The University of Melbourne

School of Computing and Information Systems

COMP30027 Assignment 1 - Project Report

Jason Tang and Bruce Doan

1 Aim

In this report, we explore the performance of a supervised machine learning model, specifically the K-Nearest Neighbors (K-NN) model, to classify the quality of wines from their chemical properties. By further analysis on the provided dataset, we also intend to gain insight on the underlying correlations between various features that influence wine quality. Furthermore, we aim to extend our model and critically analyse the performance of the basic K-NN model by altering hyperparameters such as k, distance metric and weighting. Our findings from analysing the model may have practical implications, potentially aiding in the prediction of wine qualities prior to production. Moreover, our experimentation may extend our understanding on the nuances of implementing a K-NN model, and its effectiveness in classification tasks.

2 Dataset

The main dataset that was utilised in this investigation was the modified sub-set of the “Wine Quality” dataset from the UCI Machine Learning Repository [1]. The original dataset can be found at https://archive.ics.uci.edu/dataset/186/wine+quality. Notably, the dataset consists of 11 chemical properties/attributes for 2700 wine samples, illustrated in table (1). The class label for each sample is ‘quality’, which has been encoded as a boolean variable (0 = low quality, 1 = high quality). A stratified 50/50 train-test split has been used and provided by the teaching staff. All training and testing features and instances can be found in their respective files ‘winequality-train.csv’ and ‘winequality-test.csv’. This train/test split was used in all aspects of this investigation unless specified differently.

fixedAcidity	freeSulfurDioxide
volatileAcidity	totalSulfurDioxide
citricAcid	density
residualSugar	pH
chlorides	sulphates
alcohol	

Table 1: Features used in the dataset by category

3 Analysis Methods

3.1 Methods

Within supervised machine learning, there are two main algorithms; regression and classification, with regression being used to predict continuous variables whilst classification is used to categorise variables into discrete classes or clusters. Our focus lies in classification, employing the use of the K-Nearest Neighbours (K-NN) model. By utilising continuous variables across all attributes, we were able to generate confusion matrices and leverage the in-built 'classification_report' function from the scikit-learn library to compare the accuracy, precision and recall scores across the different K-NN models we implemented. We also wanted to learn more about the features and so scatter plots were plotted between different attributes to make observations on correlation. We also conducted a comparative analysis between our top-performing normalised K-NN model and Gaussian Naive Bayes (GNB) to observe any discrepancies. Furthermore, we decided to conduct feature analysis to justify the differences in predictions for these models.

3.2 Training-test splits

It is industry standard practice to divide the dataset into training and test splits in supervised learning to evaluate model performance. This often includes extended variations using cross-validation to test multiple times on different subsets of the data and averaging the accuracies. However, for this project, a stratified 50-50 split on training and test dataset was provided, which was exclusively used for all our implementations and evaluations.

The empirical distributions of class labels in the training dataset and testing dataset would usually be a good indicator of whether a different training-test split is required. We observed that our training dataset is unbalanced and we see more '0' labels than '1' labels, therefore an extension to this experiment could involve altering the ratio for training-test splits.

Figure 1: Empirical distribution of class labels in the training and testing datasets

3.3 Modelling

In our purpose of predicting the wine quality from the 11 chemical properties, we fit a K-NN model to the training data using scikit-learn, numpy and scipy libraries. Leveraging these libraries allowed us to conduct matrix operations to enhance computational speed. Additionally, we utilised these libraries to incorporate techniques such as preprocessing, normalisation and data weighting to improve model performance. To visualise our findings, we also employed libraries such as Matplotlib, Seaborn and Tabulate. These tools were crucial in representing our data through various visualisation techniques such as heatmaps, tables, bar plots, and scatter plots. This visualisation aided in our understanding of the dataset and the insights derived from our analysis.

Our modelling process involved preprocessing steps outlined in section 3 of the project. These steps standardised the data and handled any missing values or outliers. Through this process, we aimed to ensure that all attributes contributed equally to the distance calculations within the K-NN algorithm. Specifically, we implemented min-max scaling and standard scaling on our base 1-NN model and visualised their results in confusion matrices for comparison.

Furthermore, hyperparameter tuning was performed to optimise the performance of the K-NN model. We experimented with different distance metrics in section 4.2 and visualised their impact on model accuracy using the provided test dataset both with and without normalisation. Results were illustrated via a barplot to determine the most suitable distance measure for our dataset. We also explored weighting options such as Inverse weighting for labelling new instances and experimented with different values of K for both weighting and majority voting to identify potential improvements.

Moreover, we also implemented a GNB model in section 4.1 for comparative purposes. The GNB model was trained using the same training dataset, and its performance was evaluated alongside the K-NN model using appropriate evaluation metrics. We also visualised instances where the predictions were inconsistent and conducted feature analysis on these instances to justify the differences.

Overall, the modelling process involved rigorous experimentation and finte-tuning of K-NN hyperparameters to develop extensive classification models capable of ‘generalising’ and accurately distinguishing between low and high quality wines.

4 Results and Discussion

In this section, we interpret and analyse our model's performance, drawing insights from visualisations and examining its effectiveness in classifying wine quality based on chemical properties.

Section 2: 1-NN Classification

Our 1-NN model with no feature selection, original dataset, and no majority voting was able to predict wine quality in our testing dataset with an accuracy of 76.44%. In isolation, this score suggests our model, with the respective 11 features, was relatively effective in predicting wine quality. However, due to the imbalanced nature of the dataset in both the training and testing datasets observed in figure (1), this result should be interpreted alongside the confusion matrix in figure (2) and the classification report in table (2).

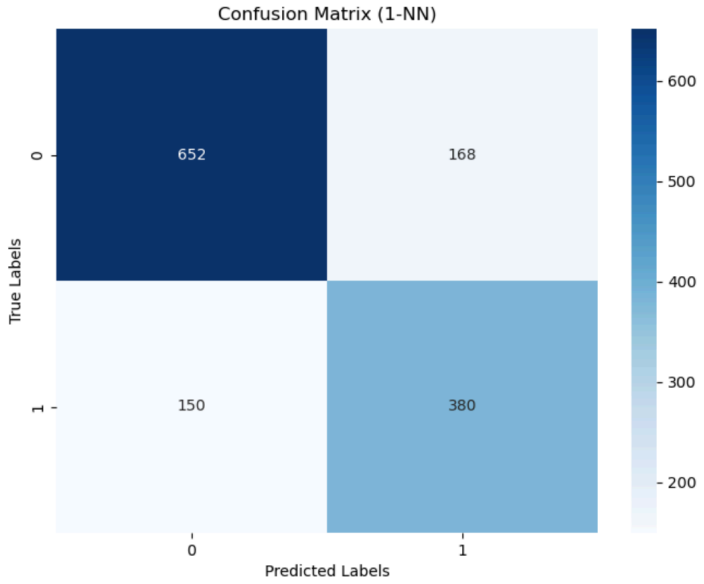

Figure 2: Confusion matrix (CM) for 1-NN model

Class	Precision	Recall	F1-score	Support
0	0.81	0.8	0.8	820
1	0.69	0.72	0.71	530
accuracy			0.76	1350
Macro avg	0.75	0.76	0.75	1350
Weighted avg	0.77	0.76	0.77	1350

Table 2: Classification report for 1-NN model

The confusion matrix suggests that a significant portion of the positive influence on the f1 score is derived from accurately predicting '0' class wines (652 true positives) compared to '1' class wines (380). This skewed predictive count in favour of label '0' over label '1' raises concerns with regards to the model's capability to generalise.

Furthermore, the observation of highly correlated features in the dataset, as depicted in figure (3), suggests that certain features in the dataset can exhibit strong relationships between each other. For a 1-NN classification, which relies on proximity of instances to make predictions, this overlap in feature space can create challenges. When instances are close together and noise is present, the model may exhibit uncertainty and become more susceptible to misclassification, especially when k is low.

Therefore, the observed correlation between features, coupled with the imbalance in class distributions can present significant challenges for a 1-NN model, and so we propose that this dataset would not be suitable for a 1-NN classification model without any improvements.

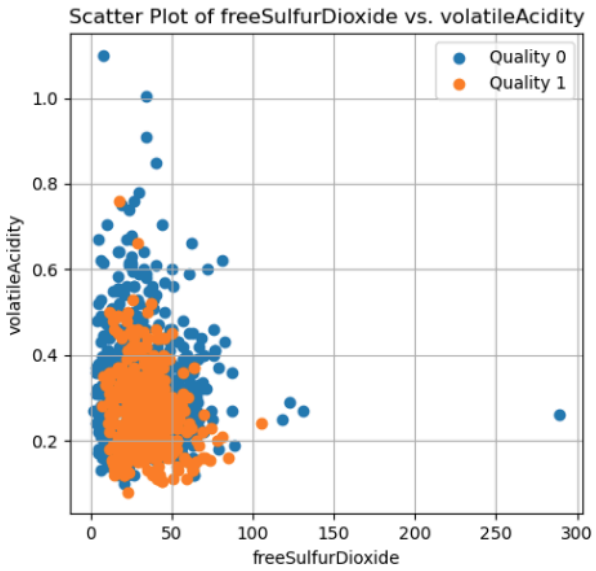

Figure 3: freeSulfurDioxide vs. volatileAcidity

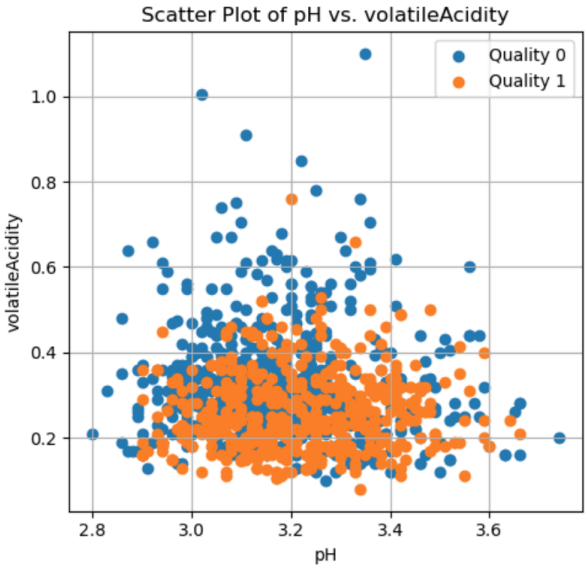

Figure 4: pH vs. volatileAcidity

Section 3: Normalisation

Our normalised models outperform the basic 1-NN model, achieving 85.04% accuracy for min-max scaled data and 86.74% for standardised data. Precision, recall, and F1 scores are consistently higher for class '0' wines. For class '0', F1 scores are 0.80 (no normalisation), 0.87 (min-max scaling), and 0.89 (standardisation). For class '1', F1 scores are 0.71 (no normalisation), 0.81 (min-max scaling), and 0.84 (standardisation). Overall, standardisation produces the best performance, followed by min-max scaling and finally our basic K-NN model.

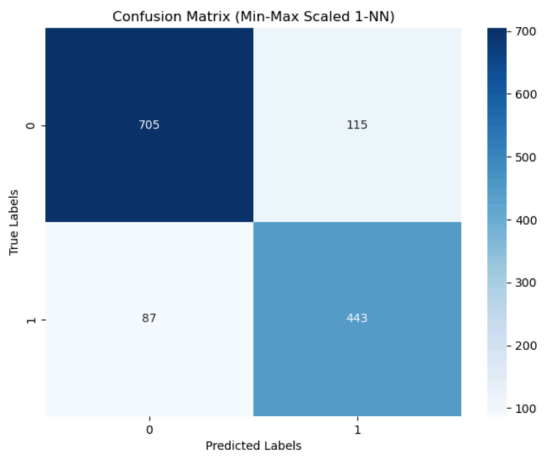

Figure 5: CM for 1-NN model (min-max scaled)

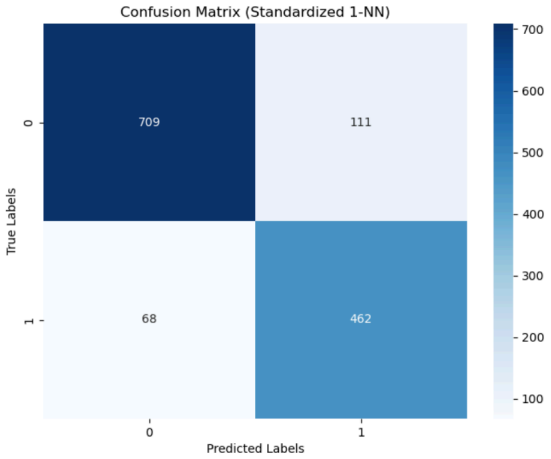

Figure 6: CM for 1-NN model (standardised)

Class	Precision	Recall	F1-score	Support
0	0.89	0.86	0.87	820
1	0.79	0.84	0.81	530
accuracy			0.85	1350
Macro avg	0.84	0.85	0.84	1350
Weighted avg	0.85	0.85	0.85	1350

Table 3: Classification report for normalised 1-NN (min-max scaling) model

Class	Precision	Recall	F1-score	Support
0	0.91	0.86	0.89	820
1	0.81	0.87	0.84	530
accuracy			0.87	1350
Macro avg	0.86	0.87	0.86	1350
Weighted avg	0.87	0.87	0.87	1350

Table 4: Classification report for normalised 1-NN (standard scaling) model

Normalisation serves as a method to scale the ranges of data and prevent features with larger scales from dominating the distance calculations. This ensures a more balanced contribution from all features, crucial for effective model learning. We visualise the effect of normalisation in figure (7) depicting free sulfur dioxide and volatile acidity levels. Although the overall distributions of data points remain consistent, there are notable discrepancies in the axis scales.

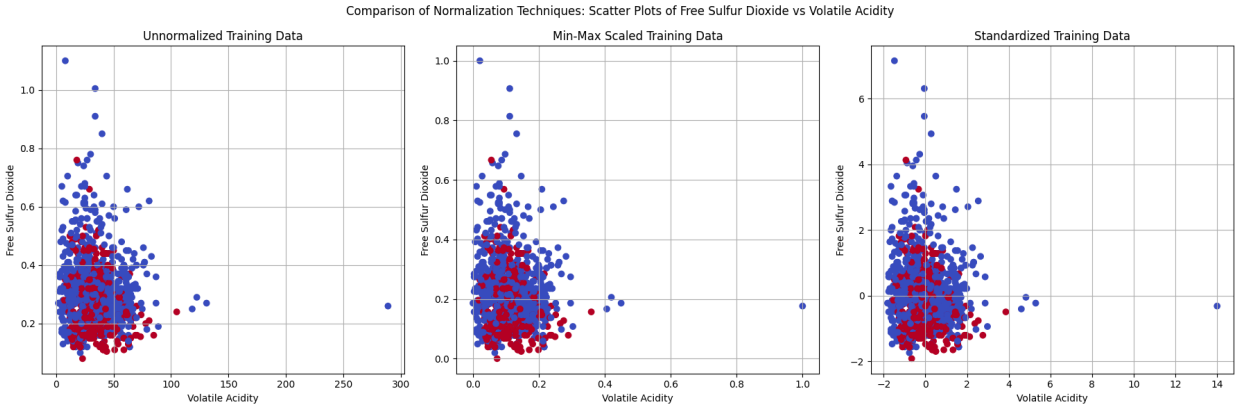

Figure 7: Comparison of 2D scatter plots of free sulfur dioxide vs volatile acidity using different normalisation techniques

While unnormalised training data exhibits wide ranges (e.g. volatile acidity from 0 to 300), min-max scaling aims to compress the values to a range of 0 to 1, facilitating better comparison and interpretation. In contrast, standardised scaling endeavours to standardise all attributes to have a mean of 0 and a standard deviation of 1 (e.g. volatile acidity from -2 to 14). This preserves the original distribution of values while mitigating the impact of large scales. This adjustment may explain why standardised data led to the best performing model.

Section 4.1: GNB vs K-NN

In this section, we conducted a comparative analysis on the performance of our standardised 1-NN model to a Gaussian Naive Bayes (GNB) model on the same dataset. Using the GNB model from sklearn, we were able to reach an accuracy of 77.78%, while our standardised 1-NN model outperformed it with an accuracy of 86.74%. Out of the 1350 test instances, we found a total of 265 inconsistent cases where the GNB model and 1-NN model classified instances differently.

Initially, we hypothesised that the comparative differences could be attributed to several factors. GNB assumes independence between features conditional on class, thus it might not perform as well in cases where features are correlated. We also recognised that the linear decision boundary learned by GNB contrasts with the more complex boundaries of the 1-NN model, which operates in a high-dimensional feature space and bases its predictions on proximity to training instances.

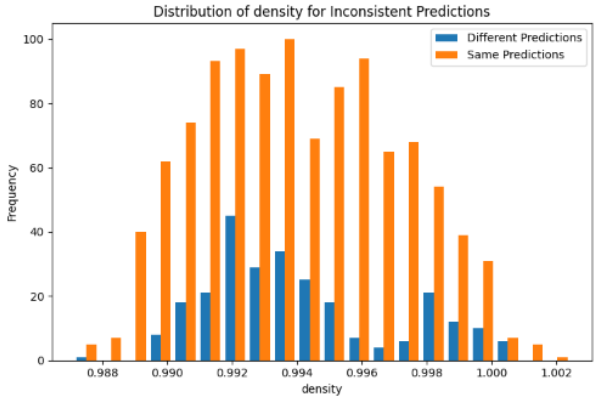

Figure 8: Inconsistent cases on density

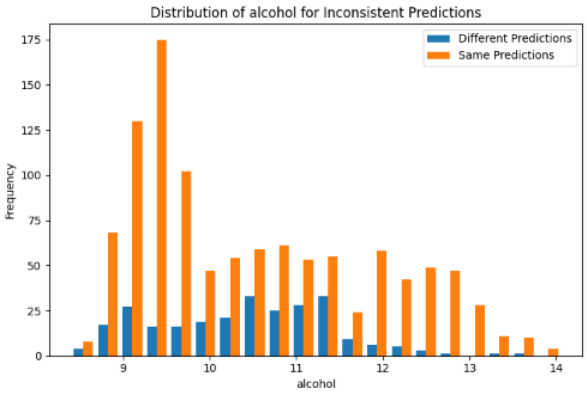

Figure 9: Inconsistent cases on alcohol

From the visualisations observed in figures (8) and (9), we observed deviations in the distributions of certain features when models disagreed. Features such as ‘density’ and ‘alcohol’ showed notable differences in distribution between instances with consistent and inconsistent predictions of wine quality. Therefore, further investigations into these features could provide valuable insights into determining the rationale behind differing model decisions. We might also choose to conduct feature correlation analysis to explore the relationships between features and how they influence model decisions.

Section 4.2: Distance metric for K-NN

In our investigation of the K-NN model, we explored the impact of another hyperparameter - the distance metric. Our basic model used euclidean distance, and here we implemented two additional measures: cosine similarity and Mahalanobis distance. Each distance measure was evaluated using these three normalisation options: no normalisation, min-max scaling, and standard scaling using a 1-NN Classifier.

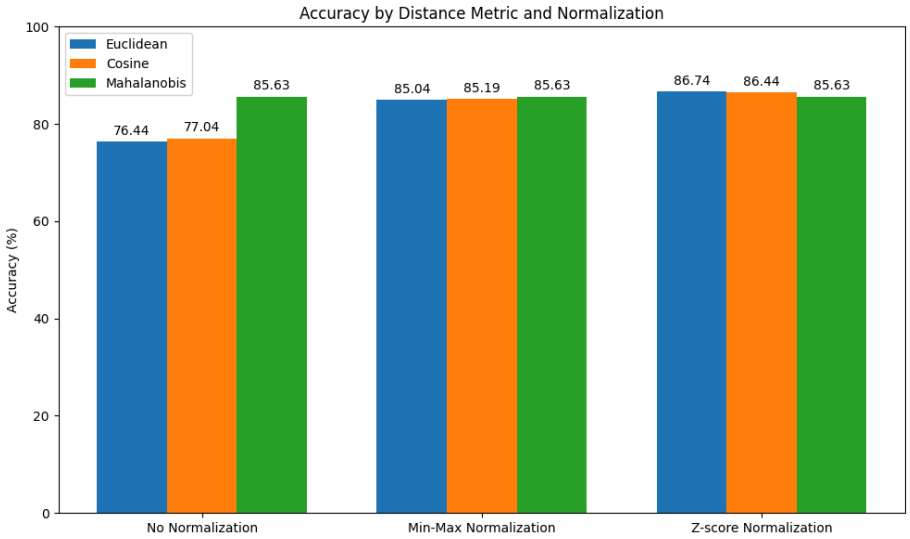

Figure 10: Comparison of distance metrics for 1-NN under different normalisation methods

From the results pictured in figure (8), we can see that Mahalanobis distance outperformed other options, achieving a very high average accuracy of 85.63%. This is followed by cosine similarity and euclidean distance attaining an average accuracy of 82.89% and 82.74%, respectively. Upon further inspection of each normalisation group, no normalisation yielded an average accuracy of 79.70%, while cosine similarity and Mahalanobis distance yielded 85.29% and 86.27%, respectively. These findings support our observations made previously in Section 3, reinforcing the effectiveness of standardised scaling. Notably, the highest accuracy of 86.74% was achieved using standard scaling and euclidean distance whereas, the lowest accuracy of 76.44% was observed where no normalisation methods were applied with euclidean distance.

Overall, our analysis reveals that models using cosine similarity and Mahalanobis distance both outperformed euclidean distance, in scenarios where no normalisation or min-max scaling was applied. However, when standard scaling was employed, euclidean distance was the top performer, suggesting that a nuanced balance between distance metric and normalisation technique is required for optimising K-NN performance.

Section 4.3: Weighting Strategy

We implemented an additional weighting strategy using inverse distance to compare its performance with our majority voting model across selected values of K. Figure (9) indicates that as K increases, the accuracy of the weighted model tends to improve (from 0.76 to 0.80), while the accuracy of the majority model decreases (from 0.76 to 0.73).

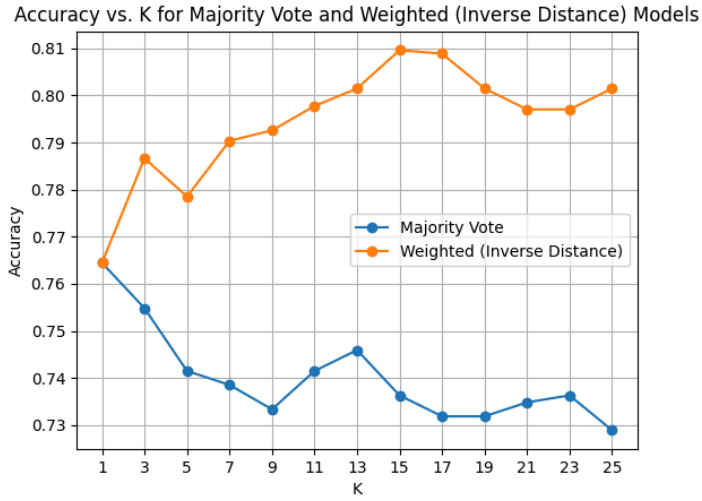

Figure 11: Comparison of weighting strategy for 1-NN across values of K

The inverse distance strategy adjusts the influence of each neighbour on the classification process based on their distance from the new instance. It assigns weights proportional to the inverse of the distance to each neighbour, thus placing more significance on closer neighbours in the process.

As K increases, the weighted model benefits from considering more neighbours, potentially capturing more nuanced patterns in the data. However, this might lead to over-smoothing or increased sensitivity to noise, especially in densely populated regions of the dataset. On the contrary, the majority vote model aggregates predictions from a fixed number of neighbours (K) without considering their distances. Hence, with increasing K, the majority vote model may incorporate conflicting predictions, leading to decreased accuracy.

In summary, the choice of weighting strategy combined with the value of K significantly impacts the performance of our K-NN model. Weighted models tend to excel with larger values of K due to their ability to assign distance-based importance to neighbours. Conversely, the majority vote model may experience heightended noise with larger values of K, negatively affecting its accuracy.

Section 4.4: Distribution of labels (Not Discussed)

5 Limitations

In our analysis, the K-NN model demonstrated commendable performance in predicting wine quality, especially with the integration of various extensions. However, we found limitations to our investigations stemming from the imbalanced distributions of classes within the dataset, introducing ambiguity to our interpretation of results. To mitigate this, future studies could employ cross-validation to vary data subsets, providing more robust insights. Additionally, we observed potential correlations between certain features utilised in wine prediction. To test this, we could look into correlation factors and implement feature selection methods to reduce dimensionality. This reduction in feature space can both increase computational efficiency and also potentially refine the model's performance by focusing on the most influential features.

6 Conclusion

Based on our results and findings, our initial K-NN model achieved a wine quality prediction accuracy of 76.44%. To extend our model, we optimised hyperparameters, resulting in improved performance. Both min-max scaled and standardised models surpassed the basic K-NN model. We further experimented with different distance metrics and weighting strategies, aiming to identify the most effective approach. However, we realised that achieving optimal performance requires a nuanced balance between all hyperparameters. In our correlation matrices and reports, we found that although certain 1-NN models exhibited high f1-scores, we were still unable to assume that the model will be able to 'generalise' well without additional testing. Regardless, this information provided valuable insights into the complexities of implementing a K-NN model, highlighting the importance of thorough parameter tuning and understanding the dataset.

7 References

[1] P. Cortez, A. Cerdeira, F. Almeida, T. Matos and J. Reis. Modeling wine preferences by data mining from physicochemical properties. In Decision Support Systems, Elsevier, 47(4):547-553, 2009. ISSN: 0167-9236.