Comparing Random Forests and KNN (K-Nearest Neighbours) in classifying Wine Quality

Description and Motivation

The primary objective of this project is compare two supervised ML algorithms, Random Forest (an ensemble method) and KNN (K-Nearest Neighbours) (an instance based/ lazy-learning algorithm) in terms of predicting Wine Quality based on its concentration of ingredients. This is a multi-classification problem, with wine quality being from 0-10. Our results are compared to two sources:

- Sarav Rajavelu (2018) for KNN
- Dr. Soumen Atta, Ph.D. (2023) for RF

Exploratory Data Analysis (EDA)

Data Intake Report and Pre-Processing

- Dataset: Wine Quality Dataset from Kaggle (Dataset link provided in references section)
- This dataset has 1143 rows and 13 columns, with 13 numeric variables. The target class is labelled "quality".
- Figure 1 displays the Data Intake Report for this dataset occupying a small amount of memory (78.06kB)
- No null value and duplicate values were observed. We employ a "minmax" scaler as we want all the data on a positive scale to standardise all the data due to high skewness for sulfur dioxide concentrations (free sulfur dioxide and total sulfur dioxide)
- The target column features values from 3-8 with a higher number representing a better quality; Multi Classification
- All the features have a correlation higher than 0.1 towards quality except "Id" as shown in Figure 3, however this is an identifier variable so we do not remove it.
- From observing the data, we set the threshold for outliers to be 3 * IQR instead of 1.5 * IQR. This is because removing outliers within the latter range results in above 30% loss in data. We want to minimise data loss because this is a small dataset.
- Figure 5 features a series of boxplots corresponding to Numerical Variables left in the dataset; we remove the outliers with red crosses, highlighting outliers which belong to the threshold above 3.0 x IQR.
- Figure 4 highlights the class distributions of the "quality" variable from 3-8 and the class distribution after employment of oversampling techniques. Synthetic sampling is used to ensure less bias and less overfitting of the model since classes have an equal chance of being predicted.
- There are 421 outliers removed, which is 14.53% of the resampled data. Majority of these wins in the outliers have very high residual sugar and chloride concentration.

Data Analysis

- Figure 2 shows the distributions for various ingredients and characteristic of wine which affect their quality. Higher wine score qualities tend to have a larger percentage of alcohol, citric acid and volatile acidity.
- We observe that variables such as residual sugar, pH, total sulfur dioxide, free sulfur dioxide, sulphates, chlorides do not have a significant impact on quality score.
- Wines with a higher density (> 1.00) tend to have a lower quality score.
- We use synthetic sampling to cater for the class imbalance shown in Figure 4.
- All data is presented as numerical values which is ideal, for input into the machine learning algorithms. There is no need for label encoding.
- Higher volatile acidity (above 1.25) suggests a lower quality score.

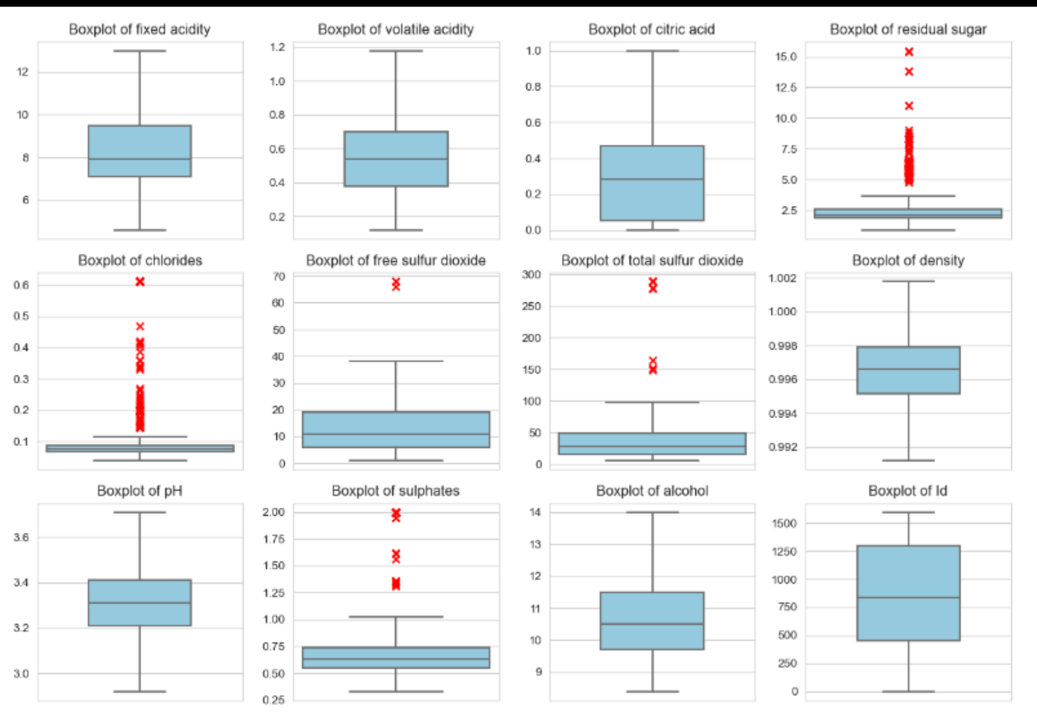


Figure 5: Boxplots with different data points for the variables

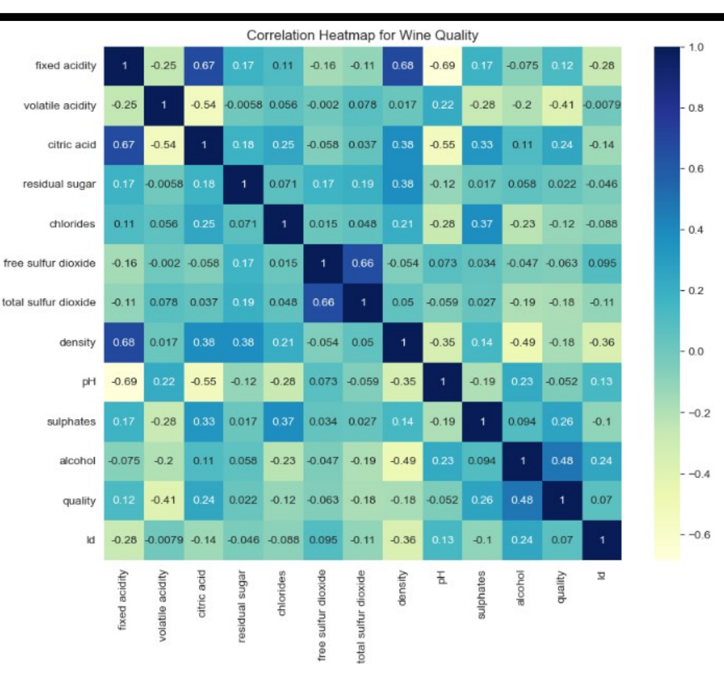
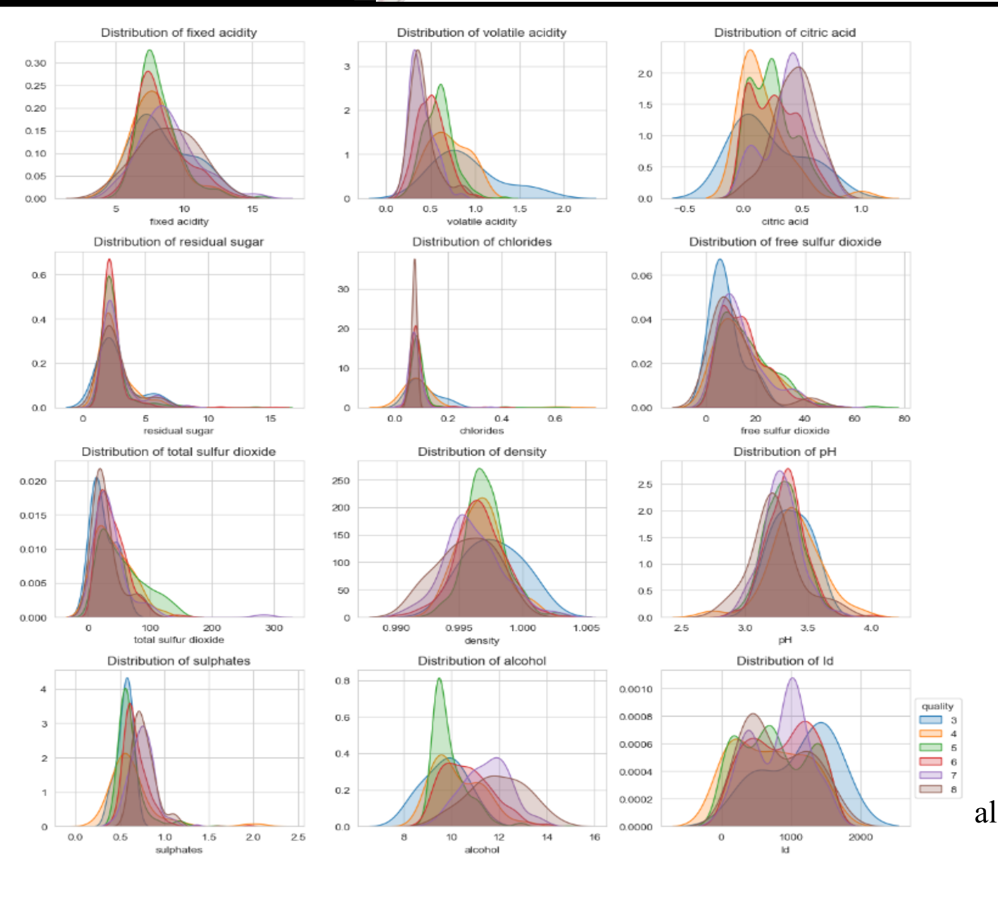


Figure 3: Correlation Matrix for all Variables



all

Figure 2: Distribution Plot of all Independent Variables with Legend as the Quality

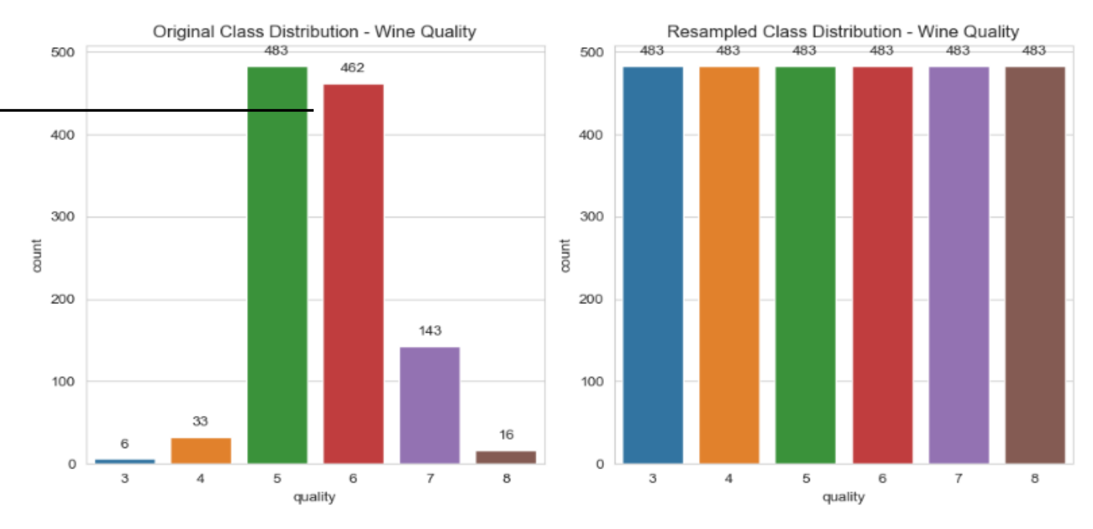


Figure 4: Wine Quality Class Distributions—Oversampling to avoid bias

Total Number of Observations	1143
Total Number of Files	1
Total Number of Features	13
Base Format of Files	.csv
Size of Data	78.06 kB
Number of Null Values	0
Number of Duplicate Values	0

Figure 1: Data Intake Report

K-Nearest Neighbours (KNN) Algorithm

- KNN is a non-parametric, instance-based learning algorithm which works by finding "k" training samples closest in distance (Euclidean Distance) to a new point and predicting the label based on these nearest neighbours.
- All known data points are transferred into a sample space, then KNN calculates the closest data points in the space which are distributed based on their characteristics.
- The "k" hyperparameter specifies the distance between data points to find the closest data points from our training data for any new data point. A small "k" making the model sensitive to noise, while a large "k" makes it computationally expensive and less precise.
- Users are given the freedom of choosing the distance metric measured between each point in the sample space, with Euclidean, Manhattan, Cosine and Hamming distance metrics. KNN calculates the distance between data points in order to find the optimal value of "k".
- We firstly select K, the number of neighbours; calculate the distance between K number of neighbours and identify the number of neighbours in each category; before assigning new data points to the category for which the neighbours are maximum; and finally having our model ready
- It is a simple algorithm; with a few hyperparameters, and is commonly within industry.

Random Forest (RF) Algorithm

- Random Forest is an ensemble learning method, using multiple decision trees during training and outputting the mode of the classes (classification). A large number of decision trees operate as an ensemble, with each tree splitting out as a class prediction and the class with the most votes becomes the model's prediction.
- Additional randomness is added to the model, whilst growing the trees. Instead of searching for the most important feature while splitting a node, it searches for the best feature among a random subset of features. This results in a wide diversity among the trees, which generally results in a more robust overall model.
- The culmination of many decision trees makes it a larger and more reliable source for larger databases. Random Forest Models are an example of a bagging technique used for Decision Trees.

Advantages and Disadvantages

- kNN struggles with larger datasets, due to its memory needed to store all the data and computation time (each prediction requires a search through the entire training set). On the other hand, kNN is a **simple** algorithm which is straightforward and easy to understand, making it a good starting point.
- Furthermore, kNN is non-parametric and makes no assumptions about the underlying data distribution, and it has the ability to capture interaction between variables which helps it understand the complex relationships within wine quality features.
- However, feature scaling is important for KNN but not the random forest model, so more extensive pre-processing is required.
- Random Forest has more hyperparameters and requires a more extensive fine-tuning process unlike KNNs which only has one key hyperparameter (K).
- KNNs are a more sensitive algorithm and are less robust than Random Forests to overfitting.
- Due to the curse of dimensionality, the performance of KNNs deteriorates in high-dimensional spaces (many features).

How Decision Trees within the Random Forest Work

- Starting with a root node is based on results from the Attribution Selection Measure, this root node will split into two nodes (the model chooses the split which maximises Information Gain) .
- The internal nodes are sub-nodes which split from the root node into further nodes, and the algorithm makes a decision on which internal node to select based on the information gain on each of the nodes for each split of the tree.
- The leaf node at the end finally makes a decision based on the target variables by learning simple decision rules inferred from root nodes and internal nodes.
- This will finally be used predict the class of a randomly picked point.
- The Random Forest model takes the average prediction of a culmination of decision trees (number of trees is a hyperparameter) as the final class output.

Hypothesis Statement

- Based on the work done by Dr. Soumen Atta and Sarav Rajavelu, Dr Atta's Random Forest model has an accuracy of 65.9% and Saray Rajavelu's KNN model had an accuracy of 75.6% when the data was not normalised and the K value = 1, whereas when the model was min-max normalised and k = 43, the accuracy was 73.1%. From these results we make two hypotheses.

Hypothesis 1: Min-Max Normalisation results in lower model accuracy in the wine quality multiclassification task than Non-Normalisation.

Hypothesis 2: The KNN will outperform the Random Forest Model.

Experimental Results, Choice of Parameters

Minimum Leaf Size	Number of Predictors to Sample	Maximum Number of Splits	Accuracy
1	1	10	80.78%
1	1	20	82.12%
1	1	30	84.54%
1	1	40	85.08%
1	3	10	82.39%
1	3	20	83.20%
1	3	30	84.95%
1	3	40	85.89%
1	5	10	81.59%
1	5	20	84.41%
1	5	30	84.27%
1	**5**	**40**	**86.02%**
1	7	10	82.80%
1	7	20	85.35%
1	7	30	85.48%
1	7	40	85.89%
5	1	10	80.11%
5	1	20	81.45%
5	1	30	84.54%
5	1	40	84.95%
5	3	10	81.59%
5	3	20	82.26%
5	3	30	83.60%
5	**3**	**40**	**86.16%**
5	5	10	82.80%
5	5	20	85.35%
5	5	30	84.81%
5	5	40	85.89%
5	7	10	83.60%
5	7	20	84.95%
5	7	30	84.95%
5	7	40	85.48%
10	1	10	81.18%
10	1	20	81.32%
10	1	30	85.22%
10	1	40	85.22%
10	3	10	82.53%
10	3	20	83.33%
10	3	30	84.54%
10	3	40	85.35%
10	5	10	82.80%
10	5	20	83.60%
10	5	30	84.68%
10	5	40	85.22%
10	7	10	83.33%
10	7	20	84.95%
10	7	30	84.68%
10	7	40	85.08%
15	1	10	80.51%
15	1	20	83.06%
15	1	30	84.27%
15	1	40	84.27%
15	3	10	82.53%
15	3	20	83.74%
15	3	30	84.01%
15	3	40	84.41%
15	5	10	83.47%
15	5	20	85.08%
15	5	30	84.14%
15	5	40	85.22%
15	7	10	84.54%
15	7	20	84.68%
15	7	30	84.68%
15	7	40	85.08%
20	1	10	80.51%
20	1	20	83.07%
20	1	30	84.27%
20	1	40	84.27%
20	3	10	82.53%
20	3	20	83.74%
20	3	30	84.01%
20	3	40	84.41%
20	5	10	83.47%
20	5	20	85.08%
20	5	30	84.14%
20	5	40	85.22%
20	7	10	84.54%
20	7	20	84.68%
20	7	30	84.68%
20	7	40	85.08%

KNN Algorithm

- We use a baseline model with the K parameter set to 3, which is the Euclidean Distance Metric.
- The KNN model had an accuracy of 79.3%, and a Cohen's Kappa Coefficient of 0.752
- We test our final KNN model on the test data and record our results

Choice of Parameters

- There are no optimal K parameters, changing this parameter has no effect on the accuracy.
- We choose k=1 in this case, as it is the simplest model saving computational resources

Random Forest

- We use a baseline model with a minimum leaf size of 10, number of predictors to sample of 3, maximum number of splits of 20.
- The Random Forest model had an accuracy of 86.02%, and a Cohen's Kappa Coefficient of 0.839
- We iterate through the sets of parameters:

Minimum Leaf Sizes = [1, 5, 10, 20]

Number of Predictors to Sample = [1, 3, 5, 7]

Maximum Number of Splits = [10, 20, 30, 40]

Normalised Data	Accuracy
KNN, No	70.6%
KNN, Yes	79.3%
Random Forest, No	82.3%
Random Forest, Yes	86.1%

Figure 12: Normalised Data

Choice of Parameters

- The most optimal parameters are minimum leaf size of 5, number of predictors to sample at 3, maximum number of splits at 40.

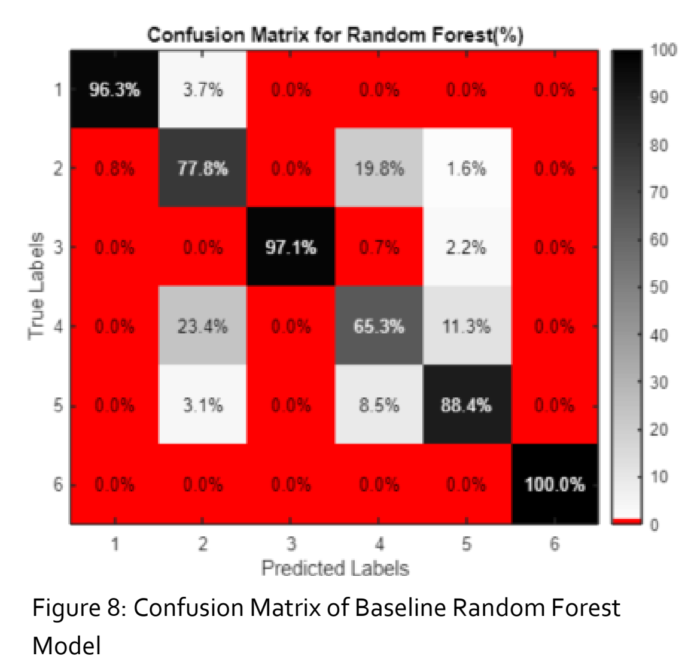


Figure 8: Confusion Matrix of Baseline Random Forest Model

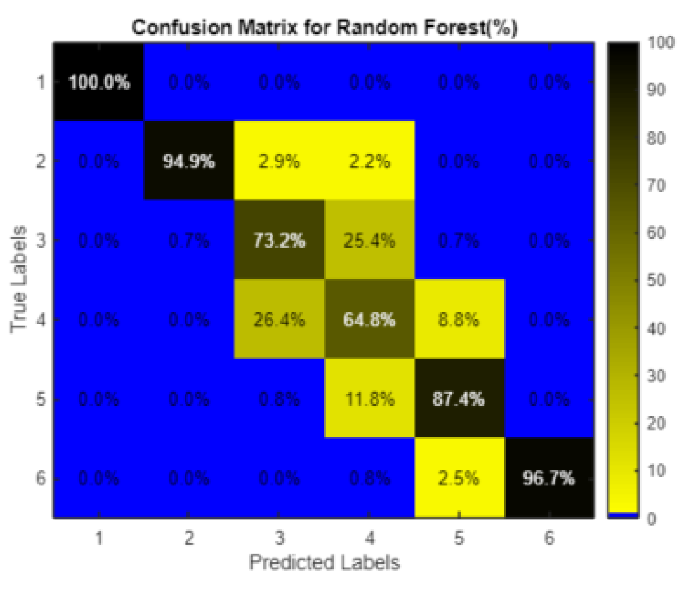


Figure 9: Confusion Matrix of Optimal Random Forest Model

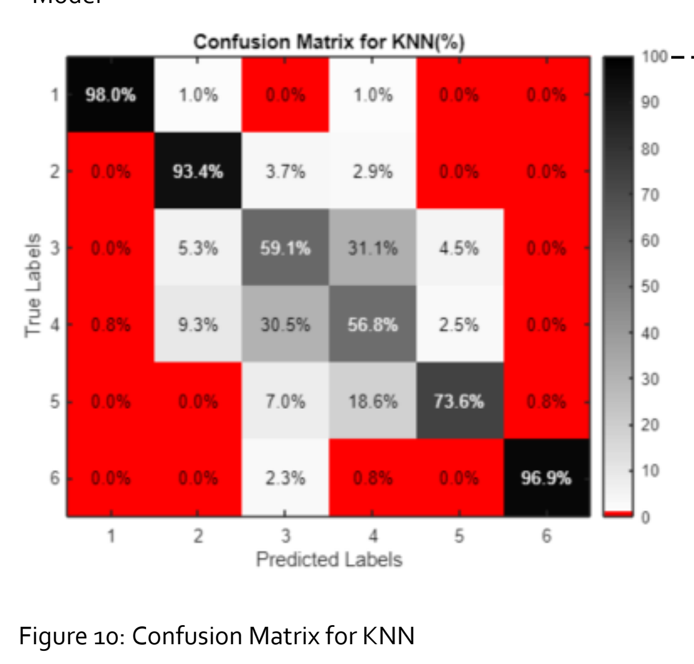


Figure 10: Confusion Matrix for KNN

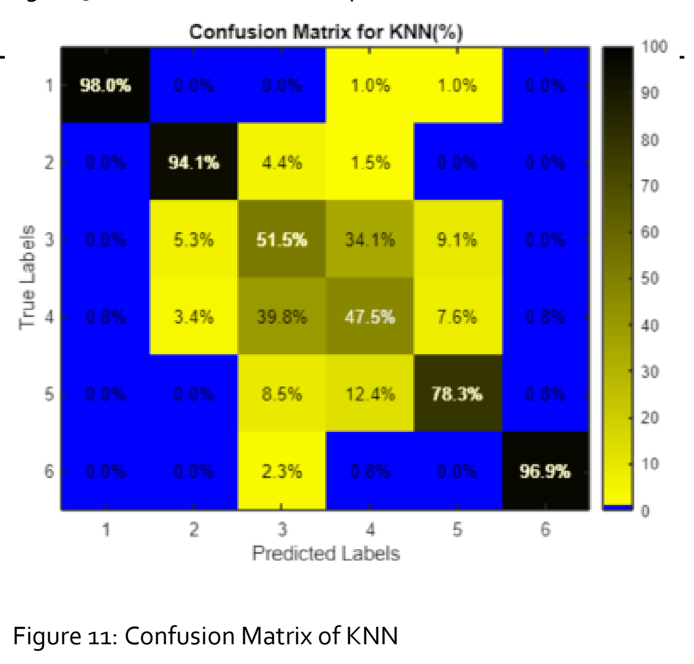


Figure 11: Confusion Matrix of KNN

K (Number of Neighbours)	Accuracy
1	79.30%
3	79.30%
5	79.30%
7	79.30%
9	79.30%

Figure 6: KNN

Metric	KNN (K-Nearest Neighbours)	Random Forest
Precision	80.8%	87.6%
Recall	79.6%	87.5%
F1-Score	80.0%	87.5%
Accuracy	79.3%	87.2%
Cohen Kappa	0.752	0.839

Figure 7: Performance Metrics for KNN and Random Forest Models

Methodology

1. After data cleaning and pre-processing, we split the data into 70:30 for train and test data using the holdout method. The test data will remain unseen till the end.
2. The model is fitted on the training data, with all hyperparameters specified for base models. **Random Forest Hyperparameters:** Depth of the Trees (1), Number of Predictors at each node (20), Number of trees (100). **K-NN Hyperparameters:** Number of Neighbours (5).
3. We evaluate these models on the training data, checking their accuracy, precision, recall and f1-score and comparing these performance metrics. We also obtain the Cohen Kappa Coefficient which measures the extent to which agreement among raters is greater than chance alone (the chance a class is obtained by chance compared to agreement among raters).
4. After this, we begin our optimisation process, where we optimise models based on their hyperparameters, iterating through different sequences of them before producing our final results. The best combination of hyperparameters is used within our final model. We also employ the use of a Classification Learner to check our results, and see any improvements or perhaps if we can use different models.
5. Measure and compare performance between original and optimized models as well as between the two final optimized models.
6. After a critical analysis of model results, we make a decision as to which model is better for the data and evaluate the hypotheses. We compare the results with the two articles we have sourced below in the references section.

Analysis and Evaluation of results:

- The Random Forest model performed better than the KNN (K-Nearest Neighbours) overall, with 86.16% accuracy compared to 79.3%. This is perhaps due to the nature of the data, since the KNN is not as efficient in large datasets, and the size of the dataset increases with oversampling.
- After iterating through all the combinations of hyperparameters for the Random Forest Model specified in the Results section, with the minimum number of leaf sizes from 1, 5, 10, 15, 20, the number of predictors to sample and the maximum number of splits including all the nodes (depth of the trees). There is only slight variation of results during the optimisation process, with all the results staying between 80.0% - 87.0%.
- The KNN model has a Cohen Kappa coefficient of 0.752 and the Random Forest model scored 0.839. Comparing these two results, we can state that the Random Forest performs better than the KNN model in the context of wine classification. This is due to the Random Forest's ability to handle complex interactions between features more effectively than the KNN. The higher score for the Random Forest Model indicates it has very good agreement with the true labels, falling into the category of almost perfect agreement, suggesting this model is accurate for classifying wines correctly. On the other hand, the KNN model with a Cohen Kappa coefficient of 0.752 indicates the model's predictions are much better than what would be expected by chance, however there is still room for improvement.
- The Random Forest model outperforms the KNN across all three metrics (precision, recall and f1-score). Random Forest's precision score of 87.6% means that it is better at minimising false positives compared to the KNN (80.8%). A high recall score of 87.5% means that it is better at minimising false negatives, compared to the KNN at 79.6% relevant instances. An F1-Score of 80% for the KNN suggests a good balance between precision and recall but the Random Forest model having a prediction of 87.5% suggests this model is more effective in terms of accuracy and completeness in classifying wine quality.
- When we look at normalised data, we can see that the KNN (K– Nearest Neighbours) has a higher accuracy increase when we normalise X (8.7% increase) whereas the Random Forest model only increases with an accuracy of 3.8%. This underscores the importance of normalising data for KNN algorithm, emphasising its feature sensitivity and sensitivity to outliers.
- From our confusion matrices, we observe that the Random Forest model only achieves a slightly higher accuracy in predicting the lower and upper classes (1,2,5 and 6) however 3 and 4 are for lower accuracies and predictions. The key difference in in classes 3 and 4, with the optimised random forest model achieving 73.2% and 64.8% accuracy compared to the KNN model at 51.5% and 47.4% respectively.
- From our results we can conclude that the Random Forest Model is the superior model with a higher predictive power and completeness in classifying wine quality. Both Hypothesis 1 and Hypothesis 2 were incorrect, perhaps due to the data being different for Dr Soumenn Atta's results [2] and Sourav Rajavelu's results [1]. The Random Forest Model outperformed the KNN model instead and normalisation improved results of the data.

Lessons Learned:

- More data is needed to make accurate predictions. Despite wine quality classes having an equal balance due to upsampling, and without synthetic sampling there is significant class imbalance. Furthermore, there is not enough data for all wine classes as this goes from 0-9, although we only have data for 6 of these, from 3-8.
- The Random Forest Model is better for most multi-classification tasks where the number of labels is greater due to the sophistication of the model. It is also more robust to outliers and can identify more complex relationships within skewed data.
- Optimisation for the Random Forest model took a significant amount of time, due to so many different parameter combinations to train.

Future Work:

- We can evaluate other models, such as Logistic Regression, Naïve Bayes, Random Forests and SVM.
- Another consideration would be pruning our final decision tree or consider boosting/ bagging.
- For KNN, more feature selection could be done since there are some factors that affect the results more than others. We could look into changing the distance metric from Euclidean distance to minkowski distance or cosine similarity.

References

[1] saravrajavelu/WineQualityAnalysis: Wine Quality analysis and prediction using a kNN classifier built from scratch using Python, Pandas & Numpy. (github.com)

[2] Building a Random Forest Classifier with Wine Quality Dataset in Python | by Dr. Soumen Atta, Ph.D. | Medium

- Dataset: Wine Quality Dataset (kaggle.com)