Predicting an Individual's Risk of Heart Disease Using Various Machine Learning Models

Authors: Phuong Vu, Yunwei Liang, Trinh Nguyen

March 16, 2020

CSE 163 Intermediate Data Programming

Table of Contents

Summary of Research Questions

1. Which machine learning algorithm can best predict the risk of heart disease?

   We implement three supervised machine learning models--Decision Tree Classifier, the Random Forest Classifier, and the Gaussian Naive Bayes Classifier--to predict an individual's risk of having heart disease. Machine learning algorithms use patients' characteristics as features to predict whether an individual is healthy or at risk. We evaluate each model based on its accuracy score and select the model with the highest average accuracy score for Part 2 of our research.

   Our findings indicate that, for our research goal, **the Gaussian Naive Bayes Classifier is most accurate** in predicting an individual's risk for heart disease based on their health characteristics. The Random Forest Classifier is the second most accurate followed by the Decision Tree Classifier.

2. What is the correlation between each feature and the prediction of one's risk for heart disease? Which feature is determinant of an individual's health risk?

   We implement the most accurate machine learning model, the Gaussian Naive Bayes Classifier, from Part 1 to find the correlation between each feature and the risk prediction. We utilize the statistical functions module to calculate the Pearson r coefficient of each feature with the prediction. The Pearson r coefficient allows us to measure the correlation between the specific feature and the prediction of risk on a scale of -1 to 1. A coefficient of -1 indicates an inverse relationship, 0 implies no correlation, and 1 indicates a proportional relationship. We also integrate an ablation study in our research to analyze the effect of a specific feature on the model's prediction. This allows us to understand the behavior of the machine learning model and allows us to determine the model's most important feature.

Table 1: Correlation of Features to Prediction

Features	Pearson's Correlation Coefficient
age	0.227
sex	0.278
cp	0.409
trestbps	0.153
chol	0.080
fbs	0.003
restecg	0.166
thalach	-0.424
exang	0.421
oldpeak	0.424
slope	0.333
ca	0.463
thal	0.527

Looking at our findings in the correlations and further feature analysis, **the determinant feature for the Gaussian Naive Bayes model is cp (chest pain type).**

Motivation and Background

The development of a machine learning model that could predict one's risk of heart disease is useful for common medical diagnosis and could even be used for preventative actions. According to the American Heart Association, heart disease is the leading cause of mortality worldwide. Even when the condition is not fatal, heart disease can diminish one's quality of life, thus being able to predict one's risk of heart disease early is important. The exploration of this learning machine model can help reduce the number of diagnosis tests and alleviate the financial burden of unnecessary testing procedures. The analysis of the learning machine's behavior also allows us to analyze the correlation between certain characteristics and an individual's risk for heart disease. This is useful for generating preventative care plans for individuals approaching an age where they are more vulnerable to heart disease.

Through the research article "Improving the accuracy of prediction of heart disease risk based on ensemble classification techniques" by C. Beulah Christalin Latha and S. Carolin Jeeva, we were inspired to find the correlation between patients' characteristics and their risk of heart disease. The authors of the research article tested the performance of various machine learning algorithms in predicting the risk of heart disease of a participant and strived to improve their models through ensemble techniques. We are utilizing the dataset from Latha and Jeeva's study to perform our own analysis. With our research, we are comparing three specific machine learning models--Decision Tree Classifier, the Random Forest Classifier, and the Gaussian Naive Bayes Classifier. Instead of improving all the models, we are analyzing the most accurate model and determining the correlation between each characteristic and an individual's risk for heart disease.

Dataset

Our main dataset is the “cleve.processed.data” file in the Data Folder in the attached link to original dataset below. This dataset was collected from Cleveland Clinic Foundation and accessed in the UCI Machine Learning Repository website. There are 14 attributes describing each of 303 observations in the dataset. It shows the relationship among an individual’s characteristics such as age, sex, chest pain type, resting blood pressure, etc. and their risks of having heart disease. We converted the .data file to a .csv file in order to perform our analysis. For readability, we modified the last column’s name from “class att” to “prediction”. In the original dataset, “class att” indicates a healthy individual with “H” and at-risk individuals with “S1”, “S2”, “S3”, and “S4”. For the purpose of our research and the binary classes nature of the Gaussian Naive Bayes, we one-hot encoded the value of “H” to be 0 and values of “S1”, “S2”, “S3”, and “S4” to be 1. Note that the data filtration is done within our code. For our machine learning models, we will use the column “prediction” as the label and the remaining columns as features. Below is an informative table of our features and labels.

Link to original dataset: Heart Disease Data Set

Clean dataset: clean_data.csv

Note that Table 2 below is of the clean Cleveland dataset, which does not contain NaN values.

Table 2: Features and Label Description

Column Index	Column Name	Column Description	Values Range
1	age	Person’s age in years	29 to 77
2	sex	Person’s gender (1: Male, 0: Female)	0, 1
3	cp	Chest pain type (1: Typical Angina 2: Atypical Angina 3: Non-angina pain 4: Asymptomatic)	1, 2, 3, 4
4	trestbps	Resting blood pressure in mmHg	94 to 200
5	chol	Serum cholesterol in mg/dl	126 to 564
6	fbs	Fasting blood sugar > 120 mg/dl (1: True, 0: False)	0, 1
7	restecg	Resting electrocardiographic results (0: Normal 1: Having ST-T wave abnormality 2: Showing probable or definite left ventricular hypertrophy by Estes’ criteria)	0, 1, 2
8	thalach	Maximum heart rate achieved	71 to 202
9	exang	Exercise induced angina (1: Yes, 0: No)	0, 1
10	oldpeak	ST depression induced by exercise relative to rest	0 to 6.2
11	slope	The slope of the peak exercise ST segment (1: Upsloping 2: Flat 3: Downsloping)	1, 2, 3
12	ca	Number of major vessels colored by fluoroscopy	0 to 3
13	thal	A blood disorder (thalassemia) (3: Normal 6: Fixed Defect 7: Reversible Defect)	3, 6, 7
14	prediction	Predicted risk of heart disease (1: Risk of heart disease 0: Healthy or no risk of heart disease) 1: At risk of or has heart disease	0, 1

Methodology

I. Which machine learning algorithm can best predict the risk of heart disease?

Goal: Determine the machine learning model with the highest average accuracy score.

1. Data Preparation
    a. Open "processed.cleveland.data" in VSCode and save as a .txt file.
    b. Open Excel and import the .txt file into Excel as a comma-separated-value import file.
    c. Insert attribute names as column names.
    d. Save Excel file as .csv file (cleveland_processed.csv).
    e. Import file as a Data Frame.
    f. Filter missing data from the dataset by removing NaNs.
    g. One-hot encode the "prediction" column to convert the value of "H" to be 0 and values of "S1", "S2", "S3", and "S4" to be 1. Refers to the **Dataset** section to see why this is necessary.
    h. One-hot encode the features columns.
    i. Select all columns as features except for "prediction" as this will be the models' label.
2. Set up and train a Decision Tree Classifier Model
    a. Split the data into training and testing sets with a 8:2 ratio relatively.
    b. Use DecisionTreeClassifier to build the model.
    c. Fit the model with the training set.
    d. Test the model with the test set and record the accuracy score
    e. Print out the decision tree.
3. Set up and train a Random Forest Classifier Model
    a. Convert the label and the features from data frames into numpy arrays.
    b. Split the data into training and testing sets with a 8:2 ratio relatively.
    c. Use RandomForestClassifier to build the model.
    d. Fit the model with the training set.
    e. Test the model with the test set and record the accuracy score.
    f. Print out a decision tree in the forest.
4. Set up and train a Gaussian Naive Bayes Model
    a. Split the data into training and testing sets with a 8:2 ratio relatively.
    b. Use GaussianNB to build the model.
    c. Fit the model with the training set.
    d. Test the model with the test set and record the accuracy score.

5. Find the average of each of the model's accuracy scores over 10 runs. Record the model with the highest average accuracy score.

II. What is the correlation between each feature and the prediction of one's risk for heart disease? Which feature is determinant of an individual's health risk?

Goal: Determine the Pearson r coefficient for each feature attribute and find the feature with the most weight on the model's predicting behavior.

1. Use the clean data and the most accurate machine learning model determined from Part 1 for Part 2.
2. Find the correlation between each feature and an individual's risk of heart disease.
    a. Set the label as a fixed "y" variable. Set a feature attribute as an "x" variable. This will allow us to compare the linear relationship between the label and each of the features. Convert the "x" and "y" to one-dimensional numpy arrays.
    b. Use the statistical functions module to implement Pearson r Correlation Coefficient.
    c. Calculate and record the Pearson's correlation coefficient.
    d. Repeat Steps 2a to 2c for each feature attribute.
3. Perform an ablation study on the data.
    a. Remove a feature from the feature attributes, the remaining is the "x" features of the data. Set "prediction" as the "y" label.
    b. Split the data into training and test sets.
    c. Train the machine learning model by fitting it with the training set of the "x" features and the "y" label.
    d. Test the model with the test set and record the accuracy score.
    e. Repeat Steps 3b to 3d but with the full feature attributes.
    f. Plot a bar chart of the accuracy scores for both results. This will help visualize the effect that a certain feature has on the model's behavior.
    g. Repeat Steps 3a to 3f for each of the features.
    h. Plot accuracy score for a set of trials with or without the feature as a bar chart faceted by model type, with x-axis as features, y-axis as accuracy score, color as "with" or "without."

Results

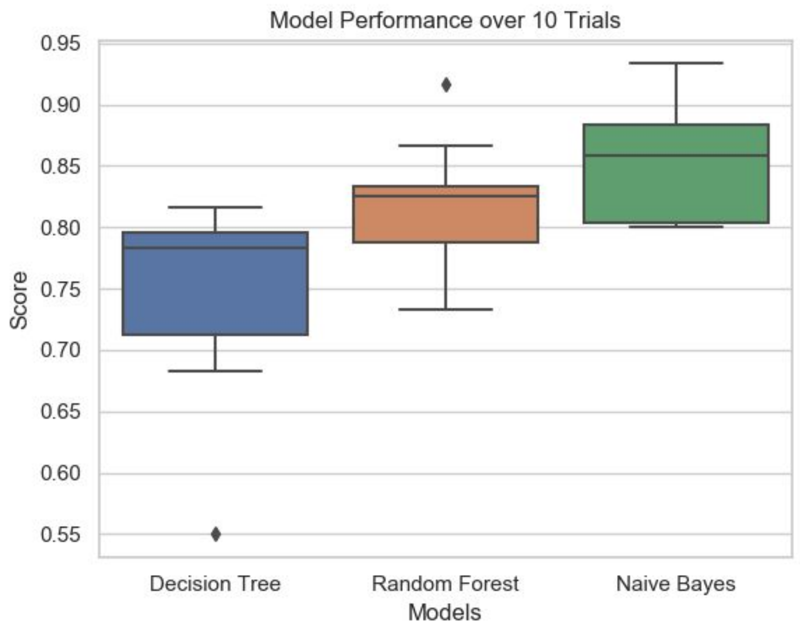
Part I

1. Which machine learning algorithm can best predict the risk of heart disease?

   Our findings indicate that, for our research goal, **the Gaussian Naive Bayes Classifier is the most accurate in predicting an individual's risk for heart disease based on all of the given health characteristics**. The Random Forest Classifier is the second most accurate followed by the Decision Tree Classifier.

   After 10 trials, the mean accuracy score for the Decision Tree Classifier is 0.747, the mean score for the Random Forest Classifier is 0.818, and the mean score for the Gaussian Naive Bayes Classifier is 0.855.

   To further visualize the distribution of data, below is a box and whisker plot representing the quartiles of accuracy scores of each learning machine model gathered from 10 trials. The averages mentioned before can also be visualized by the center of the boxes in the box plot. Note that these average values change each time the code is run as the models randomly split the dataset each time.



Plot shown in box_plot.png output
Produced by models_performances_box_plot in analyze_ml_models.py

The Gaussian Naive Bayes Classifier outperformed the Random Forest Classifier by 0.037 and outperformed the Decision Tree by 0.108.

The result confirms with our background research before the experiment. We initially selected the Naive Bayes algorithms because we found out that they are commonly used for health-care-related ML problems. We also hypothesized that the Random Forest would perform better than the Decision Tree since a Random Forest is more detailed. A forest includes multiple trees, hence a better accuracy score. Nonetheless, we were worried that a forest would overfit the data. However, it seemed that this was not the case. We are curious about why. This question can be a subject of interest in the future.

Part II

1. What is the correlation between each feature and the prediction of one's risk for heart disease?

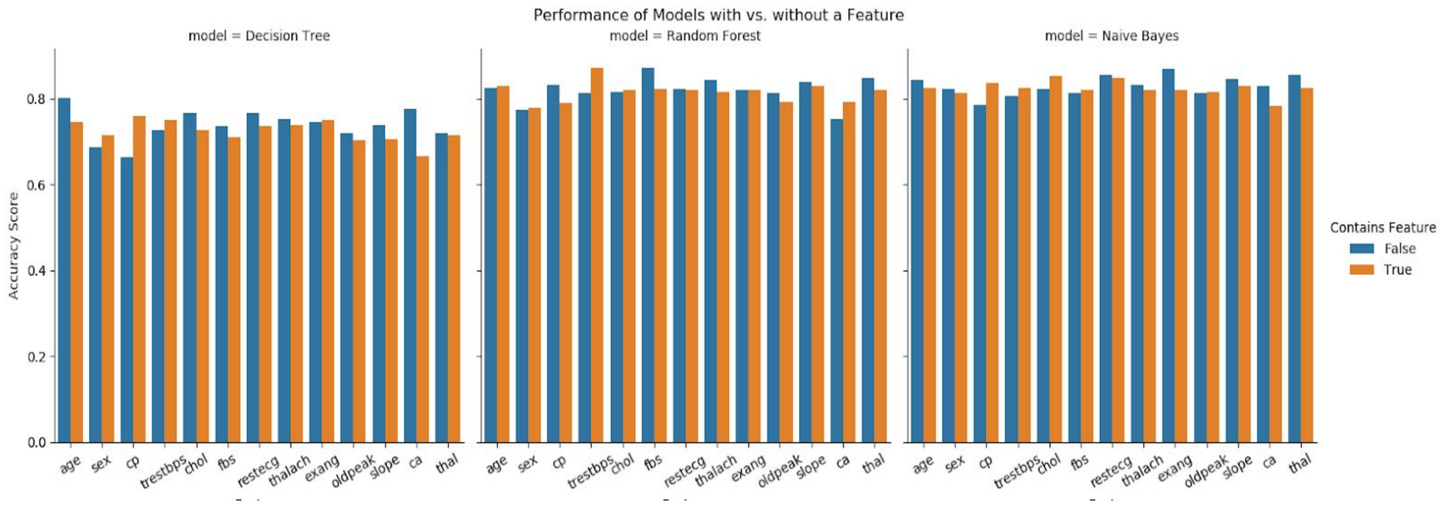
Table 1: Correlation of Features to Prediction

Features	Pearson's Correlation Coefficient
age	0.227
sex	0.278
cp	0.409
trestbps	0.153
chol	0.080
fbs	0.003
restecg	0.166
thalach	-0.424
exang	0.421
oldpeak	0.424
slope	0.333
ca	0.463
thal	0.527

Data shown in terminal output
Produced by correlation in main.py

Recall that with Pearson's Correlation Coefficient, it measures the correlation between the specific feature and the prediction of risk on a scale of -1 to 1. A coefficient of -1 indicates an inverse relationship, 0 implies no correlation, and 1 indicates a proportional relationship. Based on the data we collected, there is the most correlation between the prediction and the feature “thal”. The only inverse relationship is that of “thalach” and the prediction. To confirm this correlation is valid, we continued to the second half of our research questions for Part 2.

2. Which feature is determinant of an individual’s health risk? (Following numbers and plots produced with plot_feature_importance in main.py)



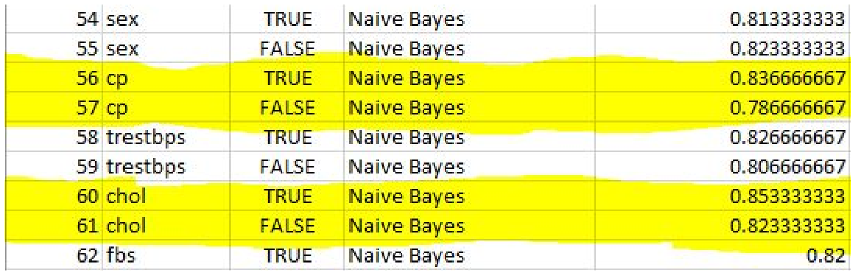
Plot shown in features_performances_in_models.png output
Produced by plot_feature_importances in main.py

There are no distinct features across all three models that significantly impact the accuracy. For the selected model from Part 1, the Naive Bayes model, the features that are more accurate after adding the feature are:

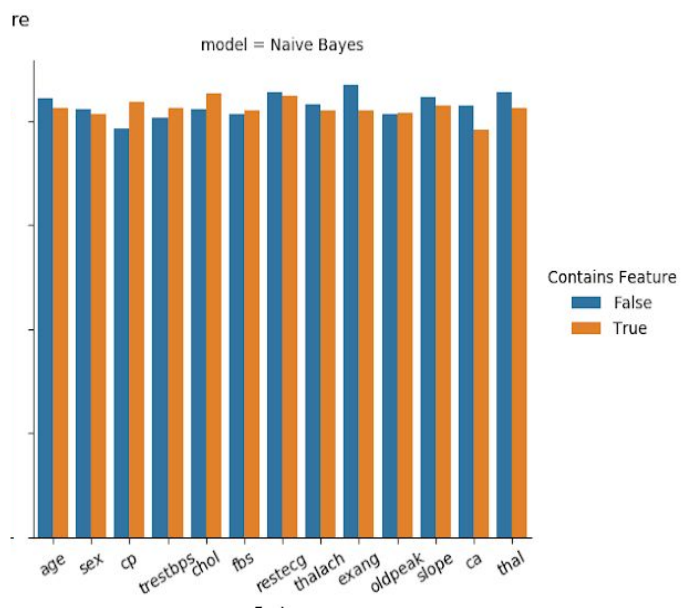
- cp: chest pain type
- trestpbs: resting blood pressure
- chol: serum cholesterol
- fbs: fasting blood sugar
- oldpeak: ST depression induced by exercise relative to rest

Of the features above, **the ones with the greatest positive difference in a Naive Bayes Model after adding the feature are:**

- **cp: chest pain type**
    - Score when removed feature: 0.7867
    - Score when added feature: 0.8367
    - Difference: + 0.0500
- **chol: serum cholesterol**
    - Score when removed feature: 0.8233
    - Score when added feature: 0.0.8533
    - Difference: + 0.0300



54	sex	TRUE	Naive Bayes	0.813333333
55	sex	FALSE	Naive Bayes	0.823333333
56	cp	TRUE	Naive Bayes	0.836666667
57	cp	FALSE	Naive Bayes	0.786666667
58	trestbps	TRUE	Naive Bayes	0.826666667
59	trestbps	FALSE	Naive Bayes	0.806666667
60	chol	TRUE	Naive Bayes	0.853333333
61	chol	FALSE	Naive Bayes	0.823333333
62	fbs	TRUE	Naive Bayes	0.82



Data shown in features_performances_in_models.csv output
Plot shown in features_performances_in_models.png
Produced by plot_feature_importances in main.py

The above two sub-questions of Part two tells us that chest pain is a common indicator of heart disease. Not only does it have a high correlation with prediction, but the Naive Bayes algorithm also increases accuracy when chest pain is added as a feature.

The results, again, confirm our previous hypothesis. This was not surprising to us because chest pain is a well-known symptom of a heart attack. The other features mentioned above, such as cholesterol, blood pressure, etc. are also common characteristics doctors test for when examining a person's heart condition. What was surprising from our three models' visualization is how little age affects the accuracy. We thought that older people are more subjected to weak heart conditions. After reflecting, we realized this might be because the majority of participants are within an older age group to begin with. The research group would have fewer reasons to select children and teenagers with excellent health. They would have contributed data with little variance that is not very interesting.

Reproduction of Results

1. Starter code and resources
    a. Download the zip file or clone the repository in our GitHub repository or download from the submission directly.
    b. You will need these from the repository/submission:
        i. main.py
        ii. analyze_ml_models.py
        iii. cleveland_processed.csv
    c. If you want to use the pre-processed dataset from the original source (instead of the cleveland_processed.csv file produced in steps a-c in Data Preparation of Methodology):
        i. Download the "Data Folder" from Heart Disease Data Set
        ii. Follow Methodology Data Preparation steps a-c to wrangle the dataset into the appropriate format.
        iii. Save as cleveland_processed.csv
2. Prepare these extra dependencies:
    a. cse163 environment
    b. Two extra graphviz packages. To install these two packages, type the following in the terminal of the Visual Studio Code:
        - conda install -c anaconda graphviz -n cse163
        - conda install python-graphviz -n cse163
3. Run the main.py by typing python main.py in code editor terminal
    a. Get these results:
        - In the terminal of Visual Studio Code:
            - Accuracy score per model in one trial
            - Mean accuracy score per model in 10 (default can be modified) trials
            - The correlation coefficient between each of the feature and the label
            - Model cross-validation values for testing purposes
        - In the current directory (of the output files produced, the following are included in the Result section as visualization for our statements):
            - decision_tree_model.pdf
            - random_forest_model.pdf
            - features_performances_in_models.png
            - features_performances_in_models.csv
    b. **Important Notes**:

i. Producing complete results will take up to 10-15 minutes. This is because we run three ML models for multiple trials to get their average accuracy score along with the rest functions in the code file to ensure correctness and more detailed analysis.

ii. **Important**: for your benefit, we added optional code for you to run our code in a shorter time. We added 2 optional lines of code in the plot_feature_importance function in main.py (noted in code).

    1. **Uncomment** these two lines to reduce the number of trials significantly and avoid running our entire experiment, which involves running *many* trials (5 per model per feature).
    2. On the other hand, **comment** these two lines to produce complete results used in the report.

Work Plan Evaluation

Original Work Plan:

1. Set up the starter files using Jupyter Notebook and GitHub and share access
    a. Starter files include main.py, cleveland.data (finish by February 28th)
    b. Break down the main.py into two main parts to address two main research questions
2. Responsibilities
    a. All contributors in meeting: first machine learning model in Part 1 finish by March 2nd
        - Finish transforming data into an accessible format and filtering the data
        - Split out the feature and label columns
        - Set up and train a Decision Tree Classifier Model
        - Record the accuracy score
    b. Trinh: Part 1
        - Finish setting up the Random Forest Classifier model by March 3rd
        - Finish splitting the data and training the model by March 5th
        - Finish recording the accuracy score of the model by March 6th
    c. Yunwei: Part 1
        - Finish setting up the Gaussian Naive Bayes Classifier model by March 4th
        - Finish splitting the data and training the model by March 6th
        - Finish recording the accuracy score of the model by March 7th
    d. Phuong: Part 2
        - Finish making bar plot of the feature importances and prediction by March 7th
        - Finish testing different sets of features using the result of Part 1 by March 9th
        - Finish calculate, record accuracy scores and make chart to compare the predicted and actual values by March 10th
3. Time estimates
    a. Finish writing all the code for transforming data, plotting graphs and building machine learning model by March 10th
    b. Finish writing report of the result by March 13th
    c. Finish preparing materials and practicing for in-class presentation by March 19th

Evaluation:

In our original work plan, we planned to have 7 days to work on setting up, cleaning the data and conducting the three machine learning models for Part 1. After Part 1, we planned to spend the next 6 days to work on finding the different set of features and plotting charts to compare the predicted and actual values for Part 2. This leaves us 3 days to write the report to submit the project on March 13th before the presentation on March 16th.

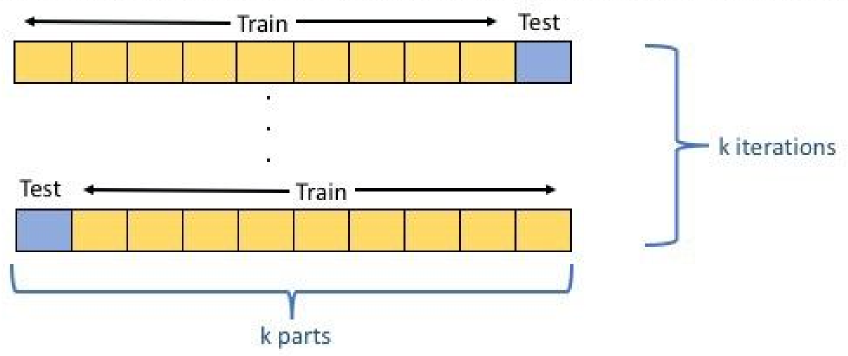
Our cleaning data process went on smoothly without any major issues. However, we did underestimate the time and workload to conduct the three machine learning models. Because one of our machine learning only takes binary classes while the other two take multiple classes. In addition, even with the machine learning models that can handle multiple classes, they still failed to accurately predict the risk of having heart disease. Therefore, we had to go back and clean the data so that the label only contains binary classes to improve the accuracy of the models' predictions.

However, this change in the data also caused us to change Part 2 research questions to make it suitable for binary class data. This change also caused us to replace the original plots with different ones and modify information regarding Part 2 in some sections throughout the report. As a result, it took us extra days to work on Part 1 and Part 2 and led us to take three late days to submit the project on March 16th. Due to the current situation of studying and having finals remotely because of the COVID-19, we have more time to catch up and finish the presentation on time to submit it on March 19th.

Testing

We ensured that our results are reliable by performing k-cross-validation on our model, creating test sets from our data, and running our code more than three times.

We utilized the 10-fold cross-validation to test the performance of our model. With this particular method, we are able to train the model using all of our data. From the model_selection module, there are methods that allow us to split our data into 10 parts. The cross-evaluation method from the same module would then test the model 10 times. Each time, a new part of the data gets to be the test set. The method continues this iteration until all sections of the data have had a chance to be the test set. Below is a visual representation of this process. We chose this validation method because it tends to generate a result that is less biased when predicting. This is desired since we are assessing our model performance.

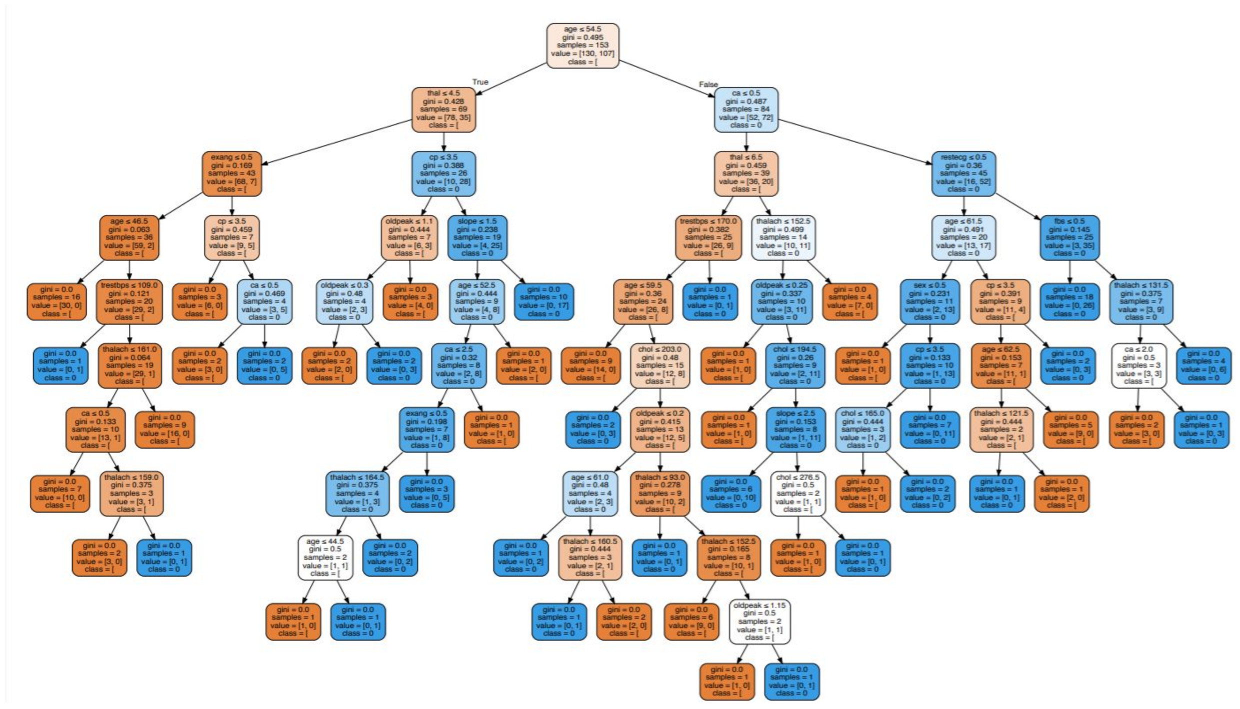


Terribile, Matthew. July 29, 2017. "Understanding Cross Validation's purpose". https://medium.com/@mtterribile/understanding-cross-validations-purpose-53490faf6a86

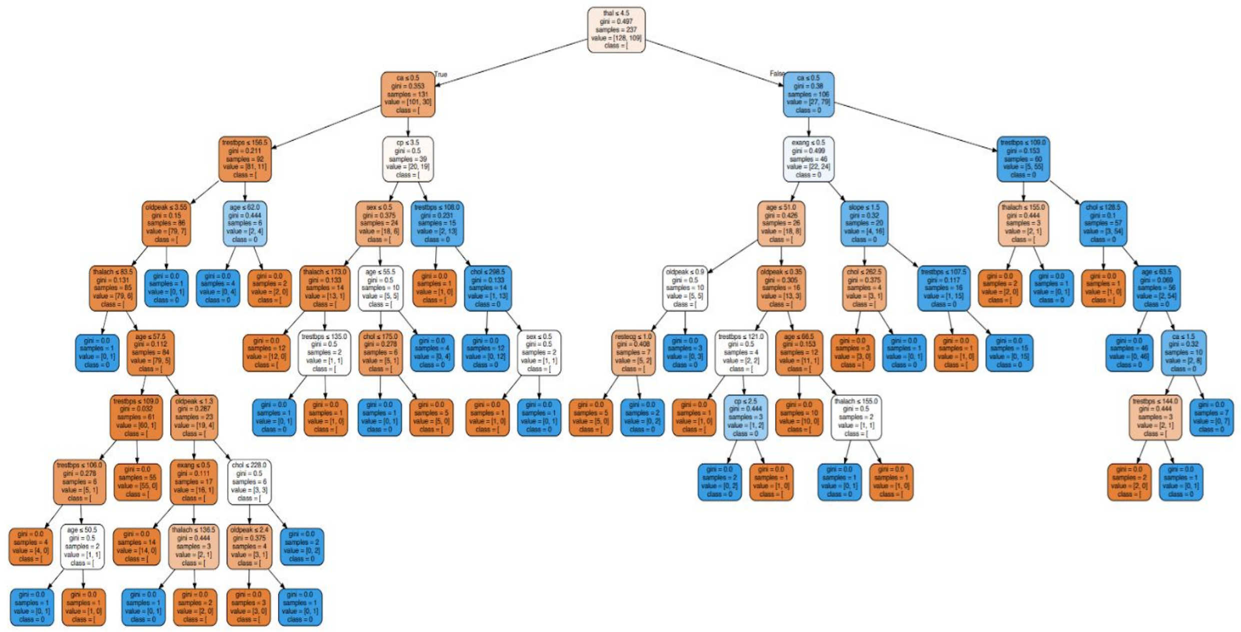
The performance metric returned was a list of accuracy scores. The range of this value aligns with the accuracy score we derived from Part I of our research. An example is for the Naive Bayes Model, the average performance metric returned was 0.838. This is in the range of 0.820 to 0.870 we have been receiving for this particular model. Therefore, we believe that our models are consistent and reliable.

Since this is a research surrounding training machine learning models, the majority of testing is done through terminal outputs and visualizations to check the correctness of the models. We compared inputting training data versus testing data to check that the model returned 1.0 for the training data and a smaller number for the testing data. It was through this test that we realized that our models were tested on a multi-class problem rather than a binary class problem and had to modify our approach.

We also produced visualizations of the trees for the Decision Tree model and the Random Forest model using graphviz. We used the function plot_tree from the lecture to export these graphs. We checked that the features and their values in each box make sense (i.e. left of “age < 54.5” would not test for “age < 54.5” or any number greater than or equal to 54.5 again)



Decision Tree Visualization produced with _plot_tree
For a more clear visualization, see decision_tree_model.pdf in the repository



Random Forest Visualization produced with _plot_tree
For a more clear visualization, see random_forest_model.pdf in the repository

Collaboration

Mitchell Estberg, our Teaching Assistant and Project Mentor, helped us to expand the project's complexity and clarify confusing project's requirements.

In addition, we also referred to the following articles for additional information:

1. Research paper by C. Beulah Christalin Latha and S. Carolin Jeeva
2. Random Forest in Python
3. How to Visualize a Decision Tree from a Random Forest in Python using Scikit-Learn
4. Graphviz Python Example
5. Naive Bayes Classification using Scikit-learn
6. Lecture 9 plot_tree function by Hunter Schafer
7. K-Cross Validation for testing
8. Facet grid for plotting feature importances by faceted by model
9. Seaborn catplot for plotting bar charts
10. Editing facet labels