Hyperparameter Optimization

Selections on Vinho Verde Red Wine

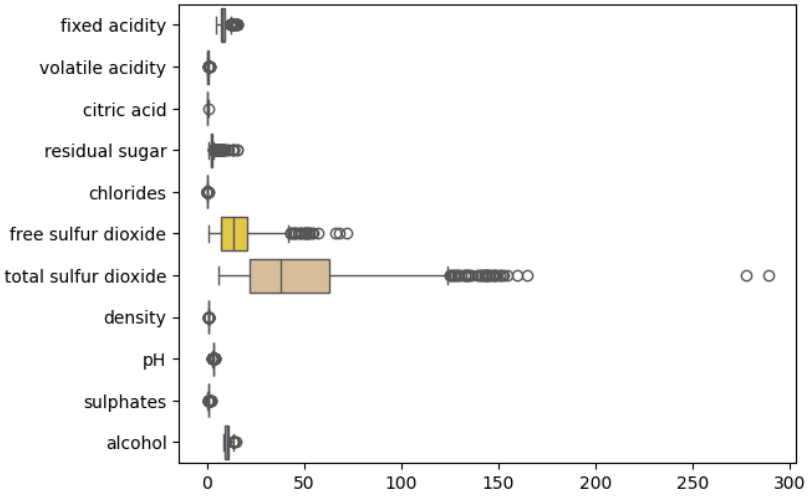
Maxie Machado
University of Wyoming

1 Introduction

The exercise being performed is Hyperparameter Optimization on the red wine quality dataset. As it's in the name we will have a collection of red wines from vinho verde which are rooted from Northern Portugal, and will be looking at different factors which will affect the quality of the red wine positively or negatively. Different machine learning algorithms and hyperparameter tuning will be done on the dataset, and will have the AUC score on them and be visualized using ROC curves. This dataset will be put through two different ML algorithms including that will be optimized is, decision tree classifier and random forest classifier.

2 Description of Red Wine Dataset

The red wine dataset consists of 1599 observations. Another thing which is important to note is the amount of features there are within the dataset and what they are. Within the red wine dataset there are 11 features, which include the following:



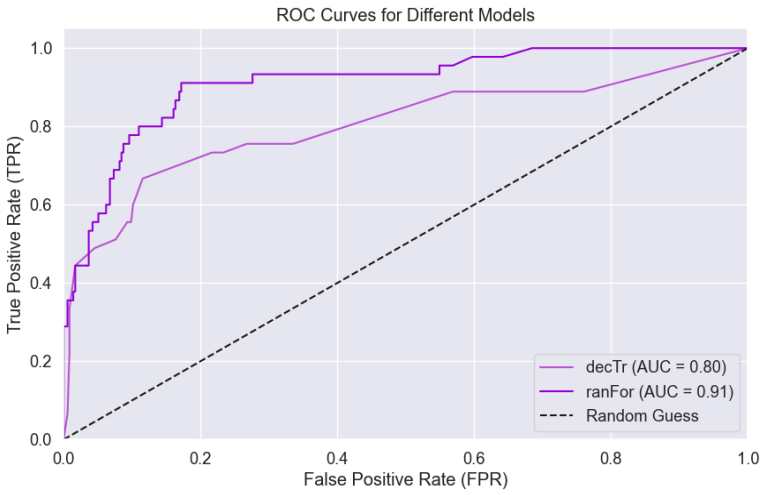
- Fixed Acidity
- Volatile Acidity
- Citric Acid
- Residual Sugar
- Chlorides
- Free Sulfur Dioxide
- Total Sulfur Dioxide
- Density
- pH
- Sulphates
- alcohol

In examining the data type of the red wine it will be shown all of these are float64 or int which mean we will not need to worry about converting objects. This data is what will be used to accomplish the task of hyperparameter optimization.

3 Experimental Setup

Now to be able to address the task of hyperparameter optimization upon the provided dataset, vinho verde red wine quality, the programming language python will be used. Along with libraries that are important to accomplish the task at hand. These libraries include, NumPy, pandas, scikit-learn, seaborn, and Matplotlib. The data we will be using will be split into the training and testing data. Although before splitting and training the data , list comprehension will

be used to create the binarization of the target variable, quality. How this will be done is by taking the quality(s) that is greater than or less than seven and grouping them into binary number one. Meaning any quality(s) that is less than seven will be grouped into the binary number 0. Once this is completed the dataset will be split and trained. The way it was done is by creating a new dataframe called X that contains all observations except for quality which is the target variable. Then creating a series called y that will hold the target variable 'quality'. Then X_train and y_train will be created to train the machine learning models and X_test and y_test will be created to evaluate how well the model performs on unseen data. Once this is completed the method train_test_split will be used on the data having 25% of the data reserved specifically for testing and the rest will be used for training. Two machine learning algorithms were chosen for the model training and performing the hyperparameter optimization. The algorithms selected were the Decision Tree Classifier and Random Forest Classifier. For this a search space was created for hyperparameters, this included:

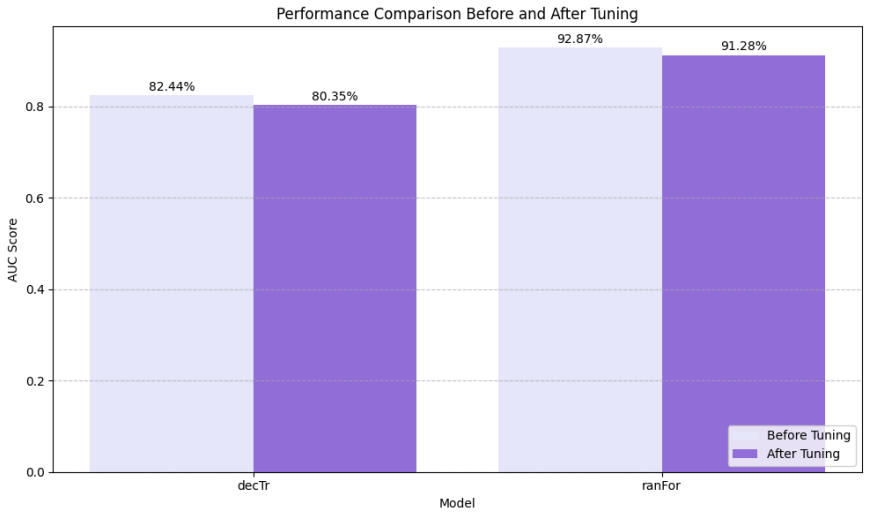


- Minimum Depth
- Minimum Features
- MinimumSamples Split
- Minimum Samples Leaf

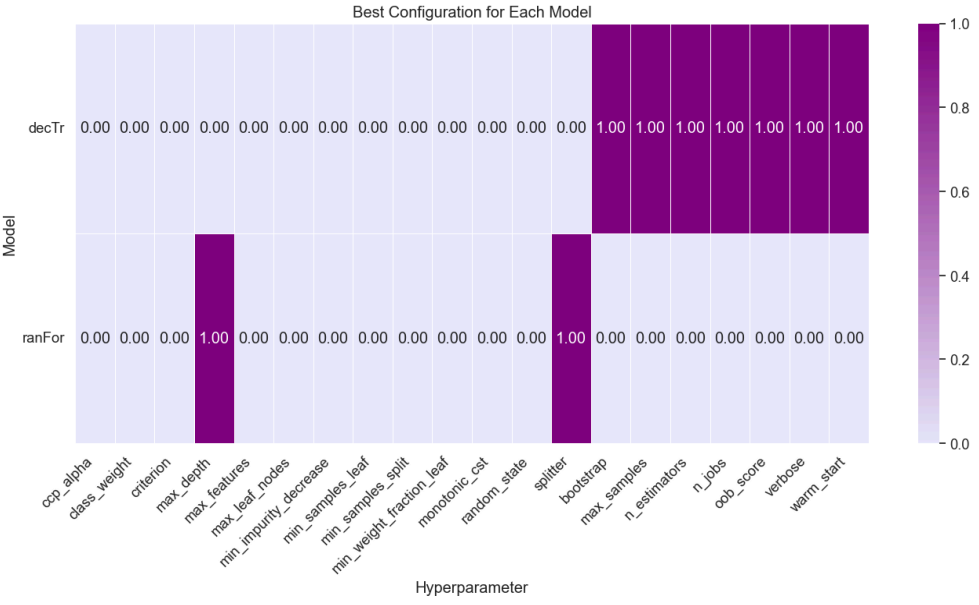
On top of this RandomSearchCV has been utilized to perform hyperparameter optimization. This was chosen due to its efficiency in the exploration of the hyperparameter space. Now, to be able to visualize this it was chosen to incorporate ROC curves. Plotting the ROC curves will utilize the Area Under the Curve (AUC) and visuale the performance of the hyperparameter tuning and each model [fig. 2]. Then a heatmap will be used in visualizing the best configuration of hyperparameters for each model.

4 Results

Once hyperparameter optimization has taken place on the red wine quality dataset we can now analyze the results. Like stated previously the machine learning algorithms that will be focused on are the Decision Tree Classifier (decTr) and Random Forest Classifier (ranFor). To analyze the effects of hyperparameter tuning a simple bar plot will be used [fig. 3]. As seen on the barplot provided, prior to hyperparameter tuning, the Decision



Tree Classifier (decTr) demonstrates an accuracy of 83.97%. Although after hyperparameter tuning it's shown that the accuracy has decreased slightly to 80.35%. Similarly, prior to the hyperparameter tuning for the Random Tree Classifier (ranFor) demonstrates an accuracy of 92.69%. Although after hyperparameter tuning it also shows a slight decrease of accuracy change to 91.11%. With this information, despite these decreases within the accuracy, these models that have been tuned still maintain competitive performance.



Once this has been visualized, we will now look at what configurations are best for each model and will visualize this using a simple heatmap [fig. 4]. These configurations will be identified by the RandomSearchCV, which will explore the hyperparameter space. First the best configuration for the Decision Tree Classifier will be discussed. For the Decision Tree Classifier, some of the factors to create the best configuration were determined to contain a maximum depth of 5, minimum samples leaf of 2, and minimum samples split of 2. On the other hand some of the factors for the best configuration for the Random Forest Classifier are determined to contain 50 estimators, maximum depth of 15, and a minimum samples leaf of 2.