

Wine feature importance and quality prediction: A comparative study of machine learning algorithms with unbalanced data

Siphendulwe Zaza¹zazasiphendulwe@gmail.com
Marcellin Atemkeng^{1,*}m.atemkeng@ru.ac.za
Sisipho Hamlomo^{1,2}s.hamlomo@ru.ac.za

¹ Department of Mathematics, Rhodes University, Grahamstown, South Africa

² Department of Statistics, Rhodes University, Grahamstown, South Africa

Abstract. Classifying wine as "good" is a challenging task due to the absence of a clear criterion. Nevertheless, an accurate prediction of wine quality can be valuable in the certification phase. Previously, wine quality was evaluated solely by human experts, but with the advent of machine learning this evaluation process can now be automated, thereby reducing the time and effort required from experts. The feature selection process can be utilized to examine the impact of analytical tests on wine quality. If it is established that specific input variables have a significant effect on predicting wine quality, this information can be employed to enhance the production process. We studied the feature importance, which allowed us to explore various factors that affect the quality of the wine. The feature importance analysis suggests that alcohol significantly impacts wine quality. Furthermore, several machine learning models are compared, including Random Forest (RF), Support Vector Machine (SVM), Gradient Boosting (GB), K-Nearest Neighbors (KNN), and Decision Tree (DT). The analysis revealed that SVM excelled above all other models with a 96% accuracy rate.

Keywords: Random Forest · Support Vector Machine · Gradient Boosting · K-Nearest Neighbors · Decision Tree · Feature selection · Wine

1 Introduction

The quality of wine is very important for both consumers and the wine industry therefore, it is imperative to determine wine quality before manufacturing or consumption. However, relying on human expert wine tasting for measuring wine quality can be a time-consuming and subjective process, posing significant challenges for experts in providing accurate predictions. According to [1], wine testing by human experts can also put them at health risk as they are exposed to a range of chemicals and other substances that may be harmful to their health. For example, the inhalation of volatile organic compounds (VOCs) such as ethanol, acetaldehyde, and ethyl acetate, during the process of wine tasting has been linked to a range of health issues, including headaches, coughs, and

respiratory problems [2, 3]. With the aid of machine learning algorithms, it is now possible to analyze the physiochemical properties of wine, which can be used to predict its quality. The aim of this paper is to use the chemical and physical properties of wine to predict its quality and to determine which features are more important for predicting good wine. We use the following algorithms: Decision Tree (DT), Random Forest (RF), Support Vector Machine (SVM), K-Nearest Neighbors (KNN), and Gradient Boosting (GB). These models are used due to the nature of the wine data we used to run the experiment. The data is of small samples, and it is also imbalanced. Shallow machine learning models have shown the potential to outperform deep learning models on small datasets. For example, [4] and [5] used some of the above-mentioned shallow machine learning models on small datasets, and these algorithms have shown exceptional performance in addressing the challenges of small sample sizes and imbalanced data.

The contribution of this work is as follows: we have trained five models and compared their performance on an unbalanced dataset, then we move further to use some sampling methods to balance the dataset and then retrain the models. Sampling methods improved the accuracy of the models with SVM resulting from 78% without sampling to 96% with sampling, thereby outperforming other models.

2 Related Work

[6] has employed a range of machine learning techniques such as linear regression to find important features for prediction and also used SVM and neural networks to predict values. The conclusion is reached that not all features are important for predicting wine quality hence one can select features that are most likely to be useful for predicting the quality of the wine. They used both the white wine and red wine datasets for their analysis, which is slightly different from our work. In our study, we focused only on the red wine dataset for our analysis and we compared our study with the work of [6] who used two datasets which are the white wine and red wine datasets. Our findings with the red wine dataset aligned with the results in [6] for predicting wine quality.

[7] employed four machine learning techniques namely RF, stochastic gradient descent, SVM, and logistic regression to forecast the quality of the wine. Out of the four techniques, RF outperformed other methods with an accuracy of 88%. In the latter work, the red wine dataset is used [5], which was then divided into two classes namely good wine and bad wine. Our research is similar to this, but we attempted to extend the problem by introducing three classes. We found that SVM was the best-performing model for predicting the quality of wine, with an accuracy of 96% compared to the 88% accuracy achieved by RF in [7]. In [4] the naive Bayes, DT, SVM, and RF are used to predict wine quality. The analysis shows that when the residual sugar is minimal the quality of the wine increases and does not change significantly, suggesting that this feature is not

as important as others such as alcohol and citric acid. We also observed in the research that our machine learning models were producing acceptable results when residual sugar was excluded. This suggests that residual sugar is not an important feature when predicting wine quality.

3 Data description and preprocessing

3.1 Data description

The red wine dataset utilized in this study is sourced from the UCI machine learning repository [8]. This dataset comprises 1599 instances of red wine, and its quality is assessed through 11 distinct input variables including Fixed acidity, Volatile acidity, Citric acid, Residual sugar, Chlorides, Free sulfur dioxide, Total sulfur dioxide, Density, PH, Sulphates, and Alcohol. The output variable quality is based on these input parameters and is rated on a scale of 0 to 10, with 0 representing poor wine and 10 signifying excellent wine. Table 1 presents the statistical summary of the red wine dataset employed in this paper.

Variable Name	Mean	Sd	Min	Max	Median
Fixed acidity	8.31	1.73	4.60	15.90	7.90
Volatile acidity	0.52	0.18	0.12	1.58	0.52
Citric acid	0.27	0.19	0.00	1.00	0.26
Residual sugar	2.52	1.35	0.90	15.50	2.20
Chlorides	0.08	0.04	0.01	0.61	0.07
Free sulfur dioxide	15.89	10.44	1.00	72.00	14.00
Total sulfur dioxide	46.82	33.40	6.00	289.00	38.00
Density	0.99	0.001	0.99	1.00	0.99
PH	3.30	0.15	2.74	4.01	3.31
Sulphates	0.65	0.17	0.33	2.00	0.62
Alcohol	10.43	1.08	8.40	14.90	10.20
Quality	5.62	0.82	3.00	8.00	8.00

Table 1: statistics for red wine dataset

3.2 Data Pre-processing

We use label encoding, a process that converts the labels into a machine-readable form. We use this method to categorize the data into good, normal, or bad categories. We label bad wine as wine with a quality score that is less than 5, normal wine as wine with a quality score that is between 5 and 6, and good wine as wine with a quality score between 7 and 10, as shown in the flowchart in Figure 1. Also as part of data pre-processing, we excluded duplicate entries and data points with missing values in the dataset to maintain the integrity of the analysis.

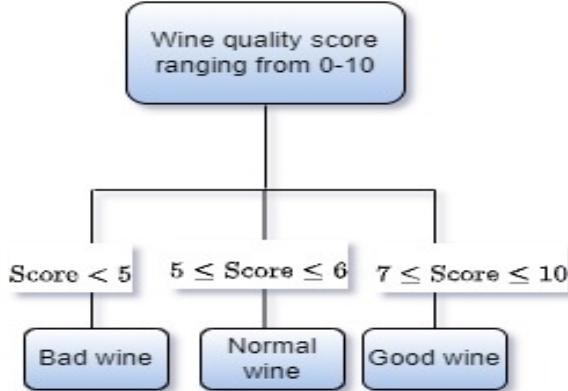


Fig. 1: Label encoding

3.3 Data analysis

The covariance matrix provides values within the range of $(-1, 1)$ which gives us information about the relationship between variables. A value of 1 indicates a strong positive linear correlation between variables whereas -1 suggests a strong negative linear correlation. On the other hand, a value of 0 indicates no relationship between the variables. This allows us to quickly understand the interconnections between the variables in our analysis. By examining the matrix, we easily identify which features have a high correlation with quality and are likely to be significant contributors to the machine learning models.

In Figure 2, we can see a correlation matrix showing a visual representation of the relationship between several variables, including "quality vs. alcohol," "volatile acidity vs. alcohol", "density vs. alcohol", and "sulphates vs. alcohol". Although the primary objective of this study is to identify features that are most indicative of good wine quality, it is evident from Figure 2 that certain features such as alcohol, volatile acidity, and chlorides, exhibit the highest correlations with quality. This suggests that these variables have the most significant impact on predicting the quality of the wine.

The feature selection process aims to reduce the number of input variables in a machine learning model by identifying and retaining only the relevant data. This can be achieved by choosing the features that are likely to be useful in finding a solution to the problem, thereby reducing noise in the data and enhancing the performance of the model [9]. One of the objectives of this study is to look into the relationship between various features through the use of Pearson's correlation coefficient to quantify the associations between the different features.

In Table 2 features are ranked according to their correlation values. According to [10] Pearson correlation coefficient ρ given a pair of random variables (X, Y)

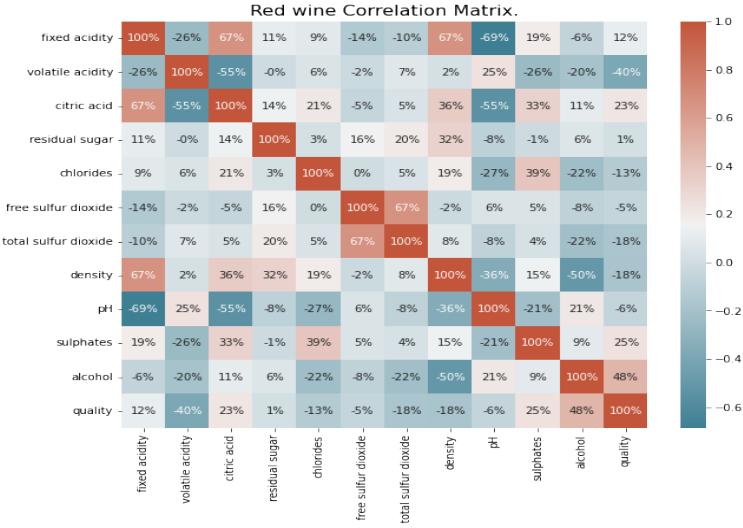


Fig. 2: Red wine correlation matrix

where X and Y are features, the formula for ρ is

$$\rho_{x,y} = \frac{\text{cov}(X, Y)}{\sigma_X \sigma_Y}, \quad (1)$$

where cov is the covariance, σ_X is the standard deviation of feature X and σ_Y is the standard deviation of feature Y .

Table 2 presents the selected features, out of which 10 were chosen for further analysis. However, following the principle of selecting essential features for improved model performance as suggested by [6], we excluded 'residual sugar' based on our machine learning model's consistently better performance without it. This decision was supported by data indicating that 'residual sugar' had a relatively minor impact on wine quality compared to other variables. Figure 3 (shown below) visually illustrates the relationship between quality and residual sugar. It is observed that quality tends to increase when residual sugar is minimal and remains relatively unchanged beyond a certain point. This finding suggests that "residual sugar" is not as crucial as other variables such as alcohol in determining the quality of the wine. Figure 4 depicts quality against alcohol, we can clearly see that alcohol is greatly contributing to the quality of wine, as the quality of wine increases we can see that the alcohol also increases. The results of the analysis revealed that the models performed better with the selected features as compared to when we used all the features.

Data standardization is a process that involves transforming data into a standardized form that will ensure that its distribution has a standard deviation of 1 and a mean of 0. The process of data standardization is essential as it helps

Rank	Name	Correlation
1	alcohol	48%
2	volatile acidity	-40%
3	sulphates	25%
4	citric acid	23%
5	total sulfur dioxide	-18%
6	density	-18%
7	chlorides	-13%
8	fixed acidity	12%
9	pH	-6%
10	free sulfur dioxide	-5%
11	residual sugar	1%

Table 2: Correlation with Quality

in equalizing the range of information [4], allowing for a more fair comparison between different features. For instance, as shown in Table 1 the overall Sulfur Dioxide readings are notably greater than chlorides. When we train machine learning models, having one variable with an exceptionally high value can mask all others, causing bias. Hence we need to standardize our data.

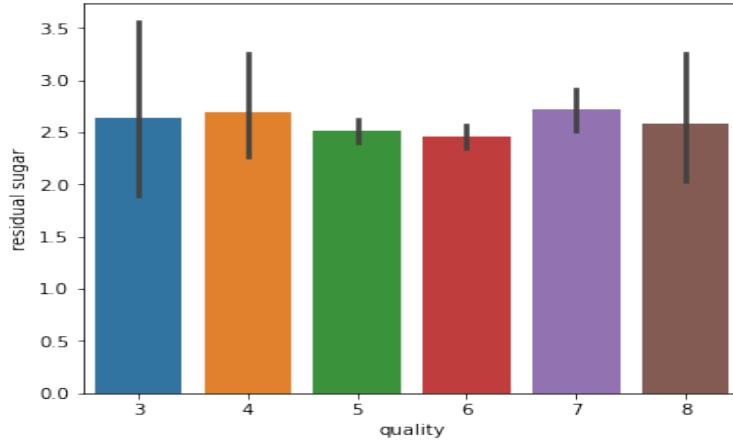


Fig. 3: Residual sugar versus quality

4 Classification Methods

4.1 Support Vector Machine

SVM is one of the most well-known supervised learning algorithms that maximizes the margin. The goal of a support vector machine is to find a hyper-

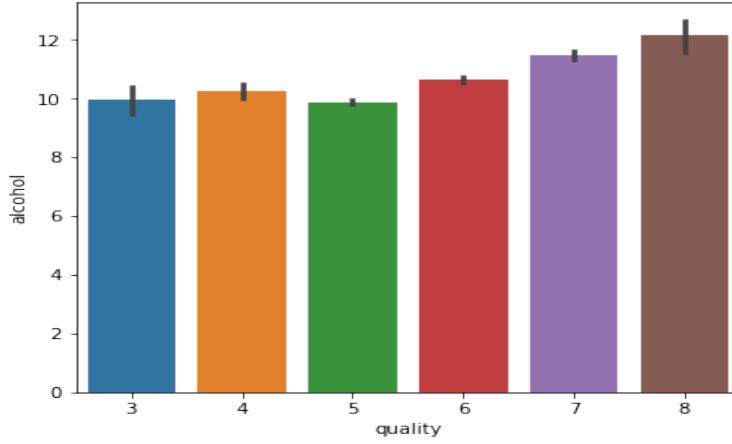


Fig. 4: Alcohol versus quality

plane that can efficiently separate various classes of data points within a high-dimensional space. This will enable us to swiftly classify new data points [11]. A hyperplane is the optimal decision boundary. The SVM algorithm takes into account the various extreme points that help in creating a hyperplane. The SVM algorithm is used for both linear (separable case) and non-linear (non-separable case) data. Let $D = \{(x_i, y_i)\}_{i=1}^N$ where (x_i, y_i) represents an individual data point and its corresponding label, and $D \in \mathbb{R}^{m \times n}$ be a training data with m rows and n columns. Here $x_i \in \mathbb{R}^n$ and $y_i \in \{0, 1, 2\}$ are indicating a multi-class classification with 0 as bad quality wine, 1 as normal wine, and 2 as good quality wine. We construct a function to classify the quality of the wine based on its features x_i .

$$f : \mathbb{R}^n \rightarrow \mathbb{R}$$

$$x_i \mapsto f(x_i) = \begin{cases} 0, & \text{if wine is bad quality, or} \\ 1, & \text{if wine is normal quality, or} \\ 2, & \text{if wine is good quality} \end{cases}$$

4.1.1 Linear SVM (separable case)

According to [11], we first assume that the training data are linearly separable and that there is a hyperplane that separates the data without error. In this case, we look for the maximum margin hyperplane:

$$f(x) = \langle w, x \rangle + b = w^T x + b. \quad (2)$$

Where $\langle \cdot, \cdot \rangle$ and w^T are the inner product and the transpose of the vector w

respectively. If x_s is a support vector and $H = \{x | w^T x + b = 0\}$, then the margin is given by:

$$\begin{aligned} \text{Marge} &= 2d(x_s, H) \\ &= \frac{2|w^T x + b|}{\|w\|}, \end{aligned} \quad (3)$$

where w is a normal vector called weight, x is the input vector and b is a bias. The parameter w and b are not unique, and kw and kb give the same area of separation:

$$\begin{aligned} kw^T x + kb &= k(w^T x + b) \\ &= 0. \end{aligned} \quad (4)$$

We then impose the normalization condition $|w^T x_s + b| = 1$ for the x_s support vectors, which leads to:

$$\text{Marge} = \frac{2}{\|w\|}. \quad (5)$$

In order to minimize the margin, we thus need to minimize $\|w\|$. Recall the normalization conditions: $wx_i + b = 1$ if x_i is a support vector of class +1 and $wx_i + b = -1$ if x_i is a support vector of class -1.:

$$\begin{cases} \text{if } y_i = 1 \text{ then } wx_i + b \geq 1 \text{ and thus } y_i(wx_i + b) \geq 1 \\ \text{if } y_i = -1 \text{ then } wx_i + b \leq -1 \text{ and thus } y_i(wx_i + b) \geq 1 \end{cases}$$

We now must solve a quadratic programming problem of optimization (called primal problem):

$$\begin{cases} \min_{w,b} \frac{1}{2} \|w\|^2 \\ \text{if } y_i = -1 \text{ then } wx_i + b \leq -1 \text{ and thus } y_i(wx_i + b) \geq 1 \end{cases}$$

The two parallel normal constraints of this optimization problem are separated by a Lagrange function. To solve this problem, we can combine the two constraints into a new Lagrangian function. We can also introduce new "slack variables" that denote α and require the derivative of the function to be zero. According to [11] the Lagrangian is given by:

$$L(w, b, \alpha) = \frac{1}{2} \|w\|^2 + \sum_{i=1}^n \alpha_i [y_i(w^T x_i + b - 1)], \quad (6)$$

where α_i represents the Lagrange multiplier introduced to solve the constrained optimization problem.

4.1.2 Linear SVM (Non-separable case)

Hyperplane cannot completely segregate binary classes of data in the majority of real-world data, hence we accept some observations in the training data on the incorrect side of the margin or hyperplane. Here, is the primal optimization problem of Soft Margin:

$$\begin{cases} \min_{w,b} \left(\frac{1}{2} \|w\|^2 - C \sum_{i=1}^n \xi_i \right) \\ y_i(wx_i + b) \geq 1 - \xi_i \text{ and } \xi_i \geq 0, i = 1, \dots, n \end{cases}$$

where ξ_i is the slack variable that allows misclassification; the penalty term $\sum_i^n \xi_i$ is a measure of the total number of misclassification in the model and C is a penalty variable for misclassified points [12]. Using the same terminology for separable SVM, we get the dual problem:

$$\begin{cases} \max_{\alpha} \left(\sum_{i=1}^n \alpha_i - \frac{1}{2} \sum_{i,j=1}^n \alpha_i \alpha_j y_i y_j (x_i x_j) \right) \\ \sum_{i=1}^n \alpha_i y_i = 0 \\ C \geq \alpha_i \geq 0, i = 1, \dots, n. \end{cases}$$

The classification of a new observation x is determined by the decision function:

$$f(x) = \sum_{i=1}^n \alpha_i y_i (x_i x) + b. \quad (7)$$

4.2 Decision Tree

A decision tree is a type of machine learning that takes into account the various inputs and outputs in a given training program. It then continuously splits the data according to a set of parameters. The two entities that comprise a decision tree are the leaves and the decision nodes [13].

Getting the correct attribute for a particular tree's root node is a huge challenge. This is why it is important to consider the various methods that are available to select attributes. There are two main methods that are commonly used to select attributes which are entropy and information gain. Let S be a sample and S_1, \dots, S_k the partition of S according to the classes of the target attribute. The Gini is denoted as $Gini(S)$ and the entropy is denoted as $Ent(S)$ are defined by [14].

$$Gini(S) = \sum_{i=1}^k \frac{|S_i|}{|S|} \times \left(1 - \frac{|S_i|}{|S|} \right) = \sum_{i \neq j} \frac{|S_i||S_j|}{|S|}, \quad (8)$$

and the entropy as:

$$Ent(S) = - \sum_{i=1}^k \frac{|S_i|}{|S|} \times \log \left(\frac{|S_i|}{|S|} \right), \quad (9)$$

where $|S_i|$ is the cardinality in the set S_i and $|S|$ is the cardinality in the sample S . The variables i, j , and k represent indices where i refers to attribute classes, j indicates different classes for the Gini formula, and k represents the total class.

4.3 Random Forest

RF is a widely used algorithm that is a part of the supervised learning framework. It can be utilized for regression and classification problems. It's based on the idea of ensemble learning, in which multiple classifiers are combined to solve a given problem and to enhance the model's performance. [13]. Figure 5 demonstrates how random forest predicts the quality of the wine.

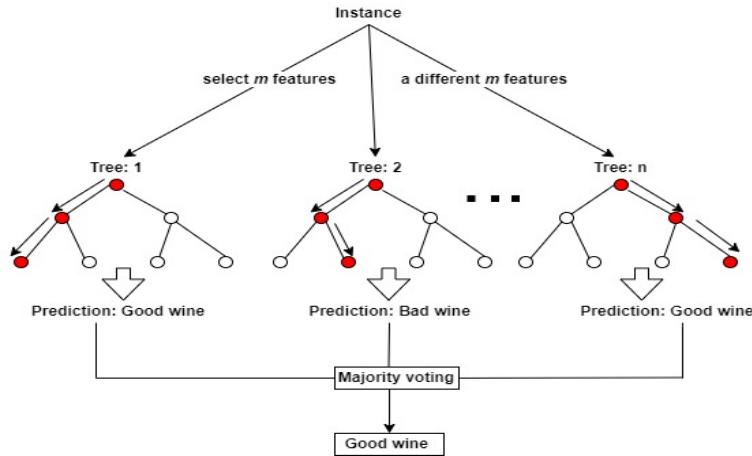


Fig. 5: Random Forest(adapted from [15])

The RF classifier combines the power of numerous decision trees. It creates several decision trees using bootstrapped datasets and randomly chooses a subset of the variables for each stage. Figure 5 shows how RF works. It aggregates the predicted outcomes from all the decision trees, and it chooses the mode that is most likely to perform well. This approach ensures that the model is more accurate and reliable, minimizing the risk that a single tree can make an error. By adopting a "majority wins" approach, RF ensures that the ultimate prediction is derived from a collective agreement among the decision trees, instead of relying solely on the outcome of an individual tree.

4.4 Gradient Boosting

A Gradient Boosting Machine is a type of tool that creates a strong learner by taking weak individuals and merging them into a single model. It can be used for classification and regression tasks. Although it is mainly utilized for tree-based models, it can also be applied to other weak individuals [16].

The fundamental concept behind GB involves incorporating new models into the ensemble, with each new model focusing on the examples that were incorrectly classified by the previous models. In order to focus on these difficult examples, GB fits each new model to the negative gradient of the loss function with respect to the current ensemble model [17]. The GB method can be used in various applications such as regression, ranking problems, and classification.

4.5 K-Nearest Neighbours

The KNN classifier is a machine learning algorithm used for classification and regression tasks that work on the premise that similar objects are usually located near each other [12]. In order for KNN to find the neighbors of a query point we need to calculate the distance between the query point and the other data points. These distance measures help in the formation of decision boundaries, which divide query points into distinct areas. One of the main drawbacks of the KNN algorithm is that it may be biased towards the majority class in datasets that are imbalanced, meaning that there are significantly more instances in one class than in another [18]. This is because KNN classifies query points by finding the k nearest neighbours in the training set and if the majority class dominates the neighbourhood of the test instance it is likely to be classified as the majority class.

Let's say we have a dataset with X representing a matrix that contains the features observed and Y representing the class label. Lets assume we have a point x which has coordinates (x_1, x_2, \dots, x_p) and point y with coordinates (y_1, y_2, \dots, y_p) [12]. The KNN algorithm is in this study because it categorizes new cases based on the Euclidean distance between the training data and the test observation. In k-NN, the optimal choice is determined by identifying the set of training data points that are closest to the given test observation in terms of Euclidean distance [19].

$$d(x_i, x_t) = \sqrt{\sum_{j=1}^d (x_{ij} - x_{tj})^2}, = \|x_i - x_t\| \quad (10)$$

where x_i represent the training data and x_t represent the test observation. Majority voting is the process of selecting the class that has the highest number of votes among the k -nearest neighbours in the K-nearest neighbours (KNN) algorithm. Majority voting is defined as follows according to [18]:

$$\hat{f}(x_t) = \operatorname{argmax}_{c \in \{c_1, c_2, c_3\}} \sum_{(x_i, y_i) \in N_k(x_t)} I(y_i = c), \quad (11)$$

where x_t represent the test observation, $\hat{f}(x_t)$ represent a forecasted class label, $N_k(x_t)$ represent a set of training instances and $I(\cdot)$ represent an indicator function that takes a value as input and returns either 0 or 1 based on whether the input satisfies a certain condition [18].

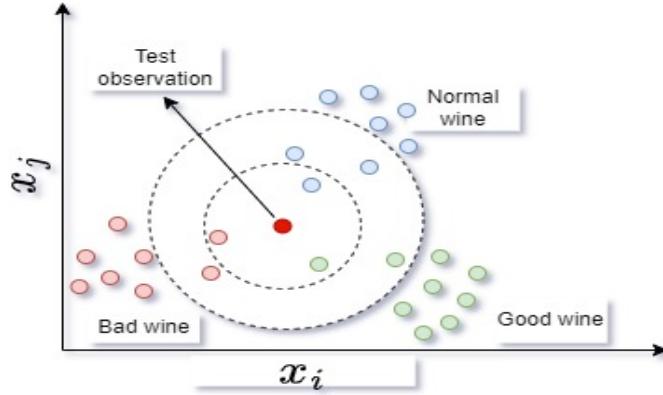


Fig. 6: KNN with different k -values (adapted from [12])

Figure 6 shows the KNN classifier with $K=3$ and $K=7$. We need to predict the class for the new observation (red circle) if it belongs to a class of Bad wine, Normal wine, or Class of Good wine respectively. If we choose $k=3$ (for a small dotted circle) then we have one observation in Class Bad wine, one observation in Class Normal wine, and one observation in Class Good wine. From this we have $Pr(\text{Bad wine})=\frac{1}{3}$, $Pr(\text{Normal wine})=\frac{1}{3}$ and $Pr(\text{Good wine})=\frac{1}{3}$ respectively. We can clearly see that we have a tie among our classes where each class has one observation. Since the number of neighbours in class Bad Wine, class Normal, and class Good Wine are the same, we cannot determine the class of the new data point based on the number of neighbours alone. According to [20], we can use different tie-breaking techniques to determine the class in case of a tie. One common method is to choose the class that has the shortest average distance to the new data point. If we choose $k=7$ (for a big dotted circle) then we have two observations in Class Bad Wine, three observations in Class Normal Wine, and two observations in Class Good Wine. From this we have $Pr(\text{Bad wine})=\frac{2}{7}$, $Pr(\text{Normal wine})=\frac{3}{7}$ and $Pr(\text{Good wine})=\frac{2}{7}$, so we can clearly see that the small red circle (test observation) belongs to class Normal wine based since class Normal wine has the highest probability as compared to other classes (majority voting). The value of a classifier determines the performance of that class. However, selecting the correct different k values can be very challenging. This is because the value of k can have a huge impact on the accuracy of the predictive

model [21].

5 Experimental Settings

5.1 Unbalanced Data

Figure 7 demonstrates the red wine quality classes, the dataset's distribution can be seen with the most significant value being 5 with the class values ranging from 3 to 8.

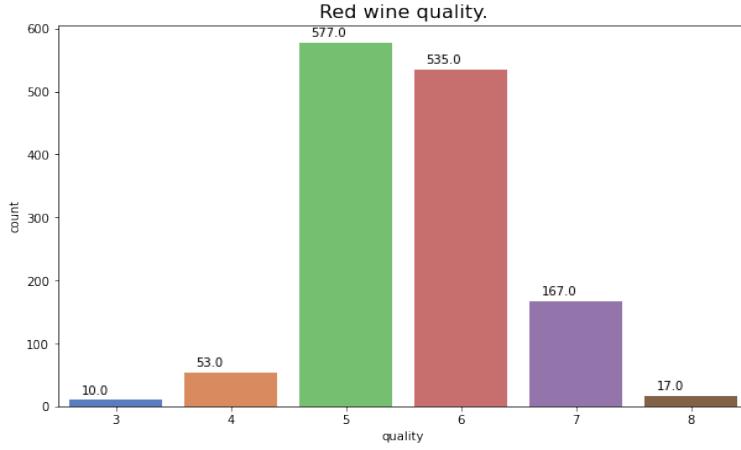


Fig. 7: Distribution of red wine quality

The dataset depicts an unbalanced distribution of red wine with other classes not being fairly represented, the instances range from 10 in the minority class to 577. As suggested by [22], sampling techniques such as undersampling, oversampling, and SMOTE are used to handle unbalanced datasets. These are further discussed in section 5.2.

5.2 Sampling Techniques

5.2.1 Undersampling and Oversampling

The oversampling method is an intuitive technique that increases the size of a minority class by creating duplicates of samples taken from the under-represented group. Undersampling on the other hand ensures that all of the data from the minority segment are kept and reduces the size of the majority segment to be

the same as the minority segment. Undersampling is usually considered to be a disadvantage as it eliminates potentially useful data. Oversampling on the other hand is more likely to cause overfitting since it duplicates existing examples [23].

5.2.2 Synthetic Minority Oversampling Technique

According to [22], using the SMOTE filter proves to be a valuable approach in addressing imbalanced wine datasets. SMOTE employs a k-nearest neighbour method to create synthetic data points. SMOTE starts by selecting K nearest neighbours from the minority samples based on the desired level of oversampling and then randomly selecting a neighbour from K nearest neighbors [22]. The selection process is not deterministic as the K nearest neighbours are chosen randomly. The selection of the K neighbors is done randomly and the random data is combined to generate synthetic data. SMOTE utilizes synthetic data points to add diversity to the minority class, mitigating the issue of overfitting that arises from random sampling techniques. According to [22] SMOTE also creates a more balanced dataset which can help improve the performance of machine learning models when dealing with imbalanced data.

5.3 Hyper parameter tuning

In machine learning, the task of selecting a set of optimal hyperparameters for a learning algorithm is known as hyperparameter tuning. The simplest approach to tuning hyperparameters is undoubtedly grid search. Using this method, we simply construct a model for every possible combination of the supplied hyperparameter values and evaluate each model, and choose the model that yields the best results [24]. According to [25], hyperparameter optimization is expressed as:

$$x^* = \arg \min_{x \in X} f(x), \quad (12)$$

where $f(x)$ represents a score that we aim to minimize, such as the error rate evaluated on the validation set. x^* refers to the set of hyperparameters that produces the lowest score value while x can take any value within the X domain. With this, we want to determine the model hyperparameters that provide the highest score on the validation set metric.

5.4 Model Evaluation

To understand how well and efficiently the model performs, we measure and evaluate its performance. There are four techniques used to determine the accuracy of predictions:

- True Positive (TP): This indicates the percentage of samples that the model correctly identifies as positive.
- False Positive (FP): It represents the percentage of samples that the model mistakenly predicts as positive when they are actually negative.

- False Negative (FN): These are the samples that the model wrongly classifies as negative while they are positive in reality.
- True Negative (TN): These are the samples that the model accurately identifies as negative.

We use the following techniques to assess the model.

1. Accuracy: It can be characterized as either the proportion of all positive classes that the model correctly predicted to be true or the number of accurate outputs that the model provides. Its formula is:

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}. \quad (13)$$

2. Precision: Precision refers to the ratio of predicted observations to the total number of expected positive observations. Its formula is:

$$\text{Precision} = \frac{TP}{TP + FP}. \quad (14)$$

3. Recall: Recall is known as the proportion of accurately predicted positive observations to all of the actual class observations. Its formula is:

$$\text{Recall} = \frac{TP}{TP + FN}. \quad (15)$$

4. F_1 Score: F_1 score is calculated as the balanced average of recall and accuracy. The test accuracy of the model is evaluated using the F_1 score. Its formula is [26]:

$$F_1\text{Score} = 2 \times \frac{\text{Recall} \times \text{Precision}}{\text{Recall} + \text{Precision}}. \quad (16)$$

According to [27], accuracy is the primary metric used to evaluate models, but when dealing with skewed class distributions and imbalanced datasets, it becomes challenging to make accurate judgments. For instance, the recall rate for minority groups has typically dropped to zero. This indicates that the model is not able to properly classify them. The reduction in recall and precision scores for minority groups is due to how the model focuses more on the majority segment instead of the minority segments. This issue is caused by the preference of the accuracy model for the majority group. As a result, the classifier tends to perform poorly on the minority groups.

6 Results and Discussion

6.1 Results

For the purpose of this study, we are using five machine learning algorithms to predict the wine quality namely SVM, DT, KNN, GB, and RF. We implemented our models in an unbalanced dataset with default parameters and the results are shown in Table 3 below that our models performed poorly with support vector machine and random forest having the highest accuracy with 78% each. Table 3 provides a comprehensive overview of our models' performance across metrics such, as accuracy, precision, recall, and F1 score.

Class	SVM			RF			KNN			GB			DT		
	Precision	Recall	F1 Score												
0	1.00	0.02	0.05	0.22	0.05	0.08	0.36	0.09	0.15	0.29	0.09	0.14	0.26	0.26	0.26
1	0.79	0.96	0.87	0.80	0.94	0.87	0.81	0.90	0.85	0.80	0.92	0.86	0.81	0.79	0.80
2	0.68	0.28	0.40	0.68	0.37	0.48	0.55	0.40	0.46	0.58	0.37	0.45	0.43	0.46	0.45
Accuracy	78%			78%			76%			77%			70%		

Table 3: Test results for the unbalanced dataset with default model parameters

We also implemented our models on a balanced dataset with tuned parameters. The results are shown in Table 4, indicating that the models perform well compared to when the models were implemented in an unbalanced dataset with default parameters. As shown in Table 4 among the five machine learning algorithms used in this research to predict wine quality, SVM shows the best performance. As mentioned in section 4.5 the KNN classifier in an unbalanced dataset tends to favour the majority class, this is evident in Table 3 as we can see that the precision, recall, and F1-score are high in the majority class (Class 1) as compared to other classes (Class 0 and Class 2). We can see that balancing the data and tuning your models increase the performance of your models as suggested by [22]. This is evident in Table 4 as we can see that the accuracy of our models increases as compared to when they were implemented our model in an unbalanced dataset with default parameters.

6.2 Feature importance

We also graphed the feature importance based on our best-performing machine learning model which is in this case the SVM. As we can see the feature importance graphed in Figure 8 alcohol is the most significant factor impacting wine quality, and this was also suggested by [28] that alcohol plays a crucial

Class	SVM			RF			KNN			GB			DT		
	Precision	Recall	F1 Score												
0	0.98	0.99	0.98	0.95	0.99	0.97	0.87	1.00	0.93	0.85	0.88	0.87	0.88	0.93	0.90
1	0.95	0.93	0.94	0.95	0.82	0.88	1.00	0.60	0.75	0.78	0.71	0.74	0.76	0.69	0.72
2	0.95	0.97	0.96	0.88	0.97	0.92	0.80	1.00	0.89	0.84	0.90	0.87	0.79	0.81	0.80
Accuracy	96%			92%			87%			83%			81%		

Table 4: Test results on the balanced dataset with tuned model parameters

role in determining wine quality. Looking at the feature importance graph it suggests that tuning features such as "alcohol", "sulphates", and "volatile acidity" may increase or decrease the wine scores. This information suggests that winemakers may benefit from tuning their models and playing around with the physio-chemical properties of wine.

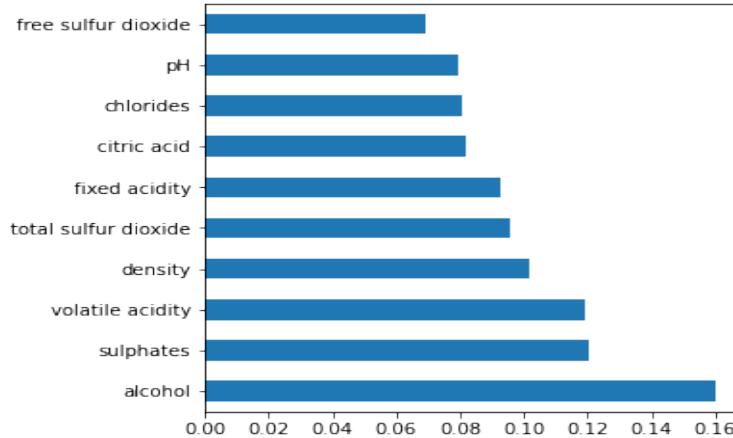


Fig. 8: Feature importance for our best performing model

6.3 Discussion

The objective of this research is to try and predict the quality of wine by analyzing the physico-chemical properties of the wine. It also looks into which features of the wine are most indicative of its quality. To achieve this goal we applied several machine learning algorithms as mentioned above, including Random Forest (RF), Support Vector Machine (SVM), Gradient Boosting (GB), K-Nearest

Neighbors (KNN), and Decision Tree (DT). We chose to use these machine learning algorithms because they are widely used algorithms for classification problems and are effective for wine quality prediction. We also dug deeper into the data and we found an interesting relationship between our feature variables and the target variable (Quality). We used the correlation coefficient matrix as shown in Figure 1 and features are ranked according to their correlation values. The results shown in Figure 1 suggest that features like "alcohol", "volatile acidity", and "sulphates" have a high correlation with quality while features like "free sulfur dioxide" and "residual sugar" do not. In Table 2, features are ranked according to their correlation values, and the first 10 features are selected during the models' ultimate implementation.

We assessed the effectiveness of the algorithms by analyzing metrics, including precision, recall, accuracy, and F_1 score as presented in both Table 3 and Table 4. We then evaluated the performance of our model by applying it to both the imbalanced dataset with default parameters and to a balanced dataset with fine-tuned parameters. The results of the analysis are presented in Table 3 and Table 4 respectively. From the performance results it is evident that the best outcome is achieved with a balanced dataset with fine-tuned parameters. As mentioned above it is evident that balancing the data and tuning your model parameters enhances the models' performance.

7 Conclusion

This study showed the importance of feature selection in understanding the impact of analytical tests on wine quality. The results of the feature selection process showed that some input variables such as Alcohol had a more significant influence on predicting wine quality than others such as Residual sugar. Applying machine learning algorithms in conjunction with the results of the feature selection process presented a valuable opportunity to improve the wine production process.

We employed five machine learning models, namely Decision Tree (DT), Random Forest (RF), Support Vector Machine (SVM), K-Nearest Neighbors (KNN), and Gradient Boosting (GB). The Support Vector Machine (SVM) outperformed the other models with an accuracy of 96%. Therefore, we conclude that not all features were equally important for predicting wine quality and that tuning your models and balancing the dataset improved the performance of the models. We also saw that in Figure 8 our feature importance graph suggested that tuning the models and playing around with physio-chemical properties such as "Alcohol" and "sulphate" may be beneficial in improving the prediction of wine quality.

Although this study presents promising results in predicting the wine quality using machine learning algorithms some limitations need to be addressed in

future work, such as the small size of the dataset and we did not use all the algorithms. In future work using larger and more diverse datasets could enhance the machine learning algorithm's performance. This will help the algorithms generalize better and reduce the risk of overfitting, thus improving the wine production process. For our study, we only used Five machine learning algorithms and there are still many other algorithms that could be explored in future work. We can evaluate the performance of the algorithms using different metrics and we can explore the impact of different preprocessing techniques such as different feature scaling techniques on the performance of the algorithms.

References

1. Balázs Ádám, Ágnes Molnár, Helga Bárdos, and Róza Ádány. Health impact assessment of quality wine production in hungary. *Health promotion international*, 24(4):383–393, 2009.
2. Adonis Saremi and Rohit Arora. The cardiovascular implications of alcohol and red wine. *American journal of therapeutics*, 15(3):265–277, 2008.
3. Meyer M. The subtle science of wine tasting. <https://winefolly.com/deep-dive/science-of-wine-tasting/>, 2019.
4. KR Dahal, JN Dahal, H Banjade, and S Gaire. Prediction of wine quality using machine learning algorithms. *Open Journal of Statistics*, 11(2):278–289, 2021.
5. Dheeru Dua and Casey Graff. UCI machine learning repository, 2017.
6. Yogesh Gupta. Selection of important features and predicting wine quality using machine learning techniques. *Procedia Computer Science*, 125:305–312, 2018.
7. Devika Pawar, Aakanksha Mahajan, Sachin Bhoithe, M Prasanna, and Kamalesh Kumar. Wine quality prediction using machine learning algorithms. *International Journal of Computer Applications Technology and Research*, 8(9):385–388, 2019.
8. Paulo Cortez, António Cerdeira, Fernando Almeida, Telmo Matos, and José Reis. Modeling wine preferences by data mining from physicochemical properties. *Decision support systems*, 47(4):547–553, 2009.
9. Kenji Kira and Larry A Rendell. A practical approach to feature selection. In *Machine learning proceedings 1992*, pages 249–256. Elsevier, 1992.
10. Yaqing Liu, Yong Mu, Keyu Chen, Yiming Li, and Jinghuan Guo. Daily activity feature selection in smart homes based on pearson correlation coefficient. *Neural Processing Letters*, 51(2):1771–1787, 2020.
11. Bin Zhang. Is the maximal margin hyperplane special in a feature space. *Hewlett-Packard Research Laboratories Palo Alto*, 2001.
12. LibreTexts. K nearest neighbors. [https://stats.libretexts.org/Bookshelves/Computing_and_Modeling/RTG%3A_Classification_Methods/3%3A_K-Nearest_Neighbors_\(KNN\)](https://stats.libretexts.org/Bookshelves/Computing_and_Modeling/RTG%3A_Classification_Methods/3%3A_K-Nearest_Neighbors_(KNN)), 2020. Accessed in 25 August 2022.
13. Nagesh Singh Chauhan. Random forest vs decision tree: Key differences. <https://www.kdnuggets.com/2022/02/random-forest-decision-tree-key-differences.html>, 2022. Accessed in 25 August 2022.
14. Wenliang Du and Zhijun Zhan. Building decision tree classifier on private data. 2002.
15. Kjell Johnson Max Kuhn. Comparison analysis of machine learning algorithms: Random forest and catboost. https://rstudio-pubs-static.s3.amazonaws.com/740098_4d48bd29722f402abf662dd33fc67794.html, 2020. Accessed in 25 August 2022.

16. Alexey Natekin and Alois Knoll. Gradient boosting machines, a tutorial. *Frontiers in neurorobotics*, 7:21, 2013.
17. Anshul Saini. Gradient boosting algorithm: A complete guide for beginners. <https://www.analyticsvidhya.com/blog/2021/09/gradient-boosting-algorithm-a-complete-guide-for-beginners/>, 2021. Accessed in 03 June 2022.
18. Wei Liu and Sanjay Chawla. Class confidence weighted k nn algorithms for imbalanced data sets. In *Advances in Knowledge Discovery and Data Mining: 15th Pacific-Asia Conference, PAKDD 2011, Shenzhen, China, May 24-27, 2011, Proceedings, Part II 15*, pages 345–356. Springer, 2011.
19. Mamad Tamamadin, Changkye Lee, Seong-Hoon Kee, and Jurng-Jae Yee. Regional typhoon track prediction using ensemble k-nearest neighbor machine learning in the gis environment. *Remote Sensing*, 14(21):5292, 2022.
20. Ashok Reddy. K nearest neighbors conceptual understanding and implementation in python. <https://www.citrusconsulting.com/k-nearest-neighbors-conceptual-understanding-and-implementation-in-python/>, 2020.
21. YuLong Ling, Xiao Zhang, and Yong Zhang. Improved knn algorithm based on probability and adaptive k value. In *2021 7th International Conference on Computing and Data Engineering*, pages 34–40, 2021.
22. Nitesh V Chawla. Data mining for imbalanced datasets: An overview. *Data mining and knowledge discovery handbook*, pages 875–886, 2009.
23. Mayuri S Shelke, Prashant R Deshmukh, and Vijaya K Shandilya. A review on imbalanced data handling using undersampling and oversampling technique. *Int. J. Recent Trends Eng. Res.*, 3(4):444–449, 2017.
24. Leila Zahedi, Farid Ghareh Mohammadi, Shabnam Rezapour, Matthew W Ohland, and M Hadi Amini. Search algorithms for automated hyper-parameter tuning. *arXiv preprint arXiv:2104.14677*, 2021.
25. Jia Wu, Xiu-Yun Chen, Hao Zhang, Li-Dong Xiong, Hang Lei, and Si-Hao Deng. Hyperparameter optimization for machine learning models based on bayesian optimization. *Journal of Electronic Science and Technology*, 17(1):26–40, 2019.
26. Kanika Kumar and Nelshan Mandan. Red wine quality prediction using machine learning techniques. In *2020 International Conference on Computer Communication and Informatics (ICCCI)*, pages 1–6. IEEE, 2020.
27. Xinjian Guo, Yilong Yin, Cailing Dong, Gongping Yang, and Guangtong Zhou. On the class imbalance problem. In *2008 Fourth international conference on natural computation*, volume 4, pages 192–201. IEEE, 2008.
28. Nuriel Shalom Mor. Wine quality and type prediction from physicochemical properties using neural networks for machine learning: A free software for winemakers and customers. 2022.