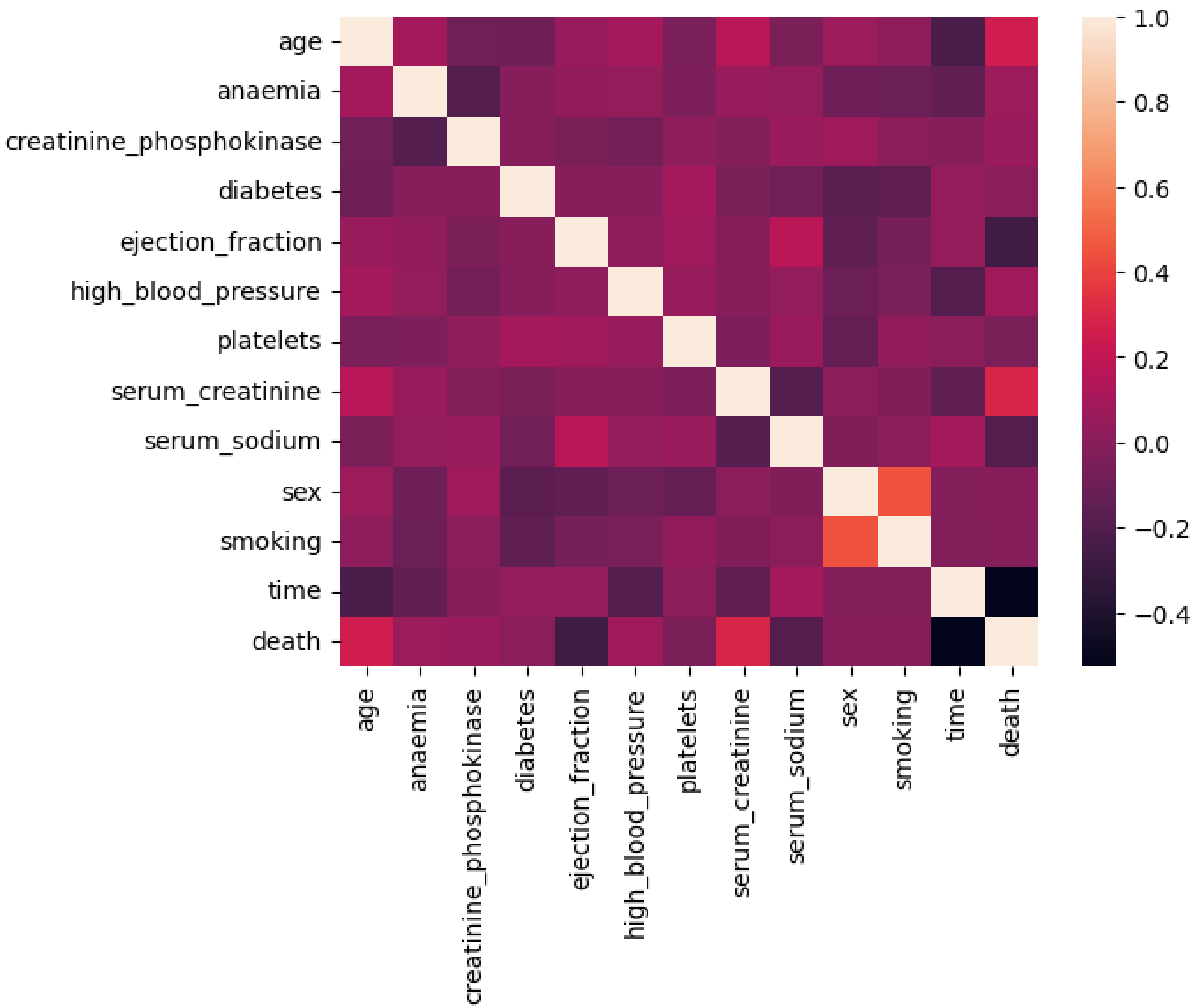
Heart failure clinical records using Decision Trees

Philmore Koung

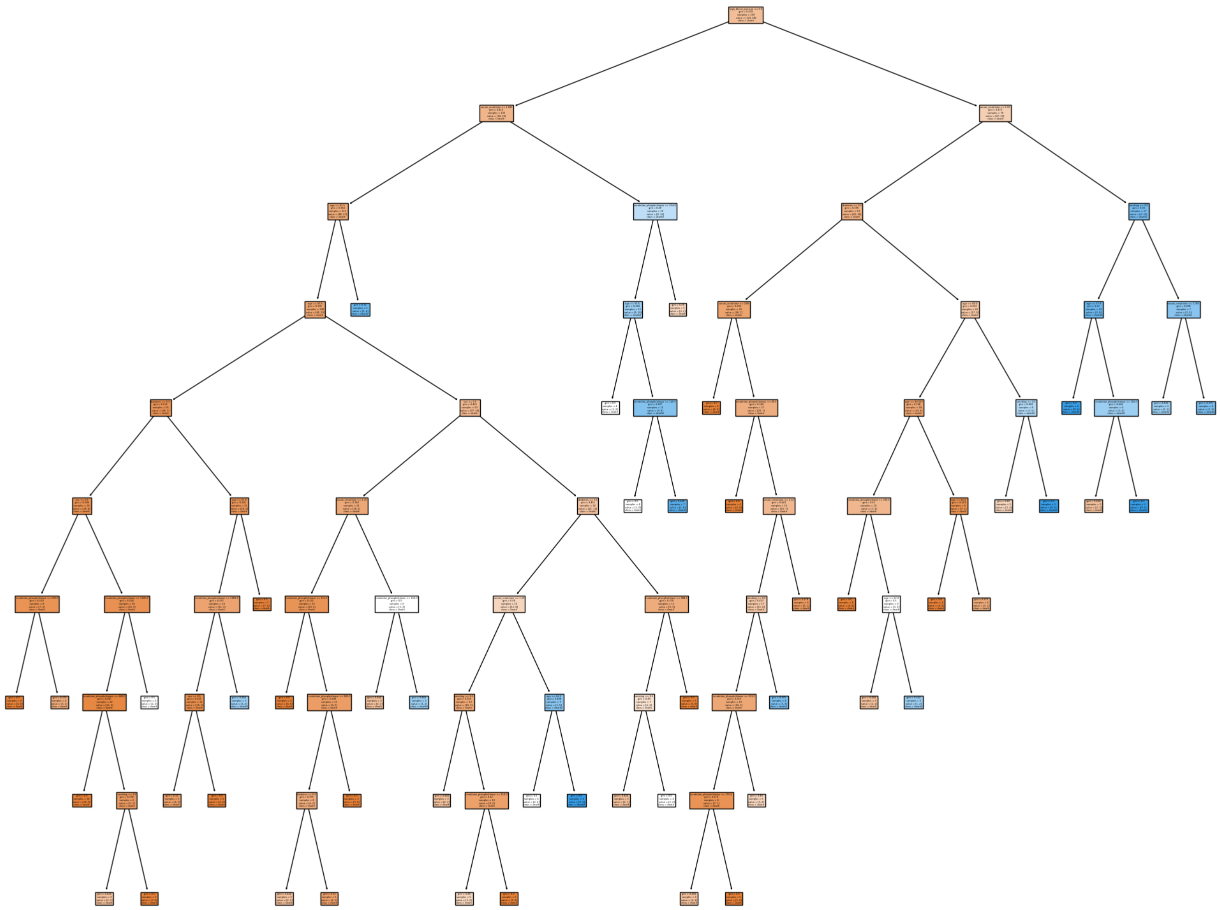
Dataset
This dataset was sourced from UCI Machine Learning Repository and contains 299 patients who had experienced heart failure, 12 features, and a target variable (of type boolean) with 0 indicating the patient did not die during the follow-up period and 1 otherwise.

Pre-Processing
The dataset contains no missing or null values and a seaborn heatmap shows us that age and serum creatinine as the two most important variables with the highest correlation to death. On the other hand, time had the lowest correlation with death. The final dataframe excluded the variables anaemia, time, ejection fraction, serum sodium, and platelets as they did not have high correlation compared to the remaining variables.



Simple Decision Tree
Using DecisionTreeClassifier from sklearn and GridSearchCV to find the best hyperparameters, I was able to obtain an accuracy of 0.667 which means that 66.7% of the time my model is able to predict if a patient dies during the follow-up period. The best parameters suggested by

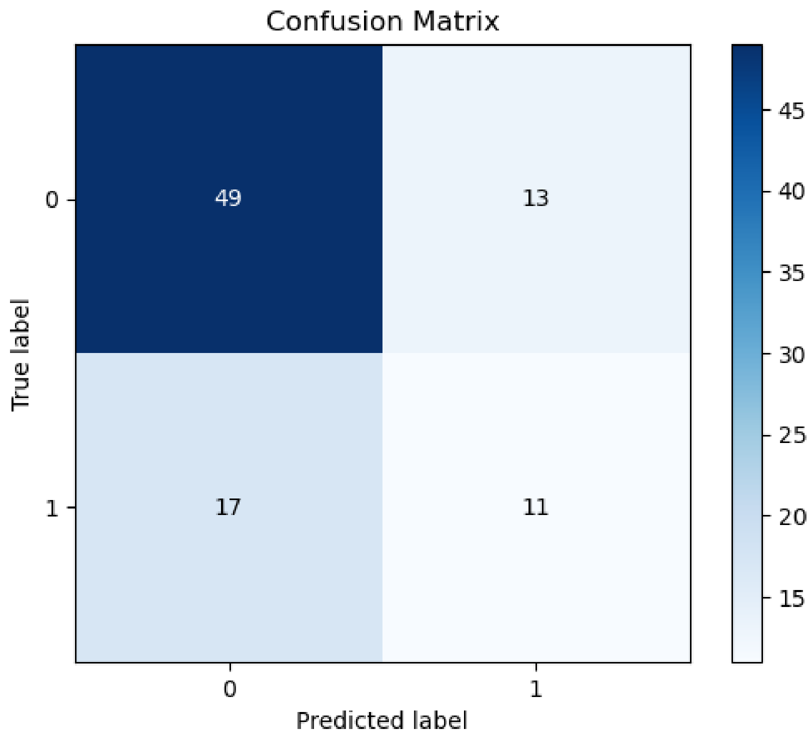
GridSearchCV were 'criterion': 'gini', 'max_depth': 10, 'max_features': 'sqrt', and 'min_samples_leaf': 3. The following classification report and plots utilize these hyperparameters.



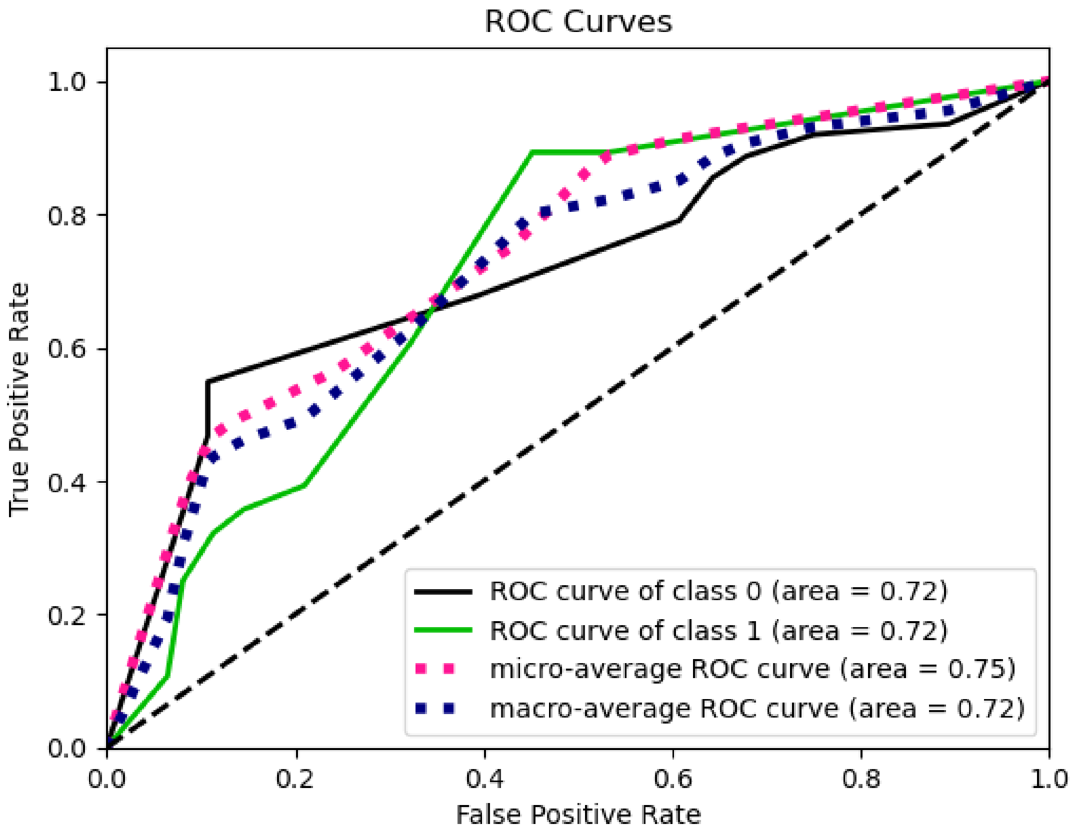
Simple Decision Tree

```
              precision    recall  f1-score   support

           0       0.74      0.79      0.77        62
           1       0.46      0.39      0.42        28

    accuracy                           0.67        90
   macro avg       0.60      0.59      0.59        90
weighted avg       0.65      0.67      0.66        90

Predicted labels:  [0 0 0 1 0 0 1 0 0 0 0 0 0 0 1 0 1 0 0 0 0 0 0 1 0 0 0 1 0 1
0 0 0 0 0 0 1
 1 0 1 0 1 0 0 0 0 0 0 0 0 0 0 1 1 1 1 0 0 0 1 0 1 1 1 0 0 0 1 0 0 1 0 0 0
 0 0 0 0 0 0 1 0 0 0 0 1 0 1 0 0]
Accuracy:  0.6666666666666666
```

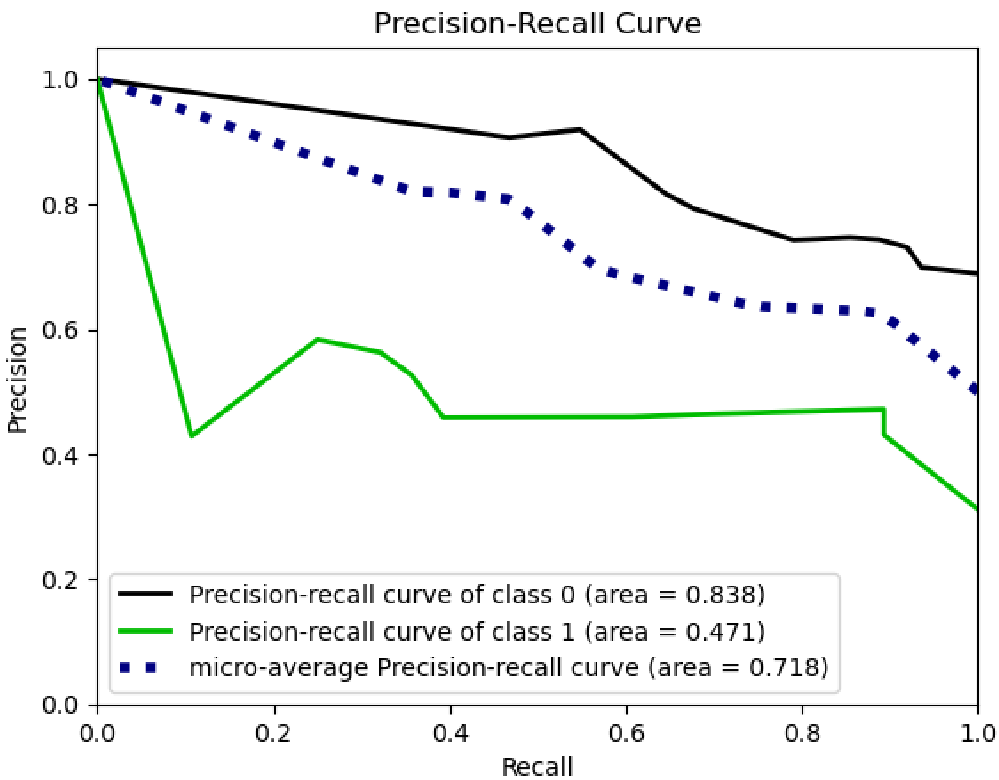
The precision and recall scores tells us that we are able to predict whether a patient does not die during the follow-up period with high accuracy; however, predicting whether the patient dies during the follow-up period is worse than random guessing as it is lower than 0.5. In general, our f1-score tells us that our overall model is better than average at predicting both outcomes at our weighted average score being 0.66



Our confusion matrix supports our precision, recall, and f1-score as it visually shows that we are able to predict true positives in the vast majority but are unable to predict true negatives at a rate better than random guessing. It can be noticed that our model predicts the wrong outcome for class 1 more often than the right outcome. Given that this is a clinical dataset, this decision tree is a poor model in practice as our accuracy is too low.



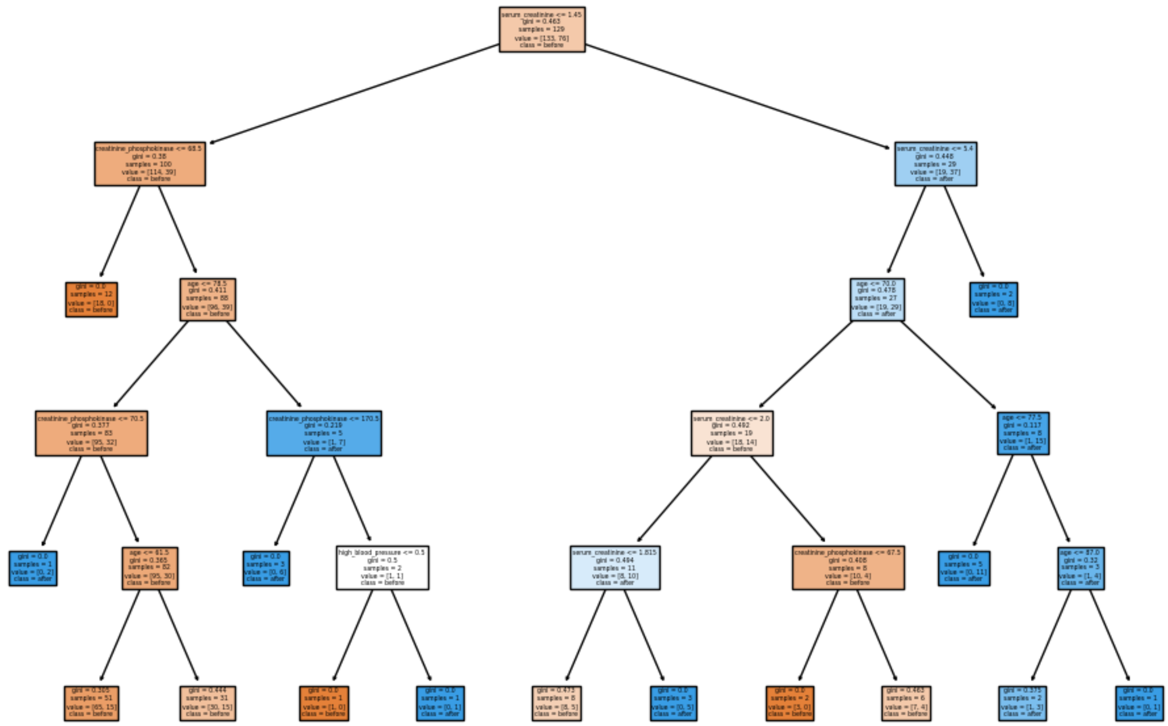
The ROC curve for this decision tree suggests that this model is a decent model and not a bad model in general as our ROC curves lie above the random classifier line. This suggests that our true positive rate value is high while our false positive rate is low at all classification thresholds. However, a better model would have high values for true positive rate creating a “right angle” (maximizing Area Under the Curve) shape on the plot.



The Precision-Recall Curve supports our findings that our model's ability to predict class 1 is greatly insufficient as it lies well below our micro-average curve. At the same time, our model is good at predicting class 0. This means that our precision and recall tradeoff is both low at all thresholds for class 1 indicating a poor fit for the model. Meanwhile, our precision recall tradeoff for class 0 cases is much better at all thresholds. Our simple decision tree model is generally a decent model, but could be better at predicting class 1 samples.

Random Forest Classifier

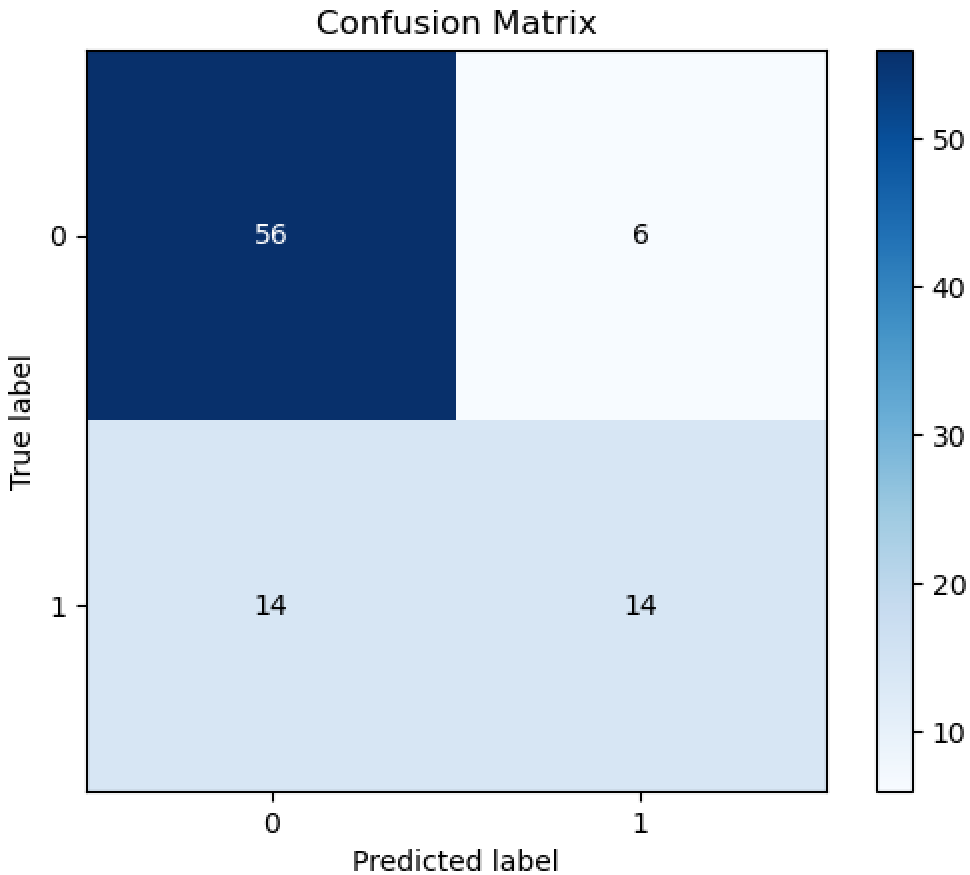
Similar to our Simple Decision Tree, I utilized GridSearchCV to finetune the best hyperparameters for our model. The hyperparameters used were 'max_depth': 5, 'max_features': 'sqrt', 'n_estimators': 300. This model produced a better overall accuracy with a score of 0.778



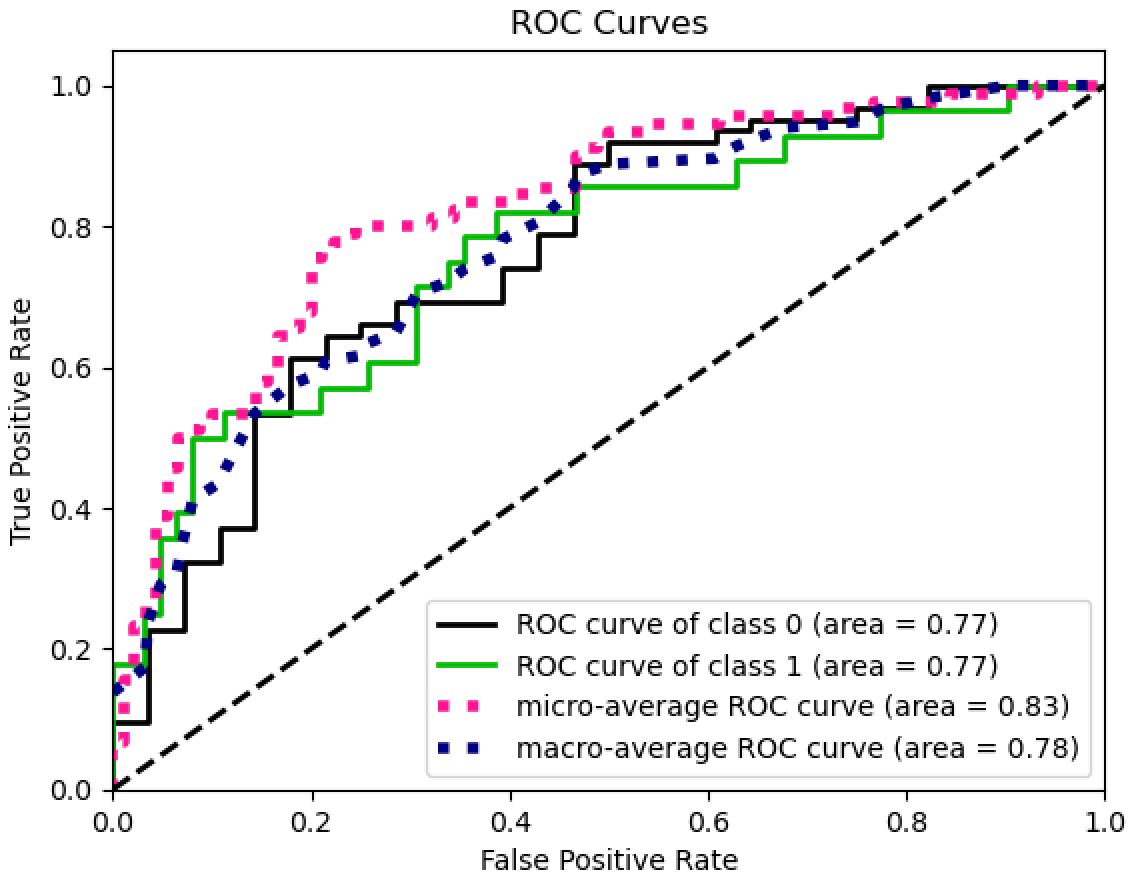
Random Forest Classifier

```
              precision    recall  f1-score   support

           0       0.80      0.90      0.85        62
           1       0.70      0.50      0.58        28

    accuracy                           0.78        90
   macro avg       0.75      0.70      0.72        90
weighted avg       0.77      0.78      0.77        90

Predicted labels:  [0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 1 0 1 0 0 0 0 1 0 0 0 0 0 1
0 0 0 0 0 1 0
 0 0 1 0 0 0 0 0 1 0 0 0 0 0 1 0 0 1 0 0 0 0 0 0 0 1 1 0 1 1 0 0 0 0 0 0 0
 0 0 0 0 0 0 1 0 1 1 0 1 0 1 0 1]
Accuracy:  0.7777777777777778
```

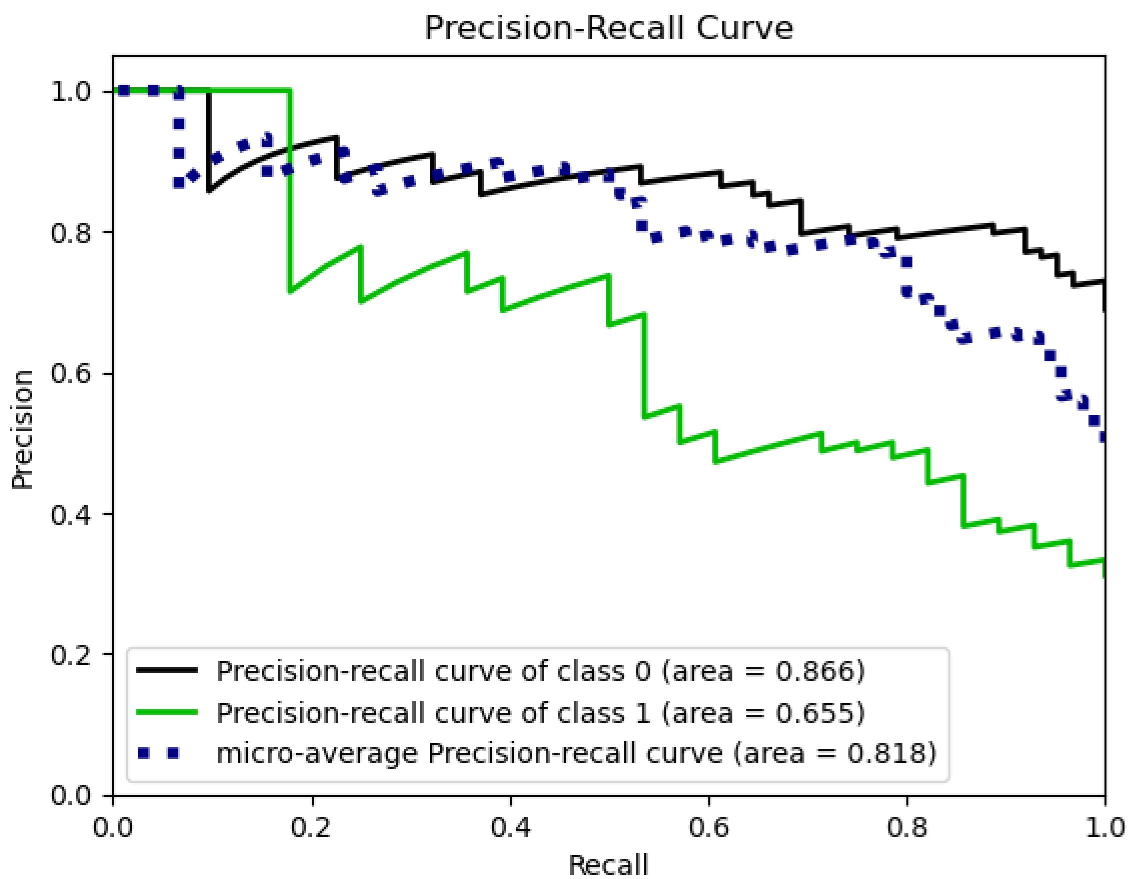
Compared to our simple decision tree, our random forest classifier is a significantly better model. However, it remains that our recall score for class 1 variables is still not better than random guessing. Furthermore, compared to our simple decision tree model our precision score is significantly higher which creates a higher f1-score and overall better accuracy in addition to higher precision and recall values for class 0 variables.



Our confusion matrix also depicts great improvement from our simple decision tree model. Our model still identifies class 0 in an overwhelmingly majority of cases while class 1 prediction is improved slightly.



Our ROC curve shows that our model is a good fit as it begins to resemble a “right angle” structure as it leans towards the top left of the plot. Each ROC curve is relatively close to each other indicating not a huge difference between one or the other.



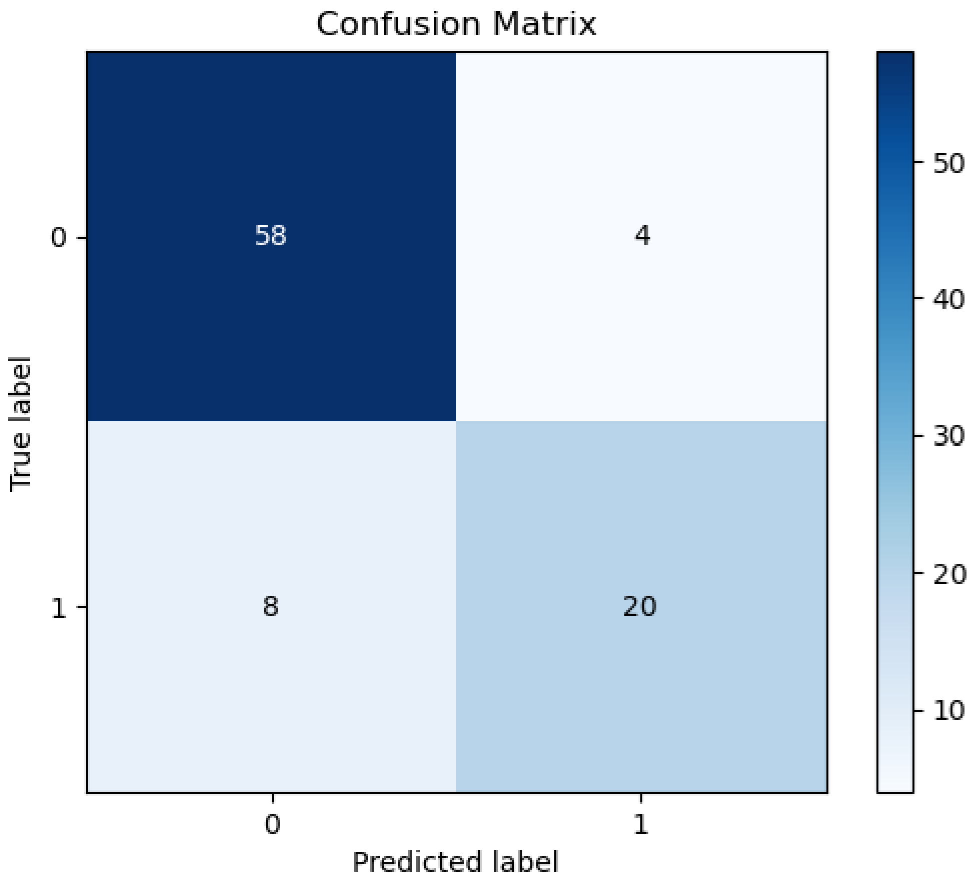
The Precision-Recall Curve is much better than our simple decision tree but our precision-recall curve for class 1 is still lacking as it is still near baseline. However, it is still sufficient to say that this model is an improvement from our original model at predicting our target variable.

Adaboost Classifier

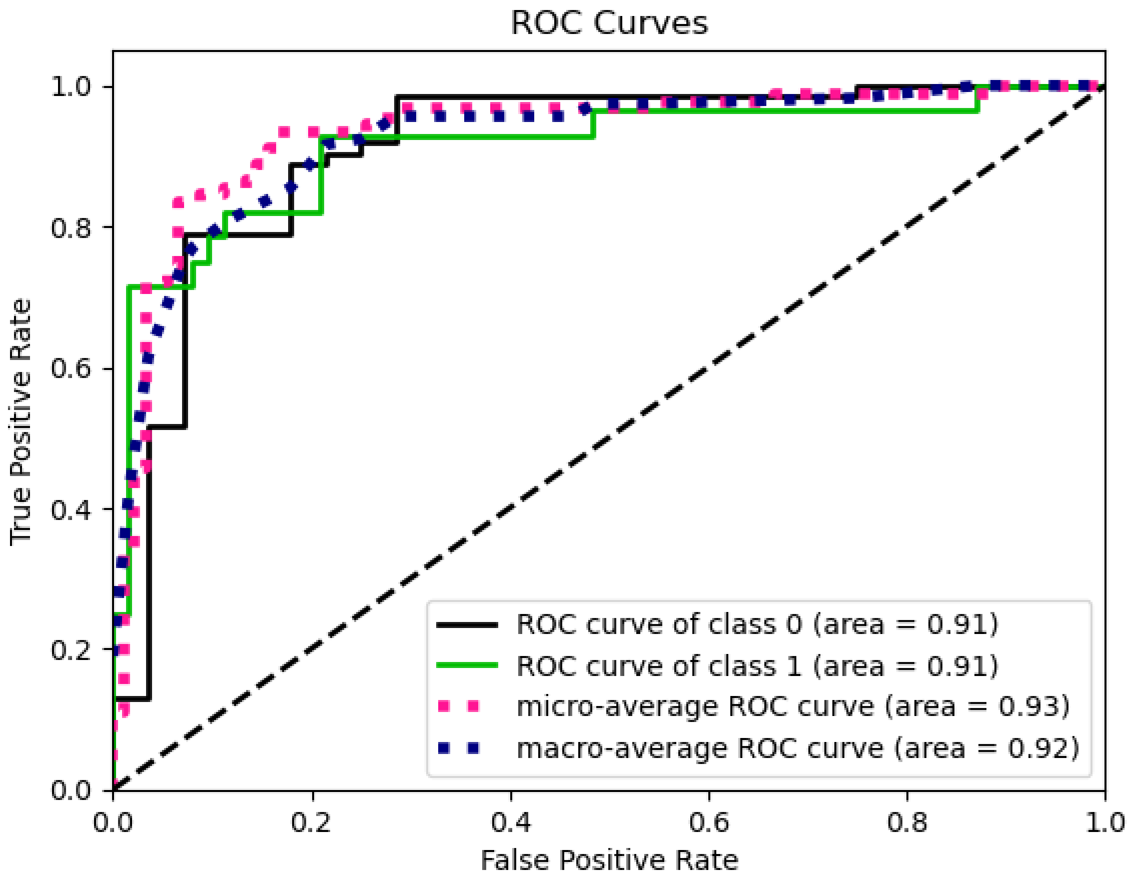
Using RandomForestClassifier as our base classifier for Adaboost, I also used GridSearchCV to identify the best hyperparameters for our model. The best hyperparameters are 'learning_rate': 1 and 'n_estimators': 10.

```
Classification Report:
              precision    recall  f1-score   support

           0       0.88      0.94      0.91        62
           1       0.83      0.71      0.77        28

    accuracy                           0.87        90
   macro avg       0.86      0.82      0.84        90
weighted avg       0.86      0.87      0.86        90
```

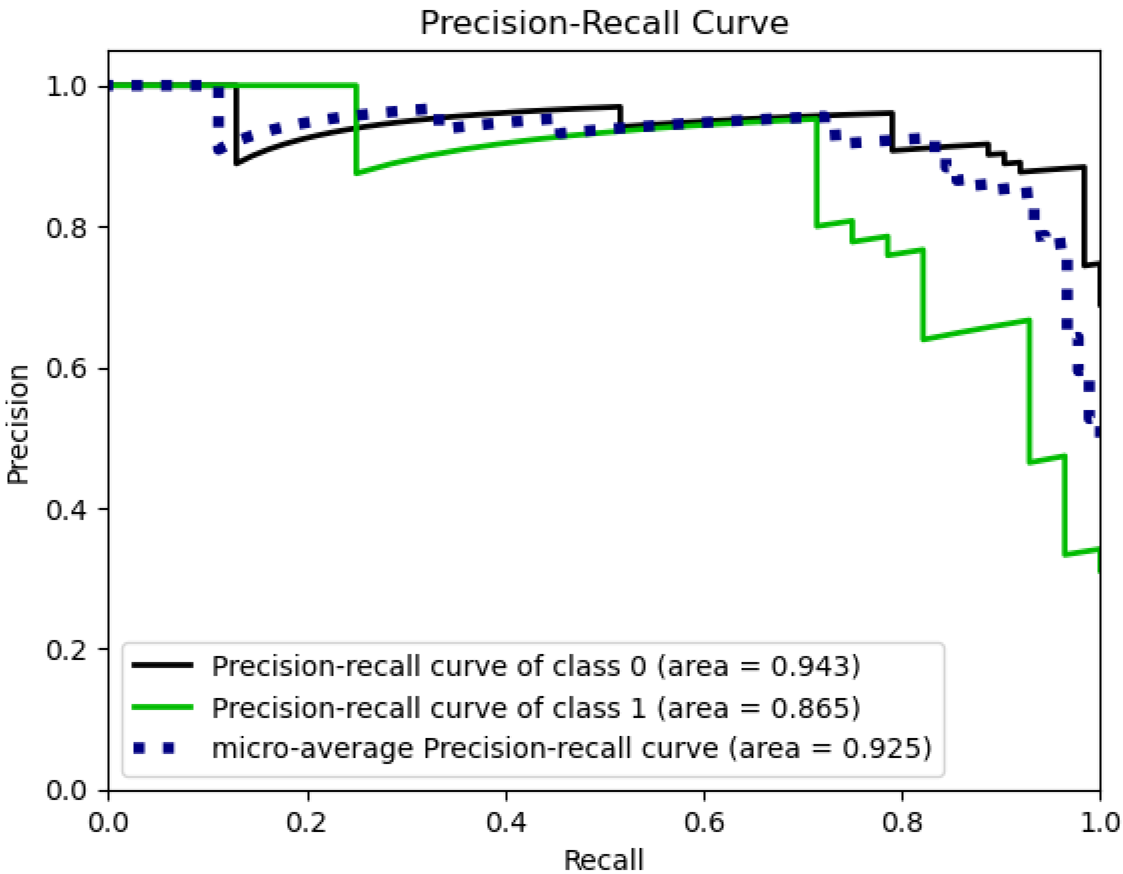
Our model produced significant improvement to our previous two with an accuracy score of 0.867. Notably, our recall and precision scores for class 1 cases increased to a value far better than random guessing unlike our other two models. As such, our f1-score reflects this improvement at 0.91 and 0.77 for both cases respectively.



Our confusion matrix visualizes this improvement greatly as we can see our model still predicts class 0 correctly in the vast majority of cases while our model now also majorly predicts class 1 cases correctly.



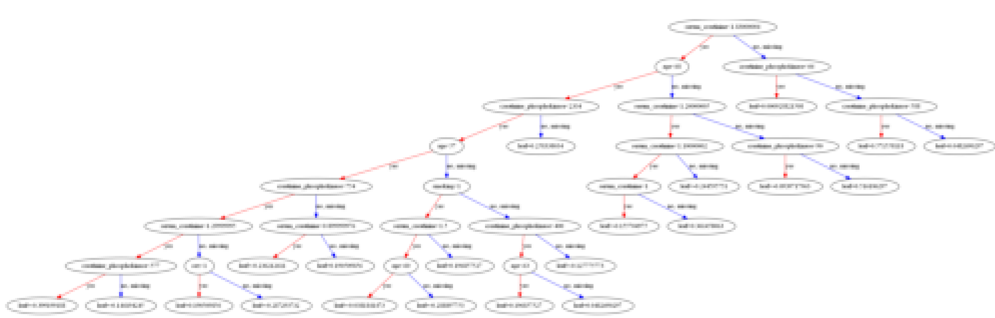
Unlike in our previous two models, our Adaboost model now reflects the “right-angle” shape of a good model in our ROC Curves plot. This suggests that our model is an extremely good fit for our data. Previously, our models tread closely to the random classifier line but now lies far above it.



Likewise, our Precision-Recall Curve plot also reflects the “right-angle” shape of a good model as our Precision-recall curve for both classes and micro-average Precision-recall curve have increased in value. This tells us that despite the tradeoff between precision and recall at different thresholds, our precision and recall values remain high; meaning we have a low false positive rate and low false negative rate.

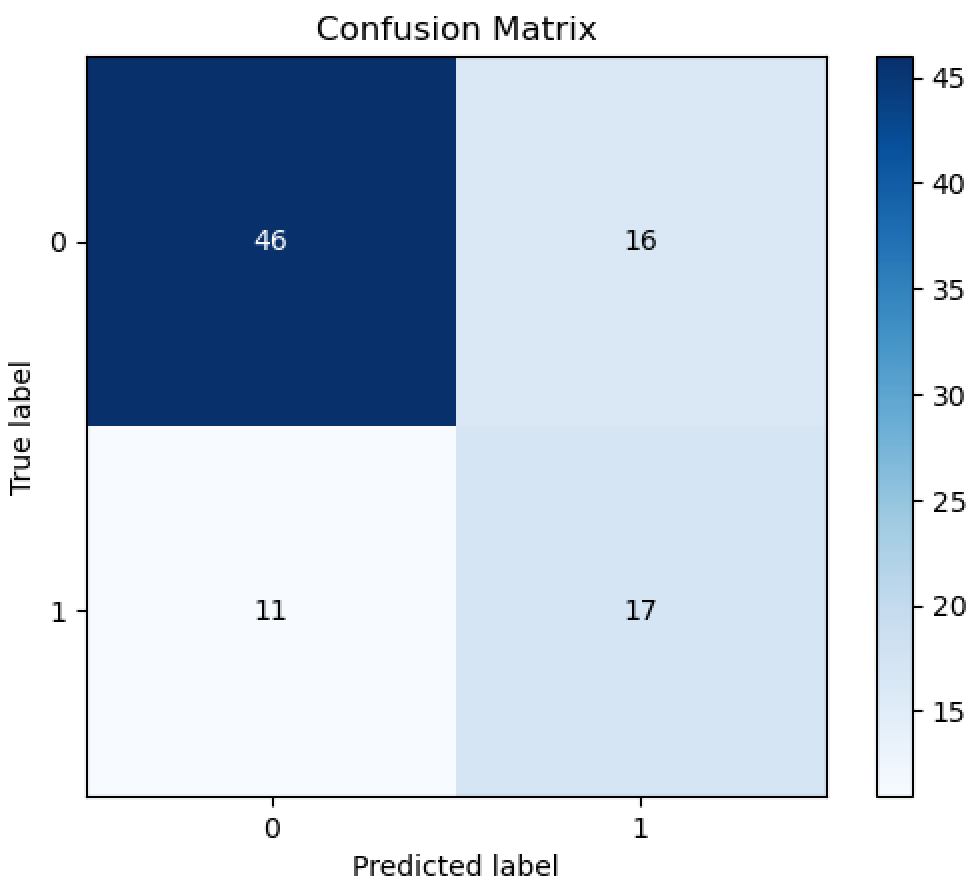
XGBoost Classifier

Similar to our other models, I utilized GridSearchCV to find the best hyperparameters for our model. The best parameters were deemed to be: 'max_depth': 12 and 'n_estimators': 100. Our tree resembles:

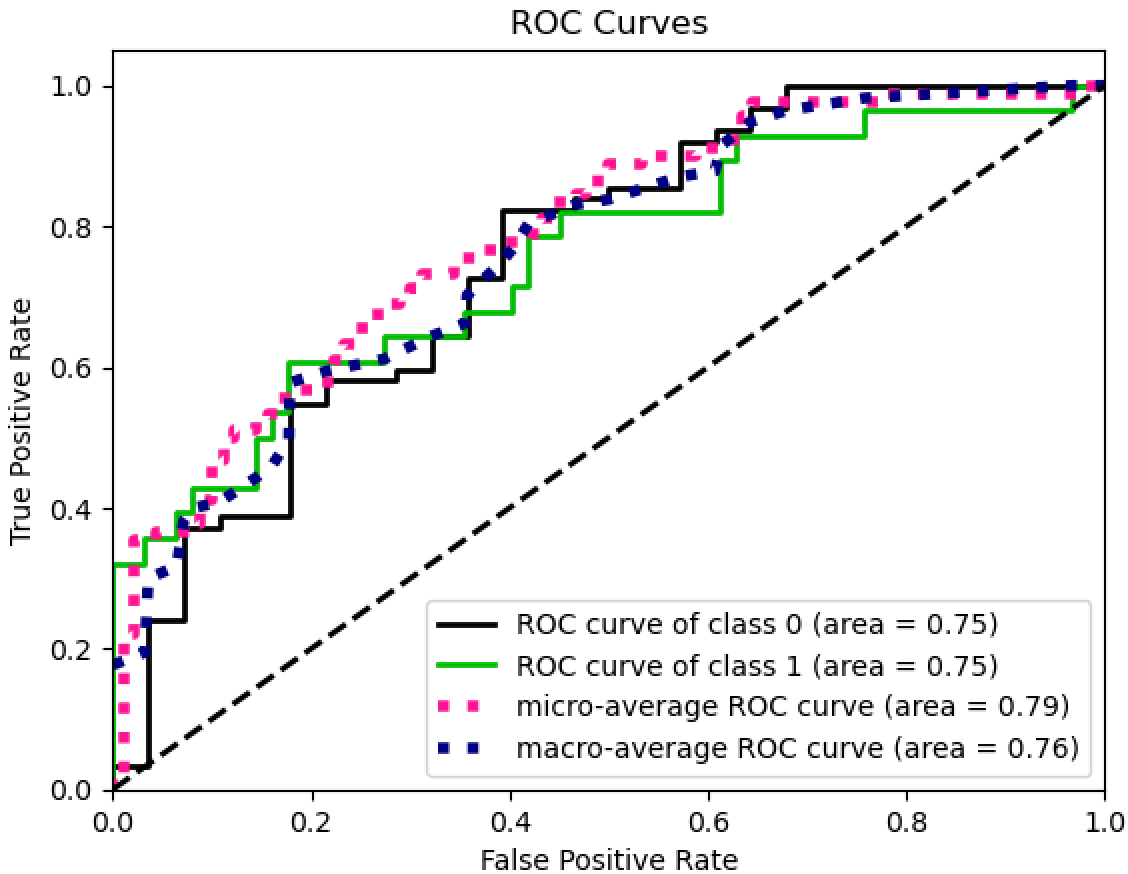


```
Classification Report:
              precision    recall  f1-score   support

           0       0.81      0.74      0.77        62
           1       0.52      0.61      0.56        28

    accuracy                           0.70        90
   macro avg       0.66      0.67      0.67        90
weighted avg       0.72      0.70      0.71        90
```

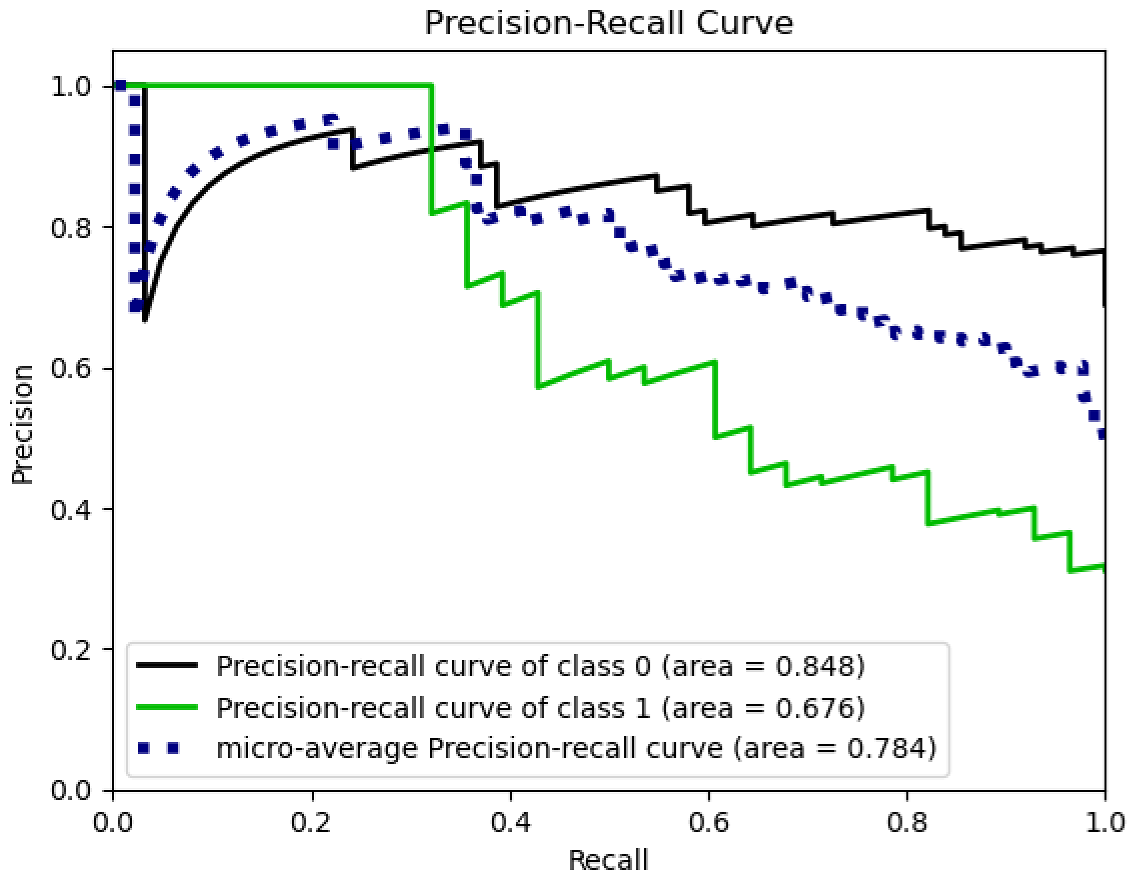
Surprisingly, our XGBoost model is worse than our Random Forest Classifier in terms of overall accuracy score (0.7) but has a higher recall score for class 1 predictions. However, the high recall score for class 0 and precision score for class 1 is not transferred. This means that the accuracy of our positive predictions is lower for class 1 but completeness of positive predictions is higher. The f1-scores for this model is still worse than that of the Random Forest Classifier indicating that the model overall is worse than the Random Forest Classifier.



As suggested by the precision and recall scores for our XGBoost model, the confusion matrix depicts an improvement in class 1 predictions but is also worse at class 0 predictions. Compared to our AdaBoost model, this model performs much worse.

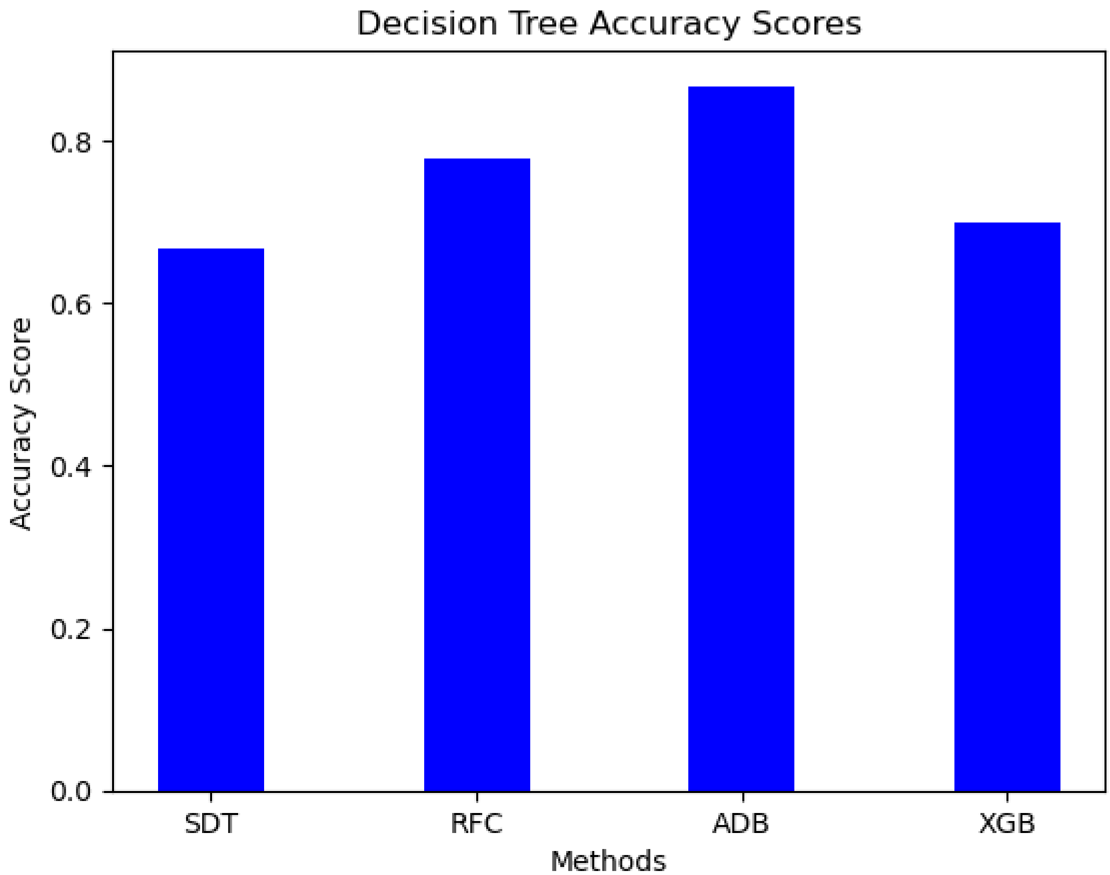


Similarly, our ROC Curves do show a trend toward the “right angle” shape but are still worse than our AdaBoost model. Our ROC curve of class 1 is still the worst performing curve similar to all of our other models at all thresholds.



The Precision-Recall Curve also reflects the fact that our model is worse than the AdaBoost model as we can see that the Precision-recall curve of class 1 performs much worse. Since the area under the curve is much lower, this indicates that we have a higher false positive rate and higher false negative rate for class 1 cases.

Method Comparison



Our best performing model was the AdaBoost Classifier. This is likely due to the fact that it is best used with weak learners that are just above or below random guessing, as seen in our simple decision tree and random forest classifier methods. As such, AdaBoost is the most suitable method for this data. XGBoost performed weaker than AdaBoost likely due to the size of our dataset. XGBoost is prone to overfitting on smaller datasets (299 variables) and thus weaker in performance. The simple decision tree method performed worse than Random Forest Classifier to our expectation as Decision Trees are also prone to overfitting whereas Random Forests can mitigate this effect; furthermore, decision trees have more limitations in general compared to random forest.

Reference: Heart failure clinical records. (2020). UCI Machine Learning Repository. https://doi.org/10.24432/C5Z89R.