Wine Quality Classification Using Machine Learning Techniques

Arnab Bannerjee and Rudrajit Dey[*]

November 24, 2024

Abstract

This report presents a comprehensive analysis of wine quality classification using machine learning techniques. The study employs various classification algorithms to predict wine quality based on physicochemical properties. The project encompasses data preprocessing, feature selection, model implementation, and performance evaluation. Results indicate that ensemble methods, particularly the Gradient Boosting and Random Forest, achieve superior performance in predicting wine quality.

Contents

[*] Roll No.: B2430040 and B2430055 (respectively).

1 Introduction

1.1 Background

Wine certification is generally assessed by physicochemical and sensory tests. Physicochemical laboratory tests routinely used to characterize wine include determination of density, alcohol or pH values, while sensory tests rely mainly on human experts. Since human taste is the least understood of all senses, thus wine classification is a difficult task. Machine learning offers an opportunity to automate and standardize this evaluation process using chemical and physical properties of wines.

1.2 Motivation

The process of the physical assessment techniques is time-consuming and expensive. Thus, the wine industry could benefit significantly from automated quality assessment tools. By developing reliable prediction models based on objective measurements, we can help wineries maintain consistent quality standards and potentially reduce the cost and time associated with traditional evaluation methods.

1.3 Objectives

Our objectives for this project is as follows:

- Develop a binary classification model to predict wine quality based on physicochemical properties.
- Identify the most influential features affecting wine quality.
- Compare the performance of various machine learning algorithms.
- Provide insights that could be valuable for wine production optimization.

2 Literature Review

Machine learning applications in wine quality assessment have gained significant attention in recent years. Previous studies have employed various approaches:

- Linear modeling techniques for wine quality prediction.
- Support Vector Machines (SVM) for classification tasks.
- Ensemble methods like Random Forests for their robustness.
- Neural Networks for complex pattern recognition in wine characteristics.

The authors of one study employed 11 physiochemical characteristics to create machine learning models for predicting red wine quality [1], using data mining methods to extract information on red wine quality from the UCL machine learning repository. According to the authors, the SVM model had a 67.25 percent accuracy, while Random Forest and Nave Bayes had 65.83% and 55.91% accuracy respectively. In [2] did a comparison of several classification algorithms and explained why some of the classification algorithms produce more accurate findings as compared to others. In a study conducted by Lee and group [3] a decision tree classifier is utilised to assess wine quality and in [4], a machine learning model based on RF and KNN algorithm is built to determine if the wine is good, average, or terrible.

The field has shown that both chemical composition and physical properties play crucial roles in determining wine quality, making it an ideal candidate for machine learning applications.

3 Dataset Description

3.1 Source

The dataset used in this study is the ”Wine Quality Red” dataset, which contains information about red variants of Portuguese ”Vinho Verde” wine. The classes are ordered and not balanced (e.g. there are many more normal wines than excellent or poor ones).

3.2 Features

The dataset includes 11 physicochemical properties:

- Fixed acidity
- Volatile acidity
- Citric acid
- Residual sugar
- Chlorides
- Free sulfur dioxide
- Total sulfur dioxide
- Density
- pH
- Sulphates
- Alcohol

3.3 Target Variable

The original quality scores (ranging from 0-10) were transformed into a binary classification problem:

- $0:$ Lower quality (score < 6)
- $1:$ Higher quality (score ≥ 6)

4 Data Preprocessing

4.1 Initial Data Assessment

4.1.1 Data Loading and Inspection

- Dataset Shape: 1599 rows × 12 columns
- Initial Features: 11 physicochemical properties + 1 target variable
- Data Type Analysis: All features were numeric (float64/int64)
- Memory Usage Assessment: Optimized data types for efficient processing

4.1.2 Missing Value Analysis

- No missing values detected in the dataset
- No imputation strategies needed

4.1.3 Statistical Summary

Key Findings:

- Alcohol content range: 8.4% to 14.9%
- pH range: 2.74 to 4.01
- Fixed acidity range: 4.6 to 15.9 g/dm^3
- Identified potential outliers in several features

4.2 Target Variable Transformation

4.2.1 Original Quality Distribution

- Original Scale : 0 - 10
- Actual range in dataset: 3-8
- Distribution analysis showed imbalance

4.2.2 Binary Class Conversion

- Threshold selection: $\geq$6 for high quality
- Class Distribution:
  - Class 0 (lower quality): 744 samples (46.53%)
  - Class 1 (higher quality): 855 samples (53.47%)

4.3 Feature Analysis and Engineering

4.3.1 Distribution Analysis

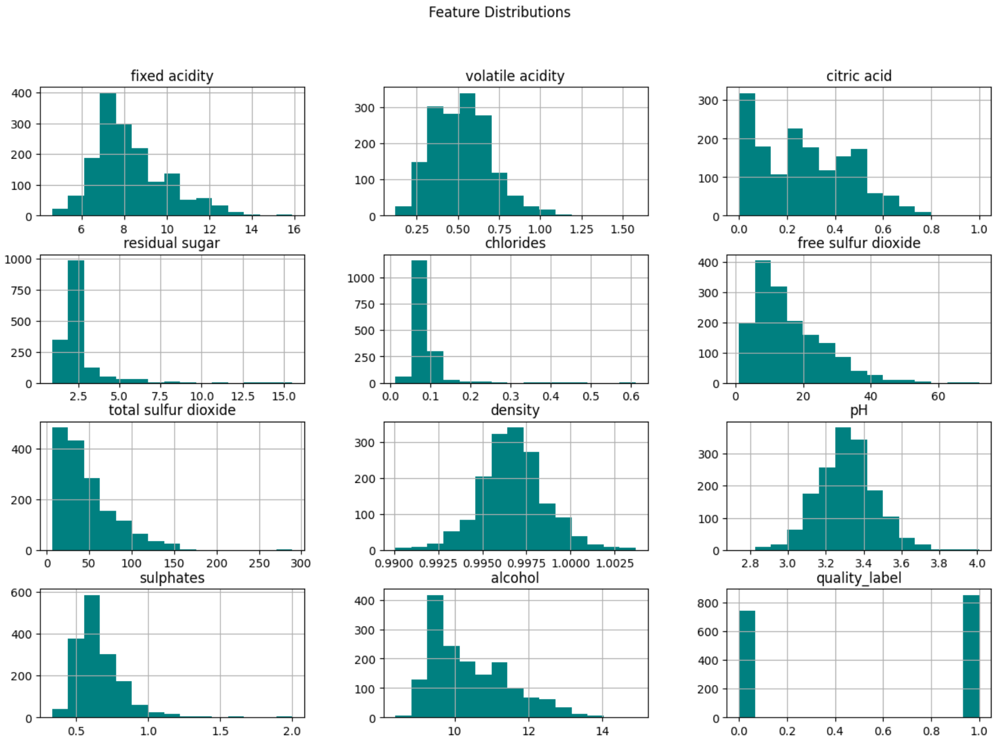


Fig 1 : Original Feature Distribution chart

- Fixed Acidity: Right-skewed
- Volatile Acidity: Approximately normal
- Citric Acid: Bimodal distribution
- Residual Sugar: Highly right-skewed, outlier treatment needed
- Chlorides: Right-skewed, outlier treatment needed
- Sulfur Dioxide (Free): Right-skewed
- Sulfur Dioxide (Total): Right-skewed
- Density: Normal distribution
- pH: Normal distribution
- Sulphates: Right-skewed
- Alcohol: Right-skewed

4.3.2 Outlier Detection and Treatment

In order to handle the outliers, we first find out the lower and upper bounds for each feature and then delete the rows with those outliers. We however reduce our no. of instances from 1599 to now 1194.

4.3.3 Feature Transformation

Standard Scaling is applied for feature transformation, since after outlier removal, skewness for each feature distribution is less than 1. Hence, no log transformation was necessary.

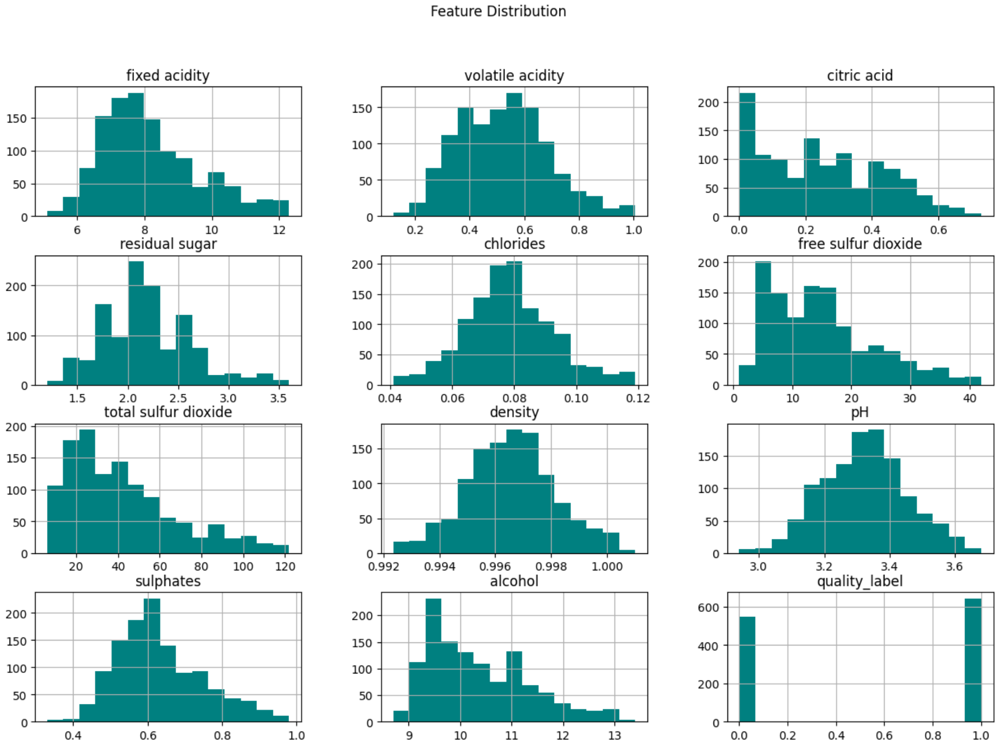
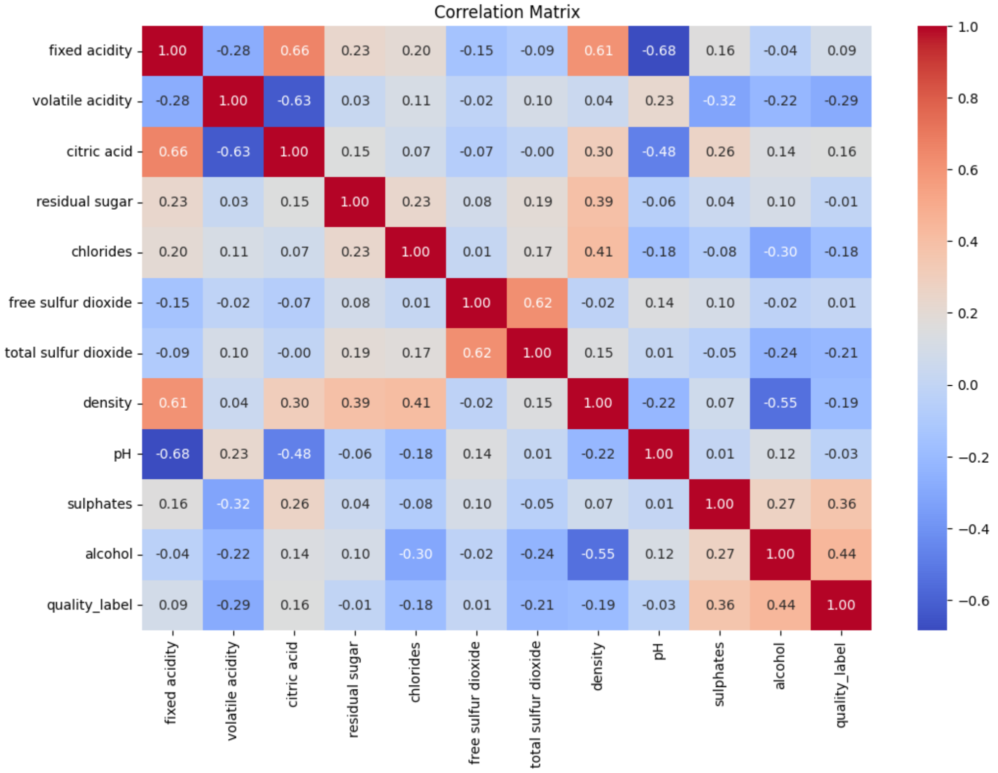


Fig 2 : Feature Distribution chart after outlier handling

4.4 Feature Selection

4.4.1 Correlation Analysis



	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	pH	sulphates	alcohol	quality_label
fixed acidity	1.00	-0.28	0.66	0.23	0.20	-0.15	-0.09	0.61	-0.68	0.16	-0.04	0.09
volatile acidity	-0.28	1.00	-0.63	0.03	0.11	-0.02	0.10	0.04	0.23	-0.32	-0.22	-0.29
citric acid	0.66	-0.63	1.00	0.15	0.07	-0.07	-0.00	0.30	-0.48	0.26	0.14	0.16
residual sugar	0.23	0.03	0.15	1.00	0.23	0.08	0.19	0.39	-0.06	0.04	0.10	-0.01
chlorides	0.20	0.11	0.07	0.23	1.00	0.01	0.17	0.41	-0.18	-0.08	-0.30	-0.18
free sulfur dioxide	-0.15	-0.02	-0.07	0.08	0.01	1.00	0.62	-0.02	0.14	0.10	-0.02	0.01
total sulfur dioxide	-0.09	0.10	-0.00	0.19	0.17	0.62	1.00	0.15	0.01	-0.05	-0.24	-0.21
density	0.61	0.04	0.30	0.39	0.41	-0.02	0.15	1.00	-0.22	0.07	-0.55	-0.19
pH	-0.68	0.23	-0.48	-0.06	-0.18	0.14	0.01	-0.22	1.00	0.01	0.12	-0.03
sulphates	0.16	-0.32	0.26	0.04	-0.08	0.10	-0.05	0.07	0.01	1.00	0.27	0.36
alcohol	-0.04	-0.22	0.14	0.10	-0.30	-0.02	-0.24	-0.55	0.12	0.27	1.00	0.44
quality_label	0.09	-0.29	0.16	-0.01	-0.18	0.01	-0.21	-0.19	-0.03	0.36	0.44	1.00

Fig 3 : Correlation Matrix after Feature Transformation and Outlier Treatment

- Identified relatively high correlated features ($|r| > 0.6$):
  - Free/Total Sulfur Dioxide
  - Fixed Acidity/Density
  - Fixed Acidity/pH
- Treatment: Kept first two features due to different chemical significance.

4.4.2 Random Forest Feature Importance and Implementation

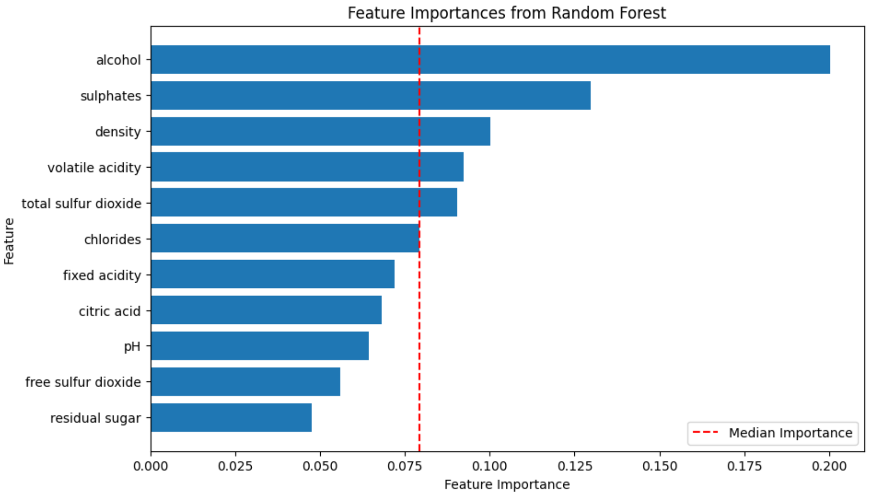


Fig 4 : Feature importance and median

Selected features according to importance:

- Alcohol (0.200)
- Sulphates (0.130)
- Density (0.100)
- Volatile Acidity (0.092)
- Total Sulfur Dioxide (0.090)

5 Methodology

5.1 Algorithms Used

Multiple classification algorithms are used:

- Logistic Regression
- Decision Tree
- Random Forest
- Support Vector Machine
- K-Nearest Neighbours
- Naive Bayes
- Gradient Boosting
- Stacking Classifier

5.2 Justification

- Logistic Regression: Provides a baseline and handles binary classification well
- Decision Tree: Captures non-linear relationships and feature interactions
- Random Forest: Reduces overfitting and handles feature importance well
- SVC: Effective for high-dimensional spaces and non-linear classification
- KNN: Useful for capturing local patterns in the data
- Naive Bayes: Efficient and works well with small datasets
- Gradient Boosting: Powerful for improving prediction accuracy
- Stacking Classifier: Combines multiple models to improve overall performance

5.3 Model Architecture

The architecture for stacking classifier is as follows:

- Estimators:
  - Logistic Regression
  - Decision Tree
  - Support Vector Classifier
- Final Estimator : Random Forest Classifier
- Hyperparameters: 5 cross-validation folds were used

6 Implementation

6.1 Tools and Libraries

The following primary libraries were used:

- pandas for data manipulation
- numpy for numerical operations
- matplotlib and seaborn for visualization
- scikit-learn for machine learning implementations
- scipy for statistical inferences

6.2 Parameters

- Hyperparameter settings maintained consistent across models for fair comparison.
- During feature importance, for random forest training, no. of estimators = 100.

6.3 Training Process/ Model Pipeline

- StandardScaler() for feature normalization.
- KFold cross-validation (5 folds) in stacking for robust evaluation.
- Pipeline implementation for consistent preprocessing.

7 Results and Discussion

In this section, we evaluate the performance of multiple classification algorithms applied to the dataset. Metrics used to compare the models:

- precision
- recall
- f1 score
- accuracy

Each algorithm was trained and tested on the dataset, and the results for the test set are presented below:

- Logistic Regression achieved an accuracy of 74%, indicating reasonable predictive performance. For Class 0, the precision was 0.71 with a recall of 0.75, while Class 1 achieved a higher precision of 0.78 but slightly lower recall of 0.74. The weighted averages for precision, recall, and f1-score were around 0.74-0.75, suggesting a balanced performance for both classes. However, the model's performance can be improved, especially in distinguishing between the two classes more effectively.

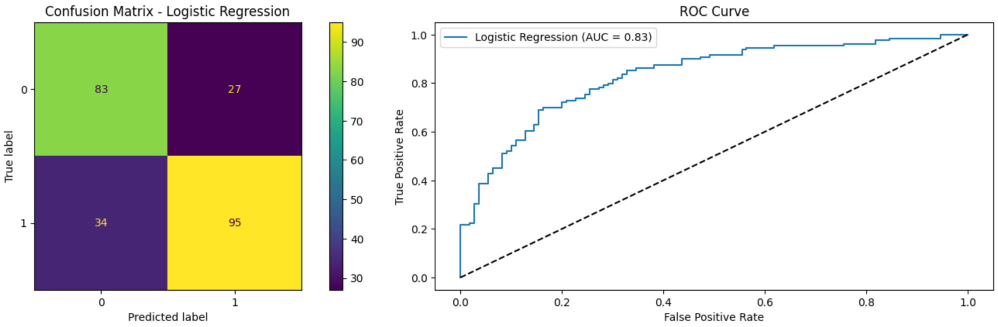


Fig 5 : Confusion matrix and ROC curve of Logistic Regression

- The Decision Tree classifier achieved an accuracy of 69%, the lowest among the tested models. Class 0 obtained a precision of 0.66 and a recall of 0.68, while Class 1 achieved slightly better precision (0.72) and similar recall (0.71). The macro average and weighted average metrics were consistently 0.69, indicating a slight bias toward Class 1 predictions. The lower accuracy and f1-scores suggest that the Decision Tree struggled to generalize well for this dataset, potentially due to overfitting.

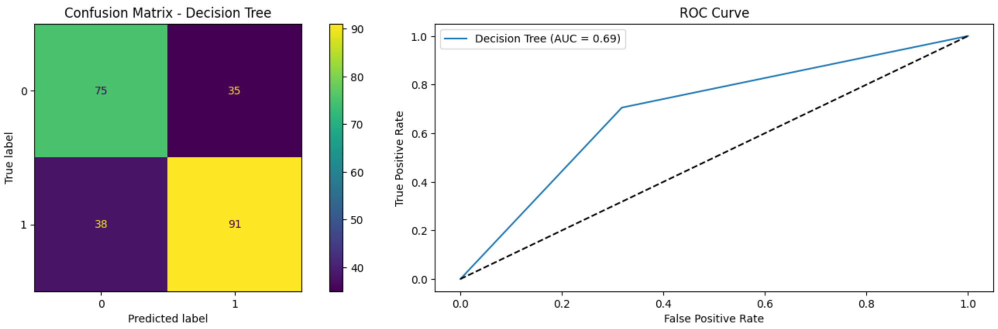


Fig 6 : Confusion matrix and ROC curve of Decision Tree

- Random Forest emerged as one of the best-performing models, achieving an accuracy of 82%. Class 0 achieved a precision of 0.82 and a recall of 0.79, while Class 1 demonstrated even higher performance with a precision of 0.83 and a recall of 0.85. The weighted averages for precision, recall, and f1-score were consistently 0.82. These results highlight Random Forest's ability to effectively capture complex patterns in the data and handle class imbalances.

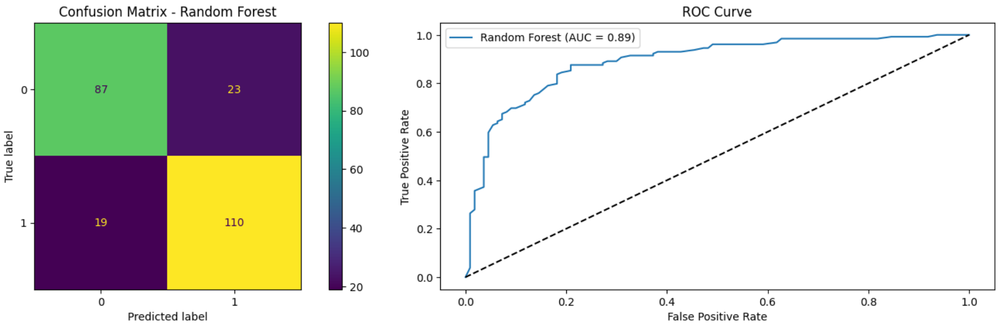


Fig 7 : Confusion matrix and ROC curve of Random Forest

- The Support Vector Classifier achieved an accuracy of 76%, reflecting strong predictive capabilities. Class 0 had a precision of 0.71 and a high recall of 0.81, while Class 1 achieved a higher precision of 0.82 but a slightly lower recall of 0.72. The macro and weighted average metrics ranged from 0.76 to 0.77, indicating a good balance between the two classes. This performance suggests that SVC effectively separates classes but may require parameter tuning for further improvements.

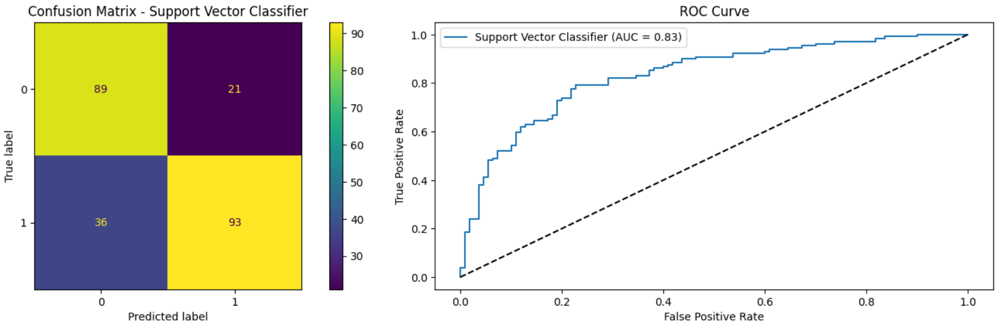


Fig 8 : Confusion matrix and ROC curve of Support Vector Classifier

- The K-Nearest Neighbors classifier achieved an accuracy of 73%, with moderate precision, recall, and f1-scores for both classes. Class 0 had a precision of 0.71 and a recall of 0.71, while Class 1 achieved a precision of 0.75 and a recall of 0.75. The macro and weighted averages were both 0.73. These results indicate that while KNN performs reasonably well, it lacks the sophistication needed to match ensemble models like Random Forest or Gradient Boosting.

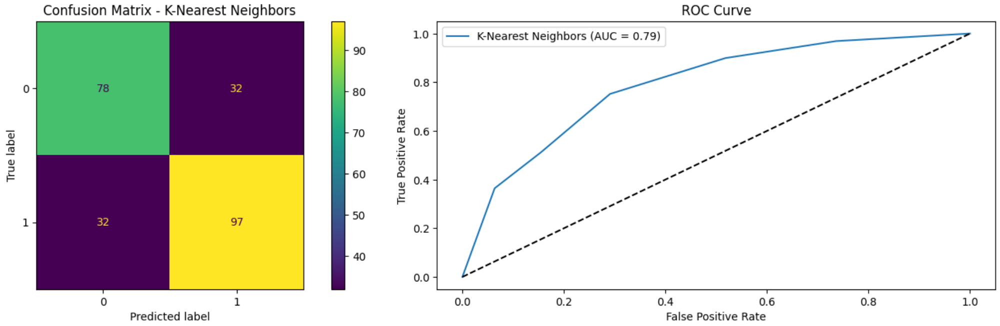


Fig 9 : Confusion matrix and ROC curve of K-Nearest Neighbours

- Naive Bayes achieved an accuracy of 74%, with performance comparable to Logistic Regression. Class 0 had a precision of 0.69 and a high recall of 0.81, while Class 1 exhibited a higher precision of 0.81 but a lower recall of 0.69. The macro and weighted averages were around 0.74-0.75. This performance reflects the simplicity and speed of Naive Bayes but also its limitations in capturing inter-feature relationships due to the independence assumption.

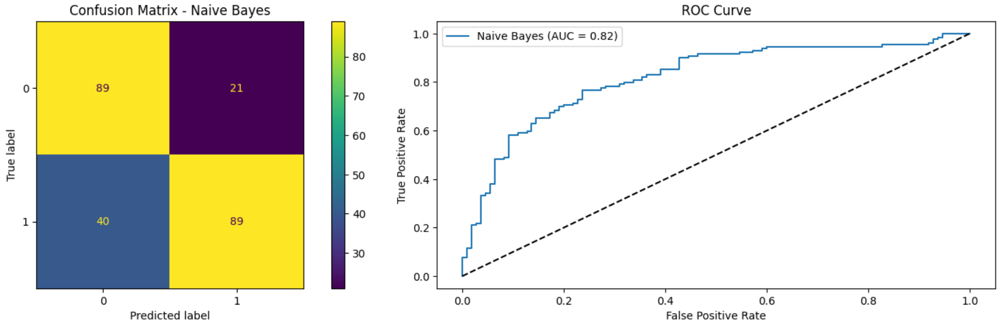


Fig 10 : Confusion matrix and ROC curve of Naive Bayes classifier

- Gradient Boosting achieved an accuracy of 77%, demonstrating strong and consistent performance. Class 0 had a precision of 0.74 and a recall of 0.75, while Class 1 achieved slightly better precision (0.79) and recall (0.78). The macro and weighted averages were consistently 0.76-0.77. These results highlight Gradient Boosting's ability to effectively balance precision and recall, making it a reliable choice for this dataset.

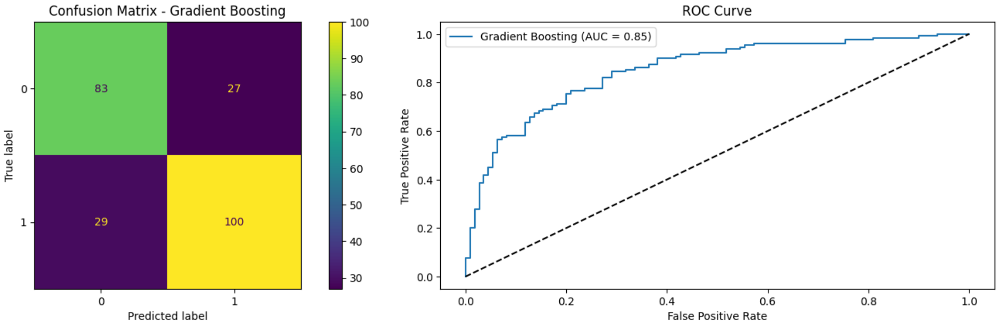


Fig 11 : Confusion matrix and ROC curve of Gradient Boosting model

- The Stacking Classifier achieved an accuracy of 77%, matching Gradient Boosting's performance. For Class 0, precision and recall were 0.74 and 0.75, respectively, while Class 1 achieved a precision of 0.79 and a recall of 0.78. The macro and weighted averages were consistently 0.76-0.77. These results demonstrate that the ensemble approach of Stacking combines the strengths of multiple base models, yielding robust and balanced predictions.

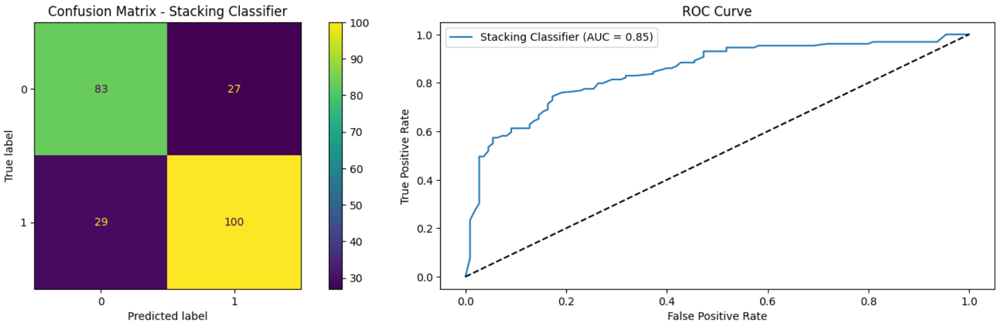


Fig 12 : Confusion matrix and ROC curve of Stacking Classifier

The performance of the classifiers varied significantly:

- Best Performers:
  - Random Forest stood out as the best model with an accuracy of 82% and consistently high scores across all metrics.
  - Gradient Boosting and Stacking Classifier followed closely, achieving 77% accuracy with balanced precision, recall, and f1-scores.
- Moderate Performers:
  - Support Vector Classifier performed well with an accuracy of 76%, particularly excelling in separating the two classes effectively.
  - Logistic Regression, Naive Bayes, and KNN achieved accuracies in the range of 73-74%, with reasonable but slightly lower performance.
- Lowest Performer:

– Decision Tree had the weakest performance, with an accuracy of 69%, indicating overfitting or limited generalization capacity.

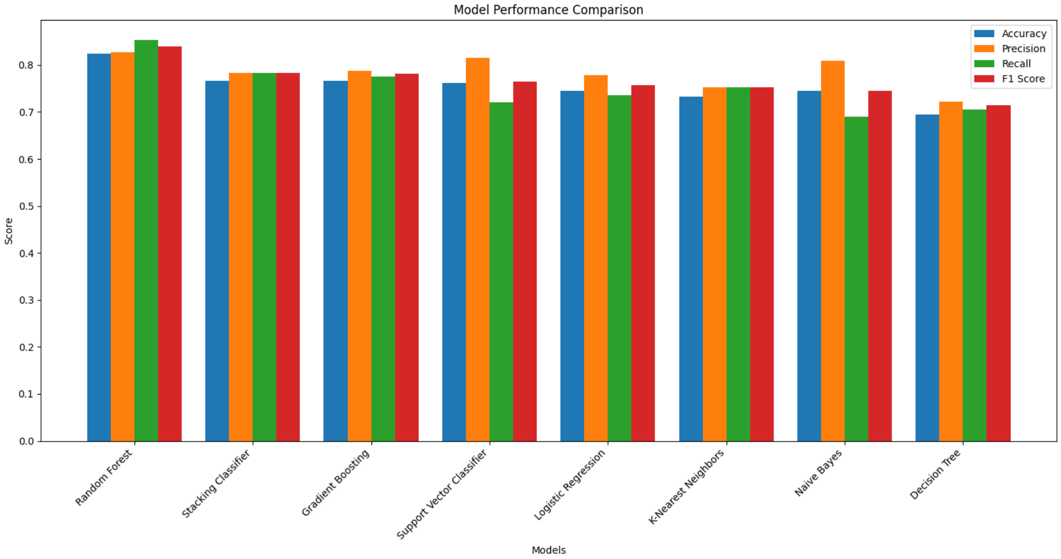


Fig 11 : Model Comparison chart

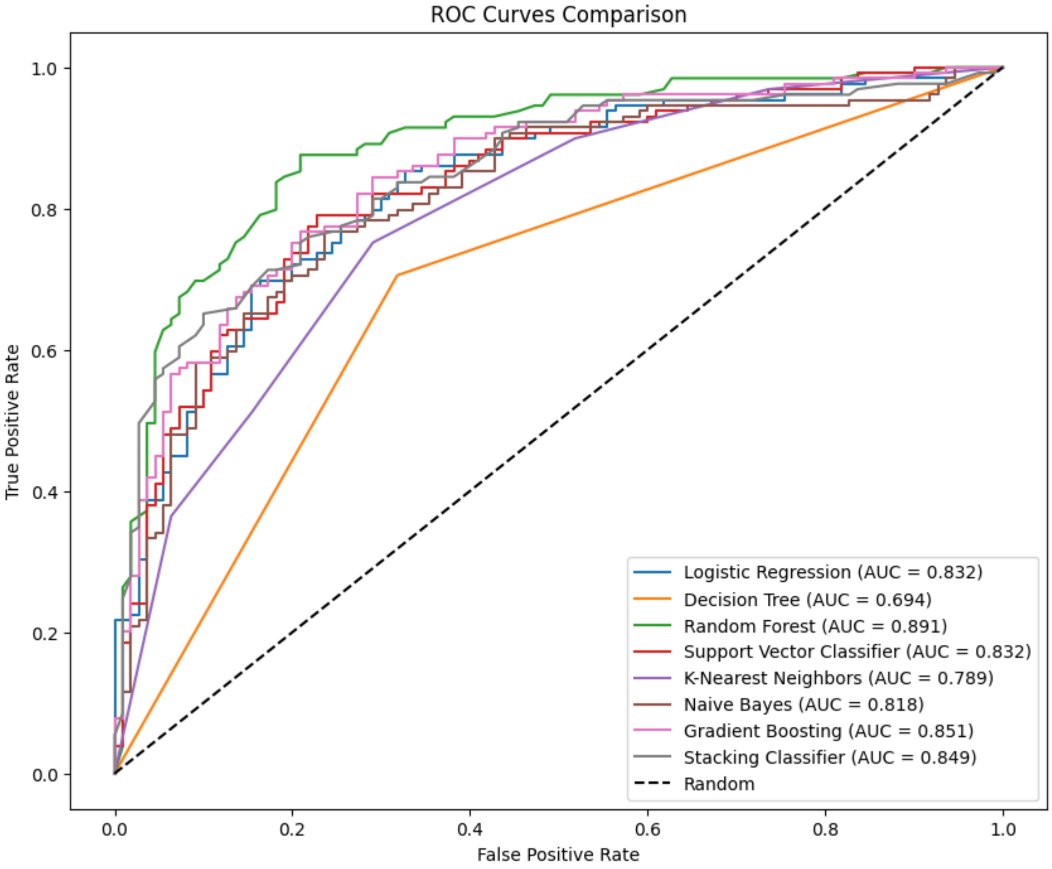


Fig 12 : ROC-AUC cumulative graph

However, we can see from the following chart that K-Nearest Neighbours and Naive Bayes consumed much less time than models like Stacking and Random Forest. The primary reason being the increased complexity of the latter models compared to the former as well as their memory utilisation.

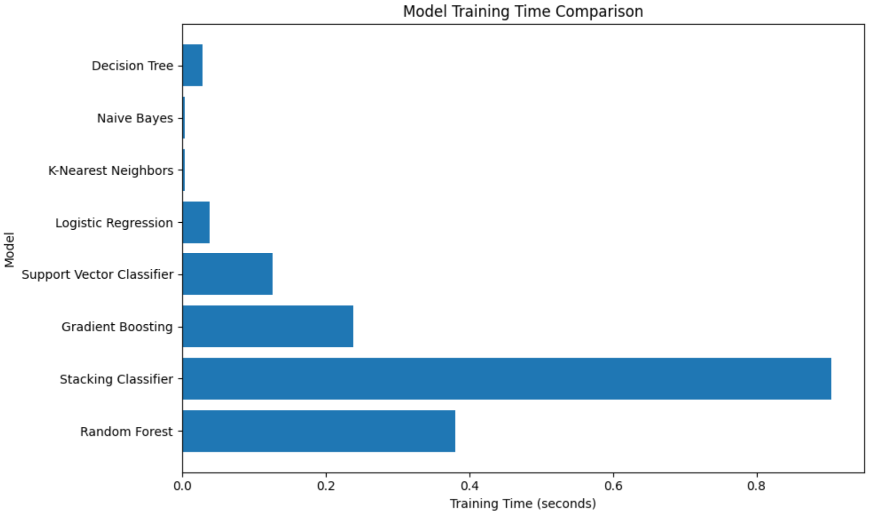


Fig 13 : Training time chart

8 Conclusion

We list our key findings from this comprehensive classification project:

- Feature importance analysis revealed alcohol content, sulphates, and density as the most significant predictors of wine quality.
- Ensemble methods (Random Forest, Gradient Boosting, Stacking) consistently outperformed single models, even though training time was relatively high.
- The correlation matrix showed significant relationships between certain features, suggesting potential for feature engineering.

Some limitations of our project:

- Binary classification may oversimplify the complex nature of wine quality.
- The dataset is specific to Portuguese red wines, potentially limiting generalizability.
- Some features show high correlation, which might affect model interpretation.

Thus, the project successfully developed a machine learning pipeline for wine quality classification. The ensemble methods proved most effective, suggesting that wine quality prediction benefits from combining multiple modeling approaches. The identification of key chemical properties influencing wine quality provides valuable insights for wine producers.

Going forward we hope to utilize and expand our knowledge in order to perform more rigorous modeling and also delve into the following:

- Experiment with deep learning approaches
- Include more features such as grape variety and aging time/
- Develop a multi-class classification model
- Create an interactive tool for real-time quality prediction

References

[1] Dahal, K. , Dahal, J. , Banjade, H. and Gaire, S. (2021) Prediction of Wine Quality Using Machine Learning Algorithms. Open Journal of Statistics, 11, 278-289. doi: 10.4236/ojs.2021.112015.

[2] S. Kumar, K. Agrawal and N. Mandan, ”Red Wine Quality Prediction Using Machine Learning Techniques,” 2020 International Conference on Computer Communication and Informatics (ICCCI), Coimbatore, India, 2020, pp. 1-6, doi: 10.1109/ICCCI48352.2020.9104095. keywords: processes;data extraction;Naïve Bayes;SVM;Random Forest;quality,

[3] S. Lee, J. Park and K. Kang, ”Assessing wine quality using a decision tree,” 2015 IEEE International Symposium on Systems Engineering (ISSE), Rome, Italy, 2015, pp. 176-178, doi: 10.1109/SysEng.2015.7302752. keywords: Decision trees;Accuracy;Data mining;Sulfur;Conferences;Predictive models;Data models.

[4] Mahima, Gupta, U., Patidar, Y., Agarwal, A., Singh, K.P. (2020). Wine Quality Analysis Using Machine Learning Algorithms. In: Sharma, D.K., Balas, V.E., Son, L.H., Sharma, R., Cengiz, K. (eds) Micro-Electronics and Telecommunication Engineering. Lecture Notes in Networks and Systems, vol 106. Springer, Singapore.