A Systematic Review and Experimental Evaluation of Feature Selection Approaches in Wine Quality Dataset

KAI-XIANG, CHANG
113423045
kaixiang@g.ncu.edu.tw

ZEI-WEI, XIE
113423036
youto201266@gmail.com

KAI-SONG, KUO
113423027
k113423027@g.ncu.edu.tw

ABSTRACT

Predicting wine quality based on physicochemical attributes is a complex yet crucial task in the wine industry. This study conducts a comprehensive review and experimental evaluation of feature selection techniques applied to wine quality prediction. Specifically, we compare the effectiveness of filter, wrapper, and embedded feature selection methods in optimizing model performance. The Random Forest algorithm is employed as the predictive model due to its robustness and interpretability. To assess the impact of feature selection on classification performance, we utilize k-fold cross-validation and evaluate the model using F1-score and accuracy metrics. The results demonstrate that appropriate feature selection significantly enhances predictive accuracy while reducing computational complexity. This study provides valuable insights into the role of feature selection in improving machine learning models for wine quality assessment, thereby supporting enhanced quality control and production efficiency in the wine industry.

Keywords

Wine Quality Prediction; Feature Selection; Filter Method; Wrapper Method; Embedded Method; K-Fold Cross-Validation; F1-Score; Accuracy; Random Forest.

1. INTRODUCTION

1.1 The Meaning of Feature Selection in Classification?

Feature selection, within the realm of classification, denotes the systematic process of identifying and preserving a subset of pertinent features from an original dataset to support the construction of a classification model. This approach is instrumental in mitigating the peril of overfitting, enhancing the model's ability to generalize to unseen data, reducing the computational burden of training, and, in certain contexts, improving the interpretability of the resulting model [1]. In classification tasks, where the primary aim is to assign categorical labels to data instances, feature selection ensures that those capable of effectively distinguishing between class boundaries are retained, thereby optimizing both predictive precision and operational efficiency. To achieve these ends, a variety of established methodologies are deployed, including filter methods, wrapper techniques, and embedded strategies, each offering distinct advantages in addressing the multifaceted objectives of classification. The importance of this process is further accentuated by its capacity to tackle the complexities posed by high-dimensional datasets, a prevalent challenge in modern machine learning applications [2].

1.2 Why is It a Problem Worth Investigating?

The assessment and prediction of wine quality constitute a multifaceted problem that warrants rigorous investigation due to its profound implications for both the viticulture industry and the advancement of machine learning methodologies. Wine quality, influenced by an intricate interplay of physicochemical properties, environmental factors, and vinification processes, presents a challenging domain for predictive modeling, necessitating the identification of salient features that underpin quality differentiation. This complexity is compounded by the high-dimensional nature of datasets typically encountered in wine quality analysis, where numerous variables must be scrutinized to distill those most predictive of sensory and commercial outcomes [3]. The investigation of this problem is thus meritorious, as it addresses a practical need within the wine industry to enhance production consistency and market competitiveness while simultaneously serving as a robust testbed for refining feature selection and machine learning techniques.

Moreover, the pursuit of accurate wine quality prediction exemplifies a broader scientific endeavor to bridge data-driven insights with domain-specific expertise, offering opportunities to evaluate the efficacy of advanced methodologies such as embedded feature selection and hyperparameter optimization. The economic stakes are considerable, given that quality directly influences consumer preference and pricing strategies, rendering precise predictive models invaluable for stakeholders. Additionally, the inherent noise and variability in wine datasets pose significant challenges to model robustness, making this an exemplary case for exploring the adversarial resilience of feature selection approaches, such as those employing LASSO [4]. Consequently, investigating wine quality prediction not only advances industrial applications but also contributes to the theoretical maturation of machine learning by tackling issues of scalability, interpretability, and generalization in real-world contexts [3].

1.3 Define the Goals of Survey

This study presents a comparative analysis of three feature selection techniques spotlighting the critical role of feature selection in boosting the accuracy and interpretability of predictive models.

The research involves:

1. Choose a suitable dataset: Select a detailed dataset that captures the essence of wine quality through variables.
2. Applying Feature Selection Techniques: Identifying and implementing methods.

The Survey Offers:

1. Exploratory Data Analysis (EDA): To characterize key features, revealing patterns, correlations, and outliers that shape wine quality.
2. Performance Framework: A structure for categorizing wine quality based on the selected features, providing a clear and intuitive system to differentiate quality levels.
3. Predictive Model: A Random Forest classifier, fine-tuned through feature selection, to forecast wine quality with precision.

2. RELATED WORK

2.1 Research Criteria

The evaluation of feature selection methodologies in classification tasks hinges upon several cardinal criteria, notably encompassing predictive performance indicators, computational efficiency, the stability of selected features, and the technique's proficiency in navigating high-dimensional data environments. These evaluative dimensions are indispensable for gauging the feasibility and robustness of feature selection approaches, particularly in complex scenarios characterized by limited sample sizes or expansive feature spaces. Rigorous assessments often leverage multi-criteria decision-making frameworks to afford a comprehensive appraisal of the techniques under scrutiny, ensuring that both theoretical efficacy and practical applicability are adequately addressed. Such criteria are further validated by their alignment with real-world challenges, as evidenced in domains requiring scalable and reliable feature selection solutions [5].

2.2 Why Wine Quality Prediction is Important?

Wine quality prediction is of paramount importance for the wine industry due to its multifaceted benefits in production, consumer protection, and research. Quality certification plays a central role in the industry, prompting companies to invest substantial resources in developing and refining advanced predictive methods [3], [6], [7]. This not only ensures that the wines available on the market consistently meet high standards but also prevents illegal adulteration that can jeopardize consumer health [3], [8]. By leveraging machine learning techniques, the process of estimating wine quality becomes more efficient and accurate, enabling producers to maintain consistent excellence while offering consumers reliable information to support their purchasing decisions [7], [9]. Furthermore, these objective and scalable predictive models serve as a robust alternative to traditional sensory evaluations—often subjective and prone to inconsistencies—thus enhancing the overall reliability of quality assessments [10]. The application of such technologies not only optimizes production processes by uncovering the critical factors that affect wine quality but also significantly reduces costs compared to time-consuming and expensive conventional chemical analyses [10], [11]. In addition, the development of chemical composition-based predictive tools provides valuable insights for winemakers and quality control specialists, while the integration of both sensory and chemical data offers a more comprehensive evaluation of wine quality [6].

2.3 What is Feature Selection?

Feature selection is a crucial task in machine learning that involves retaining a subset of the original features by eliminating irrelevant or redundant variables, thereby ensuring that only the most informative attributes contribute to prediction accuracy in both classification and regression problems [12]. Based on the availability of label information, feature selection techniques can be broadly categorized into supervised, semi-supervised, and unsupervised methods, and can also be classified into filter, wrapper, and embedded methods according to their implementation approach [13]. This process plays a vital role in enhancing model interpretability, reducing computational complexity, and mitigating overfitting by filtering out noisy and redundant data, ultimately leading to more efficient and accurate predictions.

2.3.1 Filter method for Feature Selection

In feature selection, filter methods are widely used techniques that evaluate and rank features based on statistical measures, selecting those with the highest scores. Unlike other approaches, filter methods perform feature selection independently of the machine learning model's training process, assessing each feature's relevance without depending on a specific algorithm. This independence makes them computationally efficient, particularly when dealing with datasets containing many features. Filter methods can be integrated with any machine learning model and significantly reduce the runtime of machine learning algorithms. They achieve this by calculating an independent score for each feature, ranking them accordingly, and then selecting either the top (m) features with the highest scores or all features exceeding a predefined threshold. Additionally, for many filter methods, the computation of these scores can be executed in parallel, further enhancing their efficiency [14].

2.3.1.1 Analysis of Variance (ANOVA)

Analysis of Variance (ANOVA), implemented through anova.test, is a univariate statistical testing filter method used to evaluate the importance of features. This method performs variance analysis on each feature to determine whether the categorical variable can be explained by that feature, using the F-statistic as the scoring criterion. A higher F-statistic indicates a greater difference in means of the feature across different categories. Its scoring formula is:

$$\mathbf{F} = \frac{\sum_{\mathbf{i}=1}^{\mathbf{l}} \mathbf{n_i} \left(\overline{\mathbf{x}_{\mathbf{i}\bullet}^{(\mathbf{k})}} - \overline{\mathbf{x}_{*\bullet}^{(\mathbf{k})}}\right)^2 / (\mathbf{l}-\mathbf{1})}{\sum_{\mathbf{i}=1}^{\mathbf{l}} \sum_{\mathbf{j}=1}^{\mathbf{n_i}} \left(\mathbf{x}_{\mathbf{ij}}^{(\mathbf{k})} - \overline{\mathbf{x}_{\mathbf{i}\bullet}^{(\mathbf{k})}}\right)^2 / (\mathbf{N}-\mathbf{l})}$$

Where $\mathbf{l}$ is the number of categories, $\mathbf{n_i}$ is the number of instances in category $\mathbf{i}$, $\mathbf{x}_{\mathbf{ij}}^{(\mathbf{k})}$ is the observed value of feature $\mathbf{X_k}$ in category $\mathbf{i}$, $\overline{\mathbf{x}_{\mathbf{i}\bullet}^{(\mathbf{k})}}$ is the mean of $\mathbf{X_k}$ in category $\mathbf{i}$, $\overline{\mathbf{x}_{*\bullet}^{(\mathbf{k})}}$ is the overall mean of $\mathbf{X_k}$ across all instances, and $\mathbf{N}$ is the total number of instances.

2.3.1.2 Kruskal-Wallis Test

The Kruskal-Wallis test, implemented via kruskal.test, is a univariate statistical testing filter method and a nonparametric equivalent to ANOVA. It applies a rank-sum test to each feature to assess the differences in values across categories, using the test statistic as the scoring criterion, where a higher value indicates greater differences. Its scoring formula is:

$$\mathbf{H} = \frac{\mathbf{12}}{\mathbf{N(N+1)}} \sum_{\mathbf{i}=\mathbf{1}}^{\mathbf{l}} \frac{\left(\mathbf{R_i^{(k)}}\right)^2}{\mathbf{n_i}} - \mathbf{3(N+1)}$$

where $\mathbf{N}$ is the total number of instances, $\mathbf{n_i}$ is the number of instances in category $\mathbf{i}$, $\mathbf{R_i^{(k)}}$ is the sum of ranks of the observed values of feature $\mathbf{X_k}$ in category $\mathbf{i}$, and $\mathbf{l}$ is the number of categories. This method does not assume a normal distribution of data, making it particularly suitable for nonparametric data analysis.

2.3.1.3 Mutual Information (MI)

Mutual Information (MI) is an information theory-based filter method used to measure the mutual dependence between a feature and a categorical variable, interpreted as the reduction in uncertainty about one variable given knowledge of the other. In feature selection, a higher mutual information value indicates a stronger correlation between the feature and the categorical variable. Its calculation formula is:

$$\mathbf{I(X;Y) = H(Y) - H(Y|X)}$$

where $\mathbf{H}(\mathbf{Y})$ is the entropy of $\mathbf{Y}$, and $\mathbf{H}(\mathbf{Y}|\mathbf{X})$ is the conditional entropy of $\mathbf{Y}$ given $\mathbf{X}$.

2.3.2 Wrapper Methods for Feature Selection

Wrapper feature selection methods iteratively evaluate feature subsets by training a machine learning model and measuring performance—often via metrics like accuracy, error rate, or task-specific scores—to determine the optimal combination. These methods excel in tailoring features to a specific model, optimizing its predictive power, yet falter with high-dimensional datasets due to the rapid increase in possible subsets, rendering computation resource-intensive. Common implementations include Recursive Feature Elimination (RFE), which removes less impactful features step-by-step, Forward Selection, which incrementally builds the subset, and Backward Elimination, which prunes features from the full set. [15], [16]. Figure 1 illustrates the iterative subset evaluation cycle of a wrapper-based feature selection model.

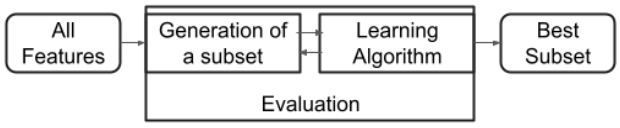

Figure 1. Wrapper method.

2.3.2.1 Genetic Algorithm (GA)

The Genetic Algorithm (GA), as a wrapper feature selection method, employs an evolutionary approach to iteratively optimize feature subsets, balancing exploration of new feature subsets with exploitation of high-performing ones based on a machine learning model's predictive performance. It begins with a population of randomly generated feature subsets, evaluating their utility using a fitness criterion to guide selection [17].

The proceeds through the following stages:

1. Initialization establishes the starting population.
2. Crossover recombines feature subsets, while mutation introduces variation by modifying them.
3. Fitness evaluation quantifies each subset's predictive utility.
4. Selection propagates the best-performing subsets to the next generation.
5. Termination occurs upon performance convergence or reaching a predefined iteration limit, yielding the optimal feature subset.

By emulating biological evolution, GA iteratively refines the feature space across generations, enhancing model performance through a dynamic interplay of exploration and exploitation.

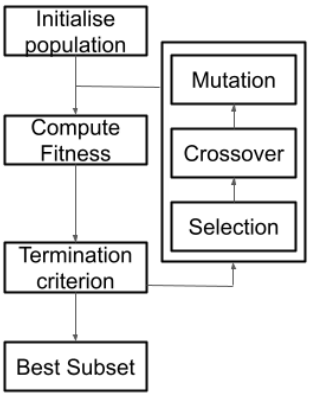

Figure 2. GA iteratively.

2.3.2.2 Recursive Feature Elimination (RFE)

RFE is a meticulous, stepwise procedure designed to prune less consequential attributes from a dataset, thereby enhancing the efficacy of predictive models. The technique commences with the full complement of features and leverages a specified estimator to assess each attribute's importance based on the model's derived weights, coefficients, or feature rankings. In each iteration, the feature with the lowest impact, as determined by the estimator's scoring, is removed, and the model is retrained on the reduced feature set. This cycle persists until a predetermined number of features remains or an optimal performance metric is achieved [15]. Mathematically, RFE can be formalized as follows: Let $\boldsymbol{F} = \{\boldsymbol{f_1}, \boldsymbol{f_2}, \dots, \boldsymbol{f_n}\}$ represent the initial feature set, and $\mathbf{M}$ denote the estimator. At iteration $\boldsymbol{k}$:

1. Compute the importance score $\boldsymbol{W_i}$ for each feature $\boldsymbol{f_i} \in \boldsymbol{F}$ using M.
2. Identify $\boldsymbol{f_j} = \mathbf{arg}\ \boldsymbol{min_{f_i}}\ |\boldsymbol{w_i}|$, the feature with the smallest absolute importance.
3. Update $\boldsymbol{F} \leftarrow \boldsymbol{F} \backslash \{\boldsymbol{f_j}\}$.
4. Retrain $\boldsymbol{M}$ on the updated $\boldsymbol{F}$ and repeat until $|\boldsymbol{F}| = \boldsymbol{m}$ (target feature count) or a performance threshold is met.

2.3.3 Embedded Methods for Feature Selection

An embedded method integrates feature selection directly into the classifier algorithm. During training, the classifier fine-tunes its internal parameters by assigning appropriate weights to each feature to maximize classification accuracy. As a result, identifying the optimal set of features and building the model occur simultaneously in one step [18]. Embedded methods strike a balance between filter and wrapper approaches by blending their strengths [19]. Specifically, like filter methods, they are computationally less demanding than wrapper methods, yet they incorporate the classifier's bias during feature selection, a characteristic of wrapper methods that typically leads to improved performance.

Some examples of embedded methods include decision tree-based algorithms (e.g., decision tree, random forest, gradient boosting), and feature selection using regularization models (e.g., LASSO or Elastic Net) [18]. The following will provide detailed explanations

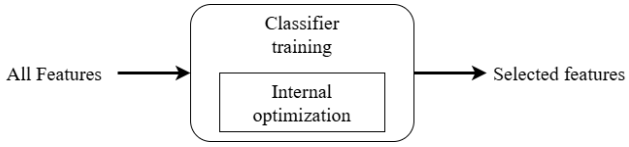

Figure 3. Embedded method.

of the principles behind the Random Forest, LightGBM, LASSO, and Elastic Net in the embedded method.

2.3.3.1 Random Forest (RF)

Random Forest (RF) is a powerful ensemble learning method that employs built-in feature importance measures to rank and prioritize features, allowing it to focus on the most relevant predictors. This helps improve model performance by reducing overfitting and effectively handling noisy data, making it well-suited for complex and high-dimensional datasets [20]. Recently, RF has gained popularity as a feature selection method due to its various advantages, including built-in error estimation, correlation analysis, and feature importance evaluation. It offers two primary approaches for computing feature importance scores: Mean Decrease Accuracy (MDA) and Mean Decrease Impurity (MDI). MDA evaluates feature importance by measuring the reduction in

prediction accuracy when a feature is removed, while MDI assesses a feature's contribution to the uniformity of nodes and leaves in a decision tree model. In both methods, higher values indicate greater feature importance, making RF an effective tool for identifying key predictors in large datasets [21].

In this study, we adopt the Random Forest MDI method because it provides a computationally efficient approximation to impurity, such as entropy [17].

In general any impurity function φ may be used. An impurity function is defined as a function over any K-tuple of non-negative numbers $(p_1, ..., p_K)$ satisfying $\sum_j p_j = 1$, with the following properties

1. φ achieves maximum only for the uniform distribution.
2. φ achieves minimum only at the points where all p_j = 1, j = 1, .., K.
3. φis symmetric with respect to $p_1, .., p_K$.

Furthermore impurity measures i(·) are related to an impurity function φ by the following relation

$$\boldsymbol{i(t) = \varphi(p(1|t),..,p(K|t))}$$

where p(j|t) is the estimated probability of class j in node t.

Its decrease Δi that results from splitting and sending the samples to two sub-nodes t_l and t_r (with respective sample fractions $p_l = \frac{N_{t_l}}{N_l}$ and $p_r = \frac{N_{t_r}}{N_r}$), the decrease in impurity is defined as

$$\boldsymbol{\Delta i(s,t) = i(t) - p_l i(t_l) - p_r i(t_r)}$$

The resulting children nodes are obtained when the data is partitioned by the parent node at s = (X_m < c). Lastly the MDI measure is defined by averaging over all trees T and all nodes *t*

$$\boldsymbol{\mathbf{VI}(X_m) = \frac{1}{Ntree} \sum_{T} \sum_{t \in T\, v(s_t)=X_m} p(t) \Delta i(s_t, t)}$$

where *p(t)* denotes the proportion $\frac{N_t}{N}$ of samples reaching node *t* and v(s_t) denotes the variable used to split node *t*.

2.3.3.2 LightGBM

LightGBM grows trees using a leaf-wise growth strategy, which helps minimize loss, whereas traditional tree-based models typically grow depth-wise. It incorporates two gradient-based techniques: Gradient-based One-Side Sampling (GOSS) and Exclusive Feature Bundling (EFB). GOSS selects features based on their gradient values, prioritizing those that provide more useful information while discarding those with minimal efficiency gains. This approach is particularly effective in selecting representative features and helps reduce overfitting, especially when training on small datasets. On the other hand, EFB reduces the total number of features by grouping mutually exclusive features, meaning features that never have nonzero values simultaneously. By bundling these exclusive features together, EFB eliminates sparse features and enhances computational efficiency [22].

2.3.3.3 LASSO

LASSO (Least Absolute Shrinkage and Selection Operator) is a regularization technique used for both variable selection and shrinkage in statistical models. In the context of linear regression, it minimizes the sum of squared errors while imposing a constraint on the sum of the absolute values of the model coefficients. This constraint encourages sparsity by penalizing large coefficients, effectively performing feature selection and reducing model complexity. As a result, LASSO is particularly useful for high-dimensional datasets. However, it is sensitive to multicollinearity, as it tends to arbitrarily select one variable among highly correlated predictors while shrinking the others toward zero [4].

The lasso estimate is defined by the solution to the l_1 optimization problem. This optimization problem is equivalent to the parameter estimation that follows

$$\boldsymbol{\hat{\beta}(\lambda) = argmin_{\beta}\left(\frac{\sum_{i=0}^{n}(Y_i - (X\beta)_i)^2}{n} + \lambda \sum_{j=1}^{k} |\beta_j|\right)}$$

$\lambda \geq 0$ is the parameter that controls the strength of the penalty, the larger the value of λ, the greater the amount of shrinkage.

When solving the optimization problem, some coefficients are reduced to zero, meaning that $\hat{\beta}(\lambda) = 0$ for certain values of j, depending on the parameter λ. As a result, the corresponding features with zero coefficients are eliminated from the model [23].

2.3.3.4 Elastic Net

Elastic Net is an extension of the LASSO method, designed to address some of its limitations.

LASSO has restrictions, such as:

- In small-n-large-p dataset the LASSO selects at most n variables before it saturates
- If there are grouped variables (highly correlated between each other) LASSO tends to select one variable from each group ignoring the others

Elastic Net overcomes these issues by combining the strengths of both LASSO and Ridge Regression. Its objective function is given by:

$$\boldsymbol{\hat{\beta}(\lambda) = argmin_{\beta}\left(\frac{\sum_{i=0}^{n}(Y_i - (X\beta)_i)^2}{n} + \lambda_1 \sum_{j=1}^{k} |\beta_j| + \lambda_2 \sum_{j=1}^{k} \beta_j^2\right)}$$

where $\lambda_1 \geq 0$ and $\lambda_2 \geq 0$ are regularization parameters. By incorporating a quadratic penalty term, Elastic Net eliminates LASSO's restriction on the number of selected variables and ensures more stable selection when handling groups of correlated features [23].

2.4 Random Forest Algorithm

Random Forest is an ensemble learning technique that combines multiple weak learners, specifically decision trees, to form a strong learner. These decision trees are typically constructed using the CART (Classification and Regression Trees) algorithm, which aims to accurately model the relationship between the dependent variable (y) and the independent variables (x) [24]. It is applicable to both classification and regression problems [25]. Additionally, it outperformed all other classifiers—such as XGBoost, KNN, and SVM—in predicting wine quality, thereby establishing itself as the best model for this task [26].

Main Steps in Building a Random Forest Classifier:

1. Bootstrap Sampling: Generate N bootstrap samples by randomly sampling the original dataset with replacement. This process, known as bootstrap aggregation (or bagging), ensures that each tree is trained on a slightly different subset of the data. Approximately 37% of the original samples are not selected in each bootstrap sample (out-of-bag samples), which can be used to evaluate model performance [27].
2. Random Feature Subset Selection: At each node of a decision tree, rather than considering all features, randomly select a subset of m features (with m typically less than the total number of features M). A common heuristic for

classification tasks is to use the rounded square root of M [27].

3. Finding the Best Split: Within the selected m features, determine the optimal split that maximizes the node purity [24], [27]. For classification, the Gini Index is a widely used criterion to measure the impurity of a node [28].
4. Growing the Decision Trees: Recursively repeat the feature subset selection and splitting process from the root node until a stopping condition is met. Common stopping conditions include reaching a pure node (a node where all samples belong to the same class) or reaching a predefined maximum depth. Unlike standalone decision trees, trees in a Random Forest are typically not pruned [24].
5. Voting for Final Classification: For a new, unseen sample, each decision tree in the forest makes a classification prediction. The final class is determined by majority voting—i.e., the class that receives the most votes across all trees is chosen as the prediction [24], [27], [28].

2.5 Evaluation Metrics

With the widespread adoption of machine learning, evaluating classification models is crucial to ensure accuracy, reliability, and performance. A thorough assessment helps identify weaknesses and optimize model selection. Various evaluation metrics provide a comprehensive analysis of predictive capabilities. The following sections will discuss key evaluation metrics that serve as standard measures for assessing classification performance.

2.5.1 Confusion Matrix

The confusion matrix, also referred to as an error matrix, is a fundamental tool for evaluating classifier performance by comparing predicted labels with actual ground truth labels. It is structured as a table, where one axis represents the true class labels and the other denotes the predicted class labels. This matrix categorizes predictions into four distinct groups: true positives and true negatives, which correspond to correct classifications, as well as false positives and false negatives, which represent misclassifications. By systematically analyzing these categories, the confusion matrix provides insights into the model's errors, facilitating targeted improvements in its performance.

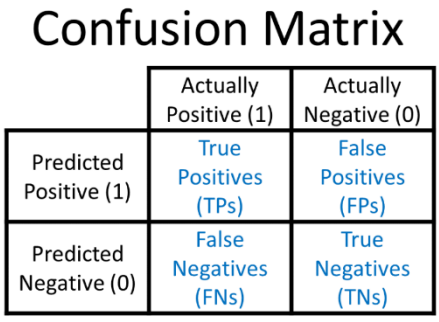

Figure 4. Confusion Matrix.

- True Positive (TP): Means that the model's prediction of a positive outcome is correct, matching the actual positive result.
- True Negative (TN): Means that the model's prediction of a negative outcome is accurate, as the actual result is negative.
- False Positive (FP): Occurs when the model incorrectly predicts a positive outcome while the true result is negative; this mistake is also referred to as a *Type I error*.
- False Negative (FN): Happens when the model mistakenly predicts a negative outcome even though the actual result is positive; this error is known as a *Type II error*.

2.5.2 Accuracy

Accuracy is a fundamental metric for evaluating a model's overall correctness by measuring the proportion of correctly classified instances. However, in imbalanced datasets, high accuracy can be misleading, as a model may predominantly predict the majority class while failing to capture minority class patterns. This highlights the need for additional evaluation metrics to ensure a comprehensive assessment of model performance.

$$\textbf{Accuracy} = \frac{\boldsymbol{TP + TN}}{\boldsymbol{TP + TN + FP + FN}}$$

2.5.3 Precision

Precision measures the proportion of correctly identified positive predictions, reflecting the reliability of the model in classifying positive instances. It is particularly crucial in applications where false positives must be minimized, such as spam detection or fraud prevention.

$$\textbf{Precision} = \frac{\boldsymbol{TP}}{\boldsymbol{TP + FP}}$$

2.5.4 Recall

Recall, on the other hand, quantifies the model's ability to detect actual positive cases by calculating the proportion of true positives among all actual positive instances. High recall is essential in scenarios where missing positive cases carries significant consequences, such as medical diagnoses.

$$\textbf{Recall} = \frac{\boldsymbol{TP}}{\boldsymbol{TP + FN}}$$

2.5.5 F1-Score

The F1-score provides a balanced measure by combining precision and recall into a single metric, making it especially useful for imbalanced datasets. It offers a comprehensive assessment when both false positives and false negatives are critical, though it assumes equal importance between precision and recall, which may vary depending on the specific application.

$$\textbf{F1} - \textbf{Score} = 2 \times \frac{\boldsymbol{Precision \times Recall}}{\boldsymbol{Precision + Recall}}$$

2.5.6 Matthews correlation coefficient (MCC)

In the binary classification setting, Pearson's correlation coefficient simplifies to a specialized form known as the Matthews Correlation Coefficient (MCC). MCC has gained prominence in machine learning applications due to its robustness, particularly in scenarios with imbalanced class distributions. The MCC value ranges from -1 to 1, where 1 indicates perfect predictions, 0 corresponds to random guessing, and -1 signifies complete disagreement between predictions and actual outcomes [29].

$$\textbf{MCC} = \frac{\boldsymbol{TP \times TN - FP \times FN}}{\sqrt{\boldsymbol{(TP + FP)(TP + FN)(TN + FP)(TN + FN)}}}$$

3. METHODOLOGY

3.1 Method Follow

The flowchart of the research methodology illustrates the entire research process, as presented in Figure 5. Each step will be explained in detail below.

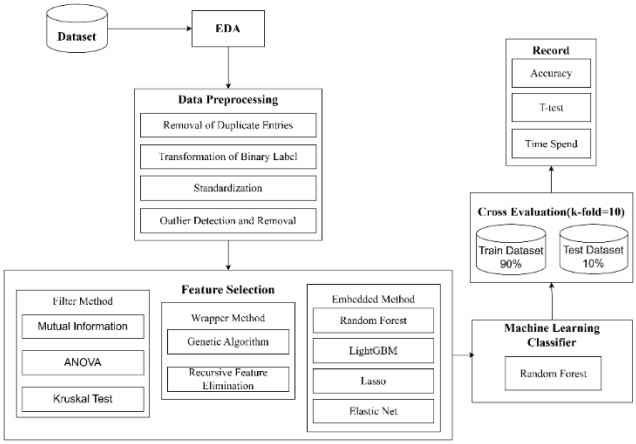

Figure 5. Flowchart of research methodology

3.2 Dataset and Exploratory Data

3.2.1 Dataset

The dataset used in this study is sourced from an open dataset on Kaggle, the "Red Wine Quality Dataset." It comprises data on 12 physicochemical properties of red wine, including fixed acidity, volatile acidity, citric acid, residual sugar, chlorides, free sulfur dioxide, total sulfur dioxide, density, pH, sulphates, alcohol content, and quality, which serves as the target variable. The data collection methods involve laboratory measurements and sensory evaluations by experts or wine tasters, assessing attributes such as color, aroma, and taste.

3.2.2 Exploratory Data Analysis (EDA)

Exploratory Data Analysis (EDA) was performed to examine the characteristics of the wine quality dataset and guide subsequent preprocessing and modeling steps, focusing on understanding feature distributions, identifying outliers, and assessing relationships among physicochemical variables to detect potential collinearity. Box plots were constructed for standardized features—such as fixed acidity, volatile acidity, citric acid, residual sugar, chlorides, free sulfur dioxide, total sulfur dioxide, density, pH, sulphates, and alcohol—illustrating their central tendency, variability, and outliers after transforming them into z-scores with a mean of 0 and a standard deviation of 1. A feature correlation heatmap was also generated, presenting correlation coefficients ranging from -1 to 1 to explore pairwise relationships and their potential impact on model performance. The EDA process, supported by these analyses, provided a foundation for identifying preprocessing needs, such as outlier removal and feature selection, to enhance the dataset for machine learning applications.

3.3 Data Preprocessing

3.3.1 Duplicate Data Removal

To ensure data integrity and avoid bias in model training, duplicate records were removed from the dataset using the function in Python's pandas library. This step eliminates rows with identical values across all features, ensuring each observation is unique. Sangodkar and Bapat [30] emphasize the importance of clean data for effective training of models like Random Forest and SVM, implicitly supporting duplicate removal to prevent skewed results. Similarly, Vazaram et al. [31] highlight preprocessing to handle uncleaned data, noting that redundancy can distort model outcomes. The primary advantage of this step is the prevention of overfitting, as duplicate entries can artificially inflate training accuracy, reducing the model's ability to generalize to unseen data [30], [31]. This ensures a more robust and unbiased predictive model.

3.3.2 Binarization of Target Variable

The target variable quality, originally a numerical score ranging from 1 to 10 in the simulation-generated dataset, was transformed into a binary classification format to simplify the prediction task. Scores from 1 to 5 were labeled as 0 (low quality), and scores from 6 to 10 as 1 (high quality). This approach aligns with Sangodkar and Bapat [30], who converted red wine quality scores (1-10) into binary categories (below 5 as bad, above 5 as good) to enhance classification clarity. Basha et al. [32] also applied label binarization (quality ≥ 7 as 1, else 0), emphasizing its role in reducing task complexity and improving interpretability. Vazaram et al. [31] similarly used binary labeling (good/bad) to streamline analysis. It simplifies the problem from multi-class to binary classification, enhancing model performance metrics like precision and recall, and it supports practical applications requiring a clear quality distinction.

3.3.3 Feature Standardization

In our wine quality prediction model, we applied Standard Scaler to all features except for the target feature quality. This scaling method ensures that feature values are transformed into a distribution with a mean of 0 and a standard deviation of 1. Dahal et al. [4] explicitly adopted standardization in their study on wine quality prediction, emphasizing the significant differences in feature value ranges within the dataset. For example, total sulfur dioxide ranges from 9 to 440 mg/dm³, while chlorides range from 0.009 to 0.35 g/dm³. Without scaling, features with larger values could dominate the model training process, particularly affecting scale-sensitive algorithms such as Support Vector Machines (SVM) and Artificial Neural Networks (ANN).

3.3.4 Outlier Handling

Previous research has underscored the need to address outliers in machine learning to prevent noise from skewing model training. Dahal et al. [33] used boxplots to detect outliers in red wine data, emphasizing their potential to mislead training, while Basha et al. [32] applied the Interquartile Range (IQR) method to stabilize statistical properties and boost accuracy, and Vazaram et al. [31] similarly employed boxplots and percentiles to enhance predictive performance by mitigating extreme value effects. Building on these insights, we implemented outlier removal for all feature columns (excluding quality) in our dataset using the IQR method. For each feature, we calculated the first quartile (Q1) and third quartile (Q3), defined the IQR as Q3 - Q1, and excluded data points falling below Q1 - 1.5 × IQR or above Q3 + 1.5 × IQR. This process yielded a cleaned dataset, aligning with established practices to reduce noise and improve model robustness for wine quality prediction.

3.4 Feature Selection

Feature selection is a vital process in machine learning, focused on pinpointing the most valuable and relevant features from the initial dataset to enhance model development and predictive performance. This step is crucial for tackling issues like the curse of dimensionality and overfitting, which can undermine a model's effectiveness. By carefully choosing appropriate feature selection

techniques, we can construct models that are both efficient and capable of generalizing well to new data

In this study, we applied a blend of feature selection strategies to a dataset comprising numerical input variables and a categorical output target variable. Leveraging insights from the feature selection literature in Sections 2.3.1, 2.3.2, and 2.3.3, we used filter methods such as ANOVA, Kruskal-Wallis Test, and Mutual Information to evaluate each feature's relevance independently. ANOVA and the Kruskal-Wallis Test assessed statistical differences in feature distributions across quality categories, while Mutual Information ranked features by measuring their dependency with the target variable. To refine the feature subset, we employed wrapper methods, including Genetic Algorithms (GA) and Recursive Feature Elimination (RFE), which iteratively trained and evaluated the model to optimize the collective impact of selected features. Additionally, we utilized embedded methods—namely Random Forest, LightGBM (LGBM), Lasso, and Elastic Net—integrating feature selection into the training process through tree-based importance scores and regularization techniques that prioritized the most influential features. This combined approach ensured a thorough assessment of feature contributions, ultimately improving the model's accuracy and robustness in predicting wine quality

3.5 Machine Learning Classifier and Model Prediction Results

Following data preprocessing, we applied a range of feature selection methods to the processed dataset to identify the most influential physicochemical attributes. These methods encompassed filter techniques, wrapper techniques, and embedded techniques. This step reduced dimensionality and enhanced model efficiency by focusing on the most predictive features. The Random Forest (RF) classifier was selected for our wine quality prediction experiment based on its demonstrated performance in a prior study [34], which highlighted RF's superior accuracy and robustness in classifying wine quality using the Red Wine Dataset (RWD). We then implemented the RF classifier with a configuration of 100 decision trees, ensuring reproducibility through a fixed random seed. To robustly assess performance, we conducted 10-fold cross-validation, dividing the dataset into 10 equal folds: 9 folds were used for training, and one-fold was reserved for evaluation in each iteration. This approach provided a comprehensive evaluation of RF's generalization ability, building on its established strengths while tailoring it to our experimental context.

To evaluate the performance of the Random Forest (RF) model, selected as the final classifier due to its superior F1 score and robustness as discussed earlier, a comprehensive assessment was planned using k-fold cross-validation, with one-fold reserved as the test set to ensure unbiased evaluation. The evaluation focuses on comparing the model's performance across different feature selection methods—Filter, Wrapper, and Embedded—and varying numbers of features: Few (3 features), Medium (6 features), Large (9 features), and Origin (all 11 features). The primary metrics for evaluation include F1-Score, Precision, Recall, Accuracy, Training Time, and Feature Selection Time, which collectively provide a balanced assessment of the model's ability to classify wine quality while accounting for class distribution and computational efficiency. Additionally, a t-test (p-value for accuracy) will be conducted to compare the accuracy of each configuration with that of the original dataset (with all 11 features), thereby providing a measure of the statistical significance of performance differences. The selection time of each feature selection method will also be analyzed to assess computational efficiency, ensuring a practical balance between performance and resource demands. Finally, we will identify the optimal combination of the number of features and selection method, and provide a detailed breakdown of performance for different quality classes (low and high) to highlight the model's effectiveness in distinguishing high-quality wines, which aligns with previous studies emphasizing the importance of feature selection in wine quality prediction.

4. EXPERIMENT AND EVALUATION

4.1 Dataset Description

This dataset, obtained from Kaggle and referred to as the "Red Wine Dataset," includes information on red wine samples from a specific region, collected through laboratory analysis and sensory evaluations. The dataset comprises 12 features with a sample size of 985 entries after outlier removal (originally 1359). The data types include both numerical and categorical variables, with the quality score serving as the target variable, originally scored from 1 to 10 based on sensory evaluation. Profile attributes encompass physicochemical properties and sensory evaluations, such as acidity levels, sugar content, and alcohol percentage (see Table 1).

The Exploratory Data Analysis (EDA) conducted on the wine quality dataset revealed critical insights into its structure, distributions, and inter-feature relationships, informing subsequent preprocessing.

Feature Name	Data Type	Data Description and Storage Method
Fixed Acidity	Numerical	Fixed acid content (g/dm^3), e.g., tartaric acid
Volatile Acidity	Numerical	Volatile acid content (g/dm^3), e.g., acetic acid
Citric Acid	Numerical	Citric acid content (g/dm^3)
Residual Sugar	Numerical	Residual sugar after fermentation (g/dm^3)
Chlorides	Numerical	Chloride salt content (g/dm^3)
Free Sulfur Dioxide	Numerical	Free SO_2 content (mg/dm^3)
Total Sulfur Dioxide	Numerical	Total SO_2 content (mg/dm^3)
Density	Numerical	Wine density (g/cm^3)
pH	Numerical	pH level (0-14)
Sulphates	Numerical	Sulphate content (g/dm^3), e.g., potassium sulphate
Alcohol	Numerical	Alcohol content (% vol)
Quality	Categorical	Origin is score (1-10), based on sensory evaluation

Table 1. Dataset Detail (After Transformation).

4.1.1 *Feature Distributions and Outliers*

Figure 6 illustrates box plots of the standardized physicochemical features prior to outlier removal, including fixed acidity, volatile acidity, citric acid, residual sugar, chlorides, free sulfur dioxide, total sulfur dioxide, density, pH, sulphates, and alcohol. The standardization process ensured that each feature has a mean of 0 and a standard deviation of 1, with the interquartile range (IQR) and median clearly depicted. Most features, such as fixed acidity, volatile acidity, and pH, exhibit relatively symmetric distributions, with medians near 0 and whiskers extending approximately from -2 to 2. However, several features demonstrate significant outliers. For example, residual sugar displays numerous outliers above the upper whisker, with standardized values reaching up to 5, indicating a right-skewed distribution. Similarly, free sulfur dioxide and total sulfur dioxide show outliers extending to 4 and 5, respectively, while chlorides exhibit outliers on both ends, ranging from -2 to 4. These findings suggest natural variability in the wine samples or potential data quality issues, necessitating outlier handling techniques such as removal to mitigate their impact on machine learning models.

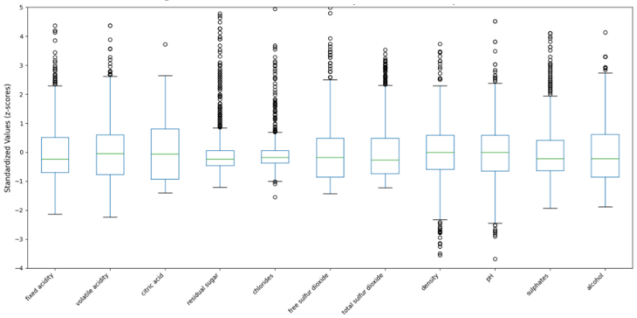

Figure 6. Box Plots of Standardized Features (Before Outlier Removal).

The original distribution of wine quality scores is presented in Figure 7, a histogram showing frequencies across scores ranging from 2 to 8. The distribution is right-skewed, with most wines concentrated at scores of 5 (approximately 550 instances) and 6 (around 500 instances). Lower scores of 3 and 4 are less frequent, with counts of about 10 and 50, respectively, while higher scores of 7 occur in approximately 200, respectively. No wines received scores of 2, 8, 9, or 10. This skewness indicates that most wines are of average quality, posing challenges for multi-class prediction due to the dominance of mid-range scores.

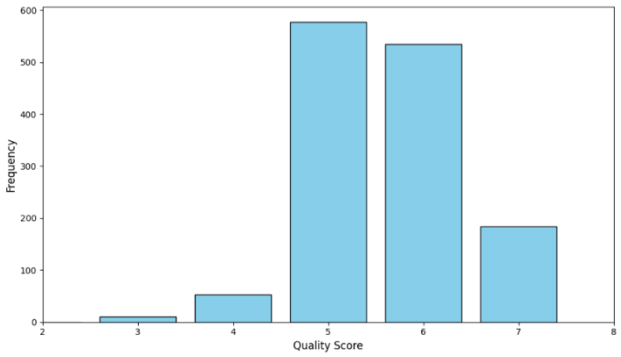

Figure 7. Distribution of Quality Scores.

To address this, the quality scores were simplified into binary categories: "low quality" (scores 1-5, encoded as 0) and "high quality" (scores 6-10, encoded as 1), a common practice in wine quality prediction research [30]. This transformation reduces the problem to a binary classification task, aligning with the goals of the machine learning algorithms employed in this study. Figure 8 displays the distribution of binary wine quality, with "low quality" comprising just over 600 samples and "high quality" slightly under 700 samples. The near-balanced distribution, differing by less than 100 instances, is advantageous for model training, as it minimizes class imbalance and reduces the risk of bias toward a dominant category.

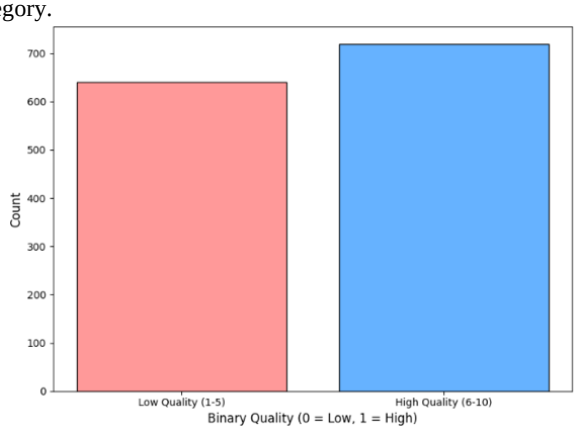

Figure 8. Distribution of Quality (After Transformation).

4.1.2 *Feature Correlations and Collinearity*

The feature correlation heatmap in Figure 9 reveals significant pairwise relationships among physicochemical features, indicating potential collinearity issues that could affect regression-based models by introducing redundancy and inflating variance. Notable correlations include a strong positive 0.63 between free sulfur dioxide and total sulfur dioxide, reflecting their compositional link, and 0.67 between fixed acidity and citric acid, showing their joint impact on wine acidity, while a strong negative -0.68 between fixed acidity and pH aligns with their inverse relationship, and -0.54 between density and alcohol suggests lower density with higher alcohol content. These pronounced correlations, such as the interdependence of free and total sulfur dioxide or among fixed acidity, citric acid, and pH, highlight the need for feature selection or dimensionality reduction to eliminate redundant variables and mitigate multicollinearity, thereby improving model stability and performance.

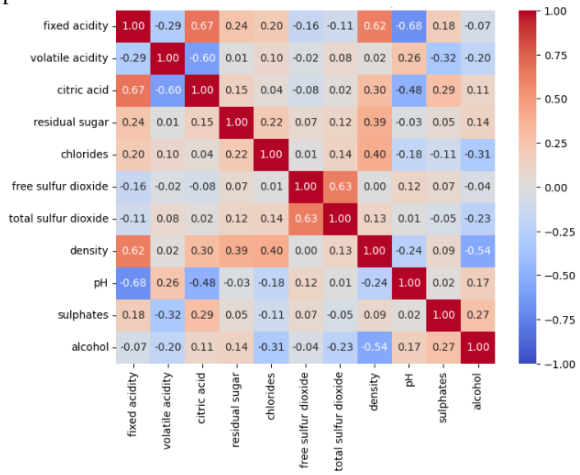

Figure 9. Feature Correlation Heatmap.

In conclusion, the EDA findings highlight the presence of outliers, a right-skewed distribution of original quality scores, a balanced binary quality distribution post-transformation, and significant

collinearity among features. These observations underscore the importance of preprocessing steps, including outlier handling and feature selection, to prepare the wine quality dataset for effective machine learning analysis.

4.2 Experiment Method

In Figure 7, a significant skew in the distribution of quality scores was observed. Figure 8 is the distribution of these binary wine quality labels after the transformation, with "low quality" comprising just over 600 samples and "high quality" slightly under 700 samples. The near-balanced distribution, with a difference of less than 100 instances between the two classes, facilitates effective model training by minimizing the risk of class imbalance, thereby reducing potential bias toward a dominant category and improving the model's generalization across both quality levels.

Firstly, data preprocessing was conducted to ensure the dataset was suitable for model training. Duplicate records were removed using function from Python's pandas library to maintain data integrity. In Figure 8, a significant skew in the distribution of quality scores was observed. To address this, the quality scores were transformed into binary categories: "low quality" (scores 1-5, encoded as 0) and "high quality" (scores 6-10, encoded as 1), as detailed in Section 3.3.2. This transformation converts the problem into a binary classification task, better suited to the machine learning algorithms used in this study. All feature columns, excluding the original quality scores and the new binary labels, were standardized using the StandardScaler to achieve a mean of 0 and a standard deviation of 1. After standardization, outliers in each feature were then processed. Outliers were detected and removed using the Interquartile Range (IQR) method, excluding data points falling below Q1 - 1.5 × IQR or above Q3 + 1.5 × IQR.

Secondly, feature selection was performed to identify the most relevant physicochemical attributes for predicting wine quality. Independent filter, wrapper, and embedded methods were applied, each generating feature subsets containing 3, 6, 9, and all 11 available features. Filter methods, including ANOVA, Kruskal-Wallis Test, and Mutual Information, ranked features based on their statistical relevance to the binary quality labels, selecting the corresponding number of features. Wrapper methods, such as Genetic Algorithms and Recursive Feature Elimination (RFE), iteratively evaluated feature combinations to optimize model performance, independently selecting the top 3, 6, and 9 features. Embedded methods, leveraging algorithms like Random Forest, LightGBM, Lasso, and Elastic Net, identified key predictors by assigning importance scores or applying regularization during their respective independent training processes, producing subsets with the same feature counts. The full set of 11 features was also retained as a baseline for comparison.

Thirdly, the Random Forest classifier was trained using the selected feature subsets. This classifier was chosen for its robustness and ability to handle complex datasets effectively. It was configured with 100 decision trees and a fixed random seed to ensure reproducibility across experiments. To assess its generalization performance, 10-fold cross-validation was employed: the dataset was divided into 10 equal folds, and in each iteration, 9 folds were used for training while the remaining fold served as the test set. This process was repeated until each fold had been used as the test set exactly once, providing a reliable estimate of the model's predictive capability.

Finally, the model's performance was evaluated using a suite of metrics to assess its effectiveness in classifying wine quality. These metrics included F1-Score, which balances precision and recall; Precision, measuring the accuracy of positive predictions; Recall, gauging the model's ability to identify all high-quality wines; and Accuracy, reflecting overall correctness. Emphasis was placed on the classifier's performance in identifying high quality wines namely class 1, given its practical significance. Additionally, the computational efficiency of the feature selection methods was evaluated by recording the time required to generate each feature subset and the model training time using the features selected by different methods, enabling a comprehensive comparison of both predictive performance and resource utilization. Lastly, a t-test was conducted on the model's accuracy to statistically validate the performance differences resulting from feature selection compared to the original dataset namely 11 features.

4.3 Results Evaluation

The 10-fold cross-validation accuracy results illustrate the performance of various feature selection methods—ANOVA, Kruskal, GA, RFE, RF, LightGBM, LASSO, Elastic Net, and Net—across datasets categorized by feature sizes: few, medium, and large. For the 'few' features dataset, accuracy ranges from approximately 0.65 to 0.80, with methods like RF and LightGBM achieving the upper bound, indicating robust performance in low-dimensional settings. In the 'medium' feature dataset, accuracy fluctuates between 0.65 and 0.75, with RFE and Elastic Net showing notable consistency, suggesting adaptability to moderate dimensionality. For the 'large' feature dataset, accuracy stabilizes between 0.65 and 0.70, with Kruskal and Net exhibiting outliers, reflecting challenges in scaling to high-dimensional data. Collectively, these findings underscore the varying efficacy of feature selection techniques, with RF and RFE demonstrating resilience across scales, while scalability remains a limiting factor for certain methods as feature numbers increase.

This section would compare the original model and the accuracy of the features selected by different feature selection methods and different feature numbers on the model at 99% confidence interval (p-value = 0.01).

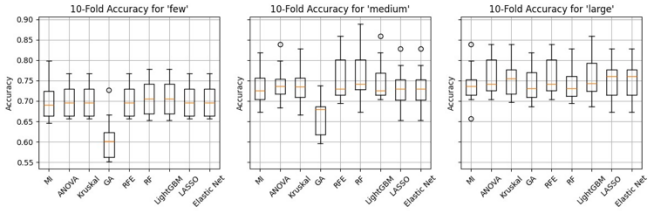

Figure 10. 10-Fold Accuracy of Feature Selection Across Dataset Sizes.

Method \ Quantity	Few	Medium	Large	Origin
MI	0.7004**	0.7359	0.7401	0.7634
ANOVA	0.6985**	0.7441	0.7583	
Kruskal	0.6985**	0.7400	0.7573	

Table 2. Accuracy Comparison of Filter-Based Feature Selection.

Method \ Quantity	Few	Medium	Large	Origin
GA	0.6080**	0.6619**	0.7431	0.7634
RFE	0.6985**	0.7532	0.7583	

Table 3. Accuracy Comparison of Wrapper-Based Feature Selection.

Method \ Quantity	Few	Medium	Large	Origin
RF	0.7106	0.7593	0.7441	0.7634
LGBM	0.7106	0.7501	0.7593	
Lasso	0.6985**	0.7329	0.7512	
Elastic Net	0.6985**	0.7329	0.7512	

Table 4. Accuracy Comparison of Embedded-Based Feature Selection.

For few features, the Filter method excels at 0.0985 seconds, outstripping the Embedded (0.5121s) and Wrapper (30.0814s) approaches. In medium feature sets, Filter's efficiency peaks at 0.0067 seconds, with Embedded at 0.3817 seconds and Wrapper at 21.4526 seconds. For large features, Filter scales optimally at 0.0032 seconds, Embedded follows at 0.1010 seconds, and Wrapper lags at 11.2407 seconds. This underscores Filter's scalability, Embedded's balanced efficacy, and Wrapper's precision-cost trade-off.

Method/Time	Few	Medium	Large
Best in Filter	0.0985s	0.0067s	0.0032s
Best in Wrapper	30.0814s	21.4526s	11.2407s
Best in Embedded	0.5121s	0.3817s	0.1010s

Table 5. Performance of Three Optimal Methods Across Varying Feature Quantities.

As the quantity of features increases, the duration required for model training correspondingly extends.

Few	Medium	Large	Origin
1.1549s	1.4125s	1.6074s	1.6698s

Table 6. Best Training Time.

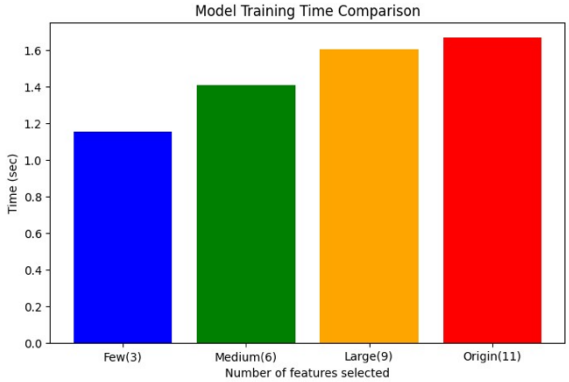

Figure 11. Training Time Comparison.

The Filter (ANOVA) method delivers an F1-score, precision, and recall of 0.76, with an accuracy of 0.7583 in just 0.0032 seconds, excelling in speed. The Wrapper (RFE) matches these metrics but requires 11.2407 seconds, reflecting its computational intensity. The Embedded (RF Medium) method slightly improves accuracy to 0.7593 with the same 0.76 metrics in 0.3817 seconds, offering balance. The Origin baseline achieves the highest accuracy (0.7634) and precision (0.77) instantly (0s), suggesting feature reduction may not always enhance performance. Filter prioritizes efficiency, Wrapper consistency, Embedded balance, and Origin raw accuracy.

Method \ Metric	F1-Score	Precision	Recall	Accuracy	Time
ANOVA	0.76	0.76	0.76	0.7583	0.0032s
RFE	0.76	0.76	0.76	0.7583	11.2407s
Random Forest	0.76	0.76	0.76	0.7593	0.3817s
Origin	0.76	0.77	0.76	0.7634	0

Table 7. Performance of Three Types of Methods in Optimal Configurations.

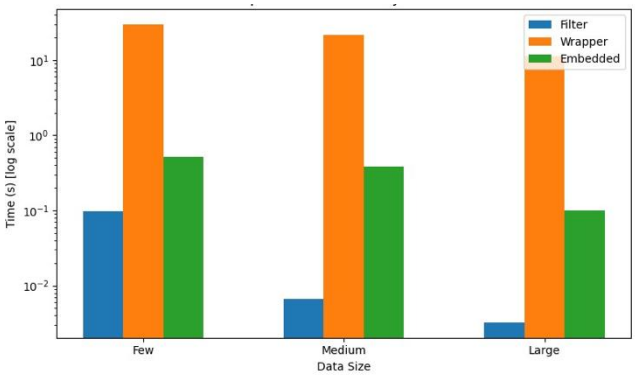

Figure 12. Selection Time Comparison.

Class 0 achieves a precision, recall, and F1-score of 0.74 with a support of 45, while Class 1 slightly outperforms with 0.78 across these metrics, supported by 54 instances, indicating a modest class imbalance favoring Class 1. Overall accuracy stands at 0.7593 across 99 instances, reflecting reliable predictive capability. Both macro and weighted averages converge at 0.76 for precision, recall, and F1-score, suggesting consistent performance across classes despite the support disparity, with the weighted average affirming robustness due to its alignment with the macro average. This balance underscores the model's effectiveness in handling the dataset's distribution.

Embedded: RF	Precision	Recall	F1-Score	Support
0	0.74	0.74	0.74	45
1	0.78	0.78	0.78	54
Accuracy			0.7593	99
Macro avg	0.76	0.76	0.76	99
Weighted avg	0.76	0.76	0.76	99

Table 8. Performance of Optimal Configuration in Three Types of Methods.

The confusion matrices for the Origin and RF (size=medium) models reveal their performance in a binary classification task, likely predicting wine quality. The Origin model correctly classifies 340 instances of Class 0 (true negatives) and 412 instances of Class 1 (true positives), with 109 false positives and 124 false negatives, yielding an accuracy of (340 + 412) / (340 + 109 + 124 + 412) ≈ 0.7634. The RF (size=medium) model slightly improves Class 1 prediction with 417 true positives but reduces true negatives to 331, while increasing false positives to 118 and false negatives to 119, resulting in an accuracy of (331 + 417) / (331 + 118 + 119 + 417) ≈ 0.7593. This suggests that while the RF model enhances Class 1 identification, the Origin model achieves higher overall accuracy, indicating that feature selection via RF may trade off some general performance for specific class improvements in this dataset.

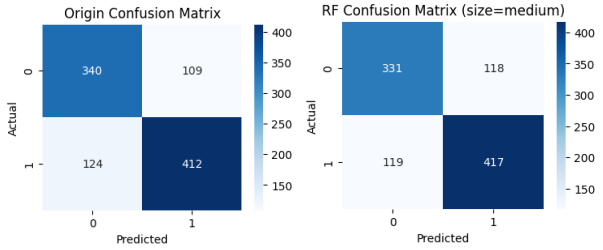

Figure 13. Comparative Evaluation of Origin and RF (Medium) Models via Confusion Matrices in Red Wine Quality Prediction.

5. CONCLUSION

Our research aims to employ feature selection methods to assist a random forest model in understanding red wine quality based on its attributes (e.g., Fixed Acidity, Alcohol, pH, etc.) and making accurate predictions. We first removed duplicate instances and outliers from the dataset, and then applied standardization. Next, we used three types of feature selection methods—Filter (MI, ANOVA, Kruskal), Wrapper (GA, RFE), and Embedded (RF, LightGBM, LASSO, Elastic Net)—to select sets of 3, 6, and 9 features. The results show that selecting 6 features using RF achieved an accuracy close to that of the model using all features, while significantly reducing training time. Finally, we compared the execution times of the different feature selection methods and found that Filter methods were the fastest, Embedded methods had moderate speed, and Wrapper methods were the slowest, which is consistent with findings in the relevant literature.

6. REFERENCES

[1] I. Guyon and A. Elisseeff, "An Introduction to Variable and Feature Selection".

[2] S. Visalakshi and V. Radha, "A literature review of feature selection techniques and applications: Review of feature selection in data mining," in *2014 IEEE International Conference on Computational Intelligence and Computing Research*, Dec. 2014, pp. 1–6. doi: 10.1109/ICCIC.2014.7238499.

[3] K. Jain, K. Kaushik, S. K. Gupta, S. Mahajan, and S. Kadry, "Machine learning-based predictive modelling for the enhancement of wine quality," *Sci. Rep.*, vol. 13, no. 1, p. 17042, Oct. 2023, doi: 10.1038/s41598-023-44111-9.

[4] "On the Adversarial Robustness of Feature Selection Using LASSO | IEEE Conference Publication | IEEE Xplore." Accessed: Mar. 12, 2025. [Online]. Available: https://ieeexplore.ieee.org/document/9231631

[5] R. J. Urbanowicz, M. Meeker, W. La Cava, R. S. Olson, and J. H. Moore, "Relief-based feature selection: Introduction and review," *J. Biomed. Inform.*, vol. 85, pp. 189–203, Sep. 2018, doi: 10.1016/j.jbi.2018.07.014.

[6] P. Bhardwaj, P. Tiwari, K. Olejar, W. Parr, and D. Kulasiri, "A machine learning application in wine quality prediction," *Mach. Learn. Appl.*, vol. 8, p. 100261, Jun. 2022, doi: 10.1016/j.mlwa.2022.100261.

[7] "A Study and Analysis of Machine Learning Techniques in Predicting Wine Quality," ResearchGate. Accessed: Mar. 14, 2025. [Online]. Available: https://www.researchgate.net/publication/352096578_A_Study_and_Analysis_of_Machine_Learning_Techniques_in_Predicting_Wine_Quality

[8] Y. Gupta, "Selection of important features and predicting wine quality using machine learning techniques," *Procedia Comput. Sci.*, vol. 125, pp. 305–312, Jan. 2018, doi: 10.1016/j.procs.2017.12.041.

[9] S. Kumar, K. Agrawal, and N. Mandan, "Red Wine Quality Prediction Using Machine Learning Techniques," in *2020 International Conference on Computer Communication and Informatics (ICCCI)*, Jan. 2020, pp. 1–6. doi: 10.1109/ICCCI48352.2020.9104095.

[10] M. Beri, K. S. Gill, and N. Sharma, "Predictive Modeling of Wine Quality using Machine Learning Techniques," in *2024 Second International Conference on Intelligent Cyber Physical Systems and Internet of Things (ICoICI)*, Aug. 2024, pp. 1017–1022. doi: 10.1109/ICoICI62503.2024.10696690.

[11] M. S. Amzad Basha, K. Desai, S. Christina, M. M. Sucharitha, and A. Maheshwari, "Enhancing red wine quality prediction through Machine Learning approaches with Hyperparameters optimization technique," in *2023 Second International Conference on Electrical, Electronics, Information and Communication Technologies (ICEEICT)*, Apr. 2023, pp. 1–8. doi: 10.1109/ICEEICT56924.2023.10157719.

[12] S. Salcedo-Sanz, L. Cornejo-Bueno, L. Prieto, D. Paredes, and R. García-Herrera, "Feature selection in machine learning prediction systems for renewable energy applications," *Renew. Sustain. Energy Rev.*, vol. 90, pp. 728–741, Jul. 2018, doi: 10.1016/j.rser.2018.04.008.

[13] J. Miao and L. Niu, "A Survey on Feature Selection," *Procedia Comput. Sci.*, vol. 91, pp. 919–926, Jan. 2016, doi: 10.1016/j.procs.2016.07.111.

[14] A. Bommert, X. Sun, B. Bischl, J. Rahnenführer, and M. Lang, "Benchmark for filter methods for feature selection in high-dimensional classification data," *Comput. Stat. Data Anal.*, vol. 143, p. 106839, Mar. 2020, doi: 10.1016/j.csda.2019.106839.

[15] N. El Aboudi and L. Benhlima, "Review on wrapper feature selection approaches," in *2016 International Conference on Engineering & MIS (ICEMIS)*, Sep. 2016, pp. 1–5. doi: 10.1109/ICEMIS.2016.7745366.

[16] "(PDF) Feature Subset Selection Problem using Wrapper Approach in Supervised Learning," *ResearchGate*, Oct. 2024, doi: 10.5120/169-295.

[17] B. H. Menze *et al.*, "A comparison of random forest and its Gini importance with standard chemometric methods for the feature selection and classification of spectral data," *BMC Bioinformatics*, vol. 10, no. 1, p. 213, Jul. 2009, doi: 10.1186/1471-2105-10-213.

[18] N. Pudjihartono, T. Fadason, A. W. Kempa-Liehr, and J. M. O'Sullivan, "A Review of Feature Selection Methods for

Machine Learning-Based Disease Risk Prediction," *Front. Bioinforma.*, vol. 2, Jun. 2022, doi: 10.3389/fbinf.2022.927312.

[19] Y. Guo, F.-L. Chung, G. Li, and L. Zhang, "Multi-Label Bioinformatics Data Classification With Ensemble Embedded Feature Selection," *IEEE Access*, vol. 7, pp. 103863–103875, 2019, doi: 10.1109/ACCESS.2019.2931035.

[20] Y. Akhiat, Y. Manzali, M. Chahhou, and A. Zinedine, "A New Noisy Random Forest Based Method for Feature Selection," *Cybern. Inf. Technol.*, vol. 21, no. 2, pp. 10–28, Jun. 2021, doi: 10.2478/cait-2021-0016.

[21] H. H. Htun, M. Biehl, and N. Petkov, "Survey of feature selection and extraction techniques for stock market prediction," *Financ. Innov.*, vol. 9, no. 1, p. 26, Jan. 2023, doi: 10.1186/s40854-022-00441-7.

[22] M. Chemmakha, O. Habibi, and M. Lazaar, "Improving Machine Learning Models for Malware Detection Using Embedded Feature Selection Method," *IFAC-Pap.*, vol. 55, no. 12, pp. 771–776, Jan. 2022, doi: 10.1016/j.ifacol.2022.07.406.

[23] "Feature Selection using LASSO." Accessed: Mar. 11, 2025. [Online]. Available: https://scholar.googleusercontent.com/scholar?q=cache:B8F-GlnIPXcJ:scholar.google.com/+LASSO+feature+selection&hl=zh-TW&as_sdt=0,5

[24] "Cervical Cancer Diagnosis Using Random Forest Classifier With SMOTE and Feature Reduction Techniques," ResearchGate. Accessed: Mar. 15, 2025. [Online]. Available: https://www.researchgate.net/publication/328110460_Cervical_Cancer_Diagnosis_Using_Random_Forest_Classifier_With_SMOTE_and_Feature_Reduction_Techniques

[25] G. Biau and E. Scornet, "A random forest guided tour," *TEST*, vol. 25, no. 2, pp. 197–227, Jun. 2016, doi: 10.1007/s11749-016-0481-7.

[26] N. Singh, A. Kumar, and Y. Yadav, "Predictive Modeling of Wine Quality Through Machine Learning Techniques," in *2024 OPJU International Technology Conference (OTCON) on Smart Computing for Innovation and Advancement in Industry 4.0*, Jun. 2024, pp. 1–6. doi: 10.1109/OTCON60325.2024.10687487.

[27] S. J. Rigatti, "Random Forest," *J. Insur. Med.*, vol. 47, no. 1, pp. 31–39, Jan. 2017, doi: 10.17849/insm-47-01-31-39.1.

[28] A. Parmar, R. Katariya, and V. Patel, "A Review on Random Forest: An Ensemble Classifier," in *International Conference on Intelligent Data Communication Technologies and Internet of Things (ICICI) 2018*, J. Hemanth, X. Fernando, P. Lafata, and Z. Baig, Eds., Cham: Springer International Publishing, 2019, pp. 758–763. doi: 10.1007/978-3-030-03146-6_86.

[29] S. A. Hicks *et al.*, "On evaluation metrics for medical applications of artificial intelligence," *Sci. Rep.*, vol. 12, no. 1, p. 5979, Apr. 2022, doi: 10.1038/s41598-022-09954-8.

[30] S. Davessar, "Wine Quality Prediction using Machine Learning," *Int. J. Res. Appl. Sci. Eng. Technol.*, vol. 12, no. 1, pp. 572–576, Jan. 2024, doi: 10.22214/ijraset.2024.58014.

[31] S. Kumar, K. Agrawal, and N. Mandan, "Red Wine Quality Prediction Using Machine Learning Techniques," in *2020 International Conference on Computer Communication and Informatics (ICCCI)*, Jan. 2020, pp. 1–6. doi: 10.1109/ICCCI48352.2020.9104095.

[32] M. S. Amzad Basha, K. Desai, S. Christina, M. M. Sucharitha, and A. Maheshwari, "Enhancing red wine quality prediction through Machine Learning approaches with Hyperparameters optimization technique," in *2023 Second International Conference on Electrical, Electronics, Information and Communication Technologies (ICEEICT)*, Apr. 2023, pp. 1–8. doi: 10.1109/ICEEICT56924.2023.10157719.

[33] "Prediction of Wine Quality Using Machine Learning Algorithms." Accessed: Mar. 11, 2025. [Online]. Available: https://www.scirp.org/journal/paperinformation?paperid=107796

[34] K. Jain, K. Kaushik, S. K. Gupta, S. Mahajan, and S. Kadry, "Machine learning-based predictive modelling for the enhancement of wine quality," *Sci. Rep.*, vol. 13, no. 1, p. 17042, Oct. 2023, doi: 10.1038/s41598-023-44111-9.

7. WORK DIVISION

Name	Work description	Contribution
KAI-XIANG, CHANG 113423045	1. Searching Wrapper method dealing imbalanced dataset 2. Design experiment 3. Implementation	33%
ZEI-WEI, XIE 113423036	1. Searching Filter method dealing imbalanced dataset 2. Design experiment 3. Implementation	33%
KAI-SONG, KUO 113423027	1. Searching Embedded method dealing imbalanced dataset 2. Design experiment 3. Implementation	33%