Analysis of Wine Quality Using Machine Learning

Morris Simons
M.Sc Ai and Machine Learning
Blekinge Tekniska Högskola
Karlskrona, Blekinge, Sweden
Mosi21@student.bth.se

***Abstract*—This document presents a comprehensive analysis of the Wine Quality dataset using various machine learning techniques. The focus is on data preprocessing, algorithm selection, model training, and evaluation.**

***Index Terms*—Wine Quality, Machine Learning, Data Preprocessing, Algorithm Selection, Model Evaluation**

I. Introduction

This project focuses on the Wine Quality dataset, a comprehensive collection of chemical wine properties, to explore the feasibility and effectiveness of machine learning techniques in predicting wine quality.

The project is structured to first undergo a thorough data preprocessing phase, where we cleanse and prepare the dataset for analysis. Following this, we then try 2 separate algorithms, where the different machine learning models are evaluated for their suitability and performance. We also explore the concepts of imbalanced data and different scaling techniques.

II. Data and Preprocessing

A. Data Overview

We start by doing some plotting of the algorithms. We also explore the distribution of our main value quality. When using '.describe()' on the dataset we find most notably that our target attribute quality ranges from 3-8 giving us 5 classes where the 2 biggest represent over 70% of the sample size. This could be a problem when training the model, as it may be biased towards the majority class. This can be solved by either oversampling the minority classes or undersampling the majority classes.

To do some more test we check for null values to look for missing data using: 'df.isnull().sum()'. We also have 5 different classes and we want to look for anomaly's in each class. To do this in the best way we can concatenate the the describe for each class to get an better view of our data.

B. Preprocessing Steps

We utilized different scaling techniques to normalize and standardize our dataset, ensuring that our machine learning algorithms perform optimally. The following steps outline our approach to preprocessing the Wine Quality dataset:

1) Standard Scaling: For features such as pH and density, which appeared to have a relatively symmetric distribution and were close to a normal distribution, we applied Standard Scaling (Z-score normalization). This scaling technique is beneficial for features that follow a Gaussian distribution. The transformation adjusts the data to have a mean of zero and a standard deviation of one.

2) MinMax Scaling: MinMax Scaling was applied to features like alcohol, fixed acidity, citric acid, and sulphates. These features displayed right-skewed distributions. MinMax Scaling transforms the data into a specific range (0 to 1 in our case), which is particularly useful for features that do not follow a normal distribution.

3) Robust Scaling: Given the presence of outliers in features such as residual sugar, chlorides, free sulfur dioxide, and total sulfur dioxide, we initially considered Robust Scaling. Log transformation was also experimented with and proved to increase preformance. But due to it not being in the instruction it was later removed to better fit the instructions of this assignement.

III. Algorithm Selection and Motivation

A. Random Forrest

The first reason is random Forrest explain-ability. We can use Feature Importance to analyze what kind of role each attribute play for role. Good Performance with Minimal Tuning: Random Forest often yields commendable results even without extensive hyperparameter tuning, making it a good choice for initial exploratory analysis in machine learning projects.

B. Gradient Boosting Classifier

When selecting Gradient Boosting Classifier(GBC) initially it was just random. But i have discovered that GBC is good at handling Imbalanced datasets like ours. But the model is also known for high accuracy and performance in a variety of tasks and are considered one of the best out-of-the-box classifiers.

IV. Training and Testing

First, we split our dataset into training and testing sets to avoid data leakage during scaling, adhering to the standard 80/20 split. This ensures our model is tested on untrained data, providing a realistic assessment of its performance.

A. Model Training

During the training phase, we employed two classification algorithms: Random Forest Classifier and Gradient Boosting Classifier. We performed Repeated k-Fold Cross-Validation to validate our models thoroughly. This approach repeats k-Fold Cross-Validation multiple times, providing a more robust estimation of our model's performance.

1) Random Forest Classifier: The Random Forest Classifier was evaluated using 3-fold cross-validation, repeated 10 times. The mean accuracy and standard deviation were calculated to assess the model's consistency and reliability.

2) Gradient Boosting Classifier: Similarly, the Gradient Boosting Classifier underwent the same cross-validation process, allowing us to directly compare its performance with the Random Forest Classifier.

Based on the cross-validation results, we determined the best-performing classifier and proceeded to train it on the entire training set. We use an random state seed to keep the randomness in each model the same creating an easier environment to measure impact.

B. Model Evaluation

We evaluated the model's performance on the testing set, focusing on our key metric accuracy. Additionally, we generated a classification report, providing detailed insights into the model's precision, recall, and F1-score.

To address the issue of imbalanced classes, we utilized SMOTE (Synthetic Minority Over-sampling Technique) to balance our training dataset. We then reassessed our models using the balanced dataset to compare their performance.

1) Feature Importance: Understanding the significance of different features in prediction was an essential part of our analysis. We extracted and reported the feature importance's from the best-performing classifier, offering insights into which features were most influential in predicting wine quality in our beast model random Forrest.

V. Results and Discussion

The result from the model was approx 65%. This is good considering that the random guess is 20% based on the fact that we have 5 class options. From Table 1 we can see what variables play what role. From table 2 and 3 we can compare the impact of smote. From the data that we gathered we can observe that accuracy went up with 1% but if we look at the macro and weighted avg we can see an significant difference this is due to our model being good in certain classes an preforming worse in other. The macro average measures the accuracy for all classes. We can see that using smote will increase the macro avg somewhat and increase the results in precision and F1 score on an weighted avg.

Feature	**Importance**
Volatile Acidity	0.143358
Sulphates	0.126009
Alcohol	0.119113
Chlorides	0.103170
Total Sulfur Dioxide	0.086624
Density	0.076502
Free Sulfur Dioxide	0.073604
pH	0.073474
Residual Sugar	0.070307
Citric Acid	0.064244
Fixed Acidity	0.063596

TABLE I
FEATURE IMPORTANCES

Category	**Precision**	**Recall**	**F1-Score**	**Support**
3	0.00	0.00	0.00	1
4	0.17	0.30	0.21	10
5	0.74	0.74	0.74	130
6	0.69	0.61	0.65	132
7	0.56	0.64	0.60	42
8	0.00	0.00	0.00	5
Avarage				
macro avg	0.36	0.38	0.37	320
weighted avg	0.66	0.65	0.65	320
Accuracy(0.65)				320

TABLE II
MODEL PERFORMANCE WITH SMOTE(BALANCED)

Grade	**Precision**	**Recall**	**F1-Score**	**Support**
3	0.00	0.00	0.00	1
4	0.00	0.00	0.00	10
5	0.71	0.75	0.73	130
6	0.62	0.69	0.65	132
7	0.63	0.52	0.57	42
8	0.00	0.00	0.00	5
Avarage				
macro avg	0.33	0.33	0.33	320
weighted avg	0.63	0.66	0.64	320
Accuracy(0.66)				320

TABLE III
MODEL PERFORMANCE WITHOUT SMOTE(IMBALANCED)

VI. Conclusion

Acknowledgment

Acknowledge any assistance or contributions from individuals or institutions.