Heart Disease Prediction
UCI Study Data

Brad Doty

November 2021

Problem Statement

1. Find a classifier to predict the target, which is heart disease as measured by angiogram, based on the 13 diagnostic test results in this UCI dataset on Kaggle.
2. Identify which diagnostic tests in this study data, formulated as features in these classification models, carry the most predictive information for these models.

Datasets

The dataset actually deviates from the documentation in several columns. Some variables don't have the same number of values as described and one column contains floats with one decimal place, rather than integer categories. Several of the column names are cryptic. In the Kaggle comments, a user posts a "corrected" dataset, which he says is more accurate than the original post. I investigated on Kaggle and UCI.

Information Digging at UC - Irvine

I could not find any corroborating evidence for either dataset on Kaggle, so I did some digging into the UC Irvine ML Data Repository at http://archive.ics.uci.edu/ml/machine-learning-databases/heart-disease/

This study was done in 1988 and analyzed by many papers over the years, BTW. There were 76 attributes considered in the study, but only 14 attributes were reported. This dataset is from the Cleveland Clinic. There were 3 other datasets from Long Beach VA Hospital, Hungary, and Switzerland.

Found these improved feature descriptions: cp: chest pain type 1 = Typical Angina, 2 = Atypical Angina, 3 = Non-Anginal Pain, 4 = Asymptomatic. But do we convert to 4 to 0, or slide the 1-based index down to the 0-based index?

Applied improved explanations in the data wrangling notebook Column Description list and improved the name set.

This is the clincher on which dataset to use. The UCI archive states that column 58 of the full set is column 14, the predicted attribute, of the reported set and it has only 2 values.

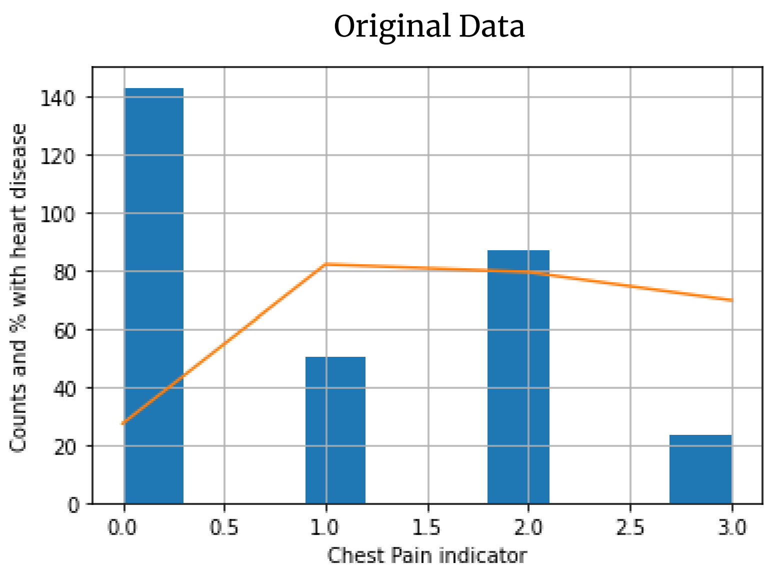
> 58 num: diagnosis of heart disease (angiographic disease status) -- Value 0: < 50% diameter narrowing -- Value 1: > 50% diameter narrowing

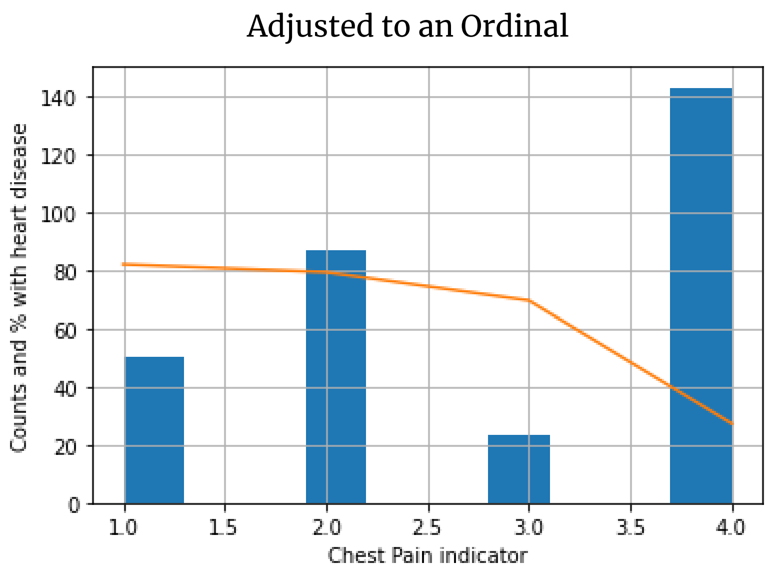
Data Science Approach

After verifying that the original dataset with the binary target variable was preferable to the alternative dataset, I performed Data Wrangling, EDA, Feature Engineering, and used each of the main supervised classification algorithms to find the best-fitting model to predict heart disease from the dataset. I tracked and compared the coefficients or feature importances available from each model for comparison of features. Each step is performed and documented in its own Jupyter notebook, available in my Github repository.

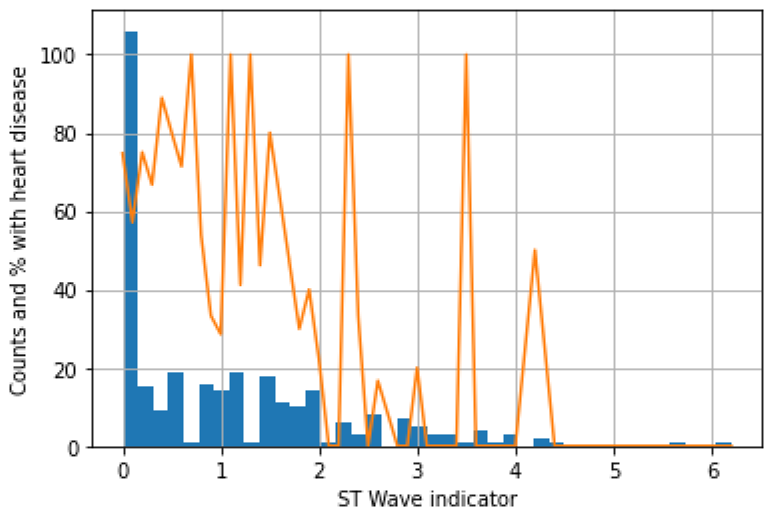
Several categorical features are ordinal, that is their correlation with the target moves up or down in a monotonic fashion with their values. Some features had a value in the opposite direction, so I moved values to make them ordinal.

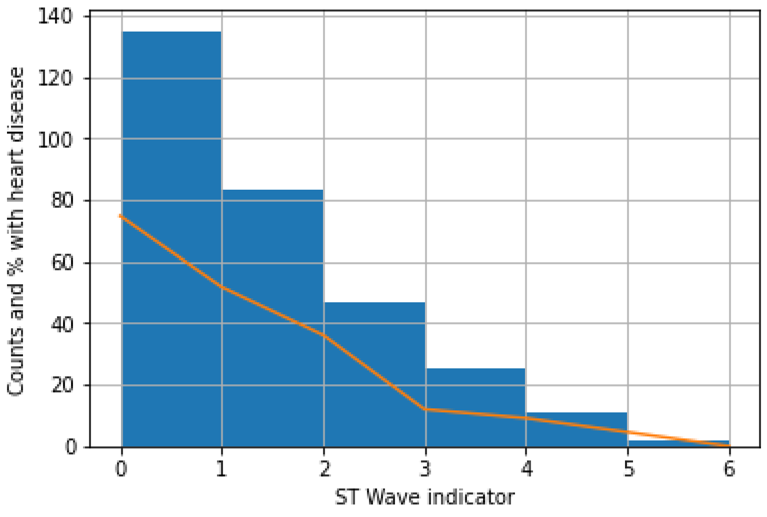
For instance, the 4 values of Chest Pain are 0: No Pain, 1: Typical Angina, 2: Atypical Angina, and 3: Other Pain. Patients reporting 0 had a much lower risk of the target, while patients with 1 through 3 had decreasing risk. So the 0 fits a decreasing risk order as a 4. I replaced 0s in this column with 4s to make it a proper ordinal.





STWave, a coding of characteristics of the ECG wave during a stress test, came in with many decimal values, rather than 4 integer categories as documented. I found that putting the histogram into bins greatly smoothed the percentage correlation with the heart disease target. So I rounded, with a couple of tweaks to balance out the categories, to make a nice ordinal which has an inverse relationship with the target.





Exploratory Data Analysis with the adjusted columns shows that none of the candidate features are highly correlated to each other. The largest correlation is -0.55 between STWave and STSlope. The correlations with the target were not that high and surprisingly low or negative in a few cases. See Fig. 3 on the next page.

I used all 13 candidate features. Split the features into training and test sets. Scaled the training set and the test set separately to avoid any feedback from the test set to the training set.

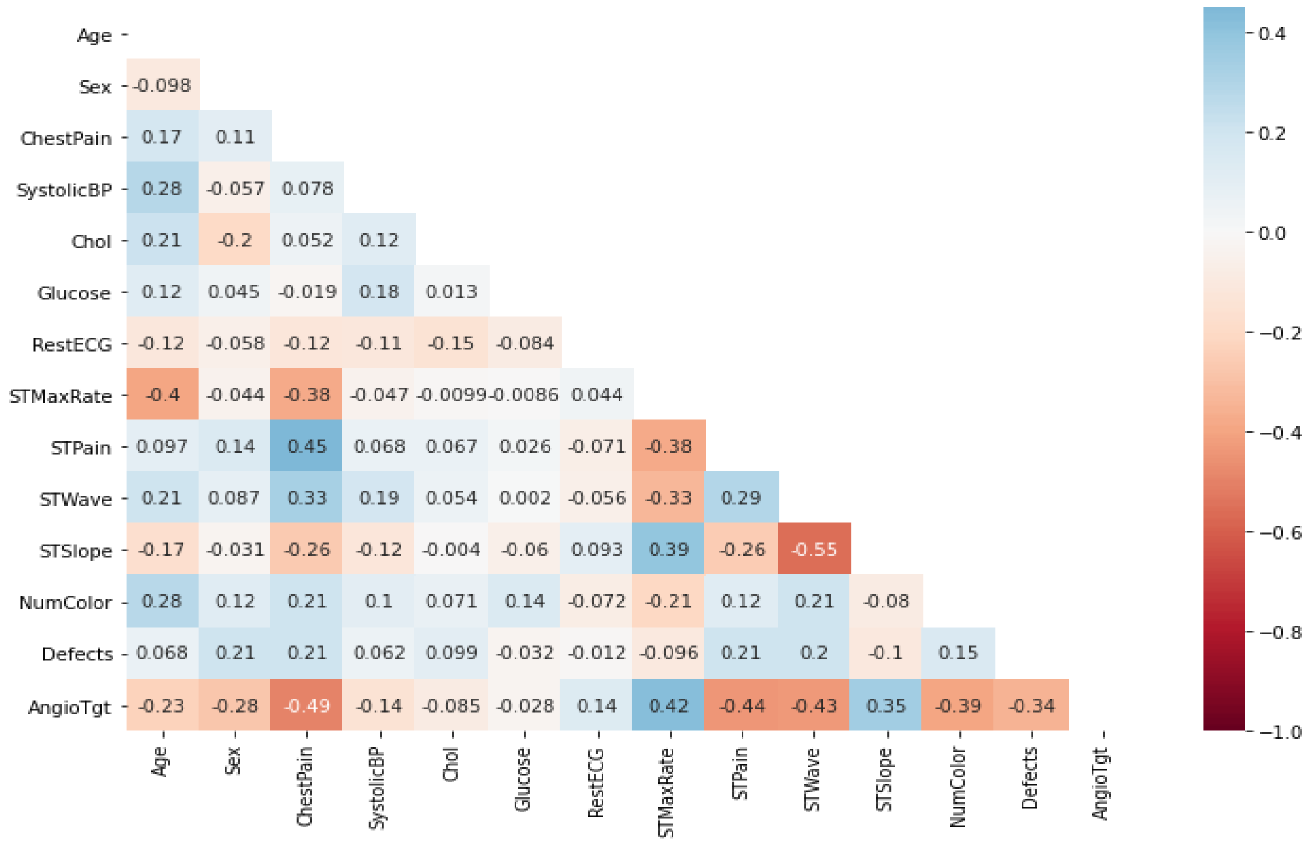


Figure 3. Correlation Matrix

Predictive Modeling

I used 5 algorithms to build 10 classification models. Here are the fit metrics of each model.

Model	Precision	Recall	Accuracy	F1	AUC of ROC
Logistic Regression	92%	79%	87%	85%	92%
Logistic with scaled data	89%	83%	87%	86%	93%
Decision Tree	83%	66%	77%	73%	77%
Decision Tree with scaled data	82%	79%	82%	81%	82%
Random Forest (max d=9, gini, 100 trees)	85%	79%	84%	82%	91%
Random Forest CV1 (md=5, entropy, 75 trees, mxfeat=5, minsplit=5)	83%	83%	84%	83%	89%
Random Forest CV2 (md=5, entropy, mxfeat=5, 80 trees, minsplit=2)	83%	83%	84%	83%	91%
K Nearest Neighbors (Scaled data) k=13	86%	83%	85%	84%	91%
Support Vector Machine (RBF kernel) (Scaled data)	86%	83%	85%	84%	91%
Support Vector Machine (linear kernel)	83%	83%	84%	83%	92%

LOGISTIC REGRESSION. This is not a distance-based algorithm, but it failed to converge with the unscaled data, even when I bumped up the iterations from 100 to 1000. I tried the scaled data and it converged. The unscaled data had better precision, the same accuracy, and slightly worse recall, F1, and ROC AUC. The scaled logistic regression has the best fit overall.

DECISION TREE. The decision tree was clearly inferior to logistic regression on this data. So I thought I'd try scaled data on it, and the performance improved, except precision ticked down from 83% to 82%. ROC AUC was still 11 points below logistic.

RANDOM FOREST. I ran one (unscaled) random forest with max depth matching the depth of the decision tree, 9, and all other values at default, including Gini criterion and 100 trees. It beat the decision tree, especially in ROC AUC (9 pct. points), but still trailed logistics.

I ran cross-validation with this grid: n trees: 25, 50, ..., 175. Gini and entropy. Max depth: 5, 9, 10, 15. Min samples split: 2, 3, 5. Max features (to consider at splits, still using all 13 features): 5, 10, 13. This yielded a best model that was near the first random forest, but 2 points under in terms of ROC AUC.   (75 trees, entropy, max depth 5, min samples split 5, max features 5)

I adjusted the grid to bracket the best model. n trees: 50, 60, ... , 110. Gini and entropy. Max depth: 5, 10, 20, 25. Min samples split: 2, 5, 6, 7. Max features: 5, 10, 15. This came back with exactly the same fit metrics, except ROC AUC ticked up 2 to 91%. (80 trees, entropy, max depth 5, min samples split 2, max features 5)

K NEAREST NEIGHBORS.

Deliverables

A.	Jupyter notebooks describing analysis and design decisions, Python code, and data visualizations for the Data Wrangling, EDA, Feature Engineering, and Predictive Modeling phases
B.	Project final report
C.	Presentation slide deck