Red Wine Quality Analysis

https://github.com/dhakaadi/cmse802_project.git

Background and Motivation:

The problem at hand involves classifying wines as either good (1) or bad (0) based on various features such as alcohol content, acidity, and other chemical properties. This type of classification problem has significant applications in the wine industry, where producers or consumers might want to predict the quality of wine without relying solely on subjective tastes. With the growing importance of predictive analytics, this task serves as a valuable case study for understanding how different machine learning models perform on a binary classification problem with structured data. The key question we set out to answer is: Which machine learning model can most accurately predict whether a wine is good or bad based on its chemical features?

Description of repository structure

cmse802_project/

├─── Data/

│ ├─── winequality-red.csv

├─── Notebook/

│ ├─── cmse802_project.ipynb

├─── Result/

│ ├─── Figure/
│ │ ├───Alcohol Content by Wine Quality.jpg

│ │ ├───Box Plots for Each Feature in the Dataset.jpg

│ │ ├───Chloride Levels by Wine Quality.jpg

│ │ ├───Citric Acid Levels by Wine Quality.jpg

│ │ ├───Confusion Matrix for Decision Tree.jpg

│ │ ├───Confusion Matrix for Logistic Regression.jpg

│ │ ├───Correlation Heatmap of Wine Features.jpg

Explanation of key files and directories

- **Data**: Contains the wine quality dataset (winequality-red.csv)

- **Notebook**: Contains Python code for data analysis and model building

- **Results/Figures**: Stores visualizations, charts, and model performance metrics

- **Report**: Contains .pdf report file.

- **README.md** contains the repository name, brief description and some basic instructions for setting up and running your code with requirements.

- **requirements.txt** contains required libraries

List of dependencies and setup instructions

The following dependencies are required:
- numpy
- pandas
- matplotlib
- seaborn
- scikit-learn

To set up the environment, use the following commands:
`pip install -r requirements.txt`

Methodology

To answer this question, we used a dataset containing chemical properties of wines, with the goal of predicting whether a wine is classified as good or bad. We employed a variety of machine learning models to explore how they handle this classification task. Specifically, we applied the following models:

- **Logistic Regression**: A simple, interpretable model suitable for binary classification.
- **Decision Tree**: A non-linear model that splits the dataset based on feature values, ideal for capturing complex relationships.
- **Random Forest**: An ensemble method based on multiple decision trees, which helps mitigate overfitting.
- **K-Nearest Neighbors (k-NN)**: A non-parametric model that makes predictions based on the proximity of data points in feature space.
- **Support Vector Machine (SVM)**: A powerful model that finds the hyperplane that best separates the classes in high-dimensional space.
- **Gaussian Naive Bayes**: A probabilistic model that assumes feature independence and works well with continuous data.
- **Multi-Layer Perceptron (ANN)**: A deep learning model that is capable of modeling complex, non-linear relationships.
- **XGBoost**: A gradient boosting method that combines weak learners (decision trees) to improve predictive performance.

We evaluated the performance of these models using the following metrics:

- **Accuracy**: The proportion of correctly classified instances.
- **Precision**: The proportion of true positives among predicted positives.
- **Recall**: The proportion of true positives among actual positives.
- **F1-Score**: The harmonic mean of precision and recall, which balances the trade-off between them.
- **Confusion Matrix**: To visualize the distribution of true positives, false positives, true negatives, and false negatives.

To ensure reliable results, the data was split into training and testing sets, and the models were evaluated using the test data. Visualizations such as confusion matrices and classification reports were used to support our evaluation.

Justification for chosen methods:

Logistic Regression: Logistic Regression was chosen due to its simplicity and interpretability. It is effective for binary classification tasks and provides insights into the relationship between features and the target variable. Though it doesn't perform as well as some other models in this case, its results offer a good baseline for comparison and are particularly useful when the goal is to model the probability of an event happening (e.g., predicting a good or bad wine based on its features).

Decision Tree: A Decision Tree was chosen for its ease of interpretability and its ability to capture non-linear relationships in the data. It splits the data into distinct regions based on feature values, which makes it easier to visualize the decision-making process. However, it can easily overfit, which is why further ensemble methods like Random Forest are considered for improving performance.

Random Forest: Random Forest was selected due to its ability to handle complex datasets and its robustness against overfitting. By averaging multiple Decision Trees, it improves classification accuracy and generalization. It is particularly well-suited for this classification problem, as it can handle both linear and non-linear relationships between the features and the target variable. Random Forest's superior performance in this case justifies its use.

K-Nearest Neighbors (k-NN): K-NN was chosen for its simplicity and non-parametric nature, meaning it doesn't make assumptions about the underlying data distribution. It is particularly useful in datasets where the relationships between the features and target are highly non-linear. However, it requires tuning the number of neighbors and may struggle with large datasets, as the computation cost increases.

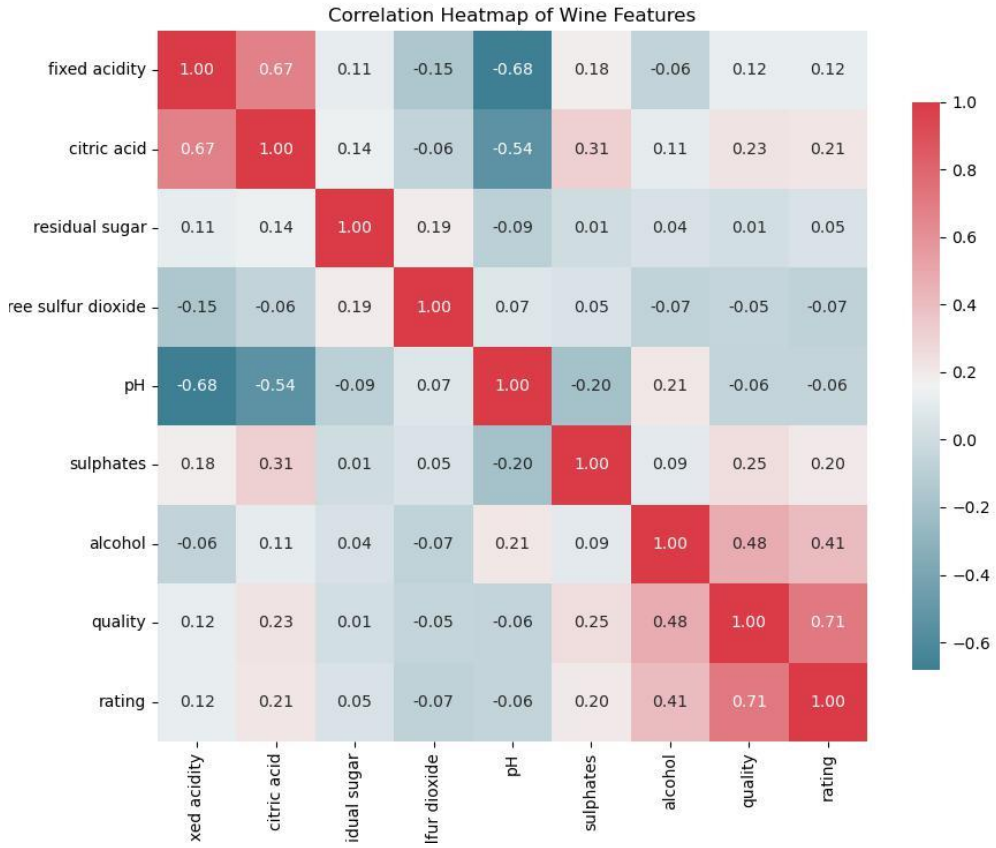
Support Vector Machine (SVM): SVM was selected due to its ability to find a hyperplane that best separates the classes in high-dimensional spaces, which makes it effective for classification tasks where classes are not linearly separable. However, in this case, its poor performance in predicting Class 1 suggests that it might not be the best fit for the dataset, especially when the data is imbalanced.

Gaussian Naive Bayes (GaussianNB): Gaussian Naive Bayes was chosen for its simplicity, efficiency, and effectiveness in handling categorical or continuous features when the assumption of feature independence holds. It works well when the classes have Gaussian-distributed features. While it performed reasonably well in terms of precision for Class 0, its inability to perform well for Class 1 makes it less suitable for this dataset.

Multi-Layer Perceptron (ANN): A Multi-Layer Perceptron (ANN) was chosen for its flexibility and capability to model complex, non-linear relationships in the data. It can capture intricate patterns and interactions between features, making it suitable for complex classification tasks. While it performed reasonably well, it struggles with the imbalanced dataset, especially in predicting Class 1.

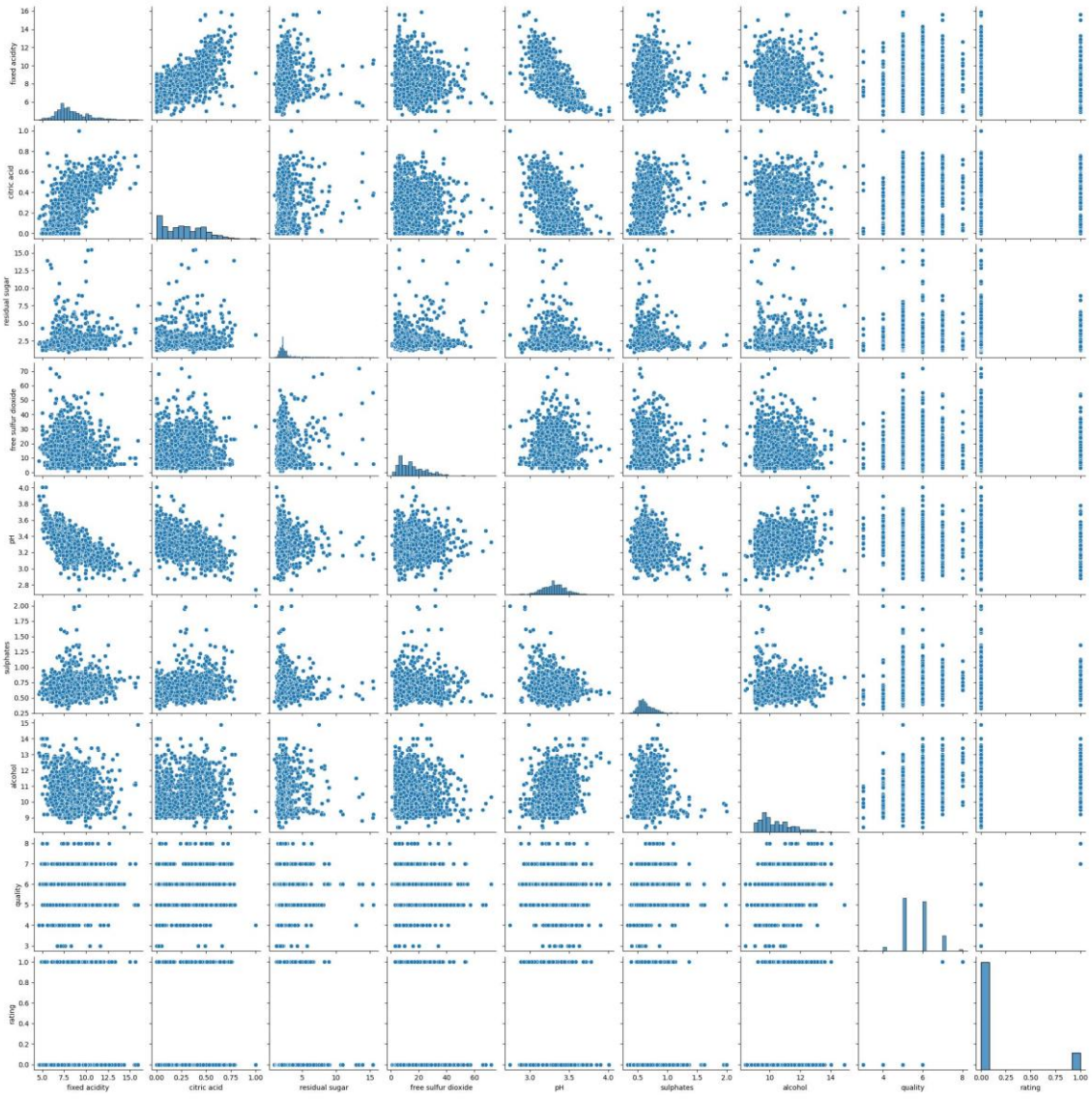
XGBoost: XGBoost was selected for its superior performance in many machine learning competitions. It is an ensemble method based on boosting, which builds weak learners (decision trees) sequentially to correct the errors made by previous models. XGBoost is highly effective in handling imbalanced data and provides high predictive accuracy, particularly when tuned properly. Its strong performance for Class 1 (bad wine) justifies its use despite its lower recall for Class 0 compared to Random Forest.

Relevant data visualizations:



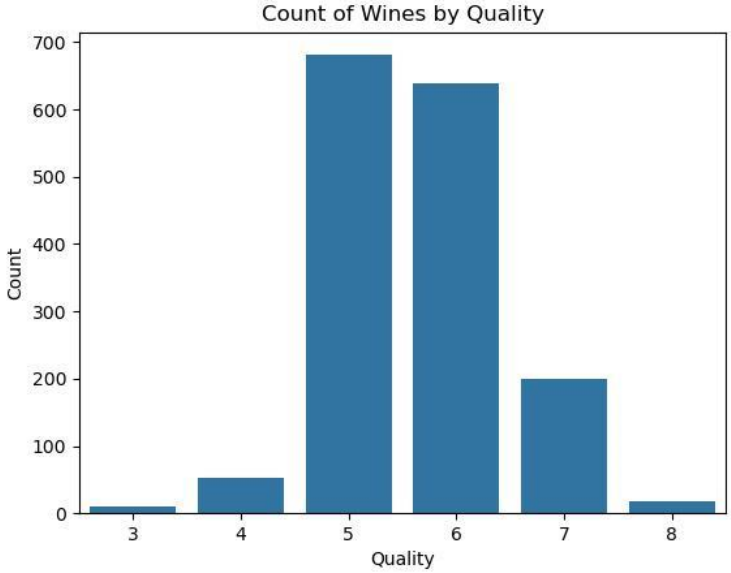
Strongest predictor of quality*: alcohol has the highest positive correlation with wine quality.*

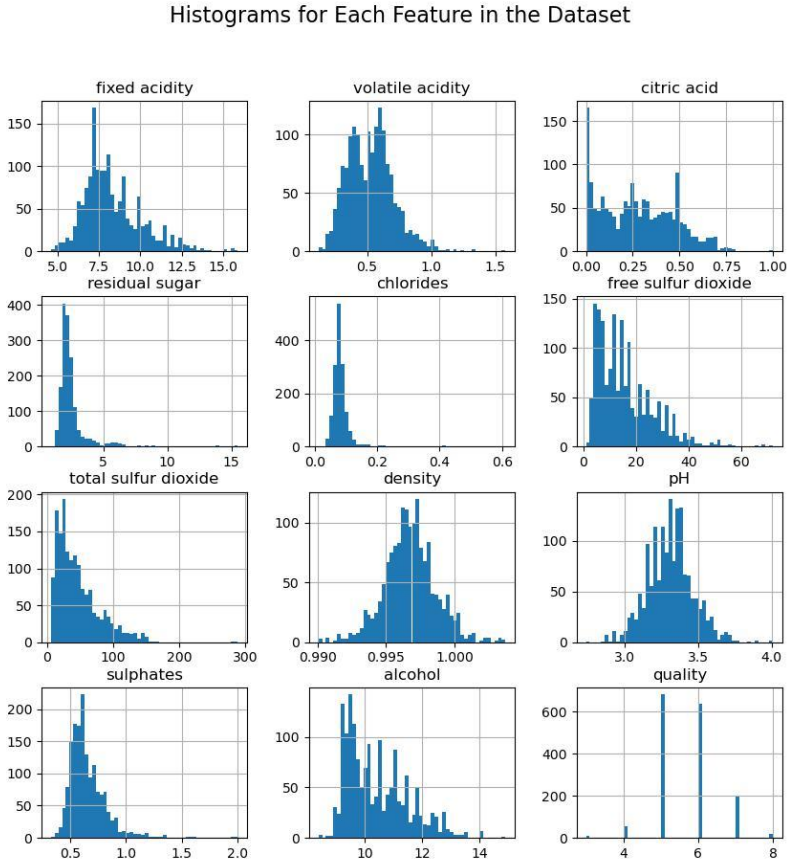
Negative predictors of quality*: volatile acidity has the most significant negative impact on quality, followed by chlorides, density, and total sulfur dioxide.*



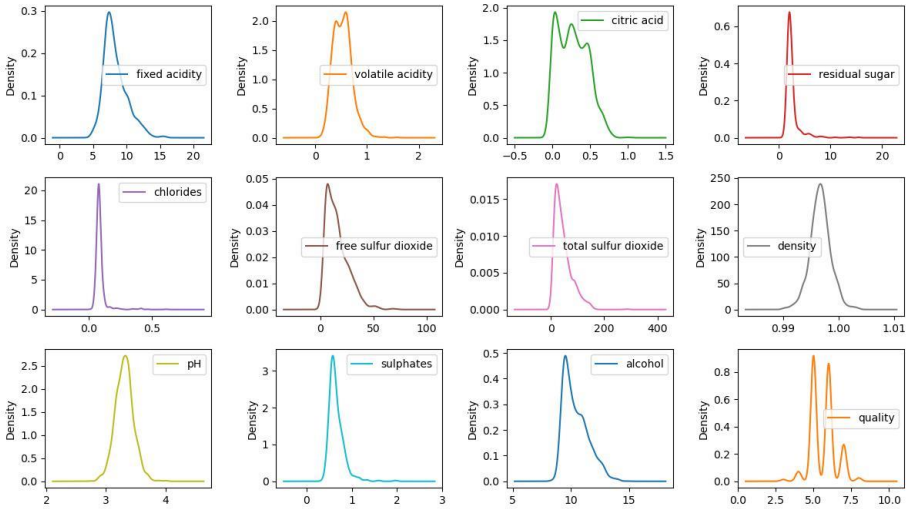
- *Alcohol, sulphates, and citric acid levels show significant variations across wine ratings. These features are likely to be important for predicting wine quality.*
- *Strongly correlated items are:*
    1. *fixed acidity and citric acid.*
    2. *free sulphur dioxide and total sulphor dioxide.*
    3. *fixed acidity and density.*
    4. *alcohol and quality*
- *Volatile acidity, total sulfur dioxide, chlorides, density are very less related to the dependent variable thus dropped*

Various plots such as box plots, violin plots, and heatmaps were created to explore data patterns.

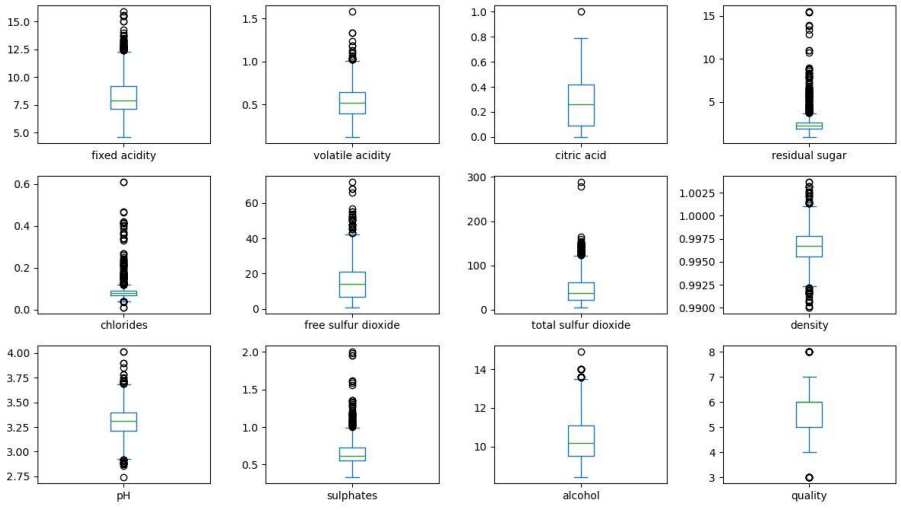


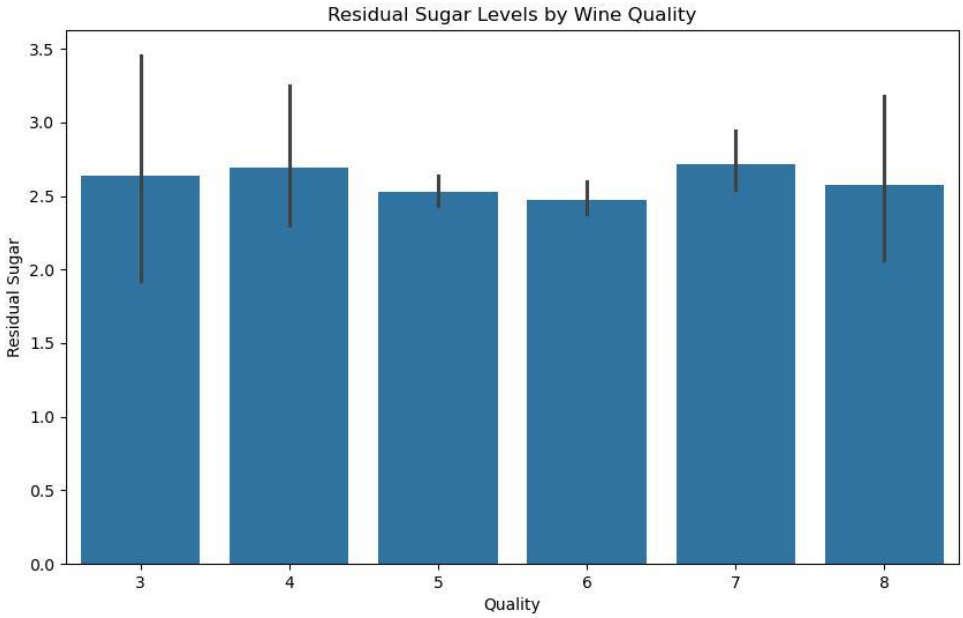


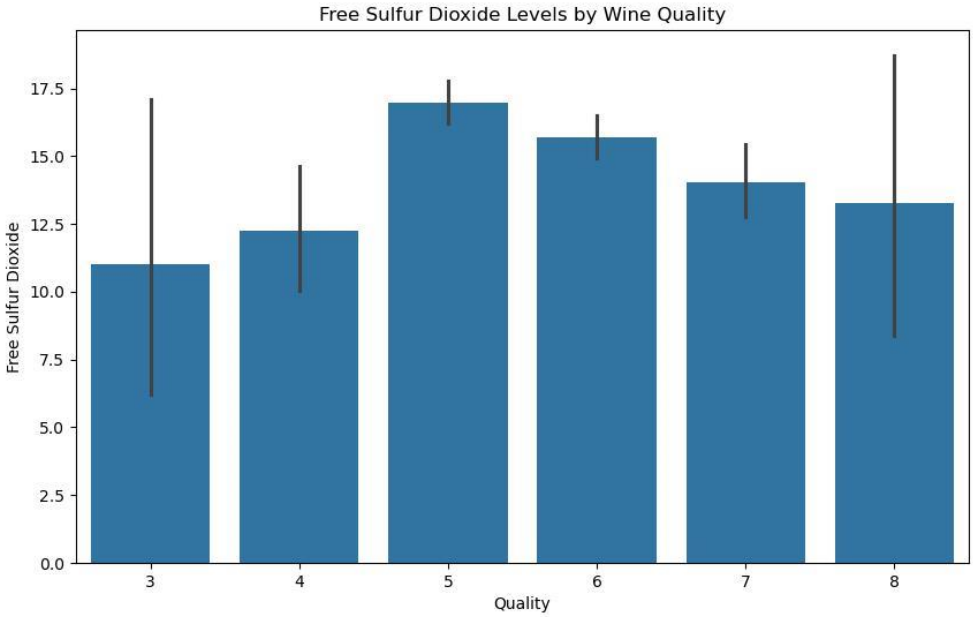
Density Plots for Each Feature in the Wine Dataset

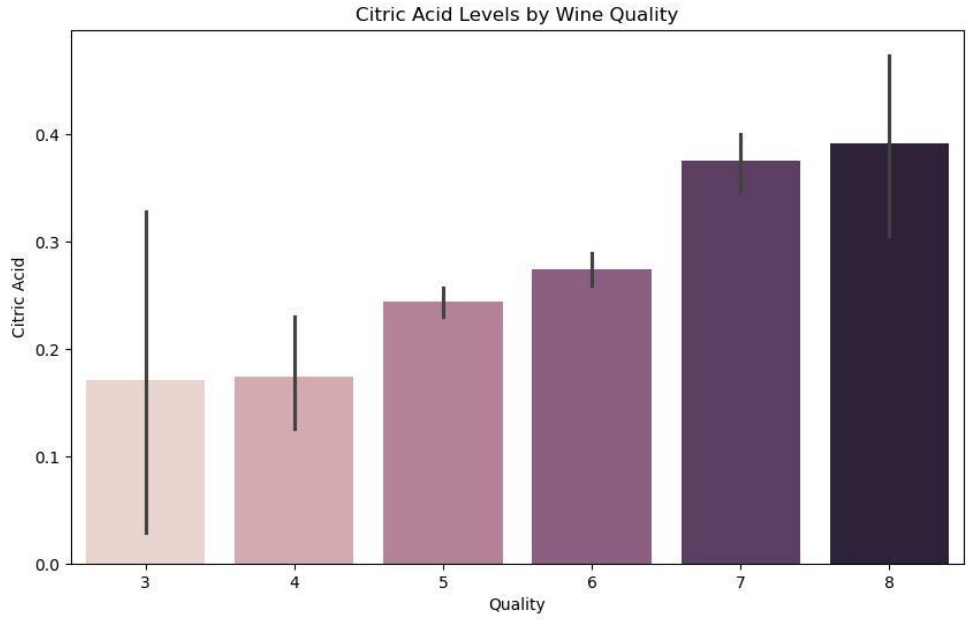


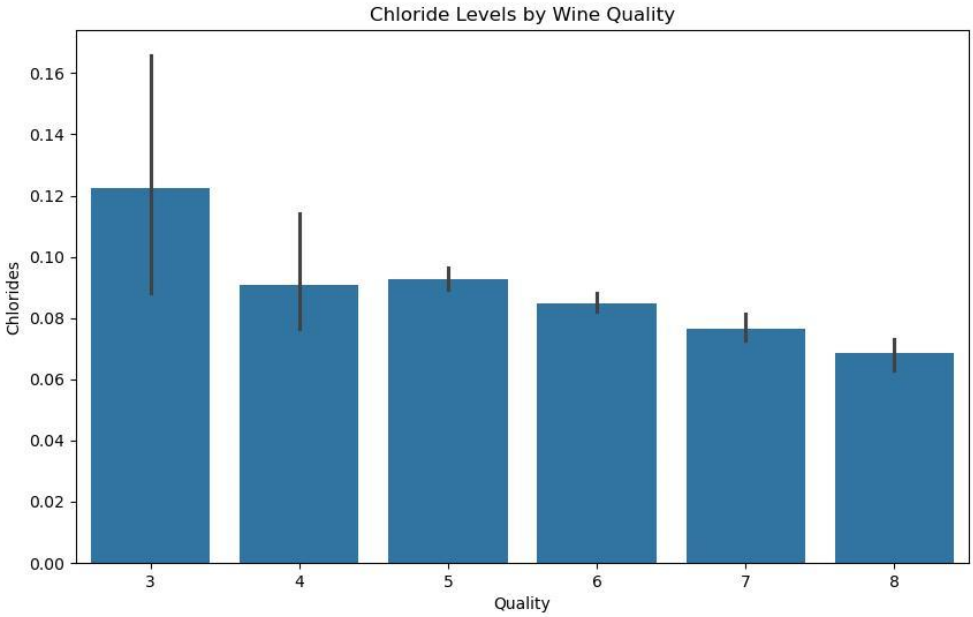
Box Plots for Each Feature in the Dataset

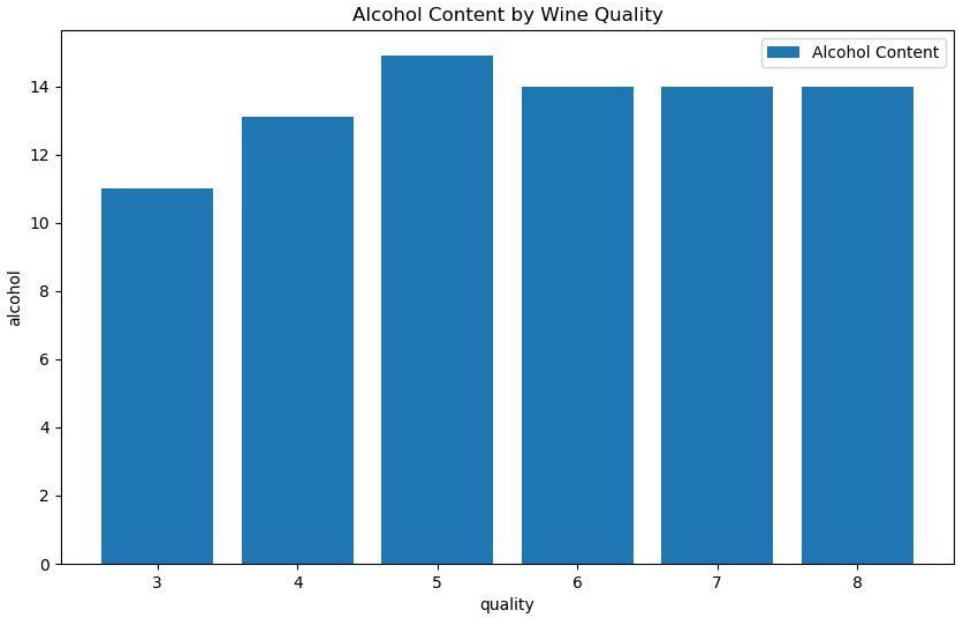


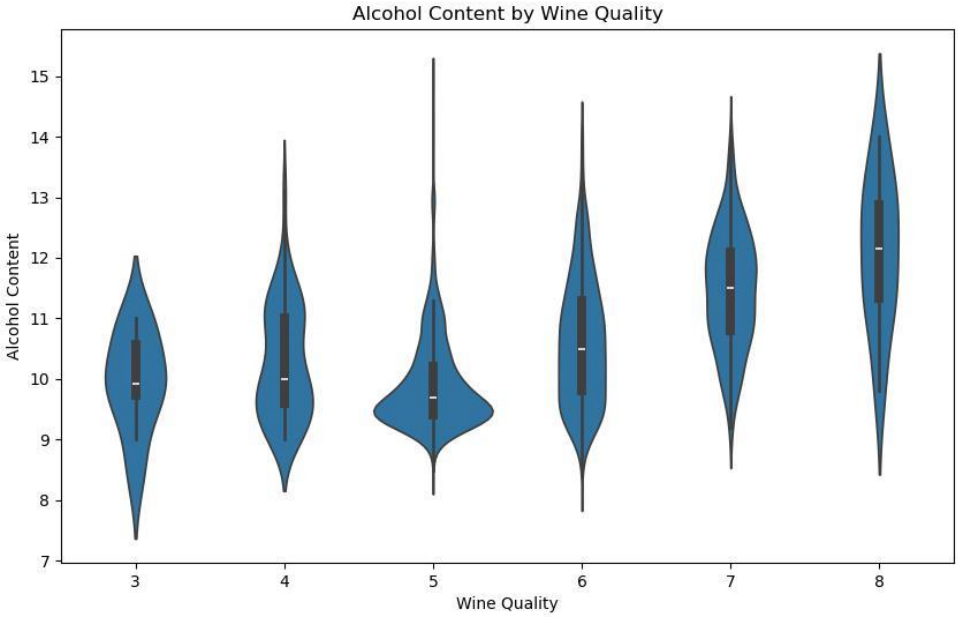

Residual Sugar Levels by Wine Quality
Residual Sugar
Quality
0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
3
4
5
6
7
8

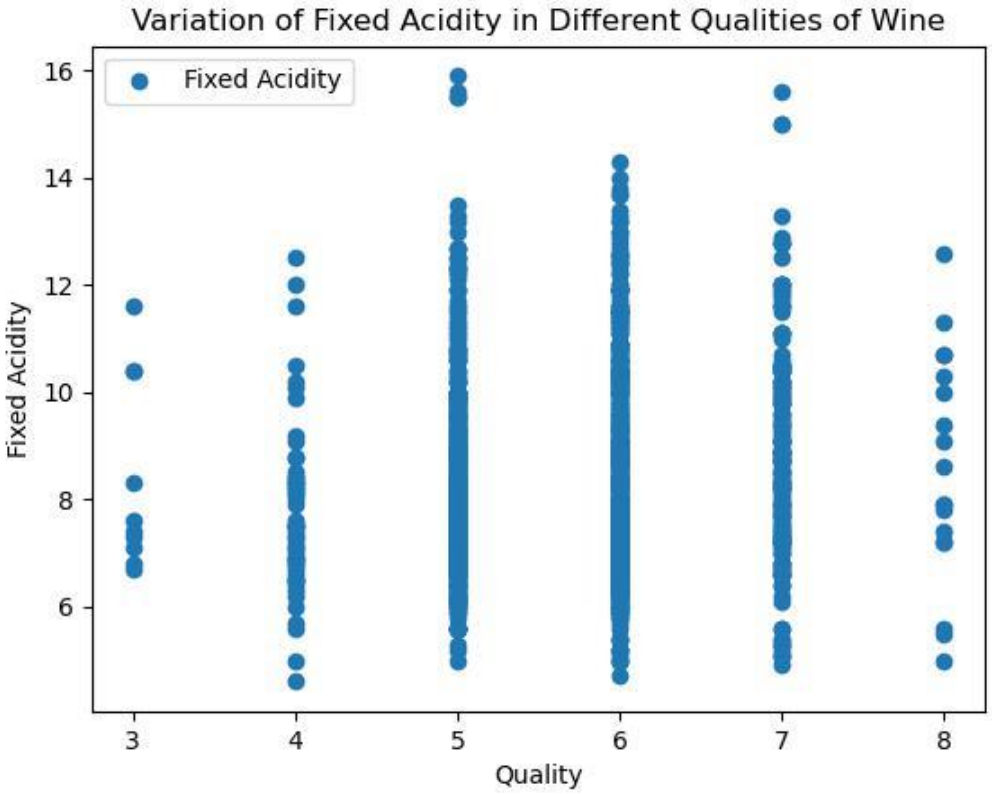

Free Sulfur Dioxide Levels by Wine Quality
Free Sulfur Dioxide
Quality
0.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5
3
4
5
6
7
8

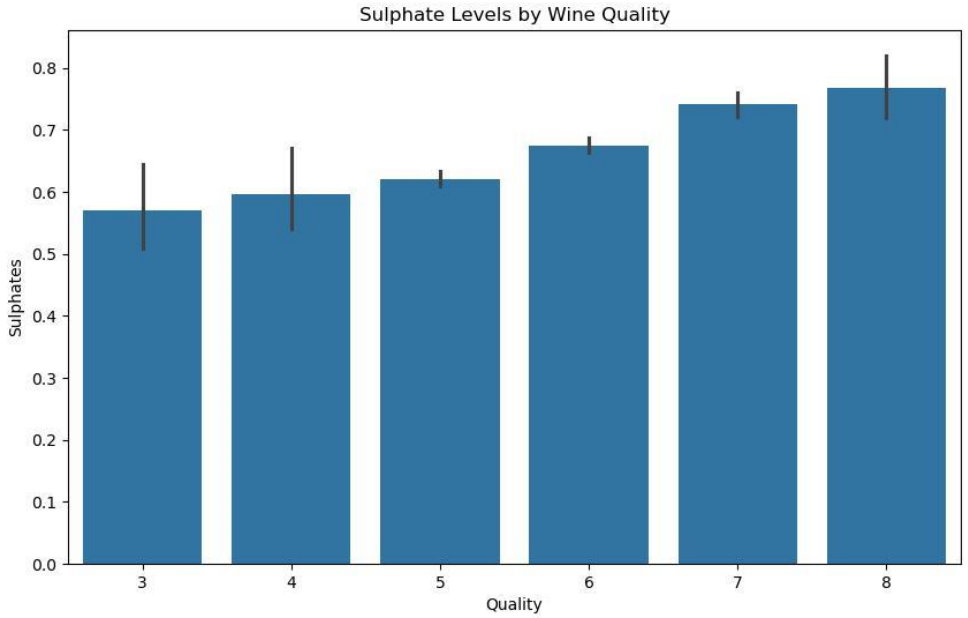

Citric Acid Levels by Wine Quality
Citric Acid
0.0
0.1
0.2
0.3
0.4
3
4
5
6
7
8
Quality


Chloride Levels by Wine Quality
Chlorides
0.00
0.02
0.04
0.06
0.08
0.10
0.12
0.14
0.16
3
4
5
6
7
8
Quality


Alcohol Content by Wine Quality
Alcohol Content
alcohol
quality
0
2
4
6
8
10
12
14
3
4
5
6
7
8


Alcohol Content by Wine Quality
Alcohol Content
Wine Quality
7
8
9
10
11
12
13
14
15
3
4
5
6
7
8


Variation of Fixed Acidity in Different Qualities of Wine
Fixed Acidity
Fixed Acidity
16
14
12
10
8
6
3
4
5
6
7
8
Quality


Sulphate Levels by Wine Quality
Sulphates
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0.0
3
4
5
6
7
8
Quality

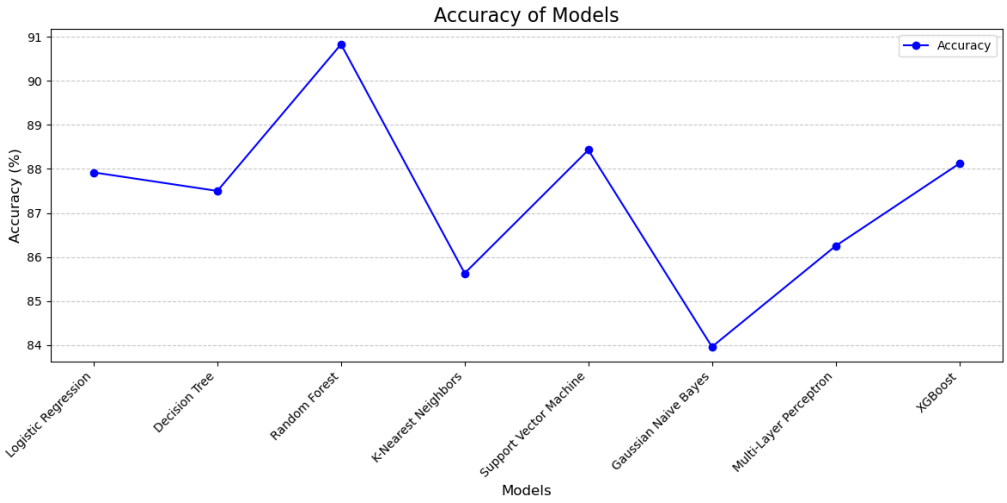
Rating:

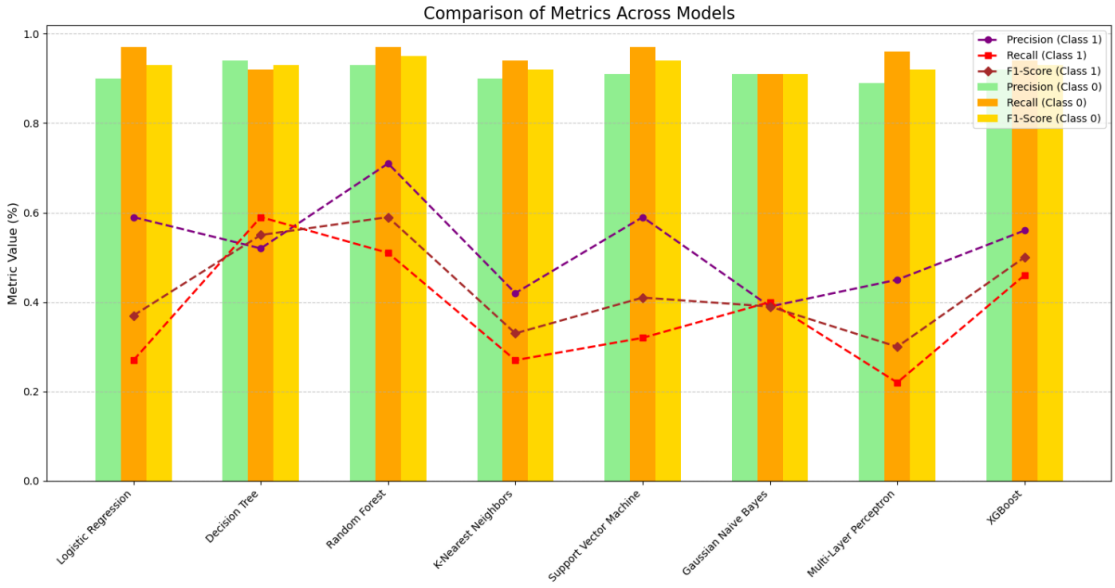
- *Bad (quality > 7) 1382*
- *Good (quality < 7) 217*

Rating	fixed acidity	citric acid	residual sugar	free sulfur dioxide	pH	sulphates	alcohol	quality
0	8.236831	0.254407	2.512120	16.172214	3.314616	0.644754	10.251037	5.408828
1	8.847005	0.376498	2.708756	13.981567	3.288802	0.743456	11.518049	7.082949

Results:

Model	Accuracy	Precision (Class 0)	Recall (Class 0)	F1-Score (Class 0)	Precision (Class 1)	Recall (Class 1)	F1-Score (Class 1)	Macro Avg (Precision)	Macro Avg (Recall)	Macro Avg (F1-Score)
Logistic Regression	87.92 %	0.90	0.97	0.93	0.59	0.27	0.37	0.74	0.62	0.65
Decision Tree	87.50 %	0.94	0.92	0.93	0.52	0.59	0.55	0.73	0.75	0.74
Random Forest	90.83 %	0.93	0.97	0.95	0.71	0.51	0.59	0.82	0.74	0.77
K-Nearest Neighbors	85.63 %	0.90	0.94	0.92	0.42	0.27	0.33	0.66	0.61	0.62
Support Vector Machine	88.43 %	0.91	0.97	0.94	0.59	0.32	0.41	0.43	0.50	0.46
Gaussian Naive Bayes	83.96 %	0.91	0.91	0.91	0.39	0.40	0.39	0.65	0.65	0.65
Multi-Layer Perceptron	86.25 %	0.89	0.96	0.92	0.45	0.22	0.30	0.67	0.59	0.61
XGBoost	88.12 %	0.92	0.94	0.93	0.56	0.46	0.50	0.74	0.70	0.72





Model Summary

Random Forest

- *Performance: Emerged as the most accurate model with an accuracy of 90.83%, providing high precision (0.93) and recall (0.97) for Class 0 (good wine).*
- *Strengths: Demonstrated significantly better metrics for Class 1 (bad wine) compared to other models, with a recall of 0.51, making it the most reliable model for predicting bad wines.*

Logistic Regression

- *Performance: Achieved an accuracy of 87.92%.*
- *Strengths: Simple and interpretable.*

- *Weaknesses: Struggled with the minority class (bad wine), with precision and recall for Class 1 being 0.59 and 0.27, respectively.*

Decision Tree

- *Performance: Achieved an accuracy of 87.50%.*
- *Strengths: Balanced precision (0.94) and recall (0.92) for Class 0 and improved recall (0.59) for Class 1 compared to Logistic Regression.*
- *Weaknesses: Class 1 precision (0.52) remains low.*

Support Vector Machine (SVM)

- *Performance: Accuracy of 88.43%.*
- *Weaknesses: Struggled to classify Class 1 effectively, with a low recall of 0.32 and precision of 0.59, indicating bias towards Class 0.*

K-Nearest Neighbors (k-NN)

- *Performance: Accuracy of 85.63%.*
- *Weaknesses: Struggled with identifying Class 1, with a recall of only 0.27, indicating difficulty in detecting bad wines.*

Gaussian Naive Bayes

- *Performance: Accuracy of 83.96%.*
- *Strengths: Good precision for Class 0 (0.91).*
- *Weaknesses: Struggled with Class 1, achieving a recall of 0.40, highlighting its limitations with imbalanced data.*

Multi-Layer Perceptron (ANN)

- *Performance: Achieved an accuracy of 86.25%.*
- *Weaknesses: Similar to k-NN, struggled with Class 1, with a low precision of 0.45 and recall of 0.22.*

XGBoost

- *Performance: Accuracy of 88.12%.*
- *Strengths: Balanced metrics for Class 0, with precision (0.92) and recall (0.94).*
- *Weaknesses: While Class 1 metrics are better than Logistic Regression, precision (0.56) and recall (0.46) still lag behind Random Forest.*

Synthesis and Discussion:

From the results, Random Forest clearly outperformed the other models due to its superior recall for the minority class (bad wine). This makes it the most reliable model for real-world applications, particularly where the cost of misclassifying bad wines is high.

Other models like Logistic Regression and Decision Tree offered a reasonable balance between accuracy and interpretability but fell short in their ability to predict the minority class effectively.

Models such as Support Vector Machine and Gaussian Naive Bayes faced significant challenges with class imbalance, leading to poor performance for Class 1.

Obstacles Encountered

- Class Imbalance: This led to difficulty in predicting the minority class, with many models exhibiting a bias toward the majority class.

- Model Complexity: Advanced models like XGBoost and ANN required fine-tuning and were computationally expensive, demanding careful optimization.

Future Recommendations

- **Class Imbalance Techniques:** Implement methods such as oversampling the minority class (e.g., SMOTE), undersampling the majority class, or using algorithms that support balanced class weights to improve recall and precision for Class 1.

- **Hyperparameter Tuning:** Perform rigorous hyperparameter optimization and cross-validation to further enhance model performance, particularly for complex models like ANN and SVM.

- **Focus on Cost-Sensitive Learning:** Employ cost-sensitive algorithms to address the high cost of misclassifying bad wines, which is critical in practical scenarios.

Conclusion

The answer to the question is that Random Forest is the best-performing model for this classification task. It provides the best trade-off between accuracy, precision, and recall, especially for the minority class (bad wine). This makes it the ideal choice for addressing the problem effectively.

References

- https://archive.ics.uci.edu/ml/machine-learning-databases/wine-quality/winequality-red.csv