


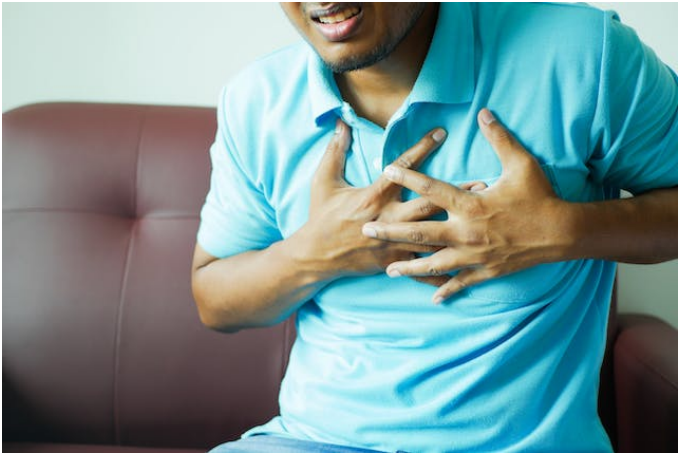
🌿 main 1 Branch 0 Tags 🔗 🔍

🔍 Go to file t Go to file + Add file Code ...

 Bella3s	citation update	36 minutes ago	...	🕒
📁 PDFs	updates	4 hours ago		
📁 images	picture sizes	1 hour ago		
📄 .gitignore	Initial commit	2 months ago		
📄 README.md	Update README.md	1 hour ago		
📄 citation.md	citation update	36 minutes ago		
📄 heart.csv	add data	2 months ago		
📄 index.ipynb	typos	1 hour ago		

📖 README

Heart Disease Classification Project



Overview

This project creates a binary classification machine learning model, using the Python library Scikit-Learn to predict whether or not a patient has heart disease. The project goes through a data exploration and cleaning process. Pipelines are utilized for both preprocessing and data as well as for iterating through different types of models. After tuning the two models with highest performance a final model is chosen and evaluated. Lastly, recommendations pulled from the model are given.

The Business + Business Problem

Heart Disease is a leading cause of death not only in the United States, but globally as well according to the World Health Organization and Center for Disease Control. Cardiovascular diseases are estimated to be the cause of death for 17.9 million lives each year globally.

This project is framed by a health care provider who would like to prevent further deaths by heart diseases with early identification and treatment. A binary classification machine learning model is created that takes in patients' information and predicts whether or not heart disease is present. The upmost goal is to identify as many people as possible who are ill prior to any serious health crises , such as a heart attack or stroke. Missing someone could be potentially fatal. However, the health care providers are also concerned with falsely categorizing healthy people with heart disease as this can place an undue burden on a healthy patient.

The data set used in this project, "Heart Failure Prediction Dataset," can be found on [Kaggle](#). The data is a combination of 5 separate datasets from the UCI Machine Learning Repository under the index of heart disease datasets.

Creating a Machine Learning Model

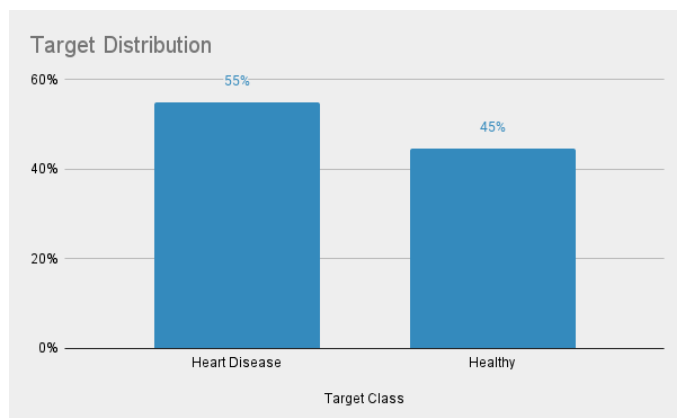
Data Exploration + Cleaning

There are 11 independent variables present in the data set:

Variable	Description
Age	Age of the patient
Sex	Sex of the patient
ChestPainType	Chest Pain Type: Typical Angina (TA), Atypical Angina (ATA), Non-Anginal Pain (NAP), or Asymptomatic (ASY)
RestingBP	Resting blood pressure (mm Hg)
Cholesterol	Serum Cholesterol (mm/dl)
FastingBS	Fasting blood sugar: 1 if FastingBS > 120 mg/dl, otherwise 0
RestingECG	Resting electrocardiogram results: Normal (Normal), Having ST-T wave abnormality (ST), or Showing probable or definite left ventricular hypertrophy by Estes' criteria (LVH)
MaxHR	Maximum heart rate achieved
ExerciseAngina	Yes if exercise-induced angina is present, otherwise No
Oldpeak	ST depression (ST relates to positions on the ECG plot)
ST_Slope	The slope of the peak exercise ST segment: Upsloping (Up), Flat (Flat), or Downsloping (Down)
HeartDisease	Target: 1 if the patient has heart disease, otherwise 0

The project first checks for missing or invalid values, finding that the Cholesterol variable has a significant amount of datapoints with a zero value. Numpy NaNs are put in place of these values such that a SimpleImputer can be used later in a Pipeline step during model creation.

Furthermore, the project explores the distribution of the target variable, finding a slight imbalance between the two classes (healthy patients and patients with heart disease). However, the imbalance is not great enough to call upon any data augmentation processes such as SMOTE.



Model Iteration

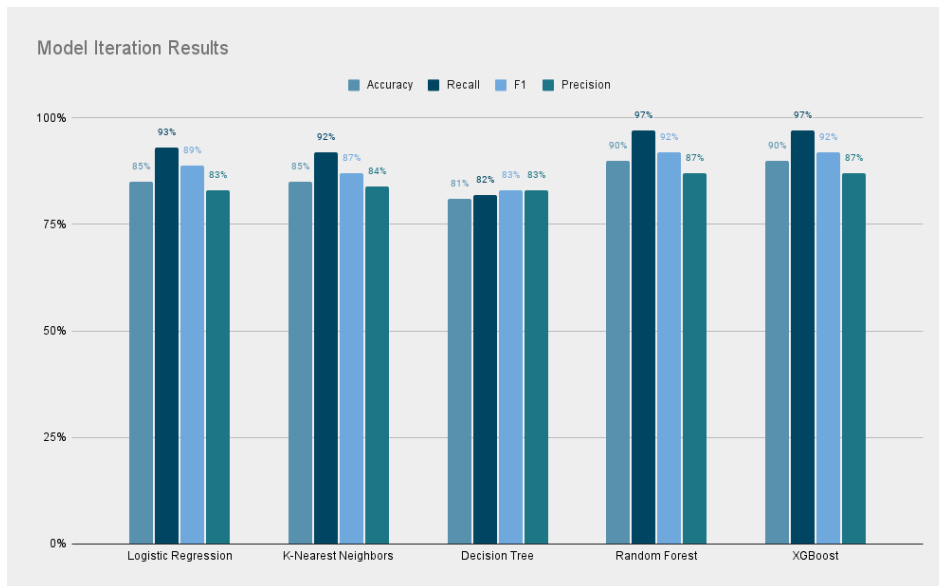
The model utilizes pipelines in order to process the data for modeling and iterate through five different models (using all of the default parameters). The preprocessing pipeline includes steps for encoding the categorical data, inputting the mean for missing values via a SimpleImputer, and scaling the data with StandardScaler. A function is created that takes in a list of models, and for each model:

- Creates a new pipeline with the preprocessing pipeline and the model as the two steps
- Fits the new pipeline
- Calls upon another helper function to retrieve evaluation metrics for the model based on evaluation data

Then the function returns a dictionary of the evaluation metrics for each model, a list of confusion matrices for each model, and a list of the fitted models themselves.

The five models that the function iterates through are:

- Logistic Regression (used as the baseline model to compare all other results to)
- K-Nearest Neighbors
- Decision Tree Classifier
- Random Forest
- XGBoost



The two most performant models were the Random Forest and XGBoost models.

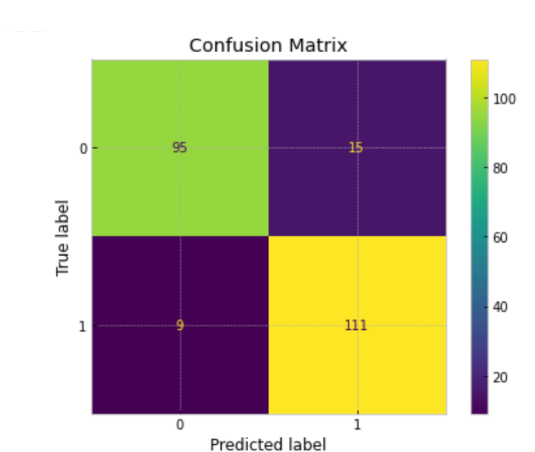
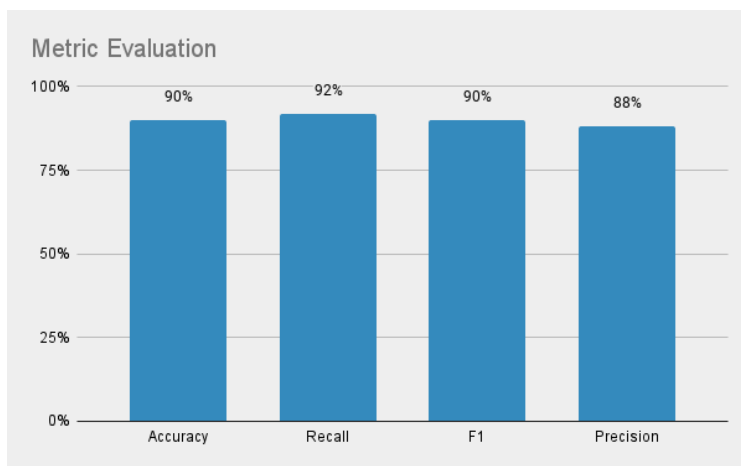
Hyperparameter Tuning

In an attempt to create even better models, the project then goes through a process of hyperparameter tuning of these two most performant models. For each model, a GridSearchCV and a RandomizedSearchCV are run with values/distributions for four different parameters. Furthermore, each search is run with a default scoring, the scoring set to Precision, and the scoring set to Recall; Twelve new models in total. This is the most time-consuming portion of the project, computationally speaking, and unfortunately none of the results beat the metrics found with the default parameters.

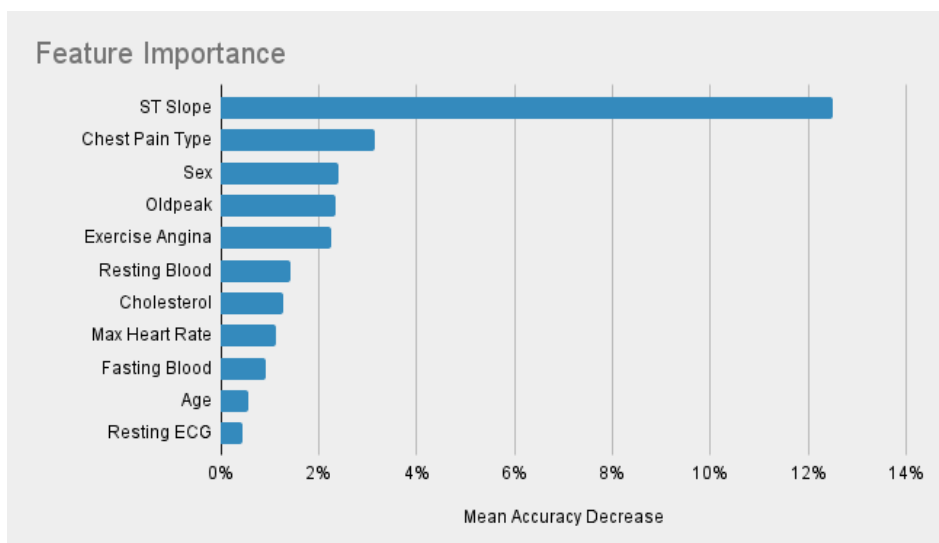
Thus, the Random Forest model with default parameters is chosen as the final model.

Final Model Evaluation

After a final model is chosen, it is trained with all of the training data and evaluated with the hold-out test data.



From the above graphs we can see that our final model is 90% accurate and has a recall score of 92%. This is slightly worse than what we saw with the validation data, but still quite a well-performing model.



The feature importance graph shows that the shape of the ST slope is by far the most important feature in our model, with the mean accuracy of the model decreasing by about 12% when this feature is not included. The second most important feature is Chest Pain Type.

Conclusion

Three recommendations are given to the health care providers in terms of what should actually be done or taken away from this project.

1. The model should be put to use. Ideally there is already a database of patient information that can be simply fed into the model. Any patients that are categorized as having heart disease can be brought in for further testing and treatment if needed. Furthermore, future patient information can be fed into the model as well for early diagnosis of heart disease. First and foremost, the model is a tool to be used.



Releases

No releases published

[Create a new release](#)

Packages

No packages published

[Publish your first package](#)

Languages

● Jupyter Notebook 100.0%