Wine Quality Classification Models: Multiclass, Binary, and Clustering Approaches

Author:
Vinícius Franklin Pedroso Mansur de Azevedo

Juiz de Fora
April 2025

Contents

List of Tables

List of Figures

1 Introduction

In machine learning classification, we deal with various types of data that can be processed to achieve our goal. In this article, we will work with a multi-class dataset, which presents several challenges, especially when compared to binary classification. In binary classification, we usually have more samples for each specific class, which makes it easier for the model to predict the class with the same number of features. I encountered this issue when trying to create a model to predict wine quality using a dataset from the UCI Machine Learning Repository. I observed low values in many metrics, so I explored different approaches to achieve better results. I used techniques such as one-vs-one, one-vs-all, and also created clusters to reduce the number of classes. Further i was curious if i could cluster the two types of wine and make the same classification after.

The codes for this application will be on this github 'https://github.com/ViniciusFranklin22/Wine Quality-Classification-Models-Multiclass-Binary-and-Clustering-Approaches'.

2 Dataset

Taking a deeper look at the two datasets, we have one for white wine and another for red wine. That being said, both have the same number of features, which are:

- fixed acidity
- volatile acidity
- citric acid
- residual sugar
- chlorides
- free sulfur dioxide
- total sulfur dioxide
- density
- pH
- sulphates
- alcohol
- quality

Our target column is the quality one, so based on these production and chemical features, we aim to predict the class. The quality score ranges from 0 to 10, but in this dataset, the available scores are 3, 4, 5, 6, 7, and 8 for red wine and 3, 4, 5, 6, 7, 8, and 9 for white wine. Since we have only discrete values, we could consider building a model that predicts either continuous or categorical data. For this model, we will use the categorical approach because a float number would not add much value; we would have to round those, which is not the best case.

Regarding the number of rows, which is an important factor in classification models, the red wine dataset contains 1,599 rows, while the white wine dataset has 4,898. Based on this information alone, we could infer that the white wine dataset might yield better scores; later, we will verify whether this assumption holds.

Before building any model, it is crucial to analyze the features and how they interact with each other. A common practice I follow is first plotting the correlation matrix to check for linear correlations between the columns, especially with the target column. If a value has an absolute correlation higher than $|0.5|$, we can consider it significant. After that, I plot a box graph comparing each feature to the target column to visually confirm any linear correlation and check for other types of relationships that may be represented mathematically.

For the red wine dataset, we obtained the following correlation matrix:

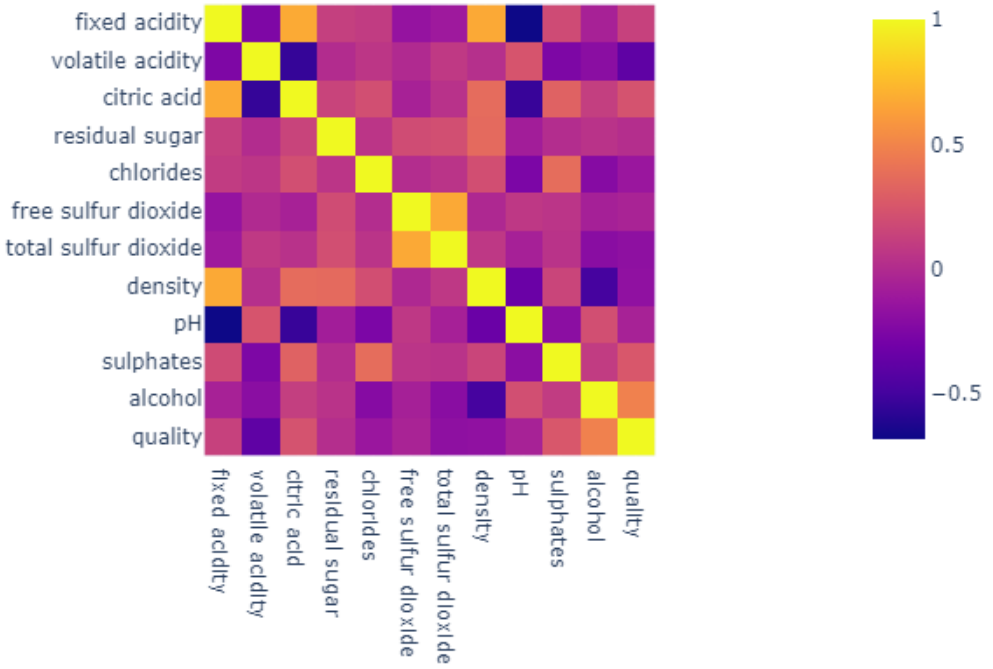

Figure 1: Red Wine Correlation Matrix (Made by the Author)

In this matrix, we observed some moderate correlations between chemical properties, which aligns with our expectations given their chemical compositions and interactions. What is most important, however, is their relationship with the target score. The feature with the highest correlation was alcohol, with a value of 0.476166, followed by volatile acidity with -0.390558. While these show some connection, they are still below the $|0.5|$ threshold, indicating moderate correlation. Therefore, we need to explore additional strategies to strengthen the model and improve its accuracy.

For the white wine dataset, we obtained the following correlation matrix:

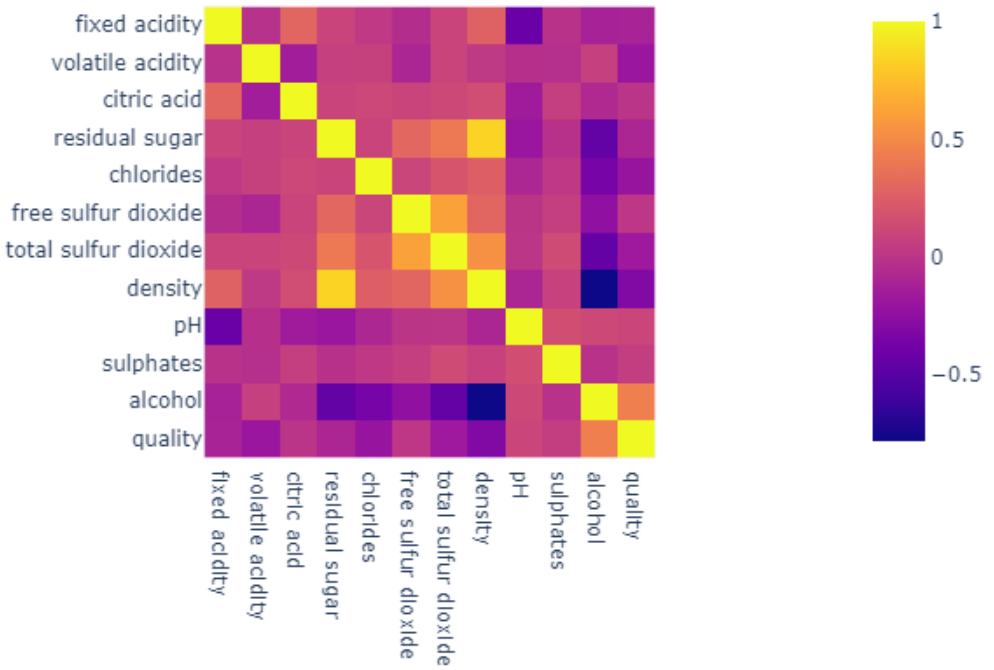

Figure 2: White Wine Correlation Matrix (Made by the Author)

Similar to the red wine dataset, we observed moderate correlations between chemical properties. The feature with the highest correlation to the target score was alcohol (0.435575), followed by density (-0.307123). Again, both values are below |0.5|, indicating moderate correlation. Notably, there are differences between the two wine types in terms of feature importance. Since none of the features exhibit strong correlation with the target score, we will need to employ additional techniques to enhance the model's predictive performance.

Now we will present the box plot graphs for various features and the target score for the red wine dataset:

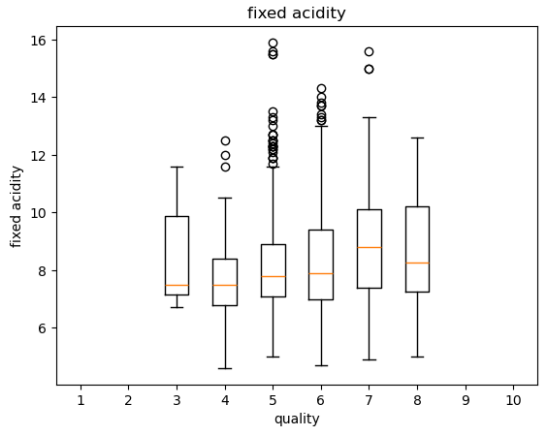

Figure 3: Red Wine - fixed acidity/quality

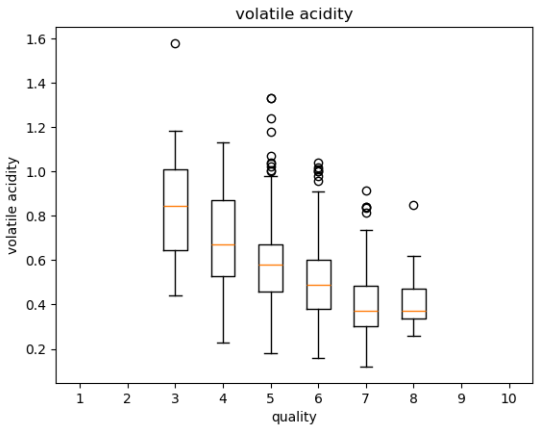

Figure 4: Red Wine - volatile acidity/quality

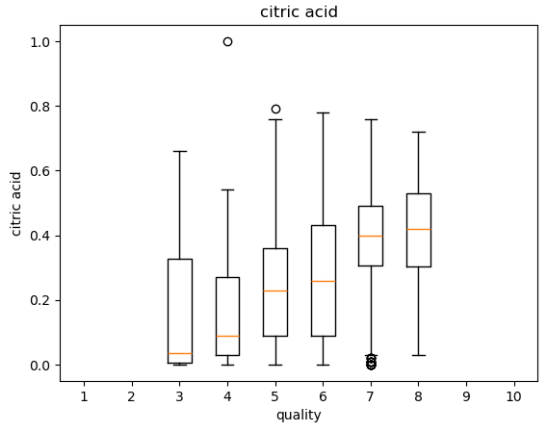

Figure 5: Red Wine - citric acid/quality

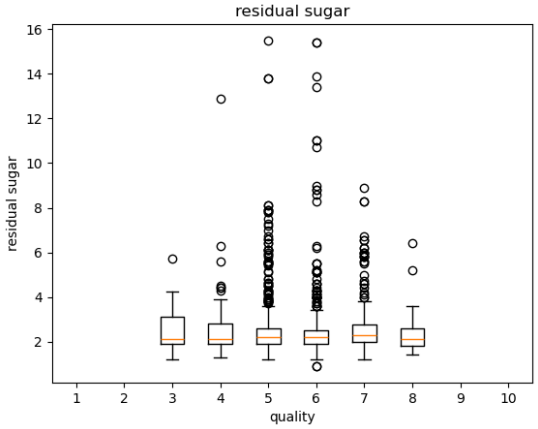

Figure 6: Red Wine - residual sugar/quality

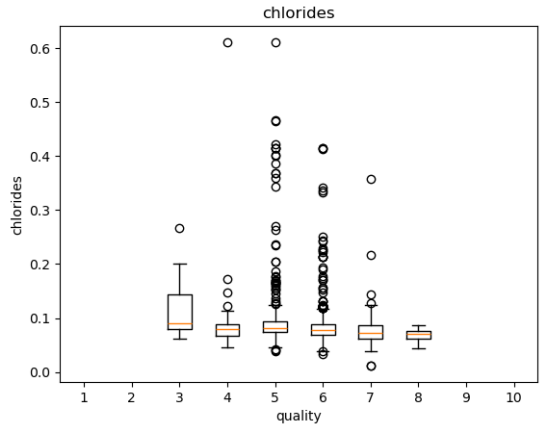

Figure 7: Red Wine - chlorides/quality

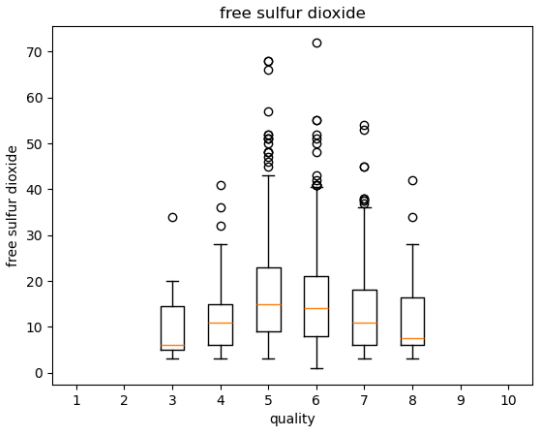

Figure 8: Red Wine - free sulfur dioxide/quality

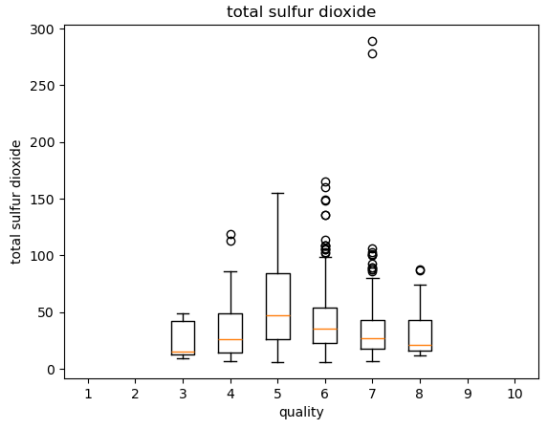

Figure 9: Red Wine - total sulfur dioxide/quality

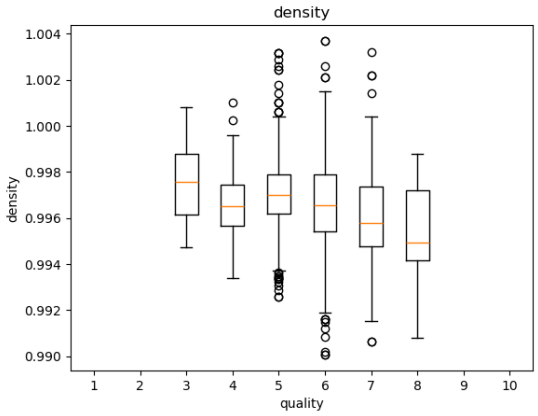

Figure 10: Red Wine - density/quality

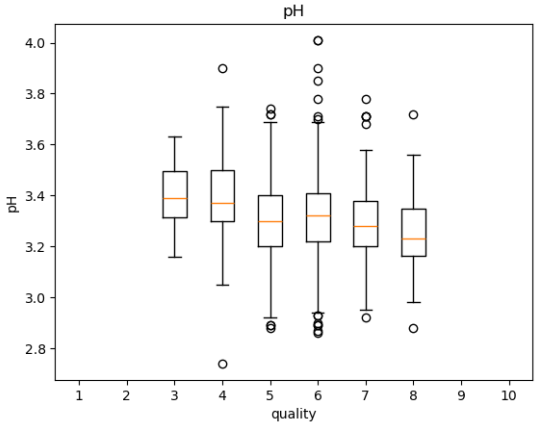

Figure 11: Red Wine - pH/quality

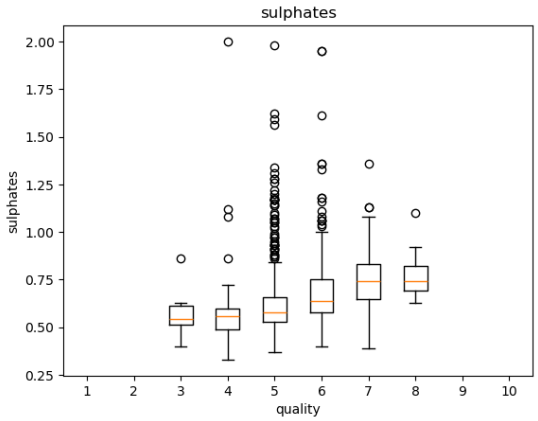

Figure 12: Red Wine - sulphates/quality

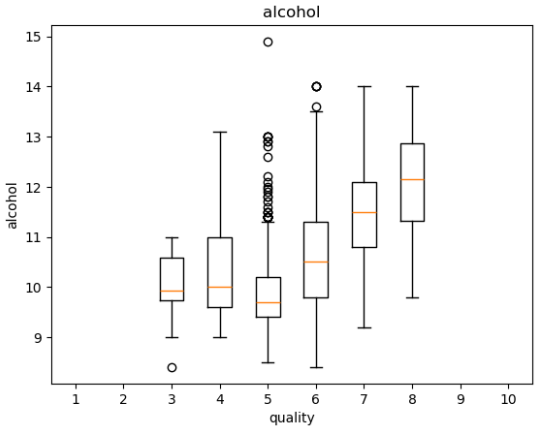

Figure 13: Red Wine - alcohol/quality

After analyzing these graphs, we can observe that the features volatile acidity and alcohol exhibit a somewhat linear relationship with the quality, although the correlation is not very strong. For the other features, the values appear more scattered without any obvious patterns, indicating that we face a bigger challenge ahead.

Next, we will present the box plot graphs for various features and the target score for the white wine dataset:

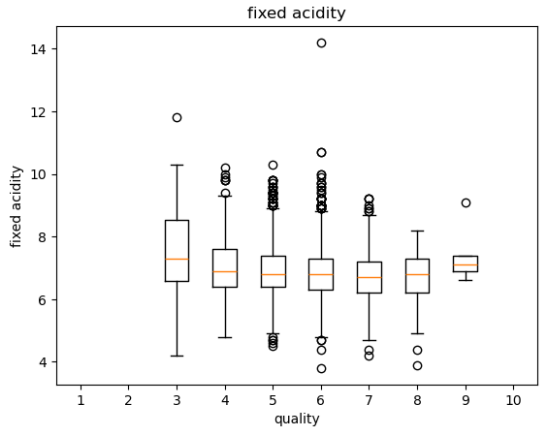

Figure 14: White Wine - fixed acidity/quality

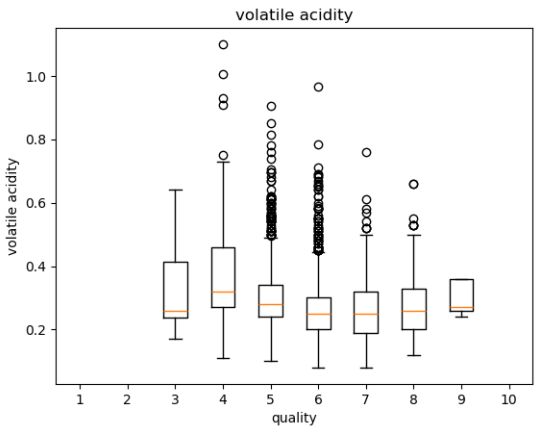

Figure 15: White Wine - volatile acidity/quality

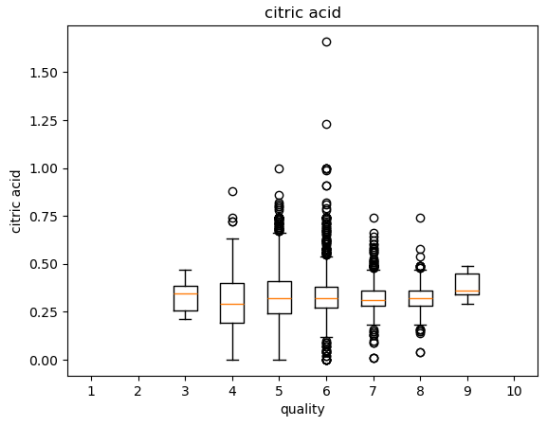

Figure 16: White Wine - citric acid/quality

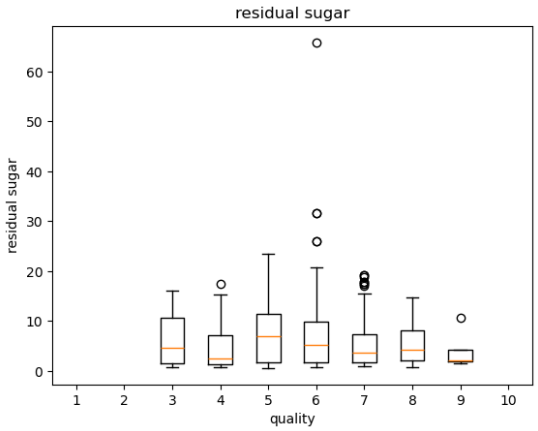

Figure 17: White Wine - residual sugar/quality

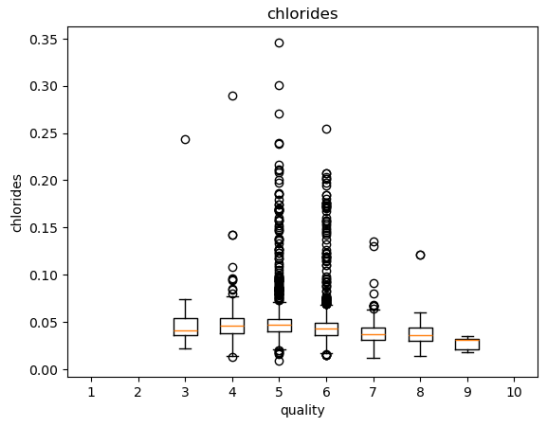

Figure 18: White Wine - chlorides/quality

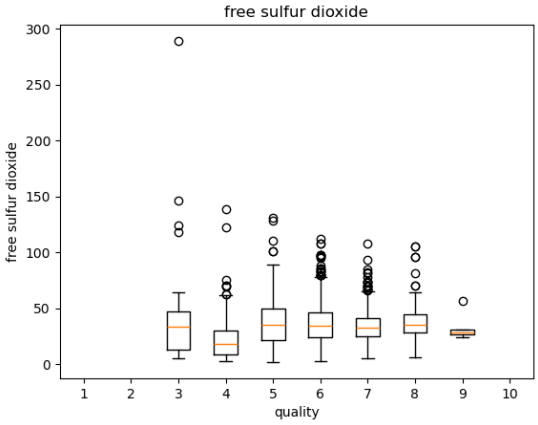

Figure 19: White Wine - free sulfur dioxide/quality

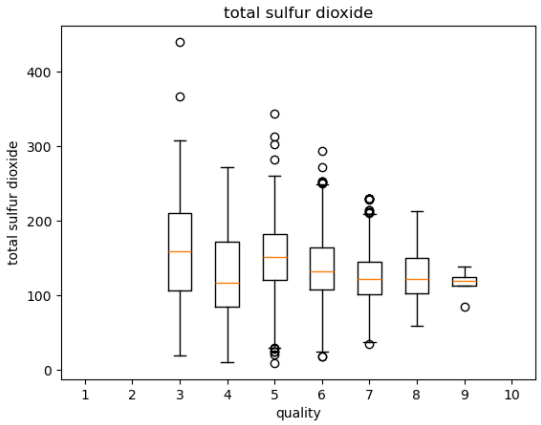

Figure 20: White Wine - total sulfur dioxide/quality

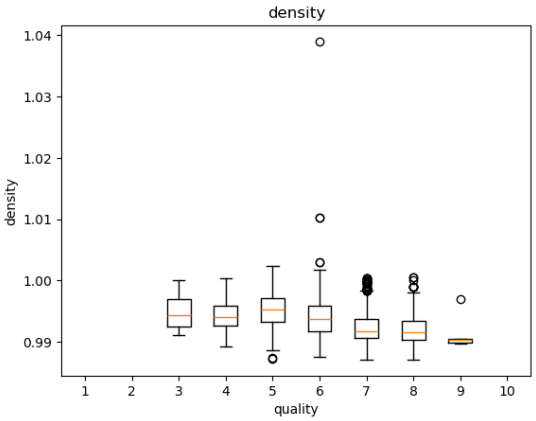

Figure 21: White Wine - density/quality

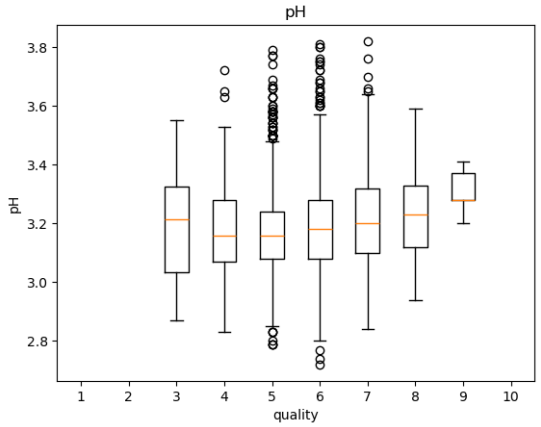

Figure 22: White Wine - pH/quality

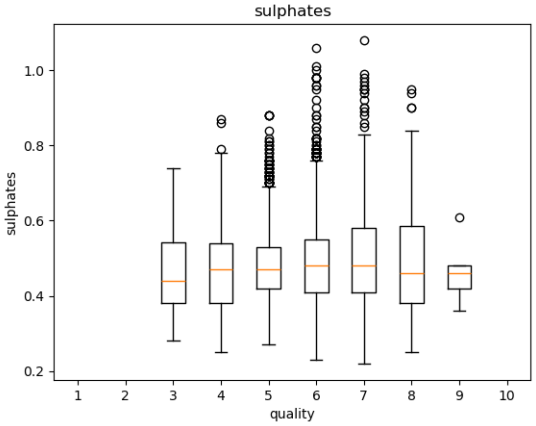

Figure 23: White Wine - sulphates/quality

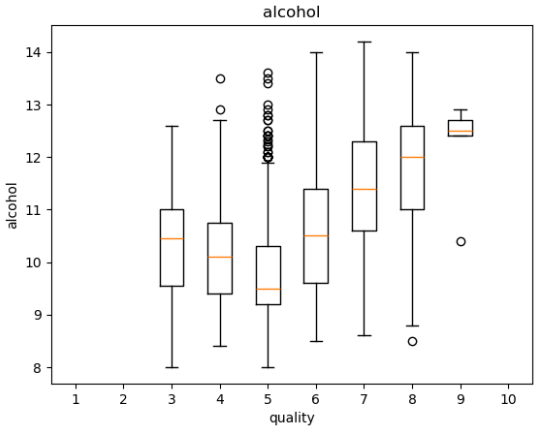

Figure 24: White Wine - alcohol/quality

Similar to the red wine, the white wine features exhibit little to no linear relationships with the quality. We only observe a slight linear trend with the alcohol

feature, although it is weak. Thus, we need to work harder to find an effective solution.

3 Pre-Processing the Data

Before training the models on this dataset, we must pre-process the data to address potential issues that may lead to misleading results.

It is common for different features, particularly those with different physical quantities, to have discrepancies in their values. For example, one feature may have much larger values than another, which could mislead the model into assigning more importance to the feature with larger values. To address this, we will use the `StandardScaler` from `sklearn.preprocessing`. This technique scales all features to a similar range, ensuring that each feature contributes equally to the final result.

Another challenge we will face is the imbalanced dataset for the target column. Some classes have a much lower number of samples compared to others. For the red wine dataset, the distribution of the target class is as follows:

- Class 3: 10 samples (0.62%)
- Class 4: 53 samples (3.31%)
- Class 5: 681 samples (42.59%)
- Class 6: 638 samples (39.99%)
- Class 7: 199 samples (12.44%)
- Class 8: 18 samples (1.12%)

For the white wine dataset, the distribution is as follows:

- Class 3: 20 samples (0.40%)
- Class 4: 163 samples (3.33%)
- Class 5: 1457 samples (29.75%)
- Class 6: 2198 samples (44.89%)
- Class 7: 880 samples (17.97%)
- Class 8: 175 samples (3.57%)
- Class 9: 5 samples (0.10%)

Both datasets exhibit significant imbalance in the target variable (quality), with most samples concentrated in the middle classes. This imbalance could lead to biased predictions. One way to address this issue is by oversampling the under-represented classes in the training data. We will use `RandomOverSampler` from `imblearn.over_sampling` to perform this oversampling.

However, it's important to note that while oversampling can help mitigate the imbalance problem, it is not a perfect solution. If the original dataset has a very low number of samples in certain classes, simply replicating those samples may

not be sufficient. This approach could still lead to poor performance for those classes, especially if the discrepancy in sample sizes is too large. We will analyze the impact of this later. In order to construct the training, validation, and test datasets, the data was randomly split into three subsets. The split was done by allocating 60% of the data to the training set, 20% to the validation set, and 20% to the test set. The `np.split` function was used after shuffling the data randomly with the `sample(frac=1)` function. This approach ensures that the data distribution is preserved while maintaining the randomness across the datasets. This method ensures that the datasets are well-represented, with each subset having a random distribution of the original data. By using this technique, we guarantee that the models will be trained on a variety of data and tested on unseen data, which is essential for evaluating their generalization ability. Moreover, the validation dataset helps fine-tune the model parameters and prevent overfitting, while the test dataset serves as an independent evaluation of the model's performance.

4 Note on Data Partitioning Across Approaches

It is important to note that although the same train/validation/test split was consistently used within each individual approach (One-vs-All, One-vs-One, and the grouped classes approach), the splits were generated independently for each script. As a result, comparisons between models within the same approach remain valid and fair, as they used the exact same data partitions. However, comparisons across different approaches may be subject to slight variations due to differences in data partitioning.

This occurred because the experiments were developed at different times, and the datasets were not saved explicitly between runs — leading to new splits being created each time. Despite this, the differences are expected to be minimal, since stratified sampling was used and the dataset is large. Thus, the overall conclusions and performance trends remain meaningful and reliable.

This is acknowledged as a minor limitation in the experimental design. In future work, this will be addressed by saving and reusing the same data splits across all experiments to ensure full consistency and reproducibility.

5 Models – One-vs-One and Native Multiclass

For classifiers that are natively binary, such as Support Vector Machines and Logistic Regression, we must use either the One-vs-One (OvO) or One-vs-All (OvA) technique to handle multiclass problems. On the other hand, classifiers that natively support multiclass classification do not require these techniques — although the OvA strategy may still be applied optionally.

In this chapter, we apply both the One-vs-All strategy and the native multiclass approach provided by the libraries, depending on the nature of each model.

The One-vs-One strategy is commonly used with natively binary classifiers to enable multiclass classification. It works by training a separate binary classifier for each pair of classes. For a total of N classes, the number of binary classifiers required is given by:

$$\text{Number of Binary Models} = \frac{N(N-1)}{2}$$

The models that typically rely on the OvO strategy for multiclass support are:

- Support Vector Machine (SVM)
- Logistic Regression
- Neural Networks (when using sigmoid activation in the output layer)

The following models, however, are natively multiclass and therefore do not require OvO. They typically handle multiple classes internally and are not compatible with the One-vs-One approach by design:

- K-Nearest Neighbors (KNN)
- Naive Bayes
- Decision Tree Classifier
- Random Forest Classifier
- Neural Networks (when using softmax activation in the output layer — as used in this section)

To ensure consistency in our comparisons, we evaluate all models using the `classification_report` from `sklearn.metrics`, which includes precision, recall, and F1-score for each class. We also consider macro and weighted averages, along with overall accuracy.

All models are implemented using Python libraries. In most cases, when a natively binary classifier is applied to a multiclass problem, the default behavior is to automatically apply the One-vs-One strategy — not One-vs-All — unless specified otherwise.

By grouping both native multiclass models and binary models with OvO into this section, we ensure broader coverage and consistency. This organization will also facilitate more intuitive comparisons later on, especially when we analyze the performance of models using the One-vs-All strategy in a separate section.

Finally, we analyze the performance of each model using the training, validation, and test datasets prepared earlier, and then compare their overall results.

5.1 K-Nearest Neighbors (KNN)

For the KNN model, we considered 1 neighbors, after testing different values. The results for the red wine dataset are presented below:

Class	Precision	Recall	F1-Score	Support
3	0.00	0.00	0.00	2
4	0.33	0.17	0.22	12
5	0.61	0.60	0.60	133
6	0.58	0.54	0.56	132
7	0.39	0.54	0.45	37
8	0.00	0.00	0.00	4
Accuracy			0.54	320
Macro avg	0.32	0.31	0.31	320
Weighted avg	0.55	0.54	0.54	320

Table 1: Classification report for the KNN model on the Red Wine dataset

The results for the white wine dataset are as follows:

Class	Precision	Recall	F1-Score	Support
3	0.50	0.25	0.33	4
4	0.22	0.15	0.18	33
5	0.57	0.58	0.58	291
6	0.61	0.62	0.62	440
7	0.48	0.48	0.48	176
8	0.42	0.49	0.45	35
9	0.00	0.00	0.00	1
Accuracy			0.56	980
Macro avg	0.40	0.37	0.38	980
Weighted avg	0.56	0.56	0.56	980

Table 2: Classification report for the KNN model on the White Wine dataset

As expected, the classes with a larger number of samples achieved better scores. The red wine dataset exhibited a relatively better average score than the white wine dataset, with a difference of approximately 17%.

5.2 Naive Bayes

For the Naive Bayes model, the results for the red wine dataset are as follows:

Class	Precision	Recall	F1-Score	Support
3	0.02	0.50	0.03	2
4	0.00	0.00	0.00	12
5	0.64	0.44	0.52	133
6	0.56	0.23	0.32	132
7	0.29	0.49	0.36	37
8	0.03	0.25	0.05	4
Accuracy			0.34	320
Macro avg	0.26	0.32	0.22	320
Weighted avg	0.53	0.34	0.39	320

Table 3: Classification report for the Naive Bayes model on the Red Wine dataset

The results for the white wine dataset are:

Class	Precision	Recall	F1-Score	Support
3	0.25	0.25	0.25	4
4	0.22	0.21	0.22	33
5	0.54	0.54	0.54	291
6	0.57	0.40	0.47	440
7	0.36	0.68	0.47	176
8	0.20	0.03	0.05	35
9	0.00	0.00	0.00	1
Accuracy			0.47	980
Macro avg	0.30	0.30	0.28	980
Weighted avg	0.50	0.47	0.47	980

Table 4: Classification report for the Naive Bayes model on the White Wine dataset

The Naive Bayes model performed better on the white wine dataset compared to the red one, which performed poorly. However, this cannot be considered an overall improvement, as the classification for underrepresented classes remained inadequate. Future work could explore methods such as data balancing or feature engineering to enhance classification performance.

5.3 Support Vector Machine

For the SVM model, the results for the red wine dataset are:

Class	Precision	Recall	F1-Score	Support
3	0.00	0.00	0.00	2
4	0.17	0.58	0.26	12
5	0.62	0.55	0.58	133
6	0.56	0.38	0.45	132
7	0.33	0.51	0.40	37
8	0.00	0.00	0.00	4
Accuracy			0.47	320
Macro avg	0.28	0.34	0.28	320
Weighted avg	0.54	0.47	0.49	320

Table 5: Classification report for the SVM model on the Red Wine dataset

For the white wine dataset, we have the following results:

Class	Precision	Recall	F1-Score	Support
3	1.00	0.00	0.00	4
4	1.00	0.00	0.00	33
5	0.60	0.01	0.02	291
6	0.45	1.00	0.62	440
7	1.00	0.00	0.00	176
8	1.00	0.00	0.00	35
9	1.00	0.00	0.00	1
Accuracy			0.45	980
Macro avg	0.86	0.14	0.09	980
Weighted avg	0.63	0.45	0.28	980

Table 6: Classification report for the SVM model on the White Wine dataset

For the SVM model, we observed relatively acceptable scores for the middle classes in the red wine dataset, which is typical since these classes have a higher number of samples. However, similar to previous results, the model performs poorly on the smaller classes. In the white wine dataset, the results were much worse. Only class 6 achieved a reasonably good score, while the other classes had very low performance. This indicates that the model is not yet reliable enough for the white wine data, as it failed to properly classify the majority of the instances.

5.4 Logistic Regression

For the Logistic Regression model, the results for the red wine dataset are:

Class	Precision	Recall	F1-Score	Support
3	0.00	0.00	0.00	2
4	0.10	0.42	0.16	12
5	0.69	0.52	0.59	133
6	0.65	0.42	0.51	132
7	0.28	0.32	0.30	37
8	0.04	0.25	0.06	4
Accuracy			0.45	320
Macro avg	0.29	0.32	0.27	320
Weighted avg	0.59	0.45	0.50	320

Table 7: Classification report for the Logistic Regression model on the Red Wine dataset

For the white wine dataset, we have the following results:

Class	Precision	Recall	F1-Score	Support
3	1.00	0.00	0.00	4
4	1.00	0.00	0.00	33
5	0.51	0.31	0.39	291
6	0.48	0.88	0.62	440
7	0.60	0.02	0.03	176
8	1.00	0.00	0.00	35
9	1.00	0.00	0.00	1
Accuracy			0.49	980
Macro avg	0.80	0.17	0.15	980
Weighted avg	0.55	0.49	0.40	980

Table 8: Classification report for the Logistic Regression model on the White Wine dataset

The Logistic Regression model performed similarly for both types of wine, with better scores for the classes with more samples. This remains an issue, as the model is biased towards the majority classes. Both datasets showed moderate performance, with accuracy being below optimal levels. Therefore, Logistic Regression does not appear to be the best candidate for the model in this project, given its limitations in handling imbalanced class distributions.

5.5 Decision Tree Classification

For the Decision Tree Classification model, the results for the red wine dataset are:

Class	Precision	Recall	F1-Score	Support
3	0.00	0.00	0.00	2
4	0.18	0.17	0.17	12
5	0.61	0.52	0.56	133
6	0.52	0.64	0.57	132
7	0.25	0.19	0.22	37
8	0.00	0.00	0.00	4
Accuracy			0.51	320
Macro avg	0.26	0.25	0.25	320
Weighted avg	0.50	0.51	0.50	320

Table 9: Classification report for the Decision Tree model on the Red Wine dataset.

For the white wine dataset, the classification results are:

Class	Precision	Recall	F1-Score	Support
3	0.00	0.00	0.00	4
4	0.28	0.27	0.28	33
5	0.60	0.61	0.61	291
6	0.64	0.60	0.62	440
7	0.52	0.52	0.52	176
8	0.38	0.60	0.46	35
9	1.00	0.00	0.00	1
Accuracy			0.58	980
Macro avg	0.49	0.37	0.36	980
Weighted avg	0.58	0.58	0.58	980

Table 10: Classification report for the Decision Tree model on the White Wine dataset.

The Decision Tree model produced moderate results, achieving its best performance on the most frequent classes, particularly those with larger sample sizes. However, some of the less frequent classes with over 30 samples also showed reasonable predictive performance. Despite these results, the model does not achieve high accuracy, indicating limitations in its ability to generalize effectively across all classes. While it provides some level of prediction, it is not the optimal choice for this classification task.

5.6 Random Forest Classification

For the Random Forest Classification model, the results for the red wine dataset are:

Class	Precision	Recall	F1-Score	Support
3	0.00	0.00	0.00	2
4	0.30	0.25	0.27	12
5	0.61	0.53	0.57	133
6	0.51	0.61	0.56	132
7	0.25	0.22	0.23	37
8	0.00	0.00	0.00	4
Accuracy			0.51	320
Macro avg	0.28	0.27	0.27	320
Weighted avg	0.51	0.51	0.50	320

Table 11: Classification report for the Random Forest model on the Red Wine dataset.

For the white wine dataset, the classification results are:

Class	Precision	Recall	F1-Score	Support
3	1.00	0.00	0.00	4
4	0.62	0.24	0.35	33
5	0.70	0.66	0.68	291
6	0.64	0.79	0.71	440
7	0.67	0.50	0.57	176
8	1.00	0.51	0.68	35
9	1.00	0.00	0.00	1
Accuracy			0.67	980
Macro avg	0.80	0.39	0.43	980
Weighted avg	0.68	0.67	0.66	980

Table 12: Classification report for the Random Forest model on the White Wine dataset.

The Random Forest Classification model also showed differences in the number of samples across different classes, which affected the scores. However, in the case of the white wine dataset, we observed an improved overall performance, particularly in class 8, which achieved an F1-Score of 0.68 despite having only 35 samples.

5.7 Neural Network

For the Neural Network model, we used a 5-layer architecture. The network consists of two hidden layers with ReLU activation functions, followed by dropout layers to help prevent overfitting. The output layer uses a softmax activation function to handle the multi-class classification task. The loss function used is sparse categorical cross-entropy, which is suitable for multi-class problems where the target labels are integers rather than one-hot encoded vectors.

The model parameters were tuned by testing various combinations to minimize the loss. To select the best-performing model, we evaluated multiple hyperparameter combinations based on validation performance (F1-score, accuracy, and loss). After testing several configurations, the best combination of hyperparameters was selected to optimize performance. These parameters were adjusted to minimize loss, maximize the F1-score, and improve the model's generalization ability on unseen data, using both validation and test sets for a more robust evaluation.

For the red wine dataset, the best-performing parameters and results were:

- **Number of nodes in each hidden layer**: 34
- **Dropout probability**: 0
- **Learning rate (lr)**: 0.005
- **Batch size**: 128
- **Number of epochs**: 100

Class	Precision	Recall	F1-Score	Support
3	0.00	0.00	0.00	2
4	0.14	0.17	0.15	12
5	0.70	0.65	0.67	133
6	0.59	0.58	0.58	132
7	0.48	0.54	0.51	37
8	0.12	0.25	0.17	4
Accuracy			0.58	320
Macro avg	0.34	0.36	0.35	320
Weighted avg	0.59	0.58	0.59	320

Table 13: Classification report for the Neural Network on the Red Wine dataset.

Similarly, for the white wine dataset, the best-performing parameters and results were:

- **Number of nodes in each hidden layer**: 16
- **Dropout probability**: 0
- **Learning rate (lr)**: 0.005
- **Batch size**: 128
- **Number of epochs**: 100

Class	Precision	Recall	F1-Score	Support
3	1.00	0.25	0.40	4
4	0.33	0.06	0.10	33
5	0.56	0.58	0.57	291
6	0.56	0.68	0.62	440
7	0.49	0.39	0.43	176
8	0.00	0.00	0.00	35
9	0.00	0.00	0.00	1
Accuracy			0.55	980
Macro avg	0.42	0.28	0.30	980
Weighted avg	0.52	0.55	0.53	980

Table 14: Classification report for the Neural Network on the White Wine dataset.

For both datasets, we encountered challenges with the underrepresented classes. While the model performed reasonably well for the more frequent classes, the overall results are still suboptimal. Despite these limitations, this model will be considered in the final evaluation for the best-performing approach in the one-vs-one strategy. However, further refinements are necessary.

5.8 Conclusion

After analyzing all models, we observed that, despite some improvements, they still struggle to predict the less frequent classes. Additionally, even for classes with a higher number of samples, the F1-score did not exceed 0.70. Due to these challenges, we will continue exploring different strategies.

For the final evaluation, we compared all models and strategies, selecting the best candidate from each. The chosen models are those that performed the best within each strategy.

- **Red Wine Dataset: Neural Network** This model achieved the highest accuracy along with the best F1-scores. Notably, it correctly classified some of the underrepresented classes, even those with only four samples. Given its performance, it will represent this technique moving forward.
- **White Wine Dataset: Random Forest Classification** This model also achieved the highest accuracy and the best F1-scores by a significant margin. Interestingly, it performed much better than the models for red wine, likely due to differences in the dataset characteristics.

An interesting observation is that, although the chosen models for each dataset were different, both are natively multiclass models. As a result, they outperformed the native binary models that used the OvO strategy.

6 Models - One vs All

In this section, we explore a different approach to multiclass classification known as **one vs all**. Instead of treating the problem as a direct multiclass classification, we transform it into multiple binary classification tasks. For each class, a separate model is trained to determine whether a given sample belongs to that class or not. If there are n classes, this process is repeated n times, training n binary classifiers.

While this technique is straightforward, it has some drawbacks. One notable issue is that a single sample may be classified as belonging to multiple classes, leading to ambiguous predictions. Additionally, there may be cases where a sample is not classified as belonging to any class, making it impossible to assign a label. As a result, the number of samples classified as true may be either higher or lower than the actual number of samples in a class.

However, if the number of samples classified as true matches the real number of samples, this often indicates that the model's false positives and false negatives are balanced. In such cases, even though the model makes classification errors, the total number of predicted positives remains aligned with the true count due to this compensation effect. For this reason, we will also present the confusion matrix to better understand the underlying classification behavior. Thus, the total number of samples classified as positive by the model can be expressed as:

$$\text{Classified Samples} = TP + FP$$

One possible approach to mitigate the overlapping and unclassified sample issues is to run the binary models in order of their performance. The model with the highest

classification accuracy is executed first, and any samples classified as positive for that class are removed from the dataset before proceeding to the next model. This ensures that each sample is assigned to at most one class and prioritizes the most reliable classifications.

However, since our focus is on evaluating the overall performance of the models on a larger dataset, we will not apply this refinement at this stage. If this method proves to be the best-performing approach, then we will consider implementing these error-handling strategies.

Just as an additional observation, in this type of approach, it is also possible to use different models for each binary classifier, aiming to maximize individual performance — a strategy sometimes referred to as a *Hybrid One-vs-All*. For instance, the best performing classifier for class 1 might be a K-Nearest Neighbors model, while for class 2 it could be a Neural Network. This flexibility can potentially yield better results when using the OVA scheme, especially when combined with the technique of identifying the true class from the test set predictions (as discussed earlier).

However, in this study, we chose to use a single model for all binary classifiers in order to simplify the experiments and facilitate later comparisons with other approaches. If this baseline proves to be the most effective, the hybrid strategy may be explored as a next step.

For evaluation, we follow the same approach as in the **one vs one** method, using multiple models to identify the best-performing one. However, due to the nature of **one vs all**, accuracy alone can be misleading. Since most samples in each binary classification problem belong to the "not the class" category, the accuracy metric tends to be inflated. Thus, it is more meaningful to analyze the **F1-score** for the positive class, ensuring a balanced assessment of precision and recall.

To evaluate the performance of the **One-vs-All** classification models, we used the **Classification Report** from Scikit-Learn, which provides the **Precision, Recall, F1-score**, and **Support** metrics for each class.

Each binary model was trained separately, treating a specific class as **positive (1)** and all others as **negative (0)**. After generating the **individual Classification Reports**, we extracted only the metrics corresponding to the **positive (1)** class from each binary model and consolidated the results into a single table. This format facilitates comparison between approaches and allows for a more objective analysis of each class's performance.

The **individual Classification Reports** for each class are available in the `wine_one_vs_all.ipynb` notebook. Since the One-vs-All structure already reflects the relationship between classes, we have chosen to present only the consolidated table in this report.

This methodology is applied consistently to **all tested models**, ensuring uniformity in the analysis and facilitating comparison.

A limitation of showing only the summarized table is that it becomes impossible to infer the number of false positives or false negatives for a given class when the number of true positives is zero. In such cases, both precision and recall for the positive class will be zero, and therefore, from the table alone, it is impossible to determine how many samples the model predicted as positive (i.e., true positives + false positives). For this reason, if a more in-depth understanding of the classification behavior is needed, please refer to the notebook mentioned above. The real number of positive samples was calculated by summing the number of predicted true samples

across all binary models.

For the weighted and macro averages, we used the precision, recall, and F1-score of the positive class (label 1) from each binary classifier. The macro average gives equal importance to each model, regardless of the class size, while the weighted average accounts for the number of true samples (support) of each class.

In addition, we calculated an overall score by summing the total number of true positives, true negatives, false positives, and false negatives across all binary classifiers. These aggregated values were then used to compute a single precision, recall, F1-score, and accuracy as if all predictions had come from a single multiclass classifier. This provides a broader perspective of the system's global performance.

The **Overclassified Samples** row indicates the number of samples that were classified as positive by one or more models.

The **Classified Samples** row represents the total number of samples predicted as positive by the full model, calculated as the sum of true (positive) predictions across all binary classifiers.

For this approach, it is crucial to perform the train/validation/test split *before* generating the binary datasets for each class. This ensures that all binary classifiers are trained and evaluated on the same data partitions, leading to more consistent and comparable results within the approach.

Next, we present the models used in this technique and later compare their performance.

6.1 K-Nearest Neighbors (KNN)

For the K-Nearest Neighbors model, the number of neighbors chosen was 1. After several tests, this configuration provided the best performance.

For the Red Wine dataset, we obtained the following results:

Class	Precision	Recall	F1-Score	Support
3	0.00	0.00	0.00	3
4	0.25	0.20	0.22	10
5	0.64	0.70	0.67	127
6	0.65	0.62	0.64	141
7	0.43	0.42	0.43	38
8	0.00	0.00	0.00	1
Macro avg	0.33	0.32	0.33	320
Weighted avg	0.60	0.61	0.61	320
Classified Samples	-	-	-	320
Over-classified Samples	-	-	-	0
Overall Avg	0.61	0.61	0.61	320
Overall Accuracy	-	-	0.87	320

Table 15: Classification report (Only True Class) for KNN (One-vs-All) on Red Wine dataset.

Class	TN	FP	FN	TP
3	316	1	3	0
4	304	6	8	2
5	144	49	38	89
6	132	47	53	88
7	261	21	22	16
8	318	1	1	0
Total	1475	125	125	195

Table 16: Confusion matrices per class (One-vs-All) for KNN and total values for the Red Wine dataset.

For the White Wine dataset:

Class	Precision	Recall	F1-Score	Support
3	0.00	0.00	0.00	4
4	0.37	0.30	0.33	33
5	0.66	0.60	0.63	291
6	0.65	0.67	0.66	440
7	0.55	0.56	0.55	176
8	0.44	0.66	0.53	35
9	0.00	0.00	0.00	1
Macro avg	0.38	0.40	0.39	980
Weighted avg	0.61	0.61	0.61	980
Classified Samples	-	-	-	980
Overclassified Samples	-	-	-	0
Overall Avg	0.61	0.61	0.61	980
Overall Accuracy	-	-	0.89	980

Table 17: Classification report (Only True Class) for KNN (One-vs-All) on White Wine dataset.

Class	TN	FP	FN	TP
3	975	1	4	0
4	930	17	23	10
5	600	89	116	175
6	380	160	147	293
7	722	82	77	99
8	916	29	12	23
9	977	2	1	0
Total	5500	380	380	600

Table 18: Confusion matrices per class (One-vs-All) for KNN and total values for the White Wine dataset.

Analyzing these results, we observe that, as expected, performance is better for the most frequent classes. For the least frequent classes, we did not correctly classify a single sample.

For both datasets, we obtained similar overall results, which are quite satisfactory considering the characteristics of the data we are working with. Furthermore, all classes were classified, and no class was predicted more than its actual count. This is because the number of over-classified samples is 0, and the number of classified samples matches the dataset size (320 and 980, respectively).

Obviously, the **Weighted avg** should always be prioritized, as the **Macro avg** gives equal weight to classes with very few samples, which may not be representative of overall model performance. When considering the model as a whole, combining all binary classifiers, the **Overall avg** should also be taken into account.

6.2 Naive Bayes

For the Naive Bayes model, we obtained the following results:

For the Red Wine dataset:

Class	Precision	Recall	F1-Score	Support
3	0.00	0.00	0.00	3
4	0.06	0.70	0.11	10
5	0.66	0.78	0.72	127
6	0.57	0.79	0.66	141
7	0.29	0.84	0.43	38
8	0.00	0.00	0.00	1
Macro avg	0.26	0.52	0.32	320
Weighted avg	0.55	0.78	0.63	320
Classified Samples	-	-	-	684
Over-classified Samples	-	-	-	243
Overall Avg	0.36	0.78	0.50	320
Overall Accuracy	-	-	0.74	320

Table 19: Classification report (Only True Class) for Naive Bayes (One-vs-All) on Red Wine dataset.

Class	TN	FP	FN	TP
3	272	45	3	0
4	202	108	3	7
5	143	50	28	99
6	96	83	30	111
7	202	80	6	32
8	250	69	1	0
Total	1165	435	71	249

Table 20: Confusion matrices per class (One-vs-All) for Naive Bayes and total values for the Red Wine dataset (Naive Bayes).

For the White Wine dataset:

Class	Precision	Recall	F1-Score	Support
3	0.01	0.25	0.02	4
4	0.13	0.61	0.21	33
5	0.48	0.69	0.57	291
6	0.48	0.77	0.59	440
7	0.28	0.80	0.42	176
8	0.06	0.71	0.11	35
9	0.00	0.00	0.00	1
Macro avg	0.21	0.55	0.27	980
Weighted avg	0.41	0.74	0.52	980
Classified Samples	-	-	-	2368
Over-classified Samples	-	-	-	864
Overall Avg	0.31	0.74	0.43	980
Overall Accuracy	-	-	0.72	980

Table 21: Classification report (Only True Class) for Naive Bayes (One-vs-All) on White Wine dataset.

Class	TN	FP	FN	TP
3	893	83	3	1
4	810	137	13	20
5	470	219	89	202
6	174	366	103	337
7	445	359	35	141
8	546	399	10	25
9	900	79	1	0
Total	4238	1642	254	726

Table 22: Confusion matrices per class (One-vs-All) for Naive Bayes and total values for the White Wine dataset.

In this model, we achieved **high recall** for both datasets, but at the cost of **low precision**. This led to a large number of **false positives** and a significant amount of **over-classified samples**—approximately **80% more classified samples** in the Red Wine dataset and **143% more** in the White Wine dataset.

It is important to note that over-classified samples refer to instances that were classified into *two or more classes*, which implies that some samples may not have been classified as true for **any class**. Therefore, it is possible for samples to be **not classified at all**.

Even with the previously discussed techniques, the model continues to perform poorly in terms of precision and is likely not suitable for this task.

6.3 Support Vector Machine

For the Support Vector Machine (SVM) model, we obtained the following results:

For the Red Wine dataset:

Class	Precision	Recall	F1-Score	Support
3	0.00	0.00	0.00	3
4	0.12	0.50	0.20	10
5	0.69	0.83	0.75	127
6	0.64	0.74	0.69	141
7	0.36	0.74	0.49	38
8	0.00	0.00	0.00	1
Macro avg	0.30	0.47	0.35	320
Weighted avg	0.60	0.76	0.67	320
Classified Samples	-	-	-	461
Over-classified Samples	-	-	-	132
Overall Avg	0.53	0.76	0.62	320
Overall Accuracy	-	-	0.85	320

Table 23: Classification report (Only True Class) for SVM (One-vs-All) on Red Wine dataset.

Class	TN	FP	FN	TP
3	313	4	3	0
4	274	36	5	5
5	146	47	22	105
6	119	60	36	105
7	233	49	10	28
8	297	22	1	0
Total	1382	218	77	243

Table 24: Confusion matrices per class (One-vs-All) for SVM and total values for the Red Wine dataset using SVM.

For the White Wine dataset:

Class	Precision	Recall	F1-Score	Support
3	0.00	0.00	0.00	4
4	0.13	0.52	0.21	33
5	0.54	0.73	0.62	291
6	0.56	0.70	0.62	440
7	0.37	0.80	0.51	176
8	0.14	0.71	0.23	35
9	0.00	0.00	0.00	1
Macro avg	0.25	0.49	0.31	980
Weighted avg	0.49	0.71	0.57	980
Classified Samples	-	-	-	1645
Over-classified Samples	-	-	-	566
Overall Avg	0.42	0.71	0.53	980
Overall Accuracy	-	-	0.82	980

Table 25: Classification report (Only True Class) for SVM (One-vs-All) on White Wine dataset.

Class	TN	FP	FN	TP
3	959	17	4	0
4	832	115	16	17
5	511	178	80	211
6	300	240	134	306
7	569	235	36	140
8	787	158	10	25
9	976	3	1	0
Total	4934	946	281	699

Table 26: Confusion matrices per class (One-vs-All) for SVM and total values for the White Wine dataset.

In this model, we achieved good recall scores for both datasets. For the Red Wine dataset, the Support Vector Machine (SVM) showed a better balance between precision and recall. Although the number of classified samples exceeded the number of test samples by 44%, the model maintained a reasonable performance, particularly for the most frequent classes (5 and 6), where both precision and recall were relatively high. This suggests that the model generalizes well for these classes and may be considered a strong candidate.

On the other hand, the performance on the White Wine dataset was notably less balanced. Although recall was still decent, precision values dropped significantly, especially for the less frequent classes. This led to a 68% increase in classified samples compared to the original test set, which indicates a considerable over-classification. This behavior may suggest that the SVM model is less suitable for datasets with a larger class imbalance, such as the White Wine dataset.

6.4 Logistic Regression

For the Logistic Regression model, we obtained the following results:

For the Red Wine dataset:

Class	Precision	Recall	F1-Score	Support
3	0.00	0.00	0.00	3
4	0.11	0.70	0.19	10
5	0.68	0.81	0.74	127
6	0.58	0.71	0.64	141
7	0.31	0.79	0.44	38
8	0.03	1.00	0.06	1
Macro avg	0.29	0.67	0.35	320
Weighted avg	0.57	0.75	0.63	320
Classified Samples	-	-	-	530
Over-classified Samples	-	-	-	171
Overall Avg	0.45	0.75	0.57	320
Overall Accuracy	-	-	0.81	320

Table 27: Classification report (Only True Class) for Logistic Regression (One-vs-All) on Red Wine dataset.

Class	TN	FP	FN	TP
3	304	13	3	0
4	254	56	3	7
5	145	48	24	103
6	106	73	41	100
7	215	67	8	30
8	287	32	0	1
Total	1311	289	79	241

Table 28: Confusion matrices per class (One-vs-All) for Logistic Regression and total values for Logistic Regression on the Red Wine dataset.

For the White Wine dataset:

Class	Precision	Recall	F1-Score	Support
3	0.01	0.50	0.01	4
4	0.09	0.61	0.16	33
5	0.49	0.73	0.59	291
6	0.51	0.65	0.57	440
7	0.33	0.74	0.46	176
8	0.07	0.63	0.13	35
9	0.04	1.00	0.08	1
Macro avg	0.22	0.69	0.29	980
Weighted avg	0.44	0.69	0.52	980
Classified Samples	-	-	-	2194
Over-classified Samples	-	-	-	780
Overall Avg	0.31	0.69	0.42	980
Overall Accuracy	-	-	0.73	980

Table 29: Classification report (Only True Class) for Logistic Regression (One-vs-All) on White Wine dataset.

Class	TN	FP	FN	TP
3	710	266	2	2
4	749	198	13	20
5	471	218	78	213
6	262	278	155	285
7	541	263	46	130
8	669	276	13	22
9	957	22	0	1
Total	4359	1521	307	673

Table 30: Confusion matrices per class (One-vs-All) for Logistic Regression and total values for the White Wine dataset.

For the red wine dataset, the model achieved decent recall and moderate precision, indicating a fair balance. Specifically, it performed well on classes 5 and 6, demonstrating good class discrimination. However, for minority classes like 3 and 8,

the performance was poor, as expected. The model over-classified, producing 65% more predictions than the number of true test samples.

For the white wine dataset, although the recall remained acceptable, precision was considerably lower due to a large number of false positives. This suggests the model often predicted the correct class but lacked confidence or consistency across classes. The number of classified samples exceeded the actual test set by 124%, indicating substantial over-classification.

6.5 Decision Tree Classifier

The Decision Tree Classifier produced the following results:

For Red Wine:

Class	Precision	Recall	F1-Score	Support
3	0.00	0.00	0.00	3
4	0.09	0.10	0.10	10
5	0.62	0.72	0.67	127
6	0.61	0.52	0.56	141
7	0.34	0.32	0.33	38
8	0.00	0.00	0.00	1
Macro avg	0.28	0.28	0.28	320
Weighted avg	0.56	0.56	0.56	320
Classified Samples	-	-	-	320
Over-classified Samples	-	-	-	57
Overall Avg	0.56	0.56	0.56	320
Overall Accuracy	-	-	0.85	320

Table 31: Classification report (Only True Class) for Decision Tree Classifier (One-vs-All) on Red Wine dataset.

Class	TN	FP	FN	TP
3	315	2	3	0
4	300	10	9	1
5	138	55	36	91
6	132	47	67	74
7	259	23	26	12
8	314	5	1	0
Total	1458	142	142	178

Table 32: Confusion matrices per class (One-vs-All) for Decision Tree Classifier and total values for the Red Wine dataset.

For White Wine:

Class	Precision	Recall	F1-Score	Support
3	0.00	0.00	0.00	4
4	0.28	0.30	0.29	33
5	0.58	0.57	0.58	291
6	0.60	0.62	0.61	440
7	0.54	0.47	0.50	176
8	0.38	0.31	0.34	35
9	0.00	0.00	0.00	1
Macro avg	0.34	0.33	0.33	980
Weighted avg	0.56	0.55	0.56	980
Classified Samples	-	-	-	958
Over-classified Samples	-	-	-	187
Overall Avg	0.57	0.55	0.56	980
Overall Accuracy	-	-	0.88	980

Table 33: Classification report (Only True Class) for Decision Tree Classifier (One-vs-All) on White Wine dataset.

Class	TN	FP	FN	TP
3	974	2	4	0
4	921	26	23	10
5	570	119	125	166
6	360	180	168	272
7	734	70	93	83
8	927	18	24	11
9	978	1	1	0
Total	5464	416	438	542

Table 34: Confusion matrices per class (One-vs-All) for Decision Tree Classifier and total values for the White Wine dataset.

The Decision Tree Classifier yielded similar performance metrics for both the red and white wine datasets. The results indicate a balanced relationship between precision and recall, although the overall performance remained moderate.

It is important to highlight that although the number of classified samples is comparable to the test dataset size, this does not imply that all samples were correctly classified. Rather, it suggests that the classifier made a confident prediction for most samples, but not necessarily accurate ones. Consequently, while precision may appear slightly higher, this does not guarantee superior overall model performance.

In summary, despite the balanced precision and recall, the Decision Tree Classifier showed limitations in handling class imbalance and generalization, which affected its effectiveness, particularly for less frequent quality scores.

6.6 Random Forest Classifier

For the Random Forest Classifier, we obtained the following results:

For the Red Wine dataset:

Class	Precision	Recall	F1-Score	Support
3	0.00	0.00	0.00	3
4	0.50	0.10	0.17	10
5	0.72	0.80	0.76	127
6	0.70	0.51	0.59	141
7	0.62	0.34	0.44	38
8	0.00	0.00	0.00	1
Macro avg	0.42	0.29	0.33	320
Weighted avg	0.68	0.58	0.62	320
Classified Samples	-	-	-	266
Over-classified Samples	-	-	-	9
Overall Avg	0.70	0.58	0.64	320
Overall Accuracy	-	-	0.89	320

Table 35: Classification report (Only True Class) for Random Forest Classifier (One-vs-All) on Red Wine dataset.

Class	TN	FP	FN	TP
3	317	0	3	0
4	309	1	9	1
5	154	39	26	101
6	148	31	69	72
7	274	8	25	13
8	319	0	1	0
Total	1521	79	133	187

Table 36: Confusion matrices per class (One-vs-All) for Random Forest Classifier and total values for the Red Wine dataset using RFC.

For the White Wine dataset:

Class	Precision	Recall	F1-Score	Support
3	0.00	0.00	0.00	4
4	0.75	0.18	0.29	33
5	0.69	0.64	0.66	291
6	0.65	0.68	0.66	440
7	0.73	0.47	0.57	176
8	0.83	0.29	0.43	35
9	0.00	0.00	0.00	1
Macro avg	0.52	0.32	0.37	980
Weighted avg	0.68	0.59	0.62	980
Classified Samples	-	-	-	864
Over-classified Samples	-	-	-	48
Overall Avg	0.67	0.59	0.63	980
Overall Accuracy	-	-	0.90	980

Table 37: Classification report (Only True Class) for Random Forest Classifier (One-vs-All) on White Wine dataset.

Class	TN	FP	FN	TP
3	976	0	4	0
4	945	2	27	6
5	604	85	104	187
6	378	162	143	297
7	773	31	94	82
8	943	2	25	10
9	979	0	1	0
Total	5598	282	398	582

Table 38: Confusion matrices per class (One-vs-All) for Random Forest Classifier and total values for the White Wine dataset.

The Random Forest Classifier achieved reasonable performance overall. It showed good precision and acceptable recall for most classes, but at the cost of a reduced number of classified samples. This trade-off led to a notable number of instances not being classified as true for any class, which can be problematic depending on the application. We will discuss this behavior in more detail later.

Looking specifically at the Red Wine dataset, class 5 stood out with particularly strong performance, balancing both precision and recall effectively. Interestingly, class 5 was not the most represented in the dataset, which makes this result even more notable.

6.7 Neural Network

In order to find the best hyperparameters for each binary Neural Network model, we spent a significant amount of time processing the data. The tested hyperparameter combinations were:

- Number of nodes: {16, 32, 64}
- Dropout probabilities: {0, 0.2}
- Learning rates: {0.01, 0.005, 0.001}
- Batch sizes: {32, 64, 128}

Since this process was repeated for each binary classification model, the total number of evaluations per class was:

$$3 \times 2 \times 3 \times 3 \times n = 54 \times n$$

where n represents the number of binary classification models. Each experiment required considerable processing time, making the overall optimization process computationally expensive.

However, once the optimal hyperparameters were obtained, there was no need to repeat the process. In theory, we now have the most suitable model in terms of hyperparameter selection.

Red Wine - Best Hyperparameters:

- Class 3: 34 nodes, dropout 0, learning rate 0.001, batch size 64

- Class 4: 34 nodes, dropout 0, learning rate 0.01, batch size 128
- Class 5: 34 nodes, dropout 0, learning rate 0.01, batch size 128
- Class 6: 34 nodes, dropout 0, learning rate 0.01, batch size 64
- Class 7: 34 nodes, dropout 0.2, learning rate 0.005, batch size 32
- Class 8: 34 nodes, dropout 0, learning rate 0.001, batch size 64

With these parameters, we obtained the following results:

Class	Precision	Recall	F1-Score	Support
3	0.00	0.00	0.00	3
4	0.33	0.20	0.25	10
5	0.63	0.67	0.65	127
6	0.62	0.56	0.59	141
7	0.47	0.53	0.49	38
8	0.00	0.00	0.00	1
Macro avg	0.34	0.33	0.33	320
Weighted avg	0.59	0.58	0.58	320
Classified Samples	-	-	-	317
Over-classified Samples	-	-	-	43
Overall Avg	0.59	0.58	0.58	320
Overall Accuracy	-	-	0.86	320

Table 39: Classification report (Only True Class) for NN (One-vs-All) on Red Wine dataset.

Class	TN	FP	FN	TP
3	316	1	3	0
4	306	4	8	2
5	144	49	42	85
6	130	49	62	79
7	259	23	18	20
8	314	5	1	0
Total	1469	131	134	186

Table 40: Confusion matrices per class (One-vs-All) and total values for the Red Wine dataset.

White Wine - Best Hyperparameters:

- Class 3: 34 nodes, dropout 0.2, learning rate 0.001, batch size 32
- Class 4: 34 nodes, dropout 0, learning rate 0.005, batch size 64
- Class 5: 34 nodes, dropout 0, learning rate 0.01, batch size 128
- Class 6: 32 nodes, dropout 0, learning rate 0.01, batch size 128

- Class 7: 34 nodes, dropout 0.2, learning rate 0.005, batch size 64
- Class 8: 34 nodes, dropout 0, learning rate 0.01, batch size 32
- Class 9: 32 nodes, dropout 0, learning rate 0.01, batch size 32

With these parameters, we obtained the following results:

Class	Precision	Recall	F1-Score	Support
3	0.00	0.00	0.00	4
4	0.44	0.42	0.43	33
5	0.58	0.56	0.57	291
6	0.62	0.55	0.58	440
7	0.46	0.65	0.54	176
8	0.54	0.54	0.54	35
9	0.00	0.00	0.00	1
Macro avg	0.38	0.39	0.38	980
Weighted avg	0.57	0.57	0.56	980
Classified Samples	-	-	-	999
Over-classified Samples	-	-	-	189
Overall Avg	0.55	0.57	0.56	980
Overall Accuracy	-	-	0.87	980

Table 41: Classification report (Only True Class = 1) for RFC (One-vs-All) on White Wine dataset.

Class	TN	FP	FN	TP
3	974	2	4	0
4	929	18	19	14
5	571	118	129	162
6	389	151	196	244
7	667	137	61	115
8	929	16	16	19
9	976	3	1	0
Total	5435	445	426	554

Table 42: Confusion matrices per class (One-vs-All) and total values for the White Wine dataset.

Overall, the Neural Network model exhibited comparable performance on both datasets. While the results were acceptable, the model did not particularly excel in any class. The extensive search for optimal hyperparameters may have contributed to a balanced but suboptimal performance across classes. Notably, in the red wine dataset, three fewer samples were classified compared to the original test set. Conversely, the white wine dataset showed a 0.2% increase in classified samples, due to over-classification.

6.8 Conclusion

After analyzing the results of all the models using the one-vs-all approach, we were able to identify the most suitable candidates to represent this strategy in further analyses across all methodologies.

To make these selections, we compared models class-by-class and considered several important aspects. For instance, a high recall paired with low precision means the model correctly identified many true samples from a class but also misclassified many others as belonging to it. This trade-off was common in our experiments. Therefore, it was generally more meaningful to evaluate the F1-score, which balances both precision and recall. In cases of similar F1 results, we favored the model with the best overall balance between these two metrics.

This approach is particularly challenging because, as discussed in the introduction of this chapter, once a model is chosen, we can rank the binary models by performance and remove samples from the test set as they are classified. Doing so avoids over-classification and could be advantageous. In such a scenario, a model that tends to classify more samples as positive may be preferred—especially if recall is already high—since this strategy would boost precision without harming recall. For this reason, we selected two models for each dataset: one for a high-reward strategy (favoring recall) and one for a low-risk strategy (favoring precision).

The chosen models were:

- **Red Wine Dataset:**
  - High Reward: **Support Vector Machine** — showed the best recall with reasonable precision. Its main competitor was the Naive Bayes model.
  - Low Risk: **Random Forest Classifier** — achieved the best precision along with average recall. Its main competitor was K-Nearest Neighbors (KNN), but the Random Forest performed better overall.
- **White Wine Dataset:**
  - High Reward: **Support Vector Machine** — also demonstrated the best recall with acceptable precision. Again, its main competitor was the Naive Bayes model.
  - Low Risk: **Random Forest Classifier** — had the best precision with average recall. Its main competitor was KNN, but overall, the Random Forest was superior.

7 Models - Clustering into 3 Classes

For the third approach, we grouped the original quality scores into three broader categories. This binning strategy helps simplify the classification task and potentially improves performance, especially when dealing with imbalanced data. The bins we used were:

- Bad Wine (0) $\rightarrow 0 \leq \text{quality} < 5.5$
- Medium Wine (1) $\rightarrow 5.5 \leq \text{quality} < 7.5$

- Good Wine (2) $\rightarrow 7.5 \leq$ quality ≤ 10

This approach can be useful depending on the goal of the classification. If the objective is to predict an exact quality score, this method may not be ideal. However, if the aim is to determine whether a wine is generally of good enough quality for purchase or recommendation (e.g., for a restaurant or household), this strategy simplifies the problem while still retaining useful information.

Unlike previous models, where predictions focused on specific scores, here we reduce the number of classes to improve model performance. It also mitigates the effect of label imbalance, which we observed previously, where over- and under-sampling strategies did not significantly enhance the results.

Additionally, a three-class division provides more nuance than a simple good/bad split, which might group together wines of very different qualities. A "medium" category allows for more flexible decisions—e.g., a medium wine might still be a good choice if the price is fair.

This strategy can be particularly beneficial in scenarios such as:

- Recommending wines for different customer profiles based on budget and quality expectations.
- Building a quick assessment tool for sommeliers or distributors.
- Designing a simplified user experience for wine apps or online stores.

This kind of classification approach is used in many other domains as well. For example, in the financial sector, it's commonly applied for credit scoring. Instead of predicting the exact credit score, institutions often group clients into categories such as:

- High Risk (0): credit score < 550
- Medium Risk (1): $550 \leq$ score < 700
- Low Risk (2): score ≥ 700

This simplifies the decision-making process, especially when the exact value is less important than the risk group. It also allows for easier automation in bank or fintech approval systems.

Other real-life applications where this kind of approach proves valuable include:

- **Weather Forecasting**: instead of needing the exact temperature, categorizing the day as Cold, Mild, or Hot is often enough to decide what to wear or plan outdoor activities.
- **Rain Prediction**: you don't need to know the exact probability of rain to decide whether to bring an umbrella—just knowing that rain is "likely" is enough.
- **Academic Performance**: grouping grades into Bad, Medium, and Good can help in generating clearer and more interpretable student performance reports.

- **Pain Level Classification in Healthcare**: grouping sensor data or subjective scales into Mild, Moderate, and Intense assists in medical triage.
- **Customer Satisfaction**: transforming a 0–10 score into categories like Unsatisfied, Neutral, and Satisfied makes dashboards and KPIs more digestible.

These use cases share the same idea: when dealing with imbalanced, fuzzy, or subjective data, clustering into fewer but meaningful classes can improve both model performance and result interpretability. Sometimes, an approximate but clear answer is not just sufficient—it's more useful than a precise but noisy prediction.

In this section, we apply the standard multiclass strategy for each model. For models that are natively binary, such as SVM and Logistic Regression, we adopt the One-vs-One (OvO) approach. Since we are now working with only three classes, it does not make much sense to compare OvO versus OvA techniques here. Instead, our focus is on analyzing the impact of clustering the target variable into broader categories. The use of OvO in this case remains appropriate and consistent with the typical handling of binary classifiers in multiclass settings.

Now, as we've done before with the other approaches, we will use the same model but with this new dataset:

7.1 K-Nearest Neighbors (KNN)

For the K-Nearest Neighbors (KNN) model, after testing different values for the number of neighbors, the best-performing configuration was found with $k = 1$. Below are the classification results for both the Red Wine and White Wine datasets.

For the Red Wine dataset:

Class	Precision	Recall	F1-Score	Support
0	0.69	0.76	0.73	140
1	0.78	0.71	0.74	176
2	0.25	0.25	0.25	4
Accuracy			0.73	320
Macro avg	0.57	0.57	0.57	320
Weighted avg	0.73	0.73	0.73	320

Table 43: Classification report for the KNN model on the Red Wine dataset

For the White Wine dataset:

Class	Precision	Recall	F1-Score	Support
0	0.67	0.65	0.66	329
1	0.79	0.80	0.80	619
2	0.43	0.41	0.42	32
Accuracy			0.74	980
Macro avg	0.63	0.62	0.63	980
Weighted avg	0.74	0.74	0.74	980

Table 44: Classification report for the KNN model on the White Wine dataset

Although this approach resulted in a well-performing model, class imbalance remains an issue, particularly for the Good Wine category (class 2), which has significantly fewer samples. While adjusting the binning thresholds to balance the class distribution could improve the classification performance, doing so would introduce bias and compromise the integrity of the classification task. Therefore, any modifications should be carefully evaluated in the context of the dataset and the problem being addressed.

7.2 Naive Bayes

For the Naïve Bayes model, we obtained the following results:

For the Red Wine:

Class	Precision	Recall	F1-Score	Support
0	0.72	0.76	0.74	140
1	0.72	0.49	0.58	176
2	0.02	0.25	0.04	4
Accuracy			0.60	320
Macro avg	0.49	0.50	0.45	320
Weighted avg	0.71	0.60	0.64	320

Table 45: Classification report for the Naïve Bayes model on the Red Wine dataset

For the White Wine:

Class	Precision	Recall	F1-Score	Support
0	0.56	0.73	0.64	329
1	0.73	0.18	0.29	619
2	0.06	0.69	0.10	32
Accuracy			0.38	980
Macro avg	0.45	0.53	0.34	980
Weighted avg	0.65	0.38	0.40	980

Table 46: Classification report for the Naïve Bayes model on the White Wine dataset

Analyzing the results, the performance of this model was not satisfactory. One particular observation stands out: class 0 achieved better scores than class 1 for both wines. This is unusual since class 1 contains a significantly larger number of samples—over 20% more in the red wine dataset and an even greater difference in the white wine dataset. Investigating the reasons behind this discrepancy could provide further insights, as this behavior contrasts with the trends observed in the rest of the study.

7.3 Support Vector Machine

For the Support Vector Machine (SVM) model, we obtained the following results:

For the Red Wine dataset:

Class	Precision	Recall	F1-Score	Support
0	0.69	0.86	0.77	140
1	0.84	0.64	0.73	176
2	0.08	0.25	0.12	4
Accuracy			0.73	320
Macro avg	0.54	0.58	0.54	320
Weighted avg	0.76	0.73	0.74	320

Table 47: Classification report for the SVM model on the Red Wine dataset

For the White Wine dataset:

Class	Precision	Recall	F1-Score	Support
0	0.60	0.76	0.67	329
1	0.81	0.55	0.66	619
2	0.13	0.59	0.21	32
Accuracy			0.62	980
Macro avg	0.52	0.64	0.52	980
Weighted avg	0.72	0.62	0.65	980

Table 48: Classification report for the SVM model on the White Wine dataset

The SVM model achieved relatively good precision across both datasets, particularly for class 1. Additionally, it demonstrated strong recall for class 0 in both cases, indicating that it correctly identified most samples of this class. However, performance for class 2 was notably poor in both datasets, with very low precision and F1-score, suggesting difficulty in distinguishing this minority class.

Overall, the model performed better on the Red Wine dataset, achieving a higher accuracy (0.73) compared to the White Wine dataset (0.62). This could be due to differences in class distribution or feature relevance between the datasets, which should be further analyzed.

7.4 Logistic Regression

For the Logistic Regression model, we obtained the following results:

For the Red Wine dataset:

Class	Precision	Recall	F1-Score	Support
0	0.70	0.84	0.77	140
1	0.79	0.52	0.63	176
2	0.03	0.25	0.05	4
Accuracy			0.66	320
Macro avg	0.51	0.54	0.48	320
Weighted avg	0.74	0.66	0.68	320

Table 49: Classification report for the Logistic Regression model on the Red Wine dataset

For the White Wine dataset:

Class	Precision	Recall	F1-Score	Support
0	0.56	0.75	0.64	329
1	0.76	0.30	0.43	619
2	0.07	0.62	0.12	32
Accuracy			0.46	980
Macro avg	0.46	0.56	0.40	980
Weighted avg	0.67	0.46	0.49	980

Table 50: Classification report for the Logistic Regression model on the White Wine dataset

The Logistic Regression model performed better on the Red Wine dataset, achieving higher accuracy. Class 0 was well classified in both datasets, but class 1 had low recall, especially in the White Wine dataset. Class 2 was the hardest to classify, likely due to its low support.

7.5 Decision Tree Classifier

For the Decision Tree Classifier model, we obtained the following results:

For the Red Wine dataset:

Class	Precision	Recall	F1-Score	Support
0	0.65	0.79	0.72	140
1	0.78	0.64	0.70	176
2	0.00	0.00	0.00	4
Accuracy			0.70	320
Macro avg	0.48	0.48	0.47	320
Weighted avg	0.71	0.70	0.70	320

Table 51: Classification report for the Decision Tree Classifier model on the Red Wine dataset

For the White Wine dataset:

Class	Precision	Recall	F1-Score	Support
0	0.59	0.61	0.60	329
1	0.76	0.74	0.75	619
2	0.31	0.38	0.34	32
Accuracy			0.68	980
Macro avg	0.55	0.57	0.56	980
Weighted avg	0.69	0.68	0.69	980

Table 52: Classification report for the Decision Tree Classifier model on the White Wine dataset

The Decision Tree model achieved 70% accuracy for Red Wine and 68% for White Wine. Class 0 had the highest recall in Red Wine, while class 1 performed best in White Wine. The model struggled with class 2 due to low support, affecting the macro averages.

7.6 Random Forest Classifier

For the Random Forest Classifier, we obtained the following results:

For the Red Wine dataset:

Class	Precision	Recall	F1-Score	Support
0	0.74	0.84	0.78	140
1	0.83	0.76	0.80	176
2	0.00	0.00	0.00	4
Accuracy			0.78	320
Macro avg	0.52	0.53	0.53	320
Weighted avg	0.78	0.78	0.78	320

Table 53: Classification report for the Random Forest Classifier on the Red Wine dataset

For the White Wine dataset:

Class	Precision	Recall	F1-Score	Support
0	0.74	0.72	0.73	329
1	0.82	0.86	0.84	619
2	0.79	0.34	0.48	32
Accuracy			0.79	980
Macro avg	0.78	0.64	0.68	980
Weighted avg	0.79	0.79	0.79	980

Table 54: Classification report for the Random Forest Classifier on the White Wine dataset

The Random Forest model performed consistently well, with solid scores across both datasets. The only notable issue was the poor performance for class 2 in Red Wine, likely due to the low number of samples. Interestingly, class 2 in White Wine achieved relatively good precision despite having fewer than 5% of the samples. Overall, the weighted metrics indicate strong performance, making this model a solid candidate.

7.7 Neural Network

For the Neural Network approach, we experimented with different hyperparameter configurations using the following ranges:

- Number of nodes: {16, 32, 64}
- Dropout probabilities: {0, 0.2}
- Learning rates: {0.01, 0.005, 0.001}
- Batch sizes: {32, 64, 128}

After experimentation, the best parameters for the Red Wine dataset were:

- Number of nodes: 34
- Dropout probability: 0
- Learning rate: 0.001
- Batch size: 32

Using these parameters, the model achieved the following results:

Class	Precision	Recall	F1-Score	Support
0	0.69	0.80	0.74	140
1	0.80	0.69	0.74	176
2	0.25	0.25	0.25	4
Accuracy			0.73	320
Macro avg	0.58	0.58	0.58	320
Weighted avg	0.74	0.73	0.73	320

Table 55: Classification report for the Neural Network model on the Red Wine dataset

For the White Wine dataset, the best parameters were:

- Number of nodes: 34
- Dropout probability: 0
- Learning rate: 0.001
- Batch size: 64

With these parameters, the results were:

Class	Precision	Recall	F1-Score	Support
0	0.61	0.67	0.64	329
1	0.78	0.73	0.75	619
2	0.31	0.41	0.35	32
Accuracy			0.70	980
Macro avg	0.57	0.60	0.58	980
Weighted avg	0.71	0.70	0.70	980

Table 56: Classification report for the Neural Network model on the White Wine dataset

Analyzing the results, we observe that the Neural Network performed better on the Red Wine dataset, achieving a more balanced performance across the classes. Notably, the model provided at least some classification for class 2, avoiding a complete failure (zero-score) as seen in some other models. However, this does not necessarily indicate strong performance for class 2, as it also introduces a higher number of false positives, which should be considered. A higher recall in this case

means more samples of class 2 were identified, but not necessarily correctly classified, which could lead to misinterpretations of the model's reliability for this class.

For the White Wine dataset, the model's overall accuracy was slightly lower, and class 2 still struggled with relatively low precision and recall. The increase in recall for class 2 in White Wine suggests that the model is making more predictions for this class, but its lower precision means many of these classifications are incorrect. This highlights a trade-off that needs to be considered when analyzing the performance of minority classes.

7.8 Conclusion

The overall results of this approach indicate a more balanced classification performance, especially after grouping some of the less representative sampled classes. This grouping improved the dataset's class distribution and likely made inter-class differences more noticeable.

However, class 2 in the red wine dataset still contained only four samples. As such, its influence on the final results is limited, especially when considering weighted averages. In cases where all models performed poorly, this class was not a determining factor.

To select the best model for each dataset, we prioritized weighted average metrics—particularly the F1-score and accuracy—while also reviewing per-class performance.

Based on these criteria, the selected models were:

- **Red Wine: Random Forest Classifier** — achieved the best overall F1-score. Although it failed to classify class 2 due to the limited data, its performance on the other classes was superior to the alternatives.
- **White Wine: Random Forest Classifier** — demonstrated the best performance across all classes, including the lower-sampled class 2.

8 Comparing the Techniques

So after showing the results for each approach we are going to compare them to each other, an interesting thing there's a better or the worst one, it will depend in each application you will use this classification technique, so it might be a little bit subjective, this will be analyzed also.

A thing that was comum for all of was obvius the imbalanced data made the task even harder for all of them, but because in this example/comparisson of the approches we are using the same datasets it happend to all of them, if the data was more balanced, more uniform, the comparisson would still be valid, cause it would increase the ratings of all of them almost proportionally.

Now we gonna present the comparissons, as we already looked for all the models, we now going to take the chosen models more into consideration, because they were the best among the each tecniques.

8.1 One-vs-One and Native Multiclass vs One-vs-All

8.1.1 Red Wine

For this dataset, we had one native multiclass model (Neural Network) as chosen in Section 5, and in Section 6, two models were selected: the high-reward model (SVM) and the low-risk model (Random Forest Classifier), which are native binary and multiclass models, respectively.

By analyzing the results, we notice that the scores for the high-reward and low-risk models were generally better in most aspects when compared to the Neural Network approach. One-vs-All is usually more suitable for unbalanced datasets. Interestingly, the One-vs-All approach still shows considerable room for improvement, as previously discussed.

It is also useful to compare the same models across different approaches to better understand their behavior and performance differences. We present the classification metric differences as follows:

$$\text{Result} = (\text{One-vs-All}) - (\text{One-vs-One or Native Multiclass})$$

SVM: A native binary model, compared here as One-vs-All vs One-vs-One:

Class	Precision Diff	Recall Diff	F1-Score Diff
3	0.00	0.00	0.00
4	-0.05	-0.08	-0.06
5	0.07	0.28	0.17
6	0.08	0.36	0.24
7	0.03	0.23	0.09
8	0.00	0.00	0.00
Macro avg	0.02	0.13	0.07
Weighted avg	0.06	0.29	0.18

Table 57: Difference in Classification Metrics: (One-vs-All) – (One-vs-One) for SVM on the Red Wine dataset.

Random Forest Classifier: Compared as One-vs-All vs Native Multiclass:

Class	Precision Diff	Recall Diff	F1-Score Diff
3	0.00	0.00	0.00
4	0.20	-0.15	-0.10
5	0.11	0.27	0.19
6	0.19	-0.10	0.03
7	0.37	0.12	0.21
8	0.00	0.00	0.00
Macro avg	0.14	0.02	0.06
Weighted avg	0.17	0.07	0.12

Table 58: Difference in Classification Metrics: (One-vs-All) – (Native Multiclass) for Random Forest on the Red Wine dataset.

Neural Network: Again compared as One-vs-All vs Native Multiclass:

Class	Precision Diff	Recall Diff	F1-Score Diff
3	0.00	0.00	0.00
4	0.19	0.03	0.10
5	-0.07	0.02	-0.02
6	0.03	-0.02	0.01
7	-0.01	-0.01	-0.02
8	-0.12	-0.25	-0.17
Macro avg	0.00	-0.03	-0.02
Weighted avg	0.00	0.00	-0.01

Table 59: Difference in Classification Metrics: (One-vs-All) – (Native Multiclass) for Neural Network on the Red Wine dataset.

From the results, we observe that the One-vs-All approach generally performs better, especially when looking at the weighted scores. The SVM, a native binary model, showed the most significant improvement when using One-vs-All compared to One-vs-One, with better results and reduced complexity in training time.

For the native multiclass models, the results were more balanced. In particular, the Neural Network had very similar scores across both approaches, with only minor variations. The Random Forest Classifier performed slightly better in the One-vs-All strategy, especially in terms of precision and overall F1-score, with a +0.12 improvement in the latter.

8.1.2 White Wine

To further explore this context, we analyze the white wine dataset. Although similar to the red wine dataset, it presents unique characteristics while suffering from the same class imbalance issue.

As chosen in Section 6, the best-performing model was the native binary SVM due to its high reward potential, and for the low risk it was the Random Forest Classifier. In Section 5, the selected model was the Random Forest Classifier.

This time, the results were different. To better understand the differences, we first present the comparison table as follows:

$$\text{Result} = (\text{One-vs-All}) - (\text{One-vs-One or Native Multiclass})$$

Random Forest Classifier: One-vs-All vs Native Multiclass

Class	Precision Diff	Recall Diff	F1-Score Diff
3	-1.00	0.00	0.00
4	0.13	-0.06	-0.06
5	-0.01	-0.02	-0.02
6	0.01	-0.11	-0.05
7	0.06	-0.03	0.00
8	-0.17	-0.22	-0.25
9	-1.00	0.00	0.00
Macro avg	-0.28	-0.07	-0.06
Weighted avg	0.00	-0.08	-0.04

Table 60: Difference in Classification Metrics: (One-vs-All) – (Native Multiclass) for Random Forest on the White Wine dataset.

SVM: One-vs-All vs Native Binary (One-vs-One)

Class	Precision Diff	Recall Diff	F1-Score Diff
3	-1.00	0.00	0.00
4	-0.87	0.52	0.21
5	-0.06	0.72	0.60
6	0.11	-0.30	0.00
7	-0.63	0.80	0.51
8	-0.86	0.71	0.23
9	-1.00	0.00	0.00
Macro avg	-0.61	0.35	0.22
Weighted avg	-0.14	0.26	0.29

Table 61: Comparison of Classification Metrics: (SVM One-vs-All) – (SVM Binary Native) on the White Wine dataset.

An important consideration is that in the classification report from Section 5 (One-vs-One and Native Multiclass), metrics are computed over **all predicted samples**, so a precision of 1.00 may occur when there are no positive predictions (i.e., precision = 0 / (0 + 0), which is undefined, but libraries like `sklearn` typically assign 1.0 in such cases to avoid errors).

In contrast, in Section 6 (One-vs-All), since metrics are computed only over the **true classes**, the precision in the same scenario is 0. This distinction significantly impacts the macro average, but not the weighted average as much, since this issue tends to affect only the least represented classes. For this reason, **F1-score** becomes a better overall metric to analyze, as it naturally drops to 0 if either precision or recall is 0:

$$\text{F1-Score} = \frac{2 \cdot \text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}$$

Regarding the SVM classifier, the One-vs-All strategy consistently outperformed the native binary implementation (One-vs-One), especially on imbalanced data. The improvements were notable even on the majority classes. It resulted in a **+0.29 increase in weighted F1-score**, which is quite significant.

On the other hand, the Random Forest classifier showed balanced behavior across both strategies. The differences were minimal and often compensated across classes. The final performance in terms of **F1-score** was almost the same, with **-0.04 in weighted** and **-0.06 in macro average**.

For the white wine dataset, the **native multiclass Random Forest classifier** is preferred. Despite the close results, it offers better performance with significantly lower computational cost, as it avoids running N separate binary models. While SVM One-vs-All provides promising results in terms of reward on highly imbalanced data, its computational expense makes it less appealing unless a substantial performance gain can be guaranteed using the techniques discussed earlier.

8.2 One-vs-One and Native Multiclass vs. Clustering into 3 Classes

In this comparison, we cannot present a classification difference table as in previous sections, since the number of classes is different and not directly comparable. Therefore, the evaluation method must be adjusted accordingly.

8.2.1 Red Wine

In this case, the model selected in Section 5 was the Neural Network (native multiclass), and the model selected in Section 7 was the Random Forest Classifier using the clustering technique.

We will compare the results using only the macro and weighted average scores for accuracy, precision, recall, and F1-score:

Result = (One-vs-One and Native Multiclass) – (Clustering into 3 Classes)

We are comparing the best-performing model from each approach. Although both are native multiclass models, they operate with a different number of target classes.

Difference Table – Neural Network (native multiclass) vs. Random Forest Classifier (clustering into 3 classes):

	Precision	Recall	F1-Score
Accuracy			-0.20
Macro avg	-0.18	-0.17	-0.18
Weighted avg	-0.19	-0.20	-0.19

Table 62: Differences in classification metrics between One-vs-One and Native Multiclass and Clustering into 3 Classes for the Red Wine dataset.

As expected, the clustering technique described in Section 7 performed significantly better than using all original classes. However, data imbalance remains a major issue — for instance, only four samples were labeled as class 8, which compromises class representation.

In the end, the appropriate method depends on your objectives. If predicting the exact score is not essential, clustering can offer considerable performance benefits. However, if exact class prediction is critical, using all original classes — despite the performance hit — becomes necessary.

8.2.2 White Wine

Now considering another dataset to support a more comprehensive conclusion: for the white wine data, the selected model in Section 5 and Section 7 was the Random Forest Classifier. This model is a native multiclass classifier, so the one-vs-one strategy was not applied in this case.

Thus, we will again compare the accuracy, macro average, and weighted average for both approaches.

Difference Table – Random Forest Classifier (native multiclass) vs. Random Forest Classifier (clustering into 3 classes):

Class	Precision	Recall	F1-Score	Support
Accuracy			-0.12	0
Macro avg	0.02	-0.25	-0.25	0
Weighted avg	-0.11	-0.12	-0.13	0

Table 63: Differences between Native Multiclass Classification and Clustering into 3 Classes on the White Wine dataset.

Once again, we encounter the previously discussed issue, where some precision values reach 1 while the recall is 0. Despite this anomaly, the clustering-based approach still performed better overall. Both methods struggled with underrepresented classes, mainly due to low recall, although they achieved good precision for those cases.

Although the difference is not as pronounced as in the red wine dataset, the clustering approach still outperformed the native multiclass model that uses all classes, with at least a 0.12 advantage in all metrics. Therefore, as always, the most suitable approach depends on the specific requirements of the task: if predicting the exact class is not essential, the clustering method is a valid—and often superior—alternative.

8.3 One vs All vs. Clustering into 3 Classes

8.3.1 Red Wine

For this dataset, using the same comparison criteria, the selected model in Section 7 was the Random Forest Classifier (a native multiclass model). In Section 6, two models were used: Support Vector Machine (a native binary model) for the high reward strategy, and Random Forest Classifier (again, native multiclass) for the low risk strategy.

To support this comparison, we will again evaluate the macro average and weighted average for both approaches. Accuracy is not considered in this case. The analysis compares both low risk and high reward OvA strategies against the clustering approach, using the following formula:

Result = (One-vs-All) – (Clustering into 3 Classes)

Difference Table – SVM (High Reward OvA) vs. Random Forest Classifier (Clustering into 3 Classes):

Metric	Precision	Recall	F1-Score
Macro avg	-0.22	-0.06	-0.18
Weighted avg	-0.18	-0.02	-0.11

Table 64: Differences between One-vs-All (High Reward) and Clustering into 3 Classes on the Red Wine dataset.

Difference Table – Random Forest Classifier (Low Risk OvA) vs. Random Forest Classifier (Clustering into 3 Classes):

Metric	Precision	Recall	F1-Score
Macro avg	-0.10	-0.24	-0.20
Weighted avg	-0.10	-0.20	-0.16

Table 65: Differences between One-vs-All (Low Risk) and Clustering into 3 Classes on the Red Wine dataset.

As observed previously with the One-vs-One and Native Multiclass comparison, the One-vs-All strategy also underperformed in all categories. As expected, the High Reward model showed larger differences in precision, while the Low Risk model had greater drops in recall. Even when applying the techniques mentioned in Section 6 Even with efforts to improve precision, the clustering approach will most likely still achieve better overall performance.

Therefore, once again, the best approach depends on the specific needs of your task. However, based purely on scoring metrics, clustering the classes remains the most effective option.

8.3.2 White Wine

To better understand the results, we now perform the same comparison using the white wine dataset. Once again, the chosen model for Section 7 was the Random Forest Classifier (natively multiclass), while for Section 6, we selected the Support Vector Machine for the high-reward scenario and the Random Forest Classifier for the low-risk scenario.

As before, we construct a difference table based on the following formula:
Result = (One-vs-All) – (Clustering into 3 Classes)

Difference Table – SVM (High Reward OvA) vs. Random Forest Classifier (Clustering into 3 Classes):

Metric	Precision	Recall	F1-Score
Macro avg	-0.53	-0.15	-0.37
Weighted avg	-0.30	-0.08	-0.22

Table 66: Difference between One-vs-All (SVM) and Clustering into 3 Classes (Random Forest) on the White Wine dataset.

Difference Table – Random Forest Classifier (Low Risk OvA) vs. Random Forest Classifier (Clustering into 3 Classes):

Metric	Precision	Recall	F1-Score
Macro avg	-0.26	-0.32	-0.31
Weighted avg	-0.11	-0.20	-0.17

Table 67: Difference between One-vs-All (Random Forest) and Clustering into 3 Classes on the White Wine dataset.

We observed the same pattern here as with the red wine dataset—but with even more significant differences, all in favor of the clustering-based approach. Once again, the precision gap in the high-reward scenario is particularly large, and the recall gap is even larger in the low-risk case.

Even applying the techniques described in Section 6, which can improve precision, it is unlikely that the OvA models could match the performance of the clustering-based method. The gap is simply too large to be compensated.

As before, the best choice depends on the specific needs of the application. However, based purely on the scoring metrics, the clustering approach outperforms the others by a considerable margin.

9 Conclusion

After conducting all tests, we observed that the results converged in the same direction, although with slight variations between datasets. Among the approaches, clustering achieved the highest overall performance, followed by One-vs-All, which was closely matched by One-vs-One and Native Multiclass classification. Despite the similarities between datasets, we could already identify differences in performance, which highlights the importance of testing multiple models and approaches to determine the most suitable one. Even though we had prior expectations about the best-performing models—such as SVM, Random Forest, and Neural Networks—empirical validation proved essential.

Another important factor to consider is processing time. Clustering was the fastest method, followed by Native Multiclass, then One-vs-All, and finally One-vs-One, which was the slowest. Depending on the application, this can significantly influence model choice. Additionally, the objective of the classification should be taken into account: if the application does not require precise prediction of the exact class, clustering becomes a strong candidate due to its simplicity, speed, and reliable scores—making it a potentially viable option for our use case.

One consistent issue across all approaches was class imbalance. Although some models handled it better than others, none delivered truly satisfactory results. This gave us valuable insight into how each model responds to this challenge. Interestingly, the amount of data was not a major factor; even though the white wine dataset had significantly more samples than the red wine dataset, both produced similar outcomes. This was likely due to the red wine dataset still having sufficient data for analysis, and both being similarly unbalanced.

Both datasets were obtained from the UCI Machine Learning Repository. However, we lack information about how the data was originally generated or how the

quality grades were assigned. It is possible that the grading process was subjective—for example, a person might have rated the wines based on sensory criteria and only afterward were the chemical properties recorded. As discussed in the data analysis section, we did not observe a clear or consistent pattern in how wines were graded, which makes it difficult to understand what exactly the models are learning.

Given that uncertainty, we cannot be sure how these models would perform on new, unseen data. This also raises concerns about overfitting, where the model may be too closely fitted to specific characteristics of the training set.

Throughout this work, we identified several potential strategies for improving model performance in future studies. For instance, feature importance analysis could be used to quantify the contribution of each variable to the final prediction. This would allow us to eliminate less relevant or disruptive features and focus on those with a direct impact on outcomes. Such analysis was not included here, as the goal was to compare different classification approaches rather than optimize model accuracy. Nevertheless, this step could proportionally enhance performance across all methods in future work.