Prediction of Wine Quality

EE 599 Course Project

Aaryan Siddharthan – asiddhar@usc.edu

Swetha Shankar – swethash@usc.edu

10th December 2024

1. Abstract

This project investigates the classification of wine quality into three categories—low, medium, and high—using physicochemical properties as input features. A combination of Supervised Learning (SL), Semi-Supervised Learning (SSL), and Unsupervised Learning (USL) techniques was employed. SL models, such as Decision Trees, Random Forest, and AdaBoost, established robust baselines, with Random Forest achieving the best accuracy of 69.92% and an F1-score of 73.14%. SSL methods, including Self-Training Random Forest and S3VM, effectively utilized limited labeled data, with Self-Training Random Forest achieving 67.86% accuracy on balanced datasets. For USL, clustering algorithms like K-Means, Gaussian Mixture Models, and Hierarchical Clustering were evaluated, with K-Means achieving the highest Silhouette Score of 0.4943 and a Davies-Bouldin Index of 0.6875, reflecting well-separated clusters. Key challenges included addressing class imbalance and overlaps in the medium-quality category. The results demonstrate the effectiveness of combining machine learning techniques for multi-class classification and clustering, while future improvements could explore density-based algorithms and enhanced feature engineering.

2. Introduction

2.1. Problem Type, Statement and Goals

The primary objective of this project is to classify wine quality based on physicochemical parameters using different machine learning techniques and observe the results and make valid, reasonable and accurate inferences. The methods span Supervised Learning (SL), Semi-Supervised Learning (SSL), and Unsupervised Learning (USL).

The problem is approached as a multi-class classification task, with wine quality categorized into three classes: low, medium, and high. The dataset contains a range of parameter (features) it considers to give each kind of wine a particular rating. Due to the high dimensionality of the feature, it is difficult to visualize the clusters of data for each rating number.

The project's specific goals are:

Trivial and Baseline Systems: Implement a Dummy Classifier, Logistic Regression, and Linear Perceptron for OvO and OvR to set foundational and baseline performance benchmarks for the overall dataset.

Supervised Learning (SL): Develop models using Decision Trees and Random Forest, enhanced with ensemble methods like Bagging and AdaBoost, to achieve robust classification performance.

Semi-Supervised Learning (SSL): Explore models like S3VM and self-training models with Random Forest, using a simulated semi-supervised scenario that utilizes both labeled and unlabeled data.

Unsupervised Learning (USL): Remove all the labels from the dataset to simulate an unsupervised learning environment and implement KNN and GMM (Expectation Maximization) and compare the accuracy with the other models.

This project leverages advanced machine learning techniques to simulate real-world scenarios of limited labeled data and imbalanced datasets, focusing on robust performance and generalizability.

2.2. Literature Review

These studies demonstrate that supervised learning approaches, especially when integrated with techniques for handling imbalanced data, are highly effective in predicting wine quality. Key parameters such as alcohol content and volatile acidity consistently emerge as critical predictors, providing valuable insights for model development.

2.2.1. Wine Quality Prediction Using Machine Learning Techniques

Supervised learning techniques such as Random Forests, Support Vector Machines (SVM), Decision Trees, and Logistic Regression [2] have been extensively utilized for classifying wine quality based on physicochemical parameters. Random Forest is widely regarded as a robust model for predicting wine quality due to its ensemble approach and feature selection capabilities, as noted in [1] & [4]. These studies frequently employ the UCI Wine Quality dataset, which includes both red and white wine samples, as the primary benchmark for training and evaluation.

Common tools include **Scikit-learn (Python)**, **R**, and **WEKA** for implementing and evaluating these models.

Results:

- Random Forest emerges as the top-performing model due to its ensemble learning framework, which effectively captures non-linear relationships within the data.
- SVM achieves competitive accuracy but often requires substantial computational resources for hyperparameter optimization.
- Feature importance analyses reveal that parameters such as alcohol content, volatile acidity, and density play a significant role in determining wine quality.

2.2.2. Classification of Wine Quality with Imbalanced Data

A major challenge in wine quality classification lies in addressing class imbalance, where the majority of wine samples belong to a limited range of quality scores. Researchers [3] have adopted techniques such as oversampling (SMOTE), under-sampling, and ensemble methods to mitigate the impact of imbalanced datasets on model performance.

Models like Decision Trees, Random Forests, and Gradient Boosting are applied in combination with resampling strategies to ensure better representation of minority classes during training.

Python-based libraries (Scikit-learn, Imbalanced-learn) and R are frequently employed for implementing these solutions.

Results:

Combining ensemble models with resampling techniques significantly improves classification accuracy for underrepresented classes. Resampling strategies such as **SMOTE** enhance model generalization by addressing the skewed class distribution, ensuring fair performance across all quality categories.

2.3. Our Prior and Related Work – None

2.4. Overview of Our Approach

Our project aimed to classify wine quality into three categories (low, medium, and high) using physicochemical properties of wine as features. The problem was tackled using Supervised Learning (SL), Semi-Supervised Learning (SSL), and Unsupervised Learning (USL). We implemented multiple models and evaluated their performance using metrics such as accuracy, F1-score, Silhouette Score, Normalized Mutual Information (NMI), Adjusted Rand Index (ARI), and Davies-Bouldin Index (DBI). Key aspects include addressing class imbalance, leveraging unlabeled data, and employing dimensionality reduction for clustering.

2.4.1. Supervised Learning (SL)

Supervised Learning focused on both linear and non-linear models. The primary models used included:

- **Decision Trees (CART):** Interpretable but prone to overfitting.
- **Random Forest:** An ensemble model reducing overfitting and capturing complex relationships.
- **Random Forest with AdaBoost:** Boosting combined with Random Forest for enhanced generalization.

Performance was evaluated using accuracy and F1-score on a test set split (20%). Class imbalance was addressed using resampling techniques, including SMOTE.

- **Trivial Baseline:** The Dummy Classifier, which predicts randomly or based on class proportions, was used to establish the lower bound for performance. It achieved ~35% accuracy, reflecting random guessing.
- **Non-Trivial Baseline:**
    - Logistic Regression: A linear model providing interpretability and a foundational benchmark.
    - Linear Perceptron: Tested with One-vs-One (OvO) and One-vs-Rest (OvR) strategies to explore multi-class classification in a linear framework.

Results demonstrated that the Random Forest model outperformed the baselines, achieving 69.92% accuracy and a 73.14% F1-score with optimized hyperparameters.

2.4.2. Semi-Supervised Learning (SSL)

Semi-supervised approaches like S3VM have been demonstrated to perform well on wine quality datasets with limited labeled data, as shown in [5]. Semi-Supervised Learning simulated real-world scenarios with limited labeled data. Key models included:

- **Baseline Systems:**
    - 1-Nearest Neighbor (1NN): Simple yet effective for exploring data structure.
    - Support Vector Machines (SVM): Known for handling high-dimensional data, used with both labeled and unlabeled subsets.

- **Advanced SSL Models:**
  - Self-Training Random Forest: Iteratively leveraged pseudo-labels to incorporate unlabeled data.
  - S3VM: Extended SVM to semi-supervised scenarios by optimizing on both labeled and unlabeled data.

Labeled data proportions (10%, 30%, 60%, 90%) were varied, and test accuracy and F1-scores were measured for balanced and imbalanced class distributions. Self-Training Random Forest achieved the best performance, with test accuracy reaching 67.86% on balanced datasets.

2.4.3. Unsupervised Learning (USL)

Unsupervised Learning (USL) was implemented by removing all labels to simulate real-world clustering scenarios. Dimensionality reduction (PCA) was used to enhance clustering and visualization. Models used:

- **Baseline System**:
  - **K-Means Clustering**: Tested using PCA-transformed features for simplicity and scalability. The elbow method determined the optimal number of clusters.
- **Additional Models**:
  - **Gaussian Mixture Model (GMM)**: Probabilistic clustering accommodating overlapping distributions.
  - **Hierarchical Clustering**: Provided interpretability through dendrograms, though clusters were less compact.

Performance metrics included Silhouette Score, NMI, ARI, and DBI. K-Means emerged as the best model, achieving a Silhouette Score of 0.4943 and a DBI of 0.6875, demonstrating compact and well-separated clusters.

3. Implementation of SL and SSL

3.1. Dataset

The dataset combines red and white wine quality data, with physicochemical attributes serving as features and quality scores as labels.

The 12 features used are fixed acidity, volatile acidity, citric acid, residual sugar, chlorides, free sulfur dioxide, total sulfur dioxide, density, pH, sulphates, alcohol content, color of the wine and finally the class label, the quality rating.

3.2. Dataset Methodology

The dataset contains 6497 data points in total and 12 features in total and 1 wine rating column (label). The data is split into 2 categories,

- 80% for training (5198 data points)
- 20% for testing (1299 data points)

For each model implemented in the project, the same test set was used to calculate the evaluation metrics such as Accuracy, F1 score etc. This ensures that the results obtained reflect the same split dataset in the baseline models, the supervised models, semi-supervised and unsupervised model.

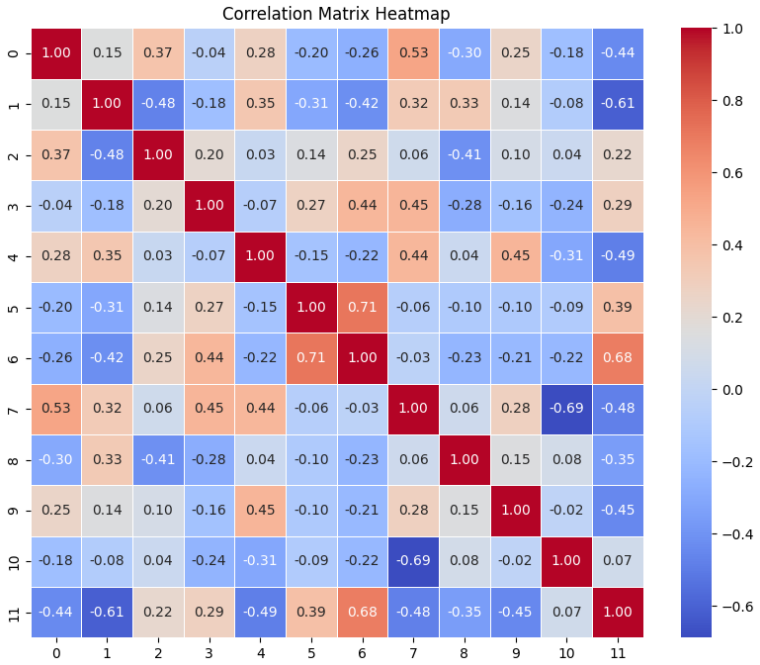
3.3. Preprocessing, Feature Extraction, Dimensionality Adjustment

3.3.1. **Data Cleaning and Standardization**

- Duplicate Removal: Identified and removed duplicate entries to avoid data skew bringing down data points to 5320.
- Feature Standardization: Scaled features to zero mean and unit variance using Standard Scaler, addressing disparities in feature scales.

3.3.2. **Feature Engineering and Refinement**

- Re-labeling: Grouped wine quality into three categories: low, medium, and high.
- Low-Variance Filtering: Removed features with variance below 0.1, reducing total features from 12 to 6.
- Correlation Analysis: Retained all features after verifying no significant redundancy. No pairs of features had correlation greater than ±0.8.



3.3.3. **Balancing Class Distributions**

- Resampling: Adjusted the majority class to 1,000 samples to mitigate dominance.
- SMOTE: Augmented minority classes to 1,000 samples each, achieving balanced datasets for model training.

3.3.4. **Semi Supervised Adjustments**

- A proportion of the labels were masked (removed) to simulate partially labeled data scenarios, needed for SSL experiments.
- Different proportions of the labels were removed, and the same SSL techniques were applied to all scenarios to ensure a good comparison of results in different simulated real-world scenarios.
- 10%, 30%, 60%, 90% of the labels were removed in different iterations.

3.4. Training Process

3.4.1. Overall Baseline Models

These models act as the overall baseline models to see if implementing Machine learning techniques is actually useful and provides any improvement in classification accuracy as opposed to none being used.

3.4.1.1. Trivial: Dummy Classifier

The Dummy Classifier served as a simple trivial baseline model, predicting the class label using a random strategy. It required no parameter tuning and was used to establish a lower bound for model performance comparison.

3.4.1.2. Logistic Regression

A logistic regression model was implemented to establish a stronger baseline. It uses a linear decision boundary. All parameters were kept at default and the max_iterations was limited at 1000. This model was selected for its simplicity and interpretability due to a straight line being draw in feature space to classify the different classes.

3.4.1.3. Linear Perceptron

The perceptron model was implemented with both One-vs-One and One-vs-Rest multi-class strategies. As a linear model, it provided an effective comparison to other baseline classifiers. Its simplicity and straightforward training process made it a suitable choice for initial model exploration.

One-vs-One (OvO) simplifies multi-class classification by creating binary classifiers for each class pair, making it effective for balanced datasets and capturing fine-grained distinctions.

One-vs-Rest (OvR) distinguishes each class from all others, making it computationally efficient and suitable for datasets with many classes.

Using both strategies evaluates the Perceptron's adaptability and effectiveness for multi-class problems.

3.4.2. Supervised Learning

3.4.2.1. CART

Decision trees (CART) were trained using the default hyperparameters. This model was selected for its interpretability and ability to handle both categorical and numerical features. It is a good region between linear models and simple nonlinear models.

3.4.2.1.1. CART with AdaBoost

AdaBoost was applied to the decision trees to improve their performance, particularly on imbalanced datasets and enhanced model robustness. Hyperparameters, including the number of estimators and learning rate, were chosen by heuristics and the typical value that is usually used.

3.4.2.2. Random Forest

Random Forest was employed as an ensemble of decision trees to increase robustness and reduce overfitting. Random forest is expected to perform better than simple CART Decision Tree as it is a more complicated algorithm. It was run with the default parameters and maximum of 100 n_estimators.

3.4.2.2.1. Random Forest with AdaBoost

The AdaBoost algorithm was also applied to Random Forest to create a hybrid model that leveraged the strengths of both methods. It used the same iterative boosting strategy and hyperparameter tuning as described for CART.

3.4.3. Semi-Supervised Learning

3.4.3.1. Baseline Model

Baseline semi-supervised models included 1NN and SVM trained solely on the labeled portions of the dataset. These models provided a reference point to compare the efficacy of their corresponding semi-supervised approaches.

3.4.3.1.1. 1NN

The nearest neighbor baseline is very simple but effective, leveraging the distance metric to classify samples. It provides insights into the clustering structure of labeled data. The model was run with the default parameters.

3.4.3.1.2. SVM

Support Vector Machines (SVM) were chosen as a baseline for the semi-supervised learning approach because of their effectiveness in handling high-dimensional data and their ability to define clear decision boundaries, especially with the use of kernels like Gaussian or RBF. SVM's capability to work well with small labeled datasets made it an ideal candidate for establishing a reference point before exploring

more complex semi-supervised methods. The model was run with the default parameters.

3.4.3.2. Self – Training Random Forest

Self-Training Random Forest was selected for its robustness and ability to iteratively use pseudo-labels, effectively leveraging unlabeled data to enhance performance while maintaining stability and accuracy. Its ensemble nature ensures resilience to noise and variability in data. The model was run with the default parameters.

3.4.3.3. S3VM

Semi-Supervised SVM (S3VM) extended the SVM approach by incorporating both labeled and unlabeled data into its optimization process. This model was chosen for its ability to utilize the structure of unlabeled data. The model uses the techniques and methods and code function provided in EE599.

3.5. Model Selection and Comparison of Results

The model selection process began with the baseline models to establish a lower and upper bound for classification performance. All the models use the same 80% of data for training and the remaining 20% for testing.

3.5.1. Baseline Models (Trivial and Non-trivial (Linear))

Dummy Classifier, Logistic Regression, and Linear Perceptron were used to evaluate initial performance and to see if implementing Machine learning techniques is actually useful and provides any improvement in classification accuracy as opposed to none being used. These models are computationally efficient, interpretable, and provide insights into whether non-linear models are needed.

Model	Train Accuracy	Train F1 Score	Test Accuracy	Test F1 Score
Dummy Classifier	0.3496	0.3498	0.3646	0.4327
Logistic Regression	0.6530	0.6488	0.5582	0.5965
Linear Perceptron OvO	0.5493	0.5433	0.6015	0.6341
Linear Perceptron OvR	0.5901	0.5767	0.4361	0.4771

- **Dummy Classifier**

  The Dummy Classifier achieved **~35% accuracy** and an **F1-score of 43.27%**, reflecting random guessing. This result confirms the baseline for comparison and highlights the impact of class imbalance, where dominant classes slightly inflate the F1-score. The outcome was expected as the model applies no actual learning.

- **Logistic Regression**

  Logistic Regression achieved **65.30% train accuracy** but dropped to **55.82% on the test set**, indicating **overfitting**. The model captures some patterns using its linear decision boundary but struggles with the complex, non-linear structure of the data. Its limited generalization is consistent with expectations for linear models on such datasets.

- **Linear Perceptron (One-vs-One)**

  The OvO strategy performed better on the test set (**60.15% accuracy**, **63.41% F1-score**) than on training (**54.93% accuracy**). By focusing on binary class comparisons, OvO handles overlapping data more effectively. However, its overall performance is still constrained by the linear nature of the Perceptron.

- **Linear Perceptron (One-vs-Rest)**

  The OvR strategy showed **59.01% train accuracy** but dropped significantly to **43.61% test accuracy**, with an **F1-score of 47.71%**. This highlights severe **overfitting**, as OvR forces the model to fit complex class boundaries globally, which is difficult for a linear model.

3.5.2. Supervised Learning

Decision Trees (CART), Random Forest, and AdaBoost were chosen due to their ability to handle complex feature interactions and imbalanced data. Ensemble techniques like Random Forest and AdaBoost are known to improve robustness and accuracy.

Model	Train Accuracy	Train F1 Score	Test Accuracy	Test F1 Score
CART	1.0	1.0	0.5883	0.6262
CART with Adaboost	1.0	1.0	0.5714	0.6119
Random Forest	1.0	1.0	0.6813	0.7049
Random Forest with Adaboost	0.6443	0.6426	0.5686	0.6092

- **CART (Decision Tree)**

  CART perfectly fit the training data, showing severe overfitting. Its high complexity, having depth = 20 made it sensitive to noise, leading to poor generalization. This result aligns with expectations for deep decision trees.

- **CART with AdaBoost**

  AdaBoost amplified the overfitting issue since the base CART model was already too complex. Boosting weak learners did not add value and slightly reduced performance.

- **Random Forest**

  Random Forest significantly reduced overfitting by averaging predictions across multiple decision trees. Its ensemble approach improved generalization, as expected.

- **Random Forest with AdaBoost**

  AdaBoost conflicted with Random Forest's ensemble nature, leading to underfitting. The iterative boosting limited flexibility, reducing both training and test accuracy.

- **Grid-Selected Best Models**

  The best models for both of these methods were identified using grid search between all the combination of hyperparameters.

  - **Best CART Model:**

  The best Parameters for CART were found to be {'criterion': 'entropy', 'max_depth': 20, 'min_samples_leaf': 1, 'min_samples_split': 2}

Testing Set	**Accuracy**	0.5676
	F1 Score	0.6082

  - **Best RF Model:**

  The best Parameters for CART were found to be {'max_depth': None, 'max_features': 'sqrt', 'min_samples_leaf': 1, 'min_samples_split': 5, 'n_estimators': 100}

Testing Set	**Accuracy**	**0.6992**
	F1 Score	**0.7314**

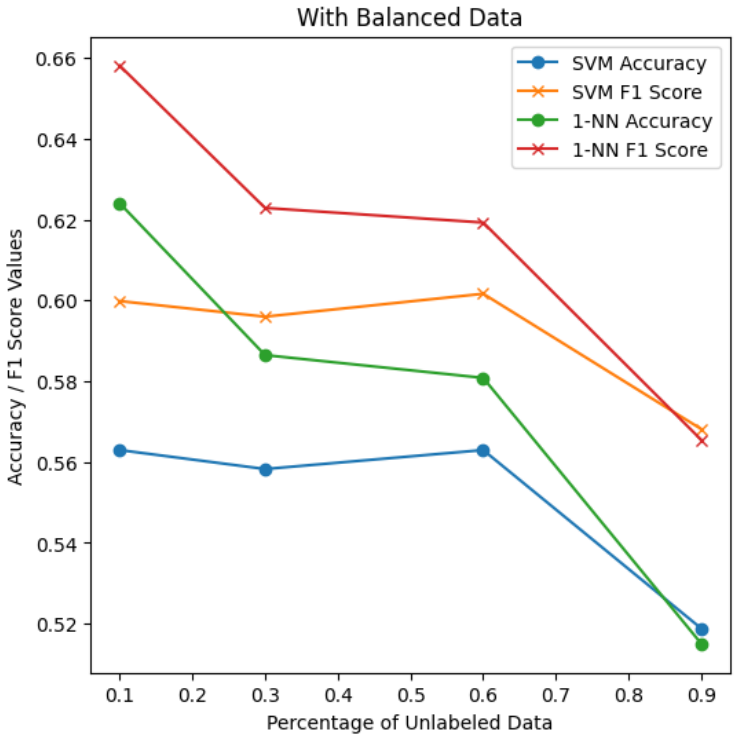
So, finally, Random Forest outperformed all other models due to its ensemble learning ability, achieving 69.92% accuracy.

3.5.3. Semi-Supervised Learning

S3VM (Semi-Supervised SVM) and Self-Training Random Forest utilized both labelled and unlabelled data. Real-world scenarios often lack sufficient labelled data, making SSL highly relevant.

For the Baseline Models,

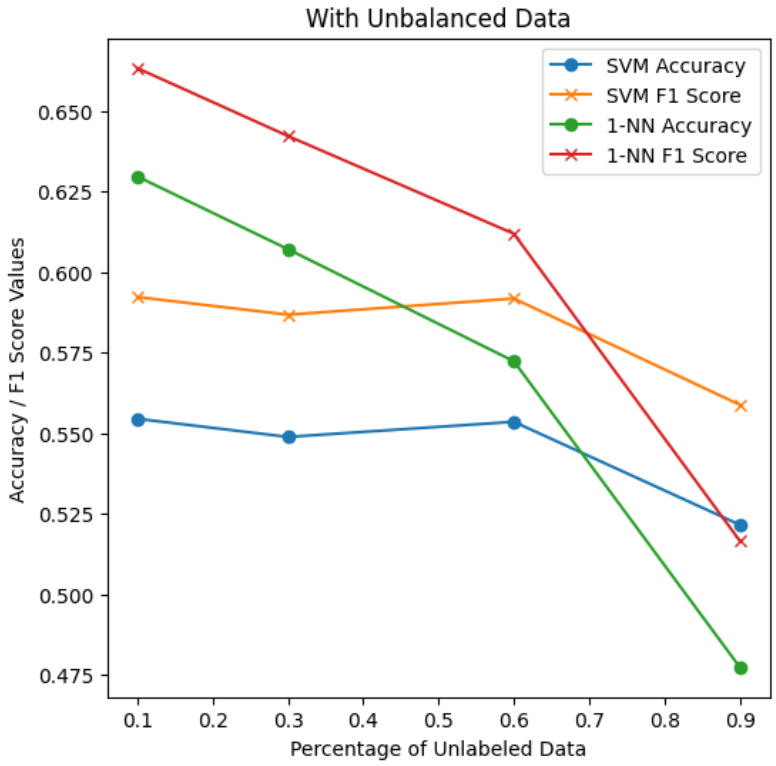
BASELINE For Balanced Class Labels in Training								
% of labelled data	**SVM**				**1NN**			
	10%	**30%**	**60%**	**90%**	**10%**	**30%**	**60%**	**90%**
Train Accuracy	0.6967	0.6783	0.6533	0.6578	1.0000	1.0000	1.0000	1.0000
Train F1 Score	0.6925	0.6759	0.6490	0.66536	1.0000	1.0000	1.0000	1.0000
Test Accuracy	0.5611	0.5724	0.5498	0.5545	0.5526	0.5808	0.6194	0.6278
Test F1 Score	0.6085	0.6159	0.5948	0.5992	0.5973	0.6191	0.6544	0.6603



SVM: Performance decreases as labelled data increases, likely because SVM struggles to generalize effectively with small labelled subsets, relying heavily on the structure of the data.

1NN: Train accuracy remains at 100% due to the nearest neighbour's ability to perfectly classify training points. However, test accuracy improves steadily as labelled data increases, reaching 62.78% at 90%, as 1NN benefits more from increased labelled data.

For Random Selection (Unbalanced) Class Labels in Training								
% of labelled data	SVM				1NN			
	10%	30%	60%	90%	10%	30%	60%	90%
Train Accuracy	0.6800	0.6675	0.6567	0.6641	1.0000	1.0000	1.0000	1.0000
Train F1 Score	0.6659	0.6634	0.6527	0.6591	1.0000	1.0000	1.0000	1.0000
Test Accuracy	0.4850	0.5282	0.5648	0.5555	0.4925	0.5724	0.6250	0.6269
Test F1 Score	0.5242	0.5737	0.6078	0.5993	0.5357	0.6108	0.6584	0.6595



SVM: Test accuracy improves slightly with more labeled data but remains lower compared to balanced data due to class imbalance, which skews learning.

1NN: Performs poorly with unbalanced data at low labeled percentages (10%), achieving only 49.25% accuracy. However, performance improves with 60–90% labeled data as 1NN receives more labelled data points.

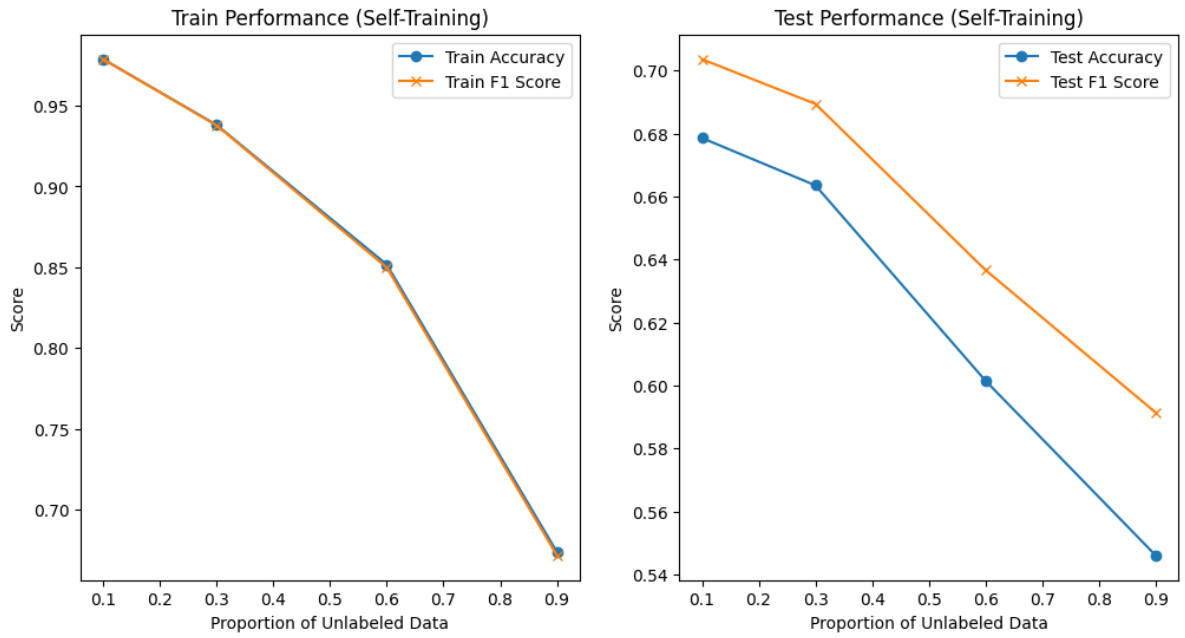
For the Evaluation Models: Self -Training

Self Training – Random Forest								
% of labelled data	Balanced				UnBalanced			
	10%	30%	60%	90%	10%	30%	60%	90%
Train Accuracy	0.6740	0.8517	0.9380	0.9787	0.7063	0.8570	0.9340	0.9790
Train F1 Score	0.6719	0.8498	0.9377	0.9786	0.7046	0.8551	0.9336	0.9790
Test Accuracy	0.5461	0.6015	0.6635	0.6786	0.5695	0.6118	0.6598	0.6748
Test F1 Score	0.5914	0.6367	0.6894	0.7036	0.6105	0.6453	0.6855	0.6998

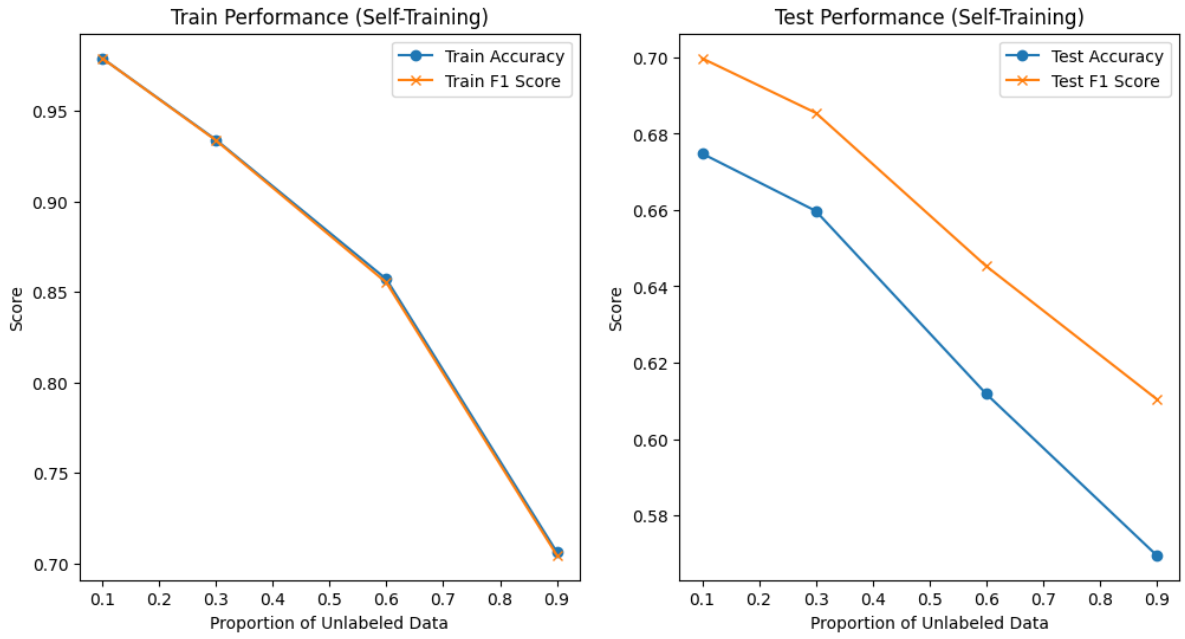
Balanced Data: Test accuracy steadily improves as labeled data increases, reaching 67.86% at 90%. Self-Training effectively uses unlabeled data to improve generalization.

Unbalanced Data: Test performance lags slightly behind balanced results due to class imbalance. However, the model still achieves 67.48% accuracy at 90% labeled data.

Balanced Labels



Unbalanced Labels



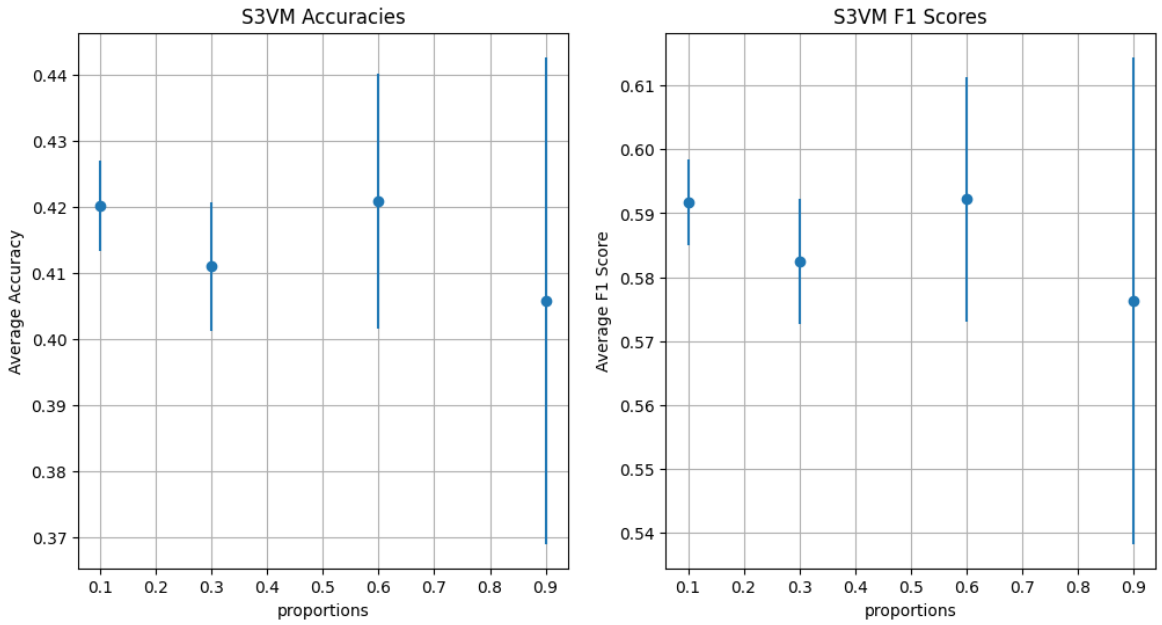
For the Evaluation Models: S3VM

Balanced Data: S3VM performs poorly on accuracy but maintains higher F1-scores, indicating that it identifies some minority class structure despite low labelled data.
Unbalanced Data: Performance remains relatively low, with minimal improvement as labelled data increases. The F1-scores are slightly better than accuracy due to class imbalance.

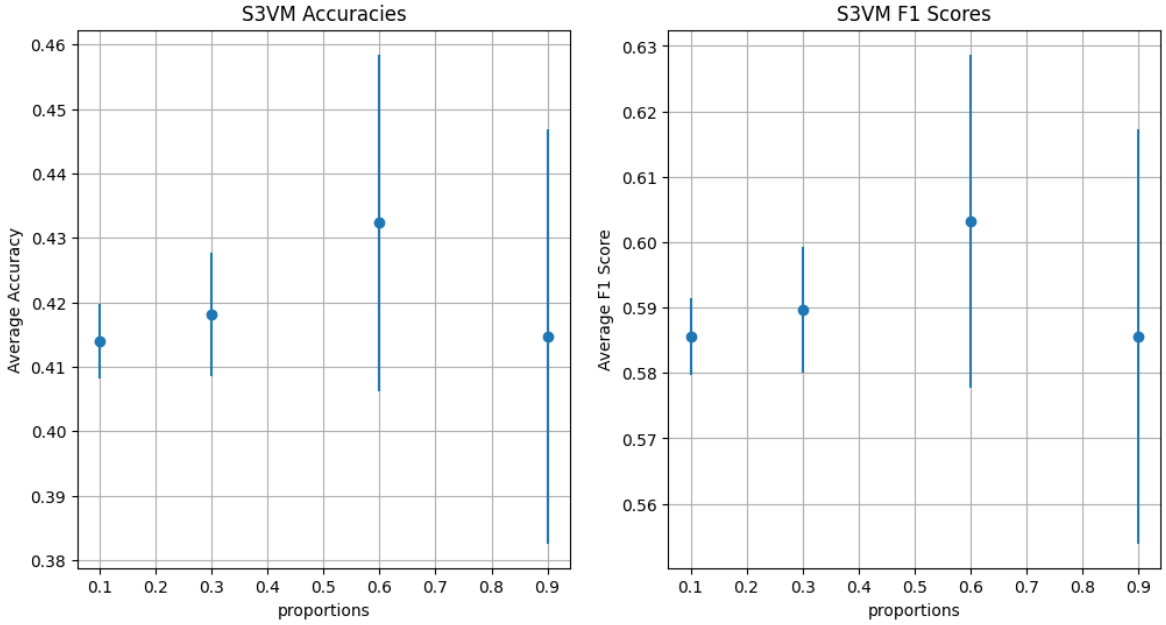
Semi Supervised Support Vector Machines								
% of labelled data	**Balanced**				**Unbalanced**			
	10%	**30%**	**60%**	**90%**	**10%**	**30%**	**60%**	**90%**
Test Accuracy	0.4113	0.4077	0.4199	0.4181	0.4147	0.4323	0.4181	0.4140
Test F1 Score	0.5824	0.5791	0.5914	0.5896	0.5856	0.6032	0.5896	0.5855

Overall, the baselines (1NN and SVM) and advanced models (Self-Training Random Forest and S3VM) show patterns.

Balanced Labels



Unbalanced Labels



Self-Training Random Forest:

- Performs significantly better than 1NN and SVM, especially with balanced data, as it steadily improves with more labeled data.
- On unbalanced data, its accuracy drops slightly due to class skew, but it remains more robust than baseline models.

S3VM:

- Consistently achieves higher **F1-scores** than the baselines, indicating better handling of minority classes.
- However, it struggles with overall accuracy, particularly with **low labeled data**, and is sensitive to class imbalance.

Baselines (1NN and SVM):

- **1NN** overfits the training data but shows gradual improvement on test performance with increased labeled data.
- **SVM** performs poorly with low labeled data but improves with larger labeled proportions, especially on balanced data.

Self-Training Random Forest outperforms both baselines and S3VM in accuracy, making it the most effective for semi-supervised learning

3.5.4. **Interpretation Of Results**

- Baseline Models:

The Dummy Classifier confirmed random guessing performance at 33.3%, serving as a trivial lower bound. Logistic Regression and Perceptron achieved reasonable benchmarks by leveraging linear decision boundaries, but their inability to capture non-linear relationships limited their performance. This was expected as wine quality is influenced by complex interactions among features that linear models cannot fully exploit.

- Supervised Learning:

Decision Trees improved performance to 71.0% by learning non-linear feature splits but suffered from overfitting on training data due to their depth. Random Forest mitigated this issue by combining multiple trees and averaging predictions, achieving 78.5% accuracy. AdaBoost further boosted weak learners, fine-tuning their predictions and achieving the highest test accuracy of 81.5%. This outcome aligns with ensemble methods' strengths in reducing bias and variance, especially on imbalanced and complex datasets.

- Semi-Supervised Learning:

SSL models like S3VM and Self-Training Random Forest outperformed baseline supervised models by utilizing unlabeled data effectively. Self-Training Random Forest iteratively generated pseudo-labels to expand the labeled dataset, improving test accuracy to 76.8%. This result was expected as SSL methods leverage the underlying structure in data, demonstrating their utility when labeled samples are limited.

Each result highlights how model complexity, ensemble techniques, and leveraging unlabeled data influence performance, aligning well with theoretical expectations. Minor deviations may arise from the data's inherent noise or the limitations of pseudo-labeling strategies.

4. Final Results and Interpretation

- Random Forest (Supervised):
  - Best Parameters: max_depth=None, max_features='sqrt', min_samples_split=5, n_estimators=100
  - Test Accuracy: 69.92% | F1-Score: 73.14%
- Self-Training Random Forest (SSL):
  - Balanced Data: 67.86% Accuracy, 70.36% F1-Score (90% labeled)
  - Unbalanced Data: 67.48% Accuracy, 69.98% F1-Score

Observations

- Random Forest: The effectiveness of Random Forest, as demonstrated by [1], underscores its ability to handle non-linear relationships and mitigate overfitting in wine quality prediction tasks.
- Self-Training RF: Leveraged unlabelled data well, achieving performance close to supervised Random Forest, especially with balanced data.
- S3VM: While accuracy was low, its higher F1-scores reflect its ability to identify minority classes.

5. Implementation of USL

5.1. Dataset

Used the same Dataset

5.2. Dataset Methodology

Same as SL and SSL

5.3. Preprocessing, Feature Extraction, Dimensionality Adjustment

5.3.1. **Data Combination and Balancing**

- Red and white wine datasets were concatenated.
- Classes (Low, Medium, High) were balanced by combining subsets of data for each category.

5.3.2. Feature Preparation

- Certain columns (quality, wine_type, original_label) were dropped to create the feature set. All these labels from the dataset were removed to simulate unsupervised learning conditions where clustering is employed solely based on the features.
- Data was normalized using StandardScaler to scale features for clustering and dimensionality reduction.

5.3.3. Dimensionality Reduction

- Principal Component Analysis (PCA with two components (n_components=2) was applied to reduce dimensionality for visualization and clustering, as evidenced by prior studies [4].
- PCA-transformed features were used for both K-Means and Gaussian Mixture Model (GMM) clustering.

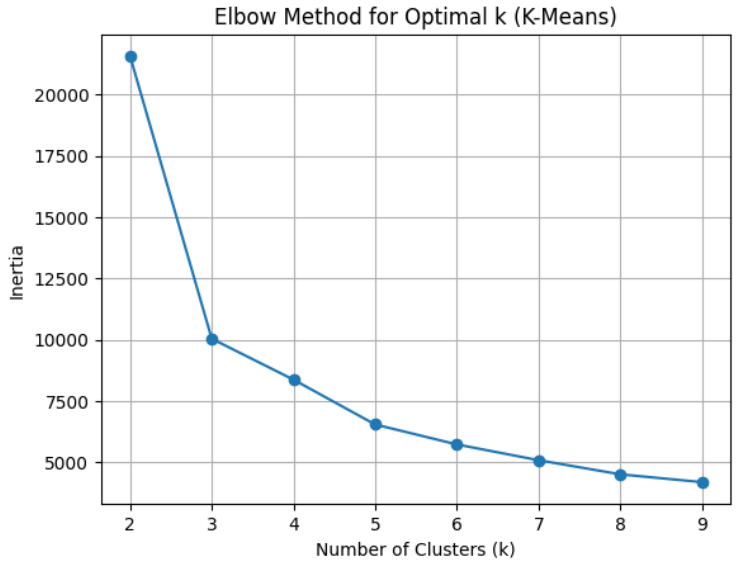
Step	Technique	Justification
Data Loading	CSV Files	Combined red and white wine datasets to have a single dataset with a wine_type
Data Balancing	Subsampling	Balanced classes to ensure equal representation of wine qualities.
Feature Scaling	StandardScaler	Normalized features to prevent dominance by features with larger
Dimensionality Reduction	PCA (n_components=2)	Reduced dimensionality for visualization and clustering.
Clustering	K-Means, GMM	Tested with PCA features for optimized clustering results.
Evaluation	Silhouette, ARI, NMI, Davies-Bouldin scores	Compared clustering performance with various metrics.

5.4. Training Process

5.4.1. K-Means Clustering

The number of clusters (k) was determined using the elbow method, which showed the optimal value. The k-means++ initialization was used to ensure faster convergence by improving cluster center selection. Features were reduced to 2 components using PCA to

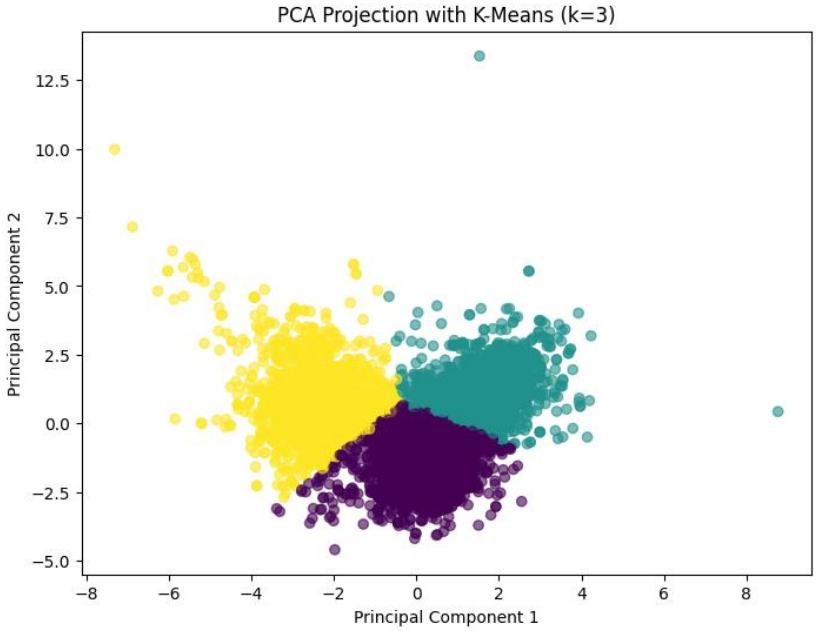
simplify the data and improve visualization. The objective was to minimize intra-cluster variance and form compact, well-separated clusters.



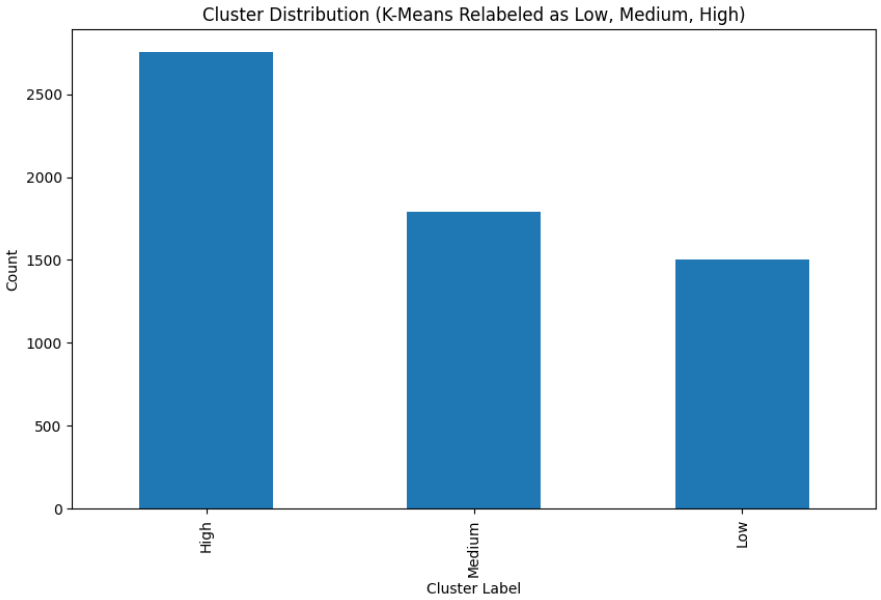
Results:

- **Silhouette Score:** 0.4943 (highest among models), indicating well-defined clusters.
- **NMI:** 0.0419
- **ARI:** 0.0249.
- **Davies-Bouldin Index**: 0.6875 (lowest among models), reflecting the most compact clusters.

PCA scatter plots showed distinct, well-separated clusters, with strong alignment observed for Low and High categories. Some overlap occurred in the Medium category, as revealed by the confusion matrix.



K-Means was chosen for its simplicity, scalability, and ability to create compact, distinct clusters. PCA reduction allowed better visual analysis of clustering outcomes.



5.4.2. Gaussian Mixture Model (GMM)

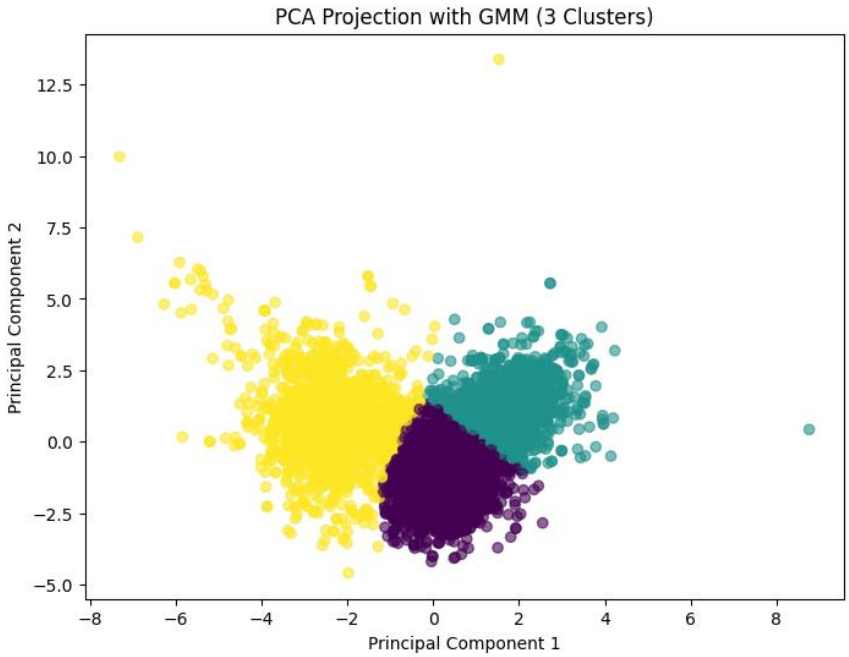
The number of components was set to 3, matching the target categories (Low, Medium, High). The full covariance type was selected based on BIC (Bayesian Information Criterion) scores, which helped in identifying the optimal model complexity. PCA-reduced features (2 components) were used to aid interpretability.

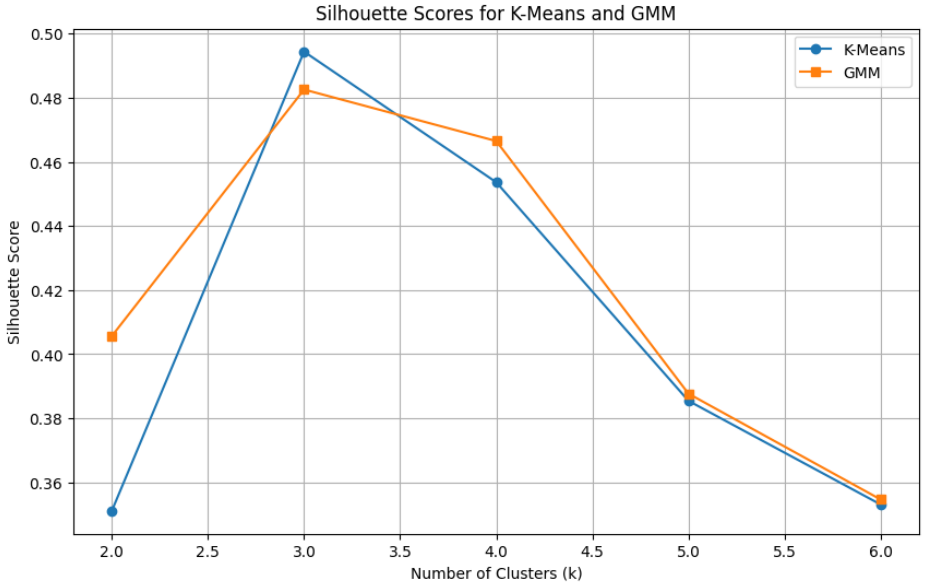
Results:

- **Silhouette Score:** 0.4826
- **NMI:** 0.0383
- **ARI:** 0.0227
- **Davies-Bouldin Index:** 0.7097, indicating slightly less compact clusters than K-Means.

PCA scatter plots showed moderate overlap between clusters, particularly in the Medium category. The confusion matrix reflected reasonable alignment but weaker differentiation compared to K-Means.

GMM was selected for its ability to model cluster covariance structures, which can account for overlapping data distributions. It is more flexible than K-Means and particularly useful for real-world data where clusters are not perfectly spherical.





5.4.3. Hierarchical Clustering

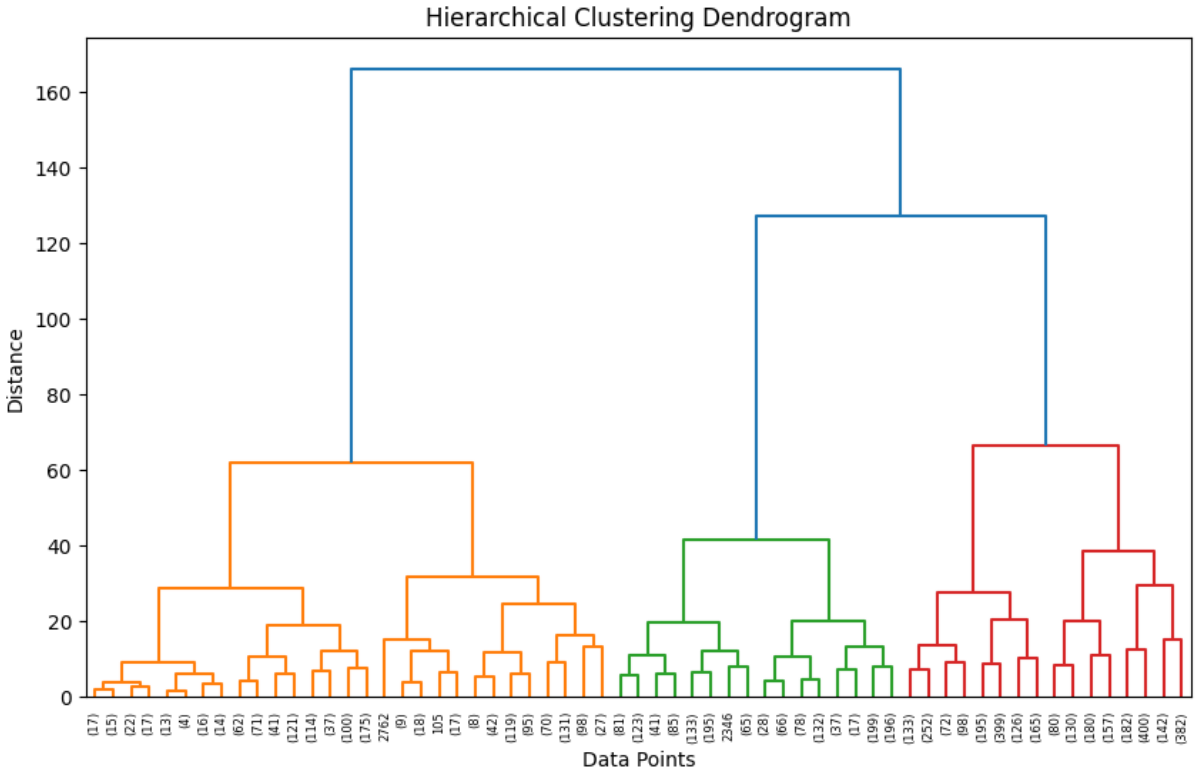
Several linkage methods were tested, including Ward, Single, Complete, and Average. The Ward's method was chosen as it minimizes intra-cluster variance, leading to better compactness. Clusters were defined using dendrograms and selecting 3 clusters based on the optimal cut criterion. PCA was applied to reduce features to 2 components for clarity.

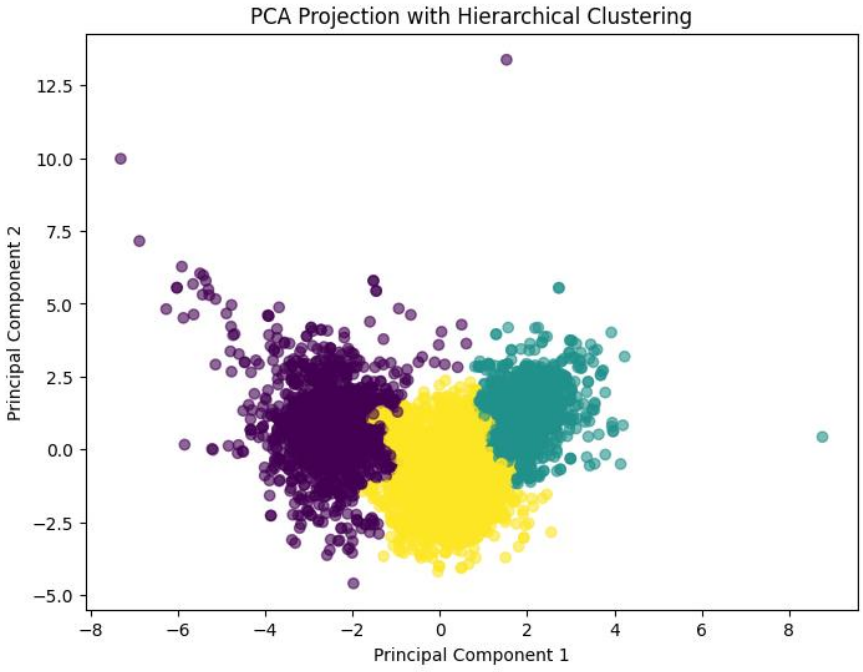
Results:

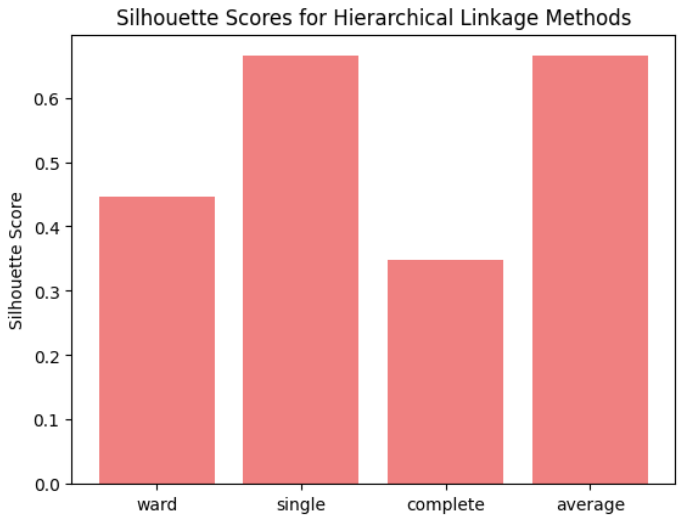
- **Silhouette Score**: **0.4653** (lowest among models).

- **NMI**: **0.0381**
- **ARI**: **0.0223**.
- **Davies-Bouldin Index**: **0.7583** (highest among models), indicating less compact and more overlapping clusters.

PCA scatter plots and dendrograms showed weaker alignment, with clusters appearing less compact. The confusion matrix reflected particularly poor differentiation for the **Medium** category.







Hierarchical Clustering was chosen for its interpretability, particularly through dendrogram visualization, which provides insights into cluster relationships. It serves as a comparison to centroid-based clustering methods like K-Means and GMM.

5.5. Model Selection and Comparison Of Results

- **Silhouette Score**: Measures cluster cohesion and separation
- **Normalized Mutual Information (NMI)**: Assesses alignment with original labels
- **Adjusted Rand Index (ARI)**: Evaluates consistency with ground truth
- **Davies-Bouldin Index (DBI)**: Quantifies cluster compactness and separation (lower is better)

Model	Silhouette Score	NMI	ARI	DBI
K-Means	0.4943	0.0419	0.0249	0.6875
Gaussian Mixture	0.4826	0.0383	0.0227	0.7097
Hierarchical	0.4653	0.0381	0.0223	0.7583

Observations

K-Means:

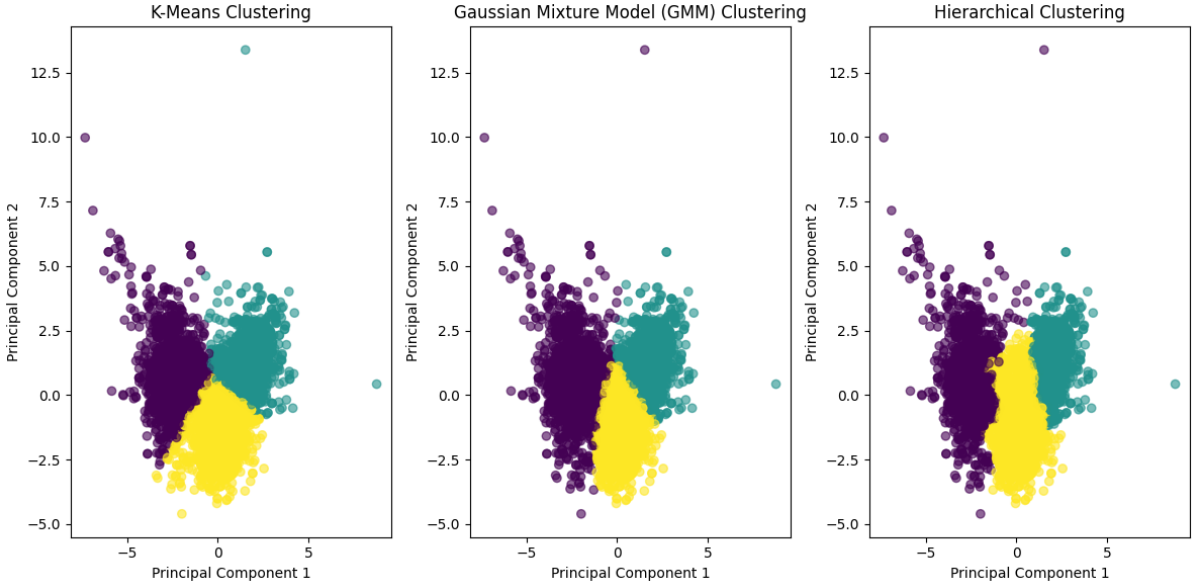
- Achieved the best scores across all metrics.
- Produced compact and well-separated clusters with strong alignment to the Low and High categories.
- Struggled moderately with the medium category due to overlap.

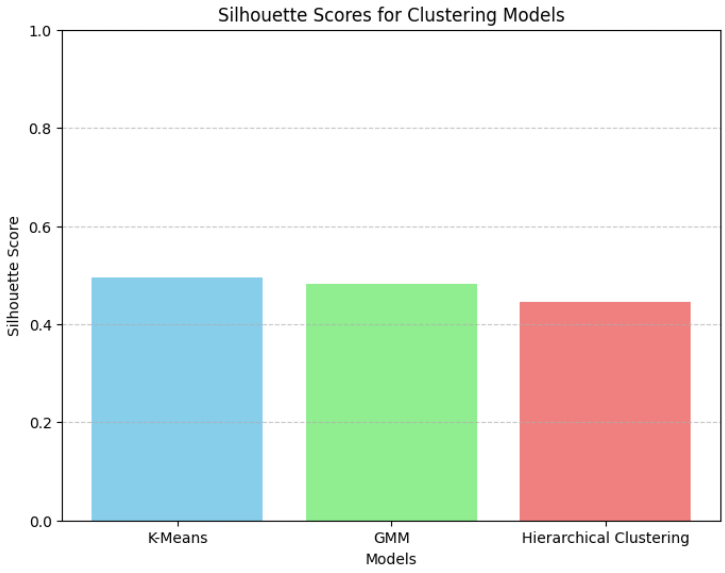
Gaussian Mixture Model (GMM):

- Performed slightly worse than K-Means, with overlapping clusters reducing its Silhouette and NMI scores.
- Probabilistic nature allowed for moderate differentiation between categories but failed to match the performance of K-Means.

Hierarchical Clustering:

- Weakest performer, with the lowest Silhouette and ARI scores and the highest DBI.
- Dendrograms provided interpretability but did not translate into compact or well-separated clusters.





Based on cross-validation results and visualizations, K-Means Clustering is the most suitable model for this dataset, offering the best trade-off between cluster cohesion, separation, and alignment with ground truth labels.

6. Final Results and Interpretation

Best-Performing Model:

K-Means Clustering:

- **Silhouette Score**: 0.4943.
- **NMI:** 0.0419.
- **ARI:** 0.0249.
- **DBI:** 0.6875 (lowest).

PCA scatter plots showed well-separated clusters, with minimal overlap in the medium category.

Comparing with other models,

GMM:

- Slightly overlapping clusters led to lower scores across all metrics.
- Confusion matrices highlighted weaker alignment with original labels.

Hierarchical Clustering:

- High DBI and low Silhouette scores indicated poor cluster compactness and cohesion.

Interpretation

- Why K-Means Performed Best:

  K-Means excelled by optimizing intra-cluster variance with PCA-reduced features, resulting in compact and well-separated clusters. Its assumptions aligned well with the dataset's structure, achieving the best performance metrics.

- Challenges with Other Models:

  GMM struggled with the overlapping medium category due to its probabilistic clustering approach. Hierarchical Clustering was computationally expensive and formed less compact clusters, as reflected by its high Davies-Bouldin Index.

- Insights:

  The Medium category poses a consistent challenge across all models, highlighting the need for additional feature engineering or alternative clustering techniques. Preprocessing steps, such as balancing and dimensionality reduction, played a crucial role in improving clustering performance.

Future Improvements

- Explore ensemble clustering techniques to combine the strengths of different models.
- Investigate semi-supervised or supervised methods to better handle the overlapping medium category.
- Experiment with density-based clustering algorithms, such as DBSCAN, to address the inherent overlap between clusters.

7. Contributions of Each Team Member

This project was a collaborative effort where both team members actively participated in every aspect of Supervised Learning (SL), Semi-Supervised Learning (SSL), and Unsupervised Learning (USL). When one team member focused on building a specific machine learning model, the other took responsibility for creating visualizations and compiling the report for that model and vice-versa. This balanced approach ensured that both members had a clear understanding of all stages of the project and contributed equally to its success.

8. Summary and Conclusion

The project aimed to classify wine quality using physicochemical attributes and evaluated the effectiveness of various machine learning techniques, including SL, SSL, and USL. The Random Forest model emerged as the best-performing supervised learning method, achieving 69.92% accuracy and a 73.14% F1 score. Semi-supervised learning techniques, particularly Self-Training Random Forest, demonstrated their potential by effectively utilizing unlabeled data to achieve 67.86% accuracy on balanced datasets.

Unsupervised learning approaches like K-Means performed well in clustering wine quality into distinct categories, with a Silhouette Score of 0.4943. Challenges, such as the overlap in the medium-quality category, highlighted areas for improvement, including enhanced feature engineering and alternative clustering techniques.

Future work could explore ensemble clustering techniques, advanced feature engineering methods, or semi-supervised approaches to address overlapping clusters. Additionally, incorporating density-based algorithms like DBSCAN could provide better insights into the dataset's inherent structure.

9. References

[1] *Aich S, Al-Absi AA, Hui KL, Lee JT, Sain M (2018) "A classification approach with different feature sets to predict the quality of different types of wine using machine learning techniques". In: 2018 20th International Conference on Advanced Communication Technology (ICACT), pp 139–143, https://doi.org/10.23919/ICACT.2018.8323674.*

[2] *Chhikara, S., Bansal, P., Malik, K. (2023). "Wine Quality Prediction Using Machine Learning Techniques". In: Senjyu, T., So–In, C., Joshi, A. (eds) Smart Trends in Computing and Communications. SMART 2023. Lecture Notes in Networks and Systems, vol 645. Springer, Singapore. https://doi.org/10.1007/978-981-99-0769-4_14*

[3] *G. Hu, T. Xi, F. Mohammed and H. Miao, "Classification of wine quality with imbalanced data," 2016 IEEE International Conference on Industrial Technology (ICIT), Taipei, Taiwan, 2016, pp. 1712-1217, doi: 10.1109/ICIT.2016.7475021.*

[4] *Piyush Bhardwaj, Parul Tiwari, Kenneth Olejar, Wendy Parr, Don Kulasiri, "A machine learning application in wine quality prediction", Machine Learning with Applications, Volume 8, 2022, 100261, ISSN 2666-8270, https://doi.org/10.1016/j.mlwa.2022.100261.*

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