Report

Introduction:
This assignment focuses on predicting the wine quality based on the various chemical properties. Predicting wine quality is important because traditional methods are slow and inconsistent. To overcome this we can use ML. The main objective of this assignment is to predict wine quality based on the different chemical properties using ML. I compared different models like KNN, logistic regression, SVM, Decision tree and random forest to find out the best performing model.

Background:
Predicting the wine quality by Traditional methods like lab tests can be slow and inconsistent. Machine learning gives a faster way to predict wine quality using the different chemical properties.

Prior Research: Many studies have compared different models for this task. Models like Random forest and SVM performed well due to their ability to find complex relationships between the features. Features like alcohol, volatile acidity, and pH have been identified as most influential features in predicting the wine quality.
For example, Muhammad Ihsan implemented SVM with RBF kernel on wine quality dataset and achieved 77% accuracy. This shows that choosing the right kernel and tuning the parameters can make the model work better.Similarly, a GitHub project titled Wine-Quality-Prediction-SVM-KNN reported slightly higher accuracy around 81% using the same kernel.

In my study, the SVM model (RBF kernel, C=50, gamma=1, SMOTE) achieved 77% accuracy, consistent with both previuos works.Meanwhile, Random Forest model (n_estimators = 128, max_depth = 20, min_samples_leaf = 4, max_features = 'log2', SMOTE) achieved a 73% accuracy with a higher macro average (0.58), providing better class balance.

Limitations: Most wines in the dataset are medium quality. It makes harder for the model to predict low and high quality wines accurately.

Methodology:
The dataset used in this project is Wine quality dataset from the UCI ML repository. It contains various chemical properties of the wine as the input features and the wine quality as the target. There are 11 features which include fixed_acidity, volatile_acidity, citric_acid, residual_sugar, chlorides, free_sulphur_dioxide, total_sulphur_dioxide, density,pH, sulphates, alcohol. The target variable is quality which is an integer representing the wine quality from 3 to 9. The features are Continuous variables.

In the dataset there are a total 1179 duplicates in X and 6490 duplicates in y. So, to remove the duplicates I concatenated the X and y and removed the duplicates using the drop_duplicates() function. This step makes sure that only the unique observations are used for model training. However, some feature values appear multiple times but their other attributes are different. For example acidity=7.8 appeared more than once but their other attributes like volatile_acidity, citric_acid etc are different from each other. Then I checked for the null values. There are no null values in the dataset.

The original dataset contains wine quality ranging from 3 to 9. However, since these continuous values are difficult to predict accurately, I grouped them into three different categories to simplify the task. I used a function named map_quality to classify the wines as Low quality(3) medium quality(for scores 5,6) and High quality(for scores greater than 7) Instead of predicting the exact numbers the models will now predict which group the wine belongs to. This makes the problem easier to understand and helps the models to give clear results.

After grouping the wine quality most of the wines fall into the Medium category which has 4074 samples. And the High quality has 1008 samples, and low quality has 236 samples. This shows the dataset is imbalanced with more medium quality samples compared to low or high quality samples.

I splitted the data into three parts to train and test the models. 70% of the data was used for training and the remaining 30% was equally divided into validation and test sets 15% each.

KNN

I used StandardScalar to scale the features for the KNN model. I first fitted the scalar on the training data to learn the mean and standard deviation and applied the same transformation to the validation and test sets. This helped to make sure that all features were on the same scale. I then trained a KNN model with k=5 using the scaled training data to predict the wine quality. This model achieved an accuracy of 78% (0.78) on the test data. It performed well in predicting the medium quality wines but it struggled to correctly identify the Low quality wines due to fewer samples. To overcome this I used SMOTE to balance the training data by generating samples for the minority classes.
After using SMOTE and distance -weighted KNN the model achieved 64%(0.64) accuracy. It identified Low and High wines but Medium wines were misclassified as High. To improve the accuracy I tested KNN with k = 1, 3, and 5 using distance weighting, trained on SMOTE-balanced data, and compared their macro and weighted F1-scores to find the best-performing k.k with 3 performed best with the

highest macro F1-score (0.52). I trained the final KNN model with k=3 and distance weighting on SMOTE data. It achieved 67%(0.67) accuracy.

Logistic Regression

I first trained a logistic regression model on the scaled training data to predict wine quality. It achieved 78% accuracy on the test set and did well with Medium quality wines, but had trouble correctly identifying Low and High quality wines, mainly because the dataset was imbalanced.

To overcome this, I trained a balanced logistic regression using class_weight='balanced'. This helped the model better recognize the minority classes, but the overall accuracy dropped to 52%, showing that balancing the classes changes how the model makes predictions.

I then applied SMOTE to create extra samples for the minority classes and retrained the model while tuning the regularization parameter C and l2 regularization using grid search. The results were similar to the balanced model, with accuracy still around 52%. The confusion matrix shows that the model now detects more High and Low quality wines, but Medium wines are still misclassified, showing that logistic regression struggles to clearly separate the classes in this dataset.

SVM

First, I tried a linear kernel with class balancing, which gave 56% accuracy on the test set. The model predicted Medium quality wines well, but struggled with Low and High quality wines.

Next, I used an RBF kernel, which improved the test accuracy to 62% and slightly better identified Low and High quality wines.

Finally, I tuned the RBF SVM hyperparameters (C and gamma) using grid search. The best parameters are C=50 and gamma=1. To handle class imbalance, I applied SMOTE oversampling on the scaled data. The final model achieved a test accuracy of 77%, with a confusion matrix showing stronger recognition of minority classes: *High* (recall = 0.46), *Low* (0.14), and *Medium* (0.88). The macro-average F1-score improved to 0.50, showing more balanced performance across all classes. Overall, the tuned RBF SVM with SMOTE gave the best balance between good accuracy.

Decision tree

I first trained a Decision Tree on the original training data and achieved around 70% validation accuracy. Adding class weights slightly improved the accuracy to 72%, but the model still struggled with minority classes like Low and High quality wines.

To handle class imbalance, I applied SMOTE to generate samples for minority classes. After retraining and tuning hyperparameters with grid search, the final Decision Tree achieved 63% accuracy. The results show better prediction of Low and High quality wines, though some Medium wines were misclassified.

Random Forest

I first trained a Random Forest on the original training data with 100 trees and class weighting to handle imbalance. This gave a validation accuracy of 79%, but the model mostly predicted Medium wines correctly and struggled with Low and High quality wines, as shown by the confusion matrix.

To improve performance on minority classes, I applied SMOTE to create samples for Low and High quality wines. After retraining and tuning hyperparameters using Halving Random Search, the final Random Forest with 128 trees, max depth 20, and other optimized settings achieved 73% test accuracy. The model now better predicts High and Low quality wines, though some Medium wines are still misclassified. Overall, Random Forest provided a good balance between handling class imbalance and overall accuracy.

Results

Model	Hyperparameters	Test Accuracy	Macro	Weighted
KNN	k=3, distance weighting,	0.67	0.52	0.70
Logistic Regression	C=1, multinomial, SMOTE	0.52	0.44	0.56
SVM	RBF kernel, C=50, gamma=1, SMOTE	0.77	0.51	0.76
Decision Tree	max_depth=None, min_samples_split=2, min_samples_leaf=1, SMOTE	0.64	0.46	0.67

Random Forest	n_estimators=128, max_depth=20, min_samples_leaf=4, max_features='log2', SMOTE	0.73	0.58	0.75

Classification results:

KNN

Class	Precision	Recall	F1-Score	Support
High	0.43	0.72	0.54	151
Low	0.19	0.39	0.25	36
Medium	0.88	0.68	0.77	611

Logistic regression

Class	Precision	Recall	F1-Score	Support
High	0.39	0.81	0.53	151
Low	0.12	0.67	0.21	36
Medium	0.91	0.44	0.59	611

SVM

Class	Precision	Recall	F1-Score	Support
High	0.54	0.46	0.50	151
Low	0.28	0.14	0.19	36
Medium	0.83	0.88	0.86	611

Decision Tree

Class	Precision	Recall	F1-Score	Support
High	0.37	0.50	0.43	151
Low	0.13	0.31	0.19	35
Medium	0.83	0.69	0.75	612

Random Forest

Class	Precision	Recall	F1-Score	Support
High	0.49	0.74	0.59	151
Low	0.28	0.44	0.34	36
Medium	0.89	0.75	0.81	611

Discussion:

Among all the models **SVM** achieved the highest test accuracy(77%) and strong performance of the Medium class. However, the **Random forest** model showed the best overall performance with accuracy(73%) giving a more balanced classification across all the classes(High,medium,low). **KNN** performed moderately well, especially in identifying minority classes after using SMOTE, while **Logistic Regression** and **Decision Tree** struggled to balance performance across all classes.

Feature importance analysis across models shows that **alcohol content, volatile acidity, and pH** are the most influential factors determining wine quality, whereas **citric acid** and **residual sugar** have relatively minimal impact. However, there are some limitations. Although the models improved in identifying High and Low quality wines, they still misclassified some Medium quality wines.

Conclusion

In this study, several machine learning models were evaluated to predict wine quality using the UCI Wine dataset. Among the models tested, **SVM** achieved the highest test accuracy (77%) and showed a strong balance between identifying medium- and high-quality wines, while **Random Forest** also performed well (73%) It gave a more balanced classification and provided insights into feature importance. **KNN** improved with SMOTE but still lagged behind SVM and Random Forest, and **Logistic Regression** and **Decision Tree** struggled with class imbalance.

Key features influencing wine quality include **alcohol, volatile acidity, and pH**, while other chemical properties had less impact. For future research, exploring more advanced ensemble methods, deep learning models, or feature engineering could further improve performance. Additionally, collecting more data for underrepresented classes may help reduce misclassification and enhance model reliability.

References:

https://imbalanced-learn.org/dev/references/generated/imblearn.over_sampling.SMOTE.html

https://www.geeksforgeeks.org/machine-learning/smote-for-imbalanced-classification-with-python/

https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.StandardScaler.html

https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV.html

https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.HalvingRandomSearchCV.html

https://emhaihsan.medium.com/wine-quality-prediction-with-support-vector-machine-ebf560c9940d

https://github.com/joelvarma/Wine-Quality-Prediction-SVM-KNN/blob/master/winequality-svm-knn.ipynb