Foundations of Machine Learning
DS 3001 - Group 11
Wadie Abboud
Julian Krese
Jenny Schilling
Elizabeth Wu

Predictive Algorithms for Coronary Heart Disease

Research Question: What model most accurately predicts the risk of individuals developing coronary heart disease?

Summary:

In this study, our objective was to identify the most effective classification model at forecasting the likelihood that an individual develops coronary heart disease (CHD). Predictive models in healthcare can help identify individuals at risk that might still benefit from early intervention strategies. Additionally, determining which factors contribute most to CHD can inform public health authorities and providers on the effectiveness of current education campaigns, screening methods, and treatment plans. To achieve such a goal, we used a subset of data from the Framingham Heart Study which contains health and demographic information across three generations of participants. The dataset was divided into predetermined train and test splits for our analysis. Our approach involved thorough data cleaning and the development and testing of a series of models. We cleaned both sets of the data, mostly by handling any missing values and data type conversions. Our team chose to focus on comparing the capabilities of linear models, k-nearest neighbors, and decision trees, alongside variations of such models. After building each primitive model based on the training data, we analyzed the R-squared value as the main indicator of predictive performance, though we also took the results of the generated confusion matrices into consideration. Furthermore, we enhanced each model with additional predictive techniques, such as polynomial expansion and feature selection. Also, we iterated through a range of hyperparameters and determined the optimal parameter corresponding to the highest R-squared value for that model, which we supported with visualizations. We found that the decision tree models performed the best across the board, followed by k-nearest neighbors, with linear models performing the worst by far. Moreover, applying feature selection to the decision tree model and retesting for the optimal hyperparameter of max depth yielded the highest R-squared value of 0.9886 trained at a max depth of 25. While this R-squared value suggests that the model effectively explains the variance in the dataset, the alarming number of false positives and negatives in the confusion matrix indicates that its classification abilities may not meet the rigorous standards expected for medical diagnoses. Despite such discrepancies in the model, these findings serve as a good starting point for data scientists to refine their predictive algorithms.

Data:

Before running the model, we had to clean up both the training and testing data sets. Both cleaning processes were very similar and their work can be found within the 'project2_data_cleaning.ipynb' file. We began by looking at the data itself, and noticed immediately an unnecessary column labeled "Unnamed: 0". This column was quickly dropped and the next glaring problem were NaN's. These NaN's were within 'education', 'cigsPerDay', 'BPMeds', 'totChol, 'BMI', and 'glucose' in the training data, and within 'education', 'cigsPerDay', 'BPMeds', 'totChol', 'BMI', 'heartRate', and 'glucose' in the testing data. After checking the background and data dictionary for possible reasons for these NaN's, none stood out and so the NaN's were simply dropped instead of imputed upon. After dropping the NaN's, the training data went from 3180 observations to 2744 observations and the testing data went from 1060 observations to 913 observations. Although losing close to 600 overall observations was much less than ideal, we found this a necessary and unavoidable choice to clean the data.

After dropping NaN's, the next immediate problem was strange typing within the columns. Certain columns were expressed as floats when they had no decimal value following the whole number and could simply be expressed as ints, which makes the data more understandable. These columns were 'cigsPerDay', 'BPMeds', 'totChol', and 'glucose'. For some reason 'heartRate' within the testing data also had to be transformed into an int as well, but not within the training data, which is a strange and unexplained incongruity. This float-to-int conversion shouldn't have radically altered the data and instead was simply for readability when viewing the raw testing and training data sets.

Lastly, there were lots of categorical variables within columns expressed as a numeric '1' or '0', but the 'education' column also had multiple values as a categorical variable. The values ranged from '1.' to '4.' to represent what level of education a person received. To improve any predictive models performance, we chose to one-hot encode this variable. This resulted in four new columns of scaling education, labeled 'edu_some_hs', 'edu_hs/ged', 'edu_some_college/voc', and 'edu_college' to correspond with the increasing float labels of '1.' through '4.'. Doing this also resolved the issue of floats being used to store categorical information.

After doing the steps above, the data was initially clean. The final check was for outliers that may need to be dropped or values with very large and skewed scaling that could be normalized in some way such as a logarithmic scaler. After mapping the kernel density plots for each (the plots can be found under the file 'project2_data_cleaning.ipynb') as well as viewing the columns' values within, no outliers were spotted and no data was deemed skewed enough to deserve a scaling function of some sort. Now that the data is officially cleaned for both the train and test set, the data was stored in 4 separate csv files, 'X_train', 'X_test', 'y_train', and 'y_test'. This was done for convenience and modularity of the model training, because any additional cleaning within 'project2_data_cleaning.ipynb' would not directly affect how we trained our models (code-wise) and would only need to be re-run.

Results:

With these new clean training and testing sets, we were able to begin experimenting with different methodologies to build the model. The three main categories that we wanted to explore were linear regression, *k* nearest neighbor (kNN), and decision trees. To start, we built several different predictive algorithms using linear models using the Scikit-learn Linear Regression python module. Using all the variables, we fit a linear regression model to the train data which resulted in an unsatisfactory R-squared value of 0.0915. Hoping to improve upon such a model, we implemented min-max normalization on the input data with the Scikit-learn MinMaxScaler, which actually dropped the R-squared value to 0.0899, revealing that this particular scaling method degrades the model's performance. The next technique we attempted was to select the most correlated feature for linear regression, which was found to be 'prevalentHyp', though this feature only achieves an R-squared value of 0.0637 (see Figure 1). Our last approach involved using polynomial expansion with linear regression though this resulted in an optimal degree of 1, which brings us back to the subpar performance of the initial model. As we can see in Figure 2, any degree over 2 resulted in R-squared values over 1, which are not meaningful and suggest overfitting of the training data.

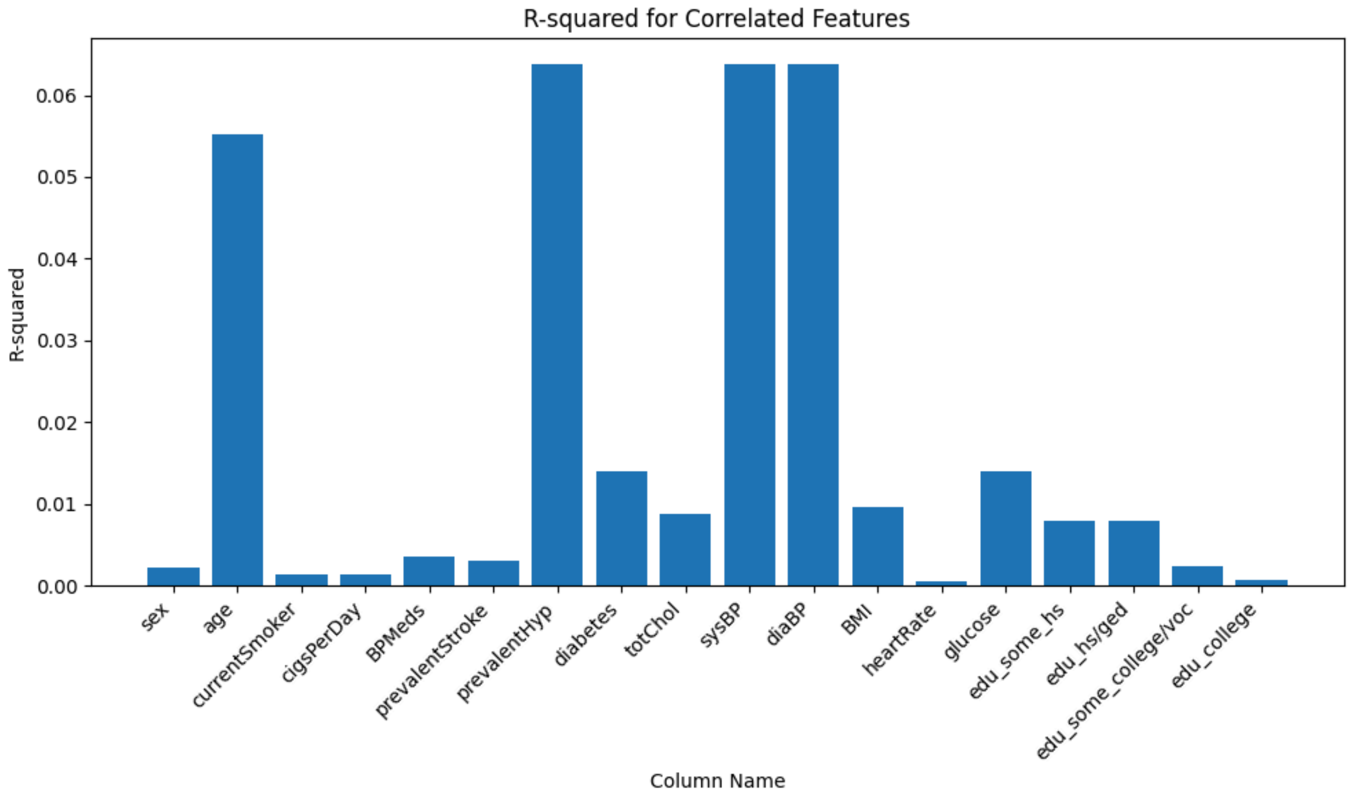

Figure 1: Linear Regression Feature Importance.

The mediocre performance of all the linear models, despite attempts for improvement, indicate that linear regression was not the best approach for our classification problem. Linear regression models typically struggle with non-continuous outputs, such as class labels, and assume that the relationships between the features and the target variable are linear, that the observations are independent, and that the residuals are normally distributed, none of which are likely true for our dataset.

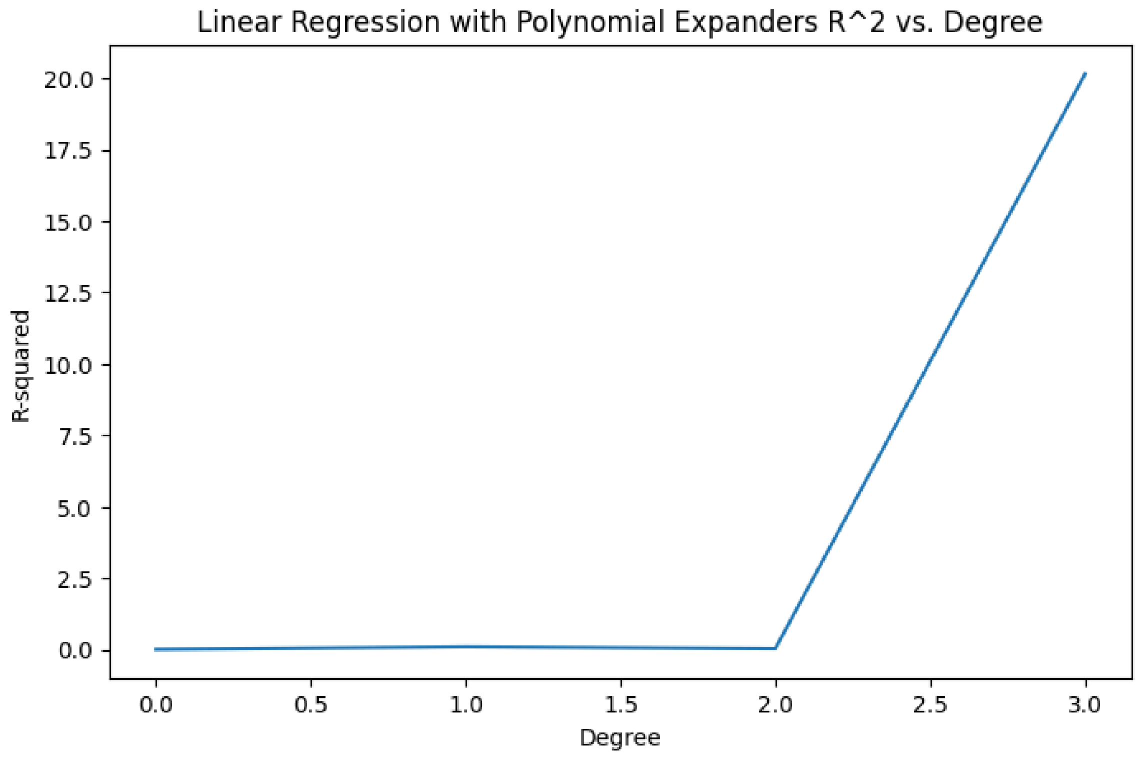

Figure 2: Visualization of Linear Regression Performance vs. Given Degree Parameter.

Next, we attempted to improve these R-squared values by using a *k* nearest neighbor classifier. The "*k*" indicates how many of the closest data points the algorithm will look at to decide how to classify a new point, so we had to first determine the best parameter configuration by testing a range of different *k* values from 1-9. We were able to accomplish this using the Scikit-learn KNeighborsClassifier module and a simple for-loop that iterated through each parameter. Then, the model with the highest R-squared value on the testing set was selected as the best performing model, and that *k* was stored as the best number of data points for the algorithm to look at. An application of this showed that a kNN model with *k* = 1 achieved the highest R-squared of 0.6806. This suggested that looking at just one nearest neighbor gave the model an optimal balance between bias and variance for predicting whether a person has CHD. This relationship between *k* and the model's R-squared performance is shown in Figure 3.

Then, to further assess the kNN model, a confusion matrix was generated as seen in Table I. This table demonstrates that there are a large number of true negatives (689), which we would expect to see, but that there are also a significant number of false positives (92), and even more concerning, false negatives (99). Providing a patient with a false negative diagnosis of CHD could have serious ethical and legal consequences if that patient were to then go untreated, so seeing this table gave us a true sense of the gravity of our research in finding a model that is as accurate as possible.

	Predicted: Negative	Predicted: Positive
Actual: Negative	689	92
Actual: Positive	99	34

Table I: Confusion Matrix for kNN Model.

The last category of algorithms to experiment with was decision tree classification. We created two models using decision trees: the first was just testing different tree depths and training using Scikit-learn's DecisionTreeClassifier module, and the second used feature selection along with the decision tree classifier. Similarly to the previous algorithm explorations, an iterative process was used to fine-tune the parameters and determine the optimal tree depth. The R-squared value was calculated for each max tree depth ranging from 1-49. Results from this process indicated that 25 was the ideal tree depth, yielding an R-squared value of 0.9534. Figure 4 depicts this relationship between the decision tree max depth and its performance as measured by R-squared.

The best decision tree model (with a depth of 24) was subsequently analyzed to obtain the most influential features driving the model formation. Feature importances were computed using the included 'feature_importances_' method, and it was determined that there were six features that were significantly more vital than the others (listed in descending order of importance): 'BMI' 'totChol', 'glucose', 'sysBP', 'age', and 'diaBP'. This essentially informs us that a patient's body mass index, total cholesterol, gluboce, and blood pressure levels contribute to their likelihood for CHD far more than their education, for example. A visual representation of this feature importance is shown in Figure 5 in the form of a horizontal bar graph. A confusion matrix was also generated to assess the predictive performance, which, similarly to the kNN model, showed a majority of accurately predicted true negatives, but still an alarming amount of false positives as well as false negatives. This data can be seen in Table II. So, while the decision tree model demonstrated a higher R-squared than the previously tested KNN model, implying better ability to capture data variation, its confusion matrix performance is not a significant improvement over the KNN model.

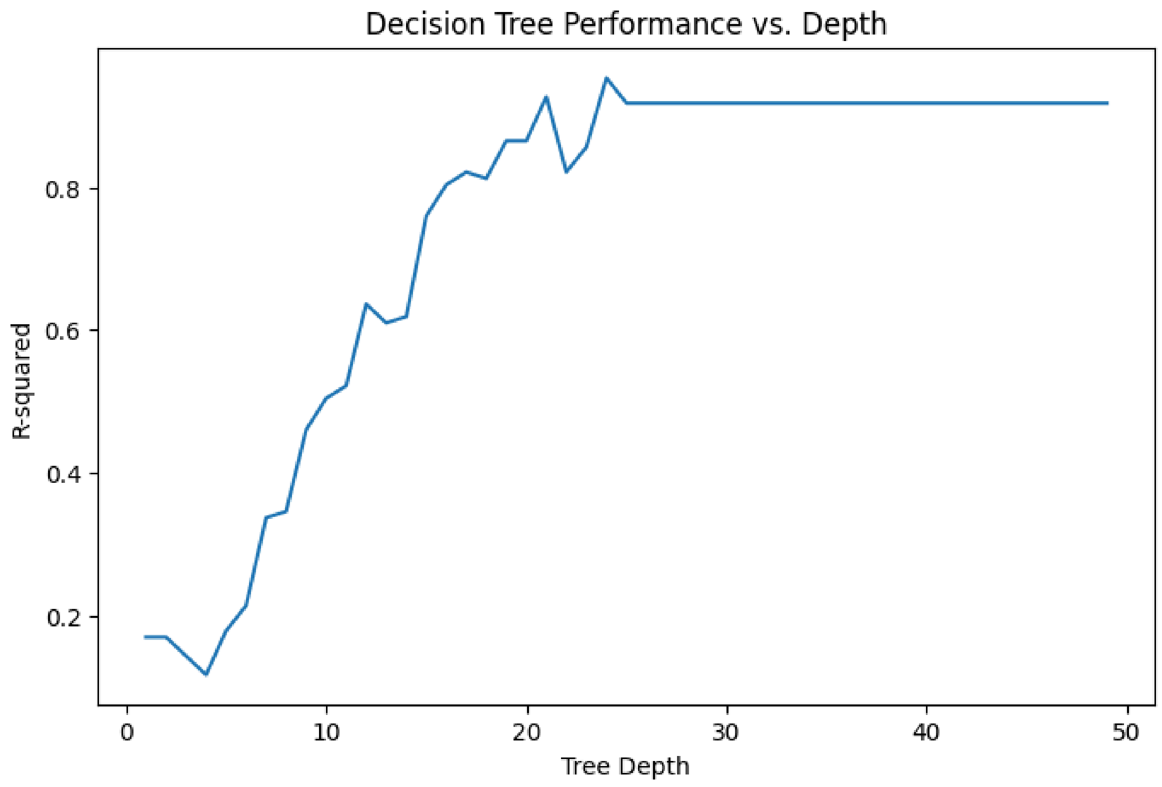

Figure 4: Decision Tree Performance vs. Depth

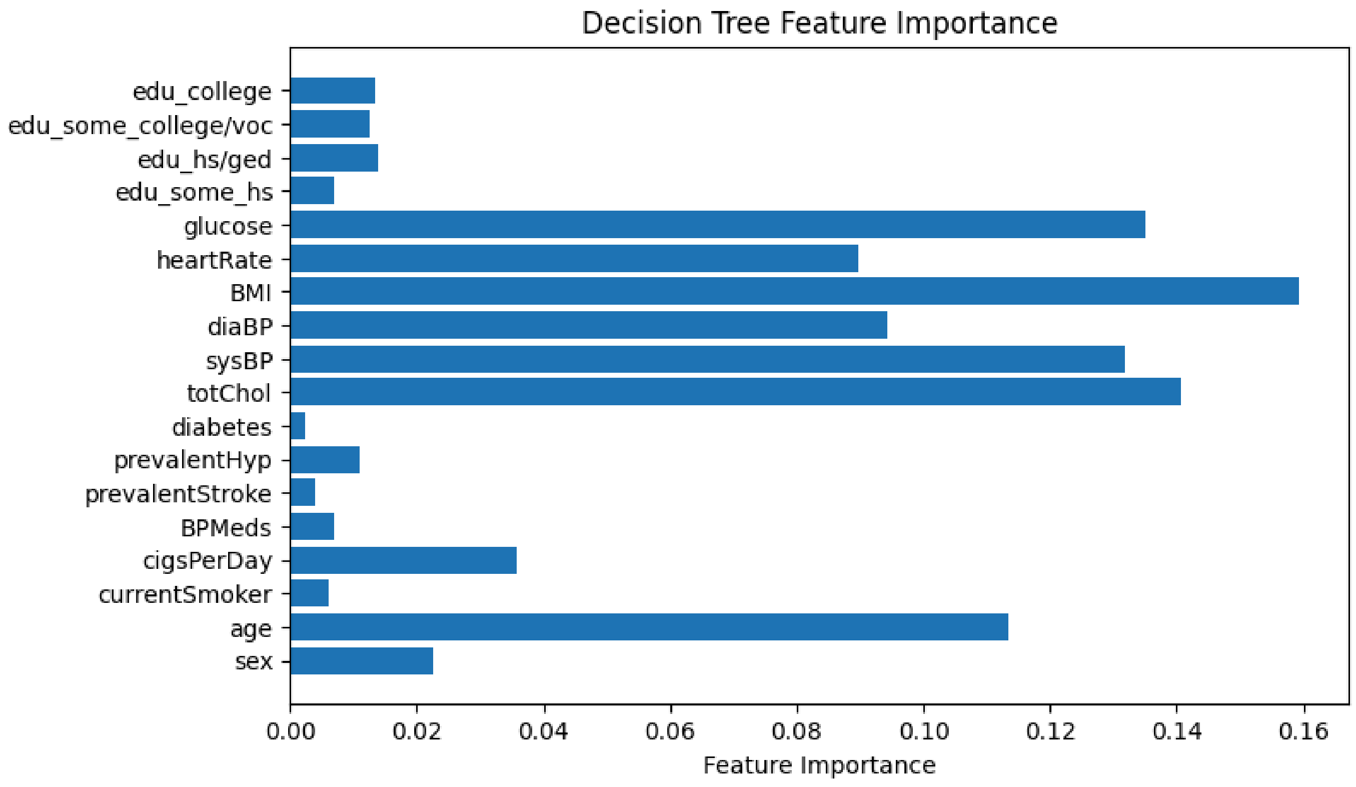

Figure 5: Decision Tree Feature Importance.

	Predicted: Negative	Predicted: Positive
Actual: Negative	660	121
Actual: Positive	101	32

Table II: Confusion Matrix for Decision Tree Model.

Finally, to even further refine the decision tree model, and hopefully improve the confusion matrix performance, feature selection was utilized. The use of feature selection would hopefully provide more accurate predictions since the algorithm would not be trained using data that simply throws the model off with outliers and features that don't actually contribute to CHD. Based on the previous decision tree model, the following features were selected as they appeared to have the strongest association with CHD: age, total cholesterol (totChol), systolic blood pressure (sysBP), and glucose. The inclusion of diastolic blood pressure (diaBP) as well as BMI was explored but actually found to decrease the model's R-squared value, so we opted to simply not use that column in the feature selection.

The decision tree model was retrained using the reduced feature set, and the parameter fine-tuning process was also repeated to identify the ideal max tree depth with this new input. A new best depth of 25 was found to yield the highest R-squared value of 0.9886. Figure 6 illustrates this relationship between decision tree depth and R-squared performance with the feature selection. This R-squared was by far the highest we had obtained with any method of model creation thus far, and since it was extremely close to 1, we knew that this model was very accurate in predicting CHD and did an excellent job at not overfitting the data.

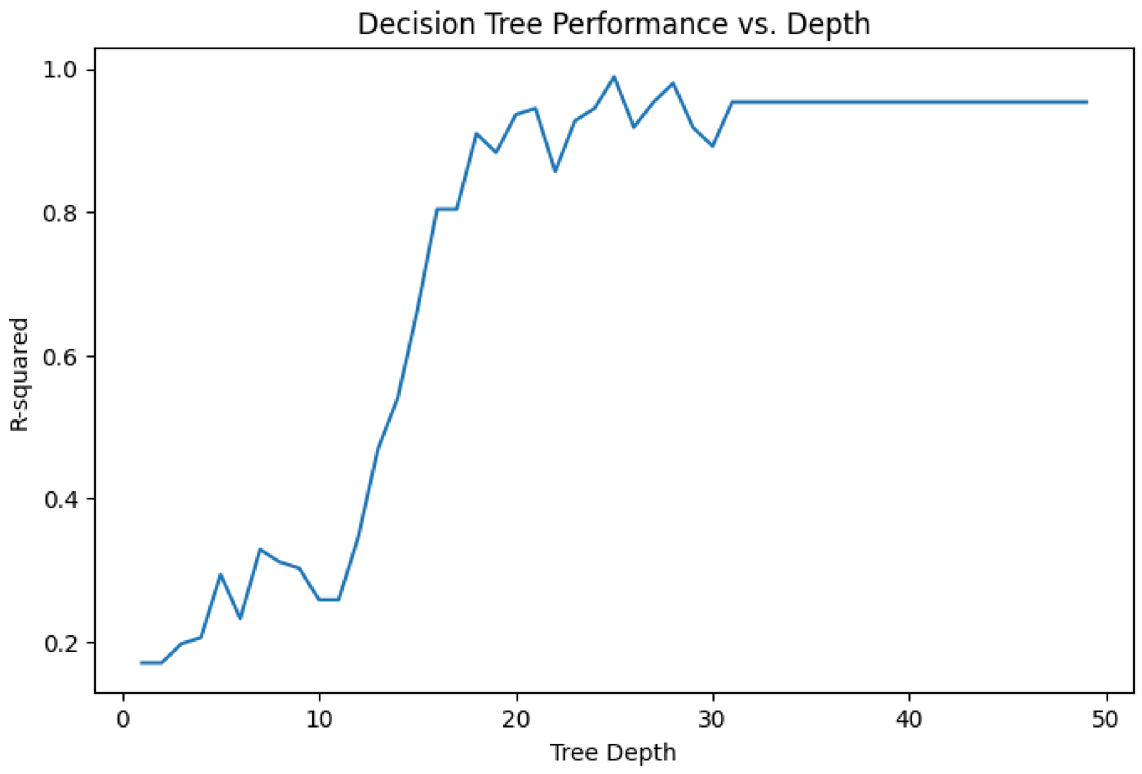

Figure 6: Decision Tree Performance vs. Depth with Feature Selection.

Conclusion:

Overall, this study concluded that decision tree classification and its variations were the most accurate predictive tools for highlighting the risk of individuals developing CHD. In order to test for the most accurate method, we played around with several different predictive models, focusing on linear models, k-nearest neighbors, and as mentioned, decision trees. By utilizing R-squared as our comparative variable, we were able to come up with what we believe are fairly conclusive results. While both decision tree methods outperformed the other models, we found exceptional results with the use of a decision tree with feature selection. With an R-squared value of 0.9886, the decision tree of 25 depth heavily surpassed our other models.

When conducting this study, we ensured that the work was done in thorough, substantial steps, with a clear process in mind. We certified that the data was reputable, sourcing it from the Framingham Heart Study. Additionally, we took great care to clean the data in a manner which both preserved the core trends and correlations, while also extinguishing the potentially distracting or "messy" data, such as missing values. More on the specifics of this process can be found in the *Data* section. While working through each respective model, we ensured we took necessary steps to test many potentially result-altering factors. For example, we iterated through all correlation variables when testing linear regression in order to save the most effective results. Additionally, we tested various degrees of polynomials. When applicable, similar steps were taken with other models, more on this in our *Results* section.

These conclusions should however, be taken with a grain of salt. There were various faults with the data in terms of smaller sample sizes, various outliers, and missing values. While this data was cleaned, it is important to understand that the results from this study can in no way be an indicator of its potential use in clinical settings. However, our findings still contribute to the understanding that of the models studied, decision trees with feature selection can be extremely useful tools when analyzing and predicting similar data.

While the testing we did was fairly concrete and conclusive, the opportunities for future research with this type of data set are truly endless. The models studied were not very complex, relative to the multitude of predictive models available. Examples could include random forest models, which would be less "sensitive" to specific sets of data. Additionally, the data these models were tested on was relatively straight forward in terms of predictive variables and contributing factors to CHD. We believe these models, if applied effectively, can be used on more abstract and complex data sets. Regardless, predictive models are and will continue to be extremely useful tools in medical settings. We are fortunate to live in a time where data of all sorts is readily available to us. With the help of these models, medical professionals can continue to make advancements, in preventative diagnosis, treatment methods, and more.