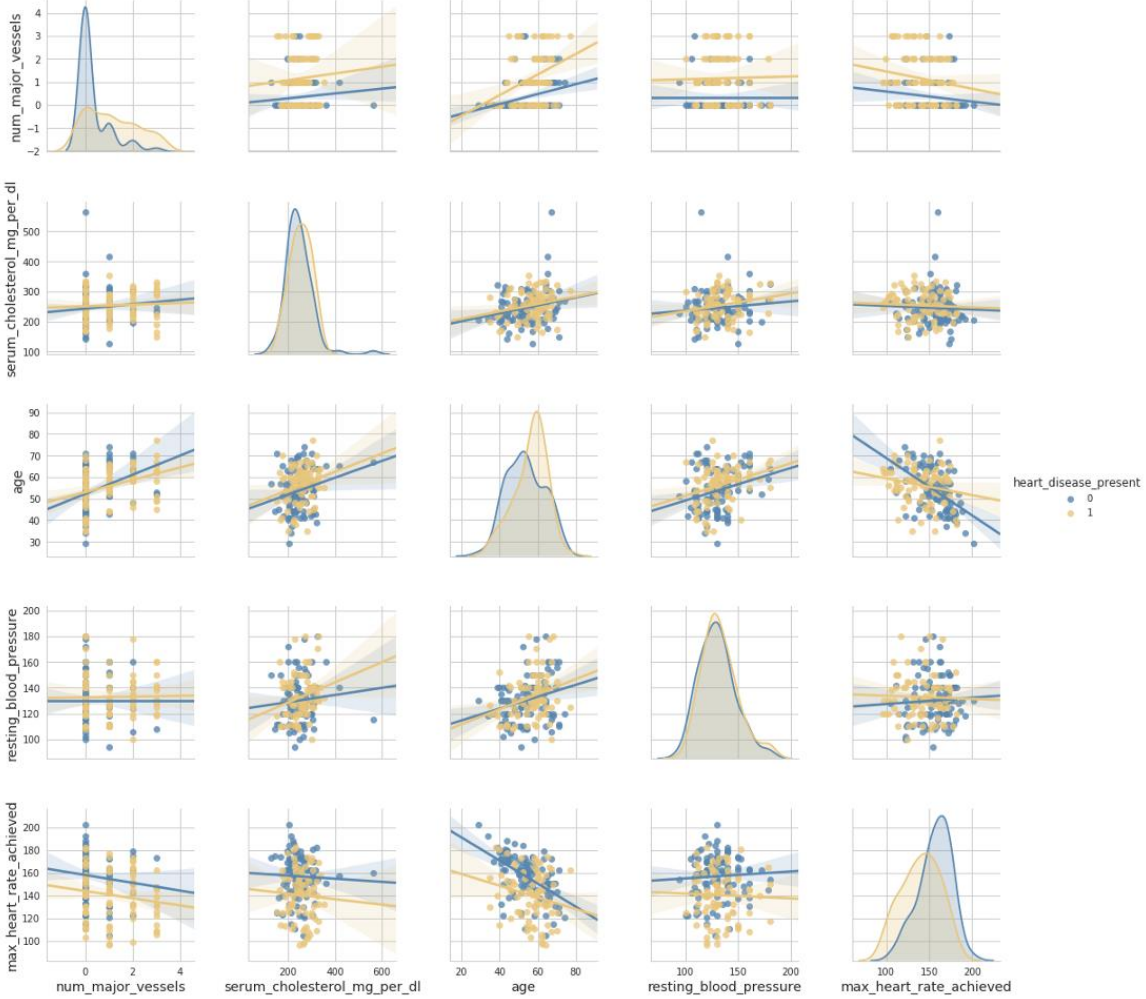
1- Download heart-disease-uci dataset

2- Exploratory analyses were conducted to get a feel for the data and visualize the differences between patients who have heart disease and those who don't on each of the independent variables.

3- Splitting dataset into train and test sets and a classification models were built and improved using gridsearch and lasso optimizer

Metrics

The evaluation metrics proposed are appropriate given the context of the data, the problem statement, and the intended solution. The performance of each classification model is evaluated using three statistical measures; classification accuracy, sensitivity and specificity. It is using true positive (TP), true negative (TN), false positive (FP) and false negative (FN). The percentage of Correct/Incorrect classification is the difference between the actual and predicted values of variables. True Positive (TP) is the number of correct predictions that an instance is true, or in other words; it is occurring when the positive prediction of the classifier coincided with a positive prediction of target attribute. True Negative (TN) is presenting a number of correct predictions that an instance is false, (i.e.) it occurs when both the classifier, and the target attribute suggests the absence of a positive prediction. The False Positive (FP) is the number of incorrect predictions that an instance is true. Finally, False Negative (FN) is the number of incorrect predictions that an instance is false.



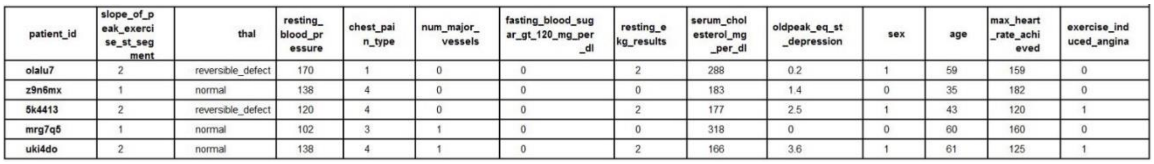
II. Analysis

Data Exploration

This database contains 76 attributes, but all published experiments refer to using a subset of 14 of them collected from 303 persons, where the patient_id column is a unique and random identifier. The remaining 13 features are described in the section below.

- slope_of_peak_exercise_st_segment (type: int): the slope of the peak exercise ST segment, an electrocardiography read out indicating quality of blood flow to the heart
- thal (type: categorical): results of thallium stress test measuring blood flow to the heart, with possible values normal, fixed_defect, reversible_defect
- resting_blood_pressure (type: int): resting blood pressure
- chest_pain_type (type: int): chest pain type (4 values)
- num_major_vessels (type: int): number of major vessels (0-3) colored by flourosopy
- fasting_blood_sugar_gt_120_mg_per_dl (type: binary): fasting blood sugar > 120 mg/dl
- resting_ekg_results (type: int): resting electrocardiographic results (values 0,1,2)
- serum_cholesterol_mg_per_dl (type: int): serum cholestoral in mg/dl
- oldpeak_eq_st_depression (type: float): oldpeak = ST depression induced by exercise relative to rest, a measure of abnormality in electrocardiograms
- sex (type: binary): 0: female, 1: male
- age (type: int): age in years
- max_heart_rate_achieved (type: int): maximum heart rate achieved (beats per minute)
- exercise_induced_angina (type: binary): exercise-induced chest pain (0: False, 1: True)

By analyzing the data I found that dataset is similar have a lot of similar occurrence and is more balanced and no missing data



patient_id	slope_of_peak_exercise_st_segment	thal	resting_blood_pressure	chest_pain_type	num_major_vessels	fasting_blood_sugar_gt_120_mg_per_dl	resting_ekg_results	serum_cholesterol_mg_per_dl	oldpeak_eq_st_depression	sex	age	max_heart_rate_achieved	exercise_induced_angina
olalu7	2	reversible_defect	170	1	0	0	2	288	0.2	1	59	159	0
z9n6mx	1	normal	138	4	0	0	0	183	1.4	0	35	182	0
5k4413	2	reversible_defect	120	4	0	0	2	177	2.5	1	43	120	1
mrg7q5	1	normal	102	3	1	0	0	318	0	0	60	160	0
uki4do	2	normal	138	4	1	0	2	166	3.6	1	61	125	1

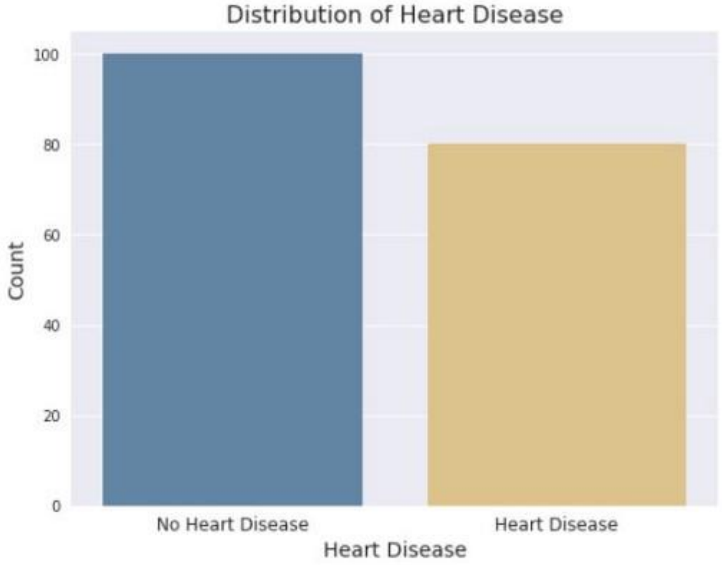
Exploratory Visualization

Analysis of the heart disease dataset is split into two parts –
1. Exploratory Analyses
2. Predictive Modeling

In the exploratory analyses part, it is first ensured that there are no missing values in the data. Then, summary statistics of all the variables are analyzed. Bivariate analyses between the independent and target variables are conducted and plotted. Specifically, for categorical independent variables, a barplot showing the split of 'target' is shown, while for continuous independent variables, a frequency histogram showing the difference in distributions for the two 'target' categories is shown

Visualizing our data with a little help from the seaborn package:

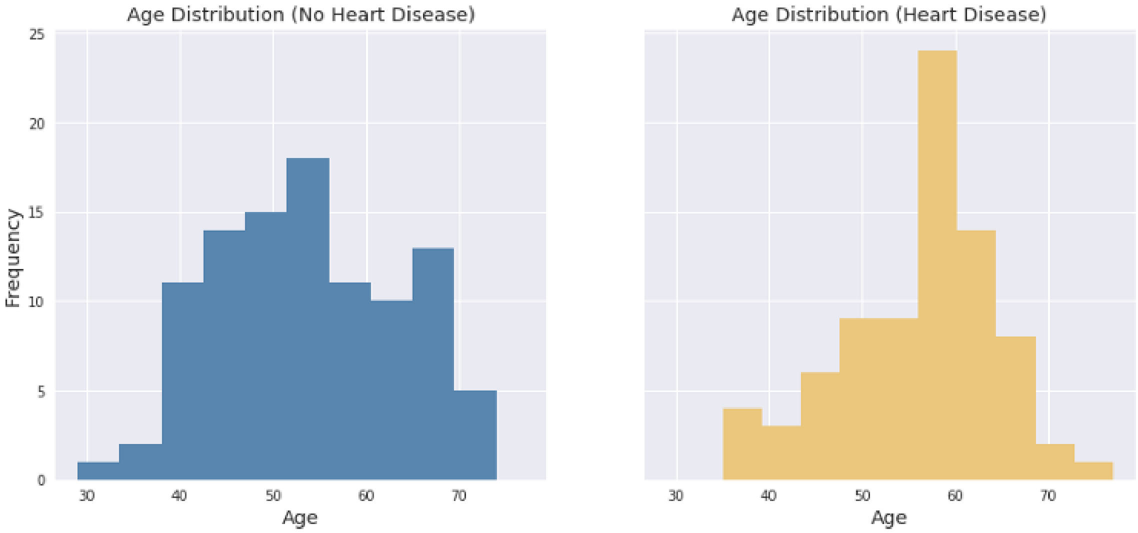
From the frequency plot of heart disease below, we see that the two classes ('Heart Disease' and 'No Heart Disease') are approximately balanced, with 45% of observations having heart disease and the remaining population not having heart disease.

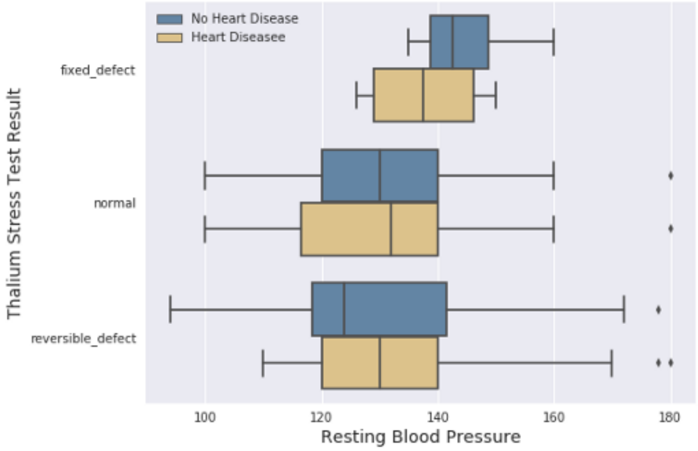


The data is relatively well-balanced, so we won't take any steps here to equalize the classes.

From The pairplot above in the metrics section allows us to see the distribution and relationship of numerical variables. The diagonal shows kernel density plots showing the rough distributions of the two populations. The scatter plots show relationship between plots. We can make a couple observations from the pairplot:

- Resting blood pressure tends to increase with age regardless of heart disease.
- We can see that max heart rates are significantly lower for people without heart disease.





Algorithms and Techniques

I used many algorithms and compared between them to find the best performance so I will explain in details the more efficient algorithmis in this problem

A Decision Tree is a robust and transparent Machine Learning model. The tree starts with a single node and then branches out, with a decision being made at every branch point. It can be used to predict whether a particular variable would have mattered in the customer's decision to subscribe or not to the bank's term deposit. The given dataset is a typical supervised learning problem for which tree type models perform a lot better than the rest. We don't know which algorithms would be fit for this problem or what configurations to use. So let's pick a few algorithms to Evaluate.

- **Logistic Regression (LR):** is the appropriate regression analysis to conduct when the dependent Variable is dichotomous (binary).
- **Classification and Regression Trees (CART)** : The CART algorithm is structured as a sequence of questions, the answers to which determine what the next question, if any should be. The result of these questions is a tree like structure where the ends are terminal nodes at which point there are no more questions.
    - **Classification Trees:** where the target variable is categorical and the tree is used to identify the "class" within which a target variable would likely fall into.
    - **Regression Trees:** where the target variable is continuous and tree is used to predict it's value.
- **Random Forests (RF):** The idea behind RF, is to build many small Trees that use a random subset of the features and then combine them into a "Forest" of Trees. Each Tree by itself is a weak predictor, but when combining the many weak predictors, you can often (but not always!) get a stronger model.
- **Adaptive Boosting (AB):** AdaBoost is a type of "Ensemble Learning" where multiple learners are employed to build a stronger learning algorithm. AdaBoost works by choosing a base algorithm (e.g.decision trees) and iteratively improving it by accounting for the incorrectly classified examples in the training set.
- **Extreme Gradient Boost (XGB):** The idea behind GBM is a more sophisticated. Roughly, the idea is to again, combine weak predictors. The trick is to find areas of misclassification and then "boost" the importance of those incorrectly predicted data points. And repeat. The result is a single Tree unlike RF.

XGBoost is a model based on tree ensemble which is a set of classification and regression trees (CART).

XGBoost classifies the members of a family into different leaves and assigns scores on corresponding leaf.

Usually, a single tree is a weak leaner which is not strong enough to use in practice. Boosted trees are typically shallow which are high in bias and low in variance. A tree ensemble model sums prediction of multiple tree.

Benchmark

The predefined benchmark from the Kaggle leaderboard or the top 10 or 20 percentile scores from there. So my goal was to achieve accuracy above 80 percent and to avoid overfiting data

III. Methodology

Data Preprocessing

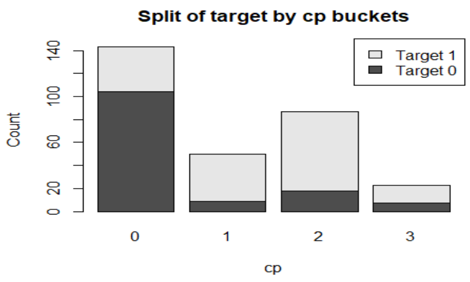
In the predictive analysis part, the dataset is first split into train and test sets such that a random 20% of the data is captured in the test set and the rest are used to train the model To work with categorical variables, we should break each categorical column into dummy columns with 1s and 0s.

Let’s say we have a column Gender, with values 1 for Male and 0 for Female. It needs to be converted into two columns with the value 1 where the column would be true and 0 where it will be false.

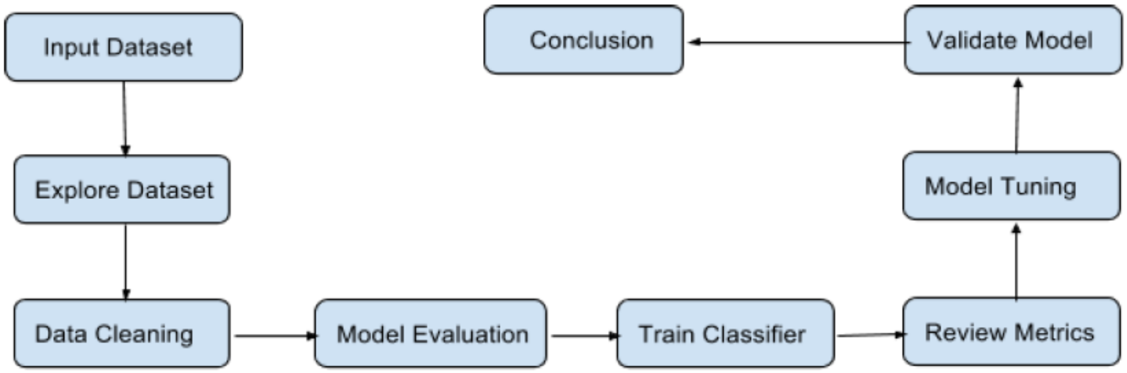
In the preprocessing stage, it was observed that there are no missing values in the data and it is useable as is. In the exploratory analysis stage, summary statistics of the variables were obtained to see how they were distributed. It was observed that there was no major class imbalance in the target variable as 165 instances corresponded to heart disease and 138 corresponded to no heart disease.

Bivariate analyses showed that certain variables may highly important to predict heart disease as they showed large variation between patients with and without heart disease. These variables are - cp, exang, slope, ca, thal, thalach

It can be seen from below plot that patients with chest pain type cp=0 are less likely to have heart disease than those with chest pain types cp=1,2 or 3



Implementation



Following the direction of the arrow as shown, with the dataset we chose, we first performed exploratory analysis by seeing distribution of some of the feature that seems to be relevant for the problem we plan to solve. We also did a check of correlations using scatterplot matrix in this process. We then performed data cleaning steps to identify columns with highest amount of null values and make a decision if we want to fill the missing values or completely drop the columns from the analysis. We processed the data through a pre-processing function to identify features that have values like 'yes', 'no' ,'unknown' and convert them to 1, 0, np.nan. For categorical features we converted them to dummy variables so that the dataset becomes fully numeric and it then is easy to process it through our chosen algorithm for meaningful outcome. Before we jump into the model evaluation, we split our full dataset into training and validation splits with a 80:20 ratio and then perform another check to make sure the splits have equal percent of responses to avoid deviations in the classification iterations. We used tree based algorithms to perform a check on accuracy and standard error metrics to pick a model that performs best. Typically Random Forest and XGBoost are well known to perform at best in usual supervised machine learning applications and we did confirm this by observing that XGBoost indeed has highest accuracy

Once we benchmarked the accuracy of untuned XGBoost model, we then performed tuning of XGBoost model’s hyperparameters in batches using GridSearchCV. This way we could keep track of best outcomes of each parameter .

Refinement

We will prepare the data by splitting feature and target/label columns and also check for quality of given data and perform data cleaning. To check if the model we created is any good, we will split the data into training and validation sets to check the accuracy of the best model. We will split the given training data in two ,80% of which will be used to train our models and 20% we will hold back as a validation set.

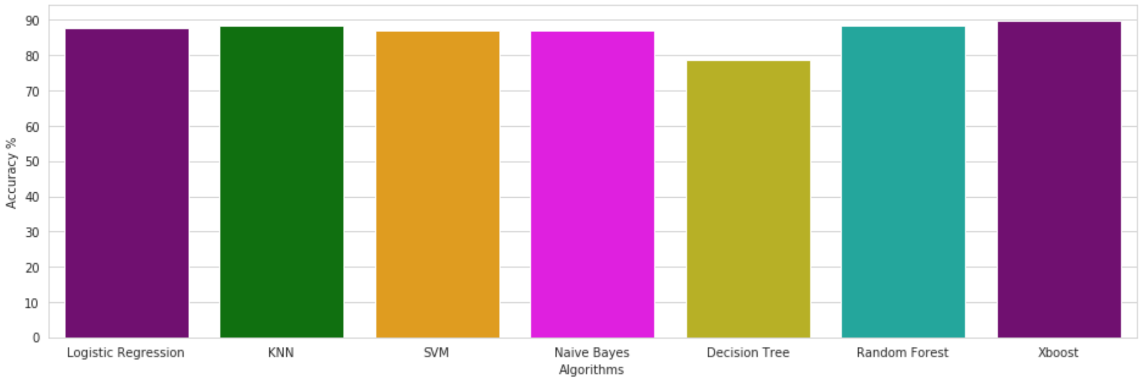
We performed hyper tuning of parameters of XGBoost wrapper from scikit-learn library and the parameters tuned were

- max_depth
- min_child_weight
- gamma
- reg_alpha L1 regularization
- reg_lambda L2 regularization

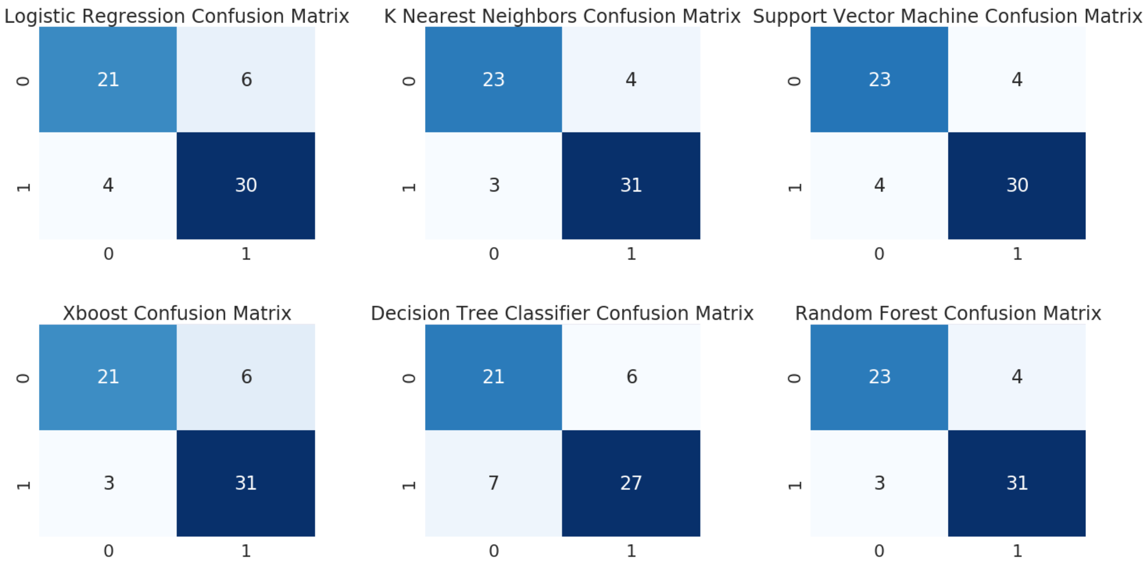
IV. Results

Model Evaluation and Validation

We applied and compared XGB model out of the box vs the rest of models. The metrics we used are calculated using sklearn wrapper so can be trusted for the model performance. Our end goal was to have a tuned model that could beat the untuned benchmark which it did by a very fine margin. So the solution described below is satisfactory to our initial expectations.



Confusion Matrixes



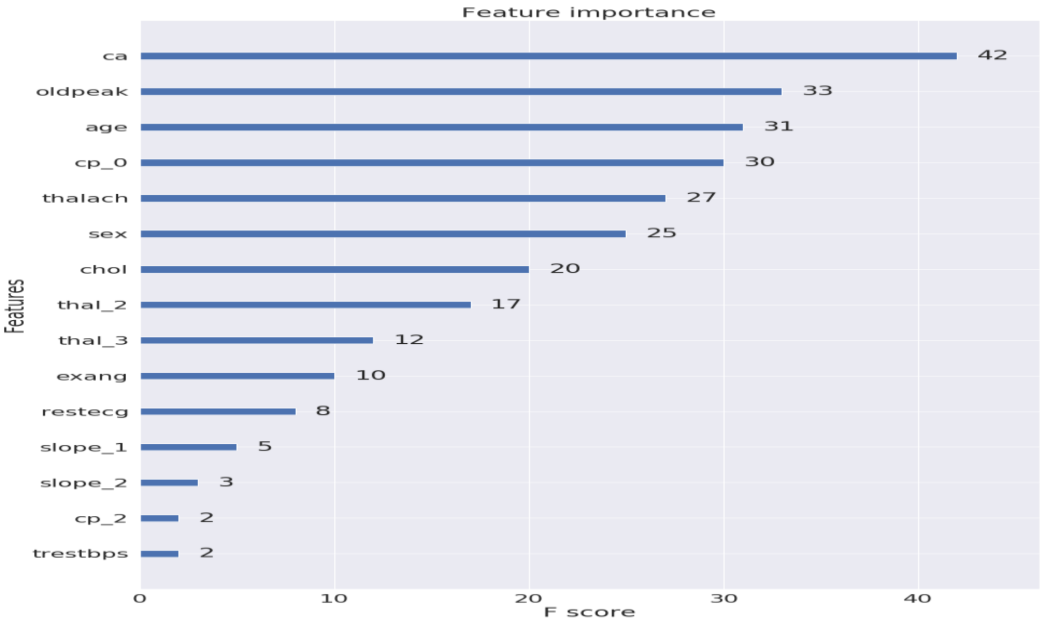
Justification

There could be more ways we could improve the score, particularly
by selecting a subset of features using the ranking obtained from observing feature importance ranking and then performing the same exercise we did as describe above. But this will be out of scope of this report for now.

V. Conclusion

Free-Form Visualization

The importance of each feature was visualized with the following plot:



Reflection

The most important and time consuming part of the problem was data cleansing and processing

Once the data was prepared and ready, the next challenge
was to pick an algorithm that could be best suited for the problem we choose to solve. With experience and prior knowledge and also the outcome of accuracy on training data, we observed that XGBoost performed the best out of others.

Improvement

One of the improvements we could do was to perform tuning to improve recall rate to improve overall prediction performance of the model. The other improvement could be to work with a subset of features that are high in variable importance. Another key aspect would be to review metrics like log-loss to understand how quick the model is able to tune.