A logo with text on it

Description automatically generated

**SC 5010 Introduction to Data Analysis**

**2023-24 Semester 2**

**Project Assignment:**

**Analysis of White Wine Quality**

**Instructor:**

**Professor Kwoh Chee Keong & Professor Wei Yang Bryan Lim**

**Group Members:**

|  |  |
| --- | --- |
| Jonathan Lu Yu Kit | U2110030L |
| Stephen Yu Kai Xuan | U2230595J |
| Wong Kai Jie | U2110245B |
| Yehuda Nathan | U2020412D |

Table of Contents

[1. Abstract 1](#_Toc165145323)

[2. Problem Description 1](#_Toc165145324)

[3. Methodology & Algorithms 2](#_Toc165145325)

[3.1 Data Preprocessing 2](#_Toc165145326)

[3.2 Linear Regression 3](#_Toc165145327)

[3.3 K-Means Clustering 4](#_Toc165145328)

[3.4 Decision Tree Classifier & Random Forest Classifier 4](#_Toc165145329)

[3.5 Neural Networks 6](#_Toc165145330)

[4. Implementation 7](#_Toc165145331)

[4.1 Data Preprocessing 7](#_Toc165145332)

[4.2 Linear Regression 9](#_Toc165145333)

[4.3 K-Means Clustering 9](#_Toc165145334)

[4.4.A Decision Tree 10](#_Toc165145335)

[4.4.B Random Forest 12](#_Toc165145336)

[4.5. Neural Networks 13](#_Toc165145337)

[5. Results & Analysis 15](#_Toc165145338)

[5.1 Linear Regression 15](#_Toc165145339)

[5.2 K-Means Clustering 16](#_Toc165145340)

[5.3. Decision Tree & Random Forest Classifiers 16](#_Toc165145341)

[5.4. Neural Networks 17](#_Toc165145342)

[5.5. Comparison Schemes *(shown in figure 5.5)* 17](#_Toc165145343)

[6. Advantages and Limitations 18](#_Toc165145344)

[6.1 Linear Regression 18](#_Toc165145345)

[6.2 K-Means Clustering 18](#_Toc165145346)

[6.3. Decision Tree & Random Forest Classifiers 19](#_Toc165145347)

[6.4. Neural Networks 19](#_Toc165145348)

[6. Conclusions 20](#_Toc165145349)

[6.1 Project achievements 20](#_Toc165145350)

[6.2 Project limitations 20](#_Toc165145351)

[6.3 Directions for improvements 20](#_Toc165145352)

[7. References 21](#_Toc165145353)

[8. Appendix 22](#_Toc165145354)

# 1. Abstract

This report aims to examine the effectiveness of various machine-learning models in analyzing the White Wine dataset, which consists of multiple parameters of wine such as density, pH level, quality, etc. The report begins by outlining the problem statement, which highlights the objective of the report and proceeds with a detailed explanation of each methodology and algorithm applied in the study. Additionally, it covered detailed implementation steps for how each machine-learning model is applied. Furthermore, the report clearly explains each model's result by providing an in-depth comparison between each model and highlighting the pros and cons of each model. Lastly, the report discusses the advantages and limitations of the study and each model, before providing direction for possible future improvements.

# 2. Problem Description

The motivation of this research is to study the application of various machine-learning models on the White Wine dataset and identify which model is the most effective in predicting wine quality based on its other parameters. This understanding will enable producers and retailers to gauge the quality of their products better and make informed business decisions should they be given the parameters.

The problem statement of this study is "Which model is more effective in predicting wine's quality with the given parameters?" To address this, the study will apply Linear Regression, K-Means clustering, Decision Tree Classifier, Random Forest Classifier and Neuron Networks Classifier to analyze and predict the wine's quality from the given parameters and compare the prediction accuracy.

# 3. Methodology & Algorithms

## 3.1 Data Preprocessing

**3.1.1 Z-score standardization**

Z-score standardization is a method that used to normalize the dataset scale by using the normal distribution theory.

The Z-score standardization formula is given:

x : Original value

: Mean of the parameter values

: Standard deviation of the parameter values

z : Standardized value

**3.1.2 Isolation Forest**

Isolation Forest is an advanced algorithm used to identify outliers in a dataset that builds upon the Random Forest model and is particularly suitable for handling datasets with high-dimensional and non-normally distributed parameters, as it does not rely on a normal distribution to identify outliers.

The Isolation Forest formula is given:

m : Number of points

x : Data Point

E(h(x)) : Average path length for isolating data point x in a tree

c(m) : Average depth of data points

is used to compare the average path length of a point in the random forest model with the average path length of all the points with a result range of 0 to 1. The closer the result to 1 indicates that the point is more likely to be abnormal and the higher the chance of being an outlier.

**3.1.3 Synthetic Minority Oversampling Technique (SMOTE)**

SMOTE is an algorithm that is used to resample imbalance dataset by generating similar data from the minority class to provide more balance dataset.

The SMOTE formula is given:

x : Parameter of minority class sample

x neighbor : Parameter of randomly selected neighbor

: Random number between 0 to 1

## 3.2 Linear Regression

Linear regression is a statistical method used for modeling the relationship between a dependent variable (target variable) and one or multiple independent variables (predictors), assuming a linear relationship. The objective is to derive and estimate the coefficients, 𝛽0 and 𝛽1, such that the error between the actual observed values and the values predicted by the model is minimized.

The formula for a simple regression with one independent variable is given:

Y : Dependent variable

X : Independent variable

𝛽0 : Intercept (bias)

𝛽1 : Coefficient (slope)

: Error term

And the estimating of coefficients in simple linear regression is by using Ordinary Least Squares method with the following formula:

and

𝛽0 : Estimated intercept

𝛽1 : Estimated coefficient for independent variable

: Mean of the independent variables

: Mean of the dependent variables

## 3.3 K-Means Clustering

K-Means clustering is a partitional clustering method that divides a dataset into non-overlapping clusters, and each data point is in exactly one cluster. The model begins by randomly selecting k number of points from the dataset to serve as initial centroids and grouping the data points similar to the initial centroids into one cluster. After forming the clusters, the center point of each cluster is recalculated and serves as the new centroids. The data points are then regrouped again based on similarity to the new centroid. This process is repeated until the centroid position is stabilized or a stop condition is met. Hence, the number of cluster forms depends on the number of initial centroids selected. 

The objective function, J for K-Means Clustering is given:

n : Number of data points

k : Number of clusters

xi : Vector representing the i-th data point

cj : Vector representing the centroid of the j-th cluster

wij : Binary variable (1 if data point i assigned to cluster j, otherwise 0)

: Euclidean distance between data point i and centroid j

The objective function, J assigns a data point to the cluster whose centroid is closest to the data point based on the Euclidean distance.

## 3.4 Decision Tree Classifier & Random Forest Classifier

**3.4.1 Gini Index & Entropy**

Both are the splitting criteria used in constructing the Decision Tree and Random Forest model by determining the best parameter to split the data at each step of the tree-building process.

**The Gini Index formula is given:**

D : Dataset for a node

n : Number of classes

: probability of being classified for a distinct class.

The Gini index is the measure of the probability of misclassification for a random instance when chosen randomly. The lower the Gini index, the lower the likelihood of misclassification.

**The Entropy formula is given:**

D : Dataset for a node

n : Number of classes

: probability of being classified for a distinct class

Entropy is the measure of disorder or impurity in a node. Thus, a node with a more variable composition would be considered higher Entropy than one with a less variable composition. Leaf nodes with all instances belonging to one class would have an entropy of zero, and nodes where all classes are divided equally would have an entropy of one. The lower the entropy, the better.

**3.4.2 Decision Tree Classifier**

The Decision Tree model is constructed via a recursive partitioning of data based on a series of conditions. The model splits the dataset into groups at each node stage based on the chosen condition. The splitting criteria mentioned previously, Gini Index and Entropy, are used to determine the best way to split according to the condition selected, and the process will continue until a stop condition is met, such as a node has homogenous classes or reached maximum depth set.

**3.4.3 Random Forest**

The Random Forest model enhanced the Decision Tree algorithm by creating multitude of uncorrelated decision tree and deriving the outcome based the majority vote from all the trees. Random Forest improved accuracy and reduce overfitting by introducing randomness in constructing each tree to ensure that each tree is uncorrelated.

## 3.5 Neural Networks

Neural networks are composed of interconnected nodes or neurons organized into layers. The main objective is to find the best set of weights to map input parameters to output predictions through repetition of forward propagation and backpropagation.

**The forward propagation is given:**

: Weight matrix of the layer

: Activated output of the layer

: Bias vector of the layer

: Linear output of the layer

: Activation function of the layer

**The backward propagation is given:**

where , , , are the gradient of cost function with respect to activated output, linear output, weights, and biases of the layer respectively. ’ is the derivative of the activation function.

# 4. Implementation

## 4.1 Data Preprocessing

**4.1.1 Visualization**

We plotted a scatter plot (*figure 4.1.1.1)* to visualize the relationship between each parameter and wine quality and a correlation metric heatmap *(figure 4.1.1.2)* to understand the relationship between each parameter.

From the scatter plots, we noticed that most of the parameters are equally distributed with the wine quality, indicating no linear relationship between the parameter and wine quality. Hence, models that build on linear relationships, such as linear regression, might not be effective for our dataset.

Besides, from the Correlation Metrix Heatmap, we noticed that most of the parameters are not independent. For instance, alcohol with density has a significant negative 0.78 correlation rate, and density with residual sugar has a significant positive 0.84 correlation rate. Hence, models that build on the assumption of independent parameters, such as Bayesian Classifiers, might be ineffective for our dataset.

**4.1.2 Missing and duplicated data**

Our dataset does not have any missing data but contains 937 rows of duplicated data, which have been removed to increase the accuracy of our analysis.

**4.1.3 Normalization**

To prevent parameters that disproportionally influence the analysis result due to the parameter scales, we have normalized our dataset with the Z-score standardization method.

**4.1.4 Removing outliers**

As shown in the histogram *(figure 4.1.4.1)* and Q-Q plots *(figure 4.1.4.2)*, most parameters did not follow the normal distribution. For instance, residual sugar is mostly distributed toward 0. Therefore, we believe that using normal distribution as the basis for removing outliers from our dataset will not be effective. Hence, we have decided to use the Isolation Forest method to identify and remove the outlier as it builds upon a Random Forest instead of a normal distribution. Due to the high dimensional of our dataset with 11 parameters, Isolation Forest is again preferred. As a result, we have identified and removed 275 rows of data from our dataset.

**4.1.5 Data resampling**

By visualizing the amount of data for each wine quality as shown in figure 4.1.5.1, we noticed a significant imbalance in our dataset, which could negatively impact our analysis due to bias toward the majority class. Hence, we have identified the minority class (wine quality of 3, 4, 8, and 9) and adopted the SMOTE technique to increase the minority class by 4 times to improve the balancing of our dataset. The resampled dataset was visualized in figure 4.1.5.2.

**4.1.6 Data reclassifying­­**

However, after the data resampling, our dataset remains significantly imbalanced, and it is inadvisable to continue increasing minority group with the SMOTE technique, as it could negative impact the accuracy of our analysis result. Hence, we decided to reclassify the wine quality into three groups (Low: quality of 3, 4 & 5; Medium: quality of 6; High: quality of 7, 8 & 9) as shown in figure 4.1.6.

## 4.2 Linear Regression

Linear Regression is finding a best-fit line with one or multiple coefficients. It is done using the “LinearRegression” module under “sklearn.linear\_model”, which utilizes the Ordinary Least Squares (OLS) method: estimate the coefficient of independent variable such that it forms a best-fit line, minimizing the sum of squared errors between the predicted values and actual values.

To start off, we first have to split the feature (Wine quality) from the target variables, followed by splitting the data entries into training and testing sets. Predictions can then be easily made on the testing set after training the model using the LinearRegression module. Lastly, we will be using R-squared and Mean Squared Error (MSE) from sklearn.metrics to evaluate the results.

Improvements like standard scaler and ridge regression have also been tried to improve the predictions.

A standard scaler helps to standardize input data to ensure that data points have a balanced scale by removing the mean and scaling to unit variance.

Ridge Regression helps regularize linear regression models by reducing errors caused by overfitting on training data, especially for multicollinearity variables, where two or more independent variables are highly correlated.

## 4.3 K-Means Clustering

Before applying the K-Means model, we would like to identify which parameters are most effective for the model. We used a combination module imported from “itertools” to generate all the possible combinations of the parameters and run the K-Means model with all the combinations identified. It is done using the “KMeans” module under “sklearn.cluster”. The value of ‘k’ is set to 3 to ensure we can split the dataset into three clusters as we have reclassified the wine quality into 3 groups. We computed the overall purity for each result and compared it to obtain the combination that could successfully separate the data into three groups according to the wine quality and highest purity. As a result, the combination of “alcohol” and “chloride“ produced the output with the highest purity, as shown in Figure 4.3.1.

Additionally, to address the limitation of K-Means, which tend to form a globular shape, we have increased the value of k to 6. This adjustment results in a larger number of smaller clusters forming, potentially reducing the tendency towards globularity, with a post-processing adjustment to combine the clusters according to their average quality.

Finally, we plotted a PCA graph to visualize the clustering result of the K-Means model and computed the mean square error (MSE), the averaged distance between centroids, overall purity (accuracy), and silhouette score for evaluation. The mean square error is used to assess the similarity within the cluster, while the average distance between centroids is used to evaluate the dissimilarity between each cluster. An effective clustering result will exhibit high intra-cluster and low inter-cluster similarities. Additionally, the silhouette score is a metric considering both intra-cluster and inter-cluster similarities.

## 4.4.A Decision Tree

The decision tree algorithm is first allowed to train to its fullest potential. This is done by choosing not to use any parameters in the DecisionTreeClassifier and intentionally allowing the algorithm to overfit. Therefore, by comparing the accuracy of each model, Gini and Entropy, we can know which criterion to use that has the best predictive ability. In our case, because Gini has an accuracy of 60.13% while Entropy has 53.90%, we have opted to use Gini as a criterion for the DecisionTreeClassifier. To control for the randomness in the generation of decision trees, we have set a random\_state = 0 in the algorithm to act as an initial seed.

Next, we rank the feature importance of the wine quality dataset through the use of feature\_importances\_ method built into scikit-learn. feature\_importances\_ ranks the importance of each feature in predicting wine quality based on the predictive capabilities of the overfitted model.

As shown in *Figure 4.4.A.1*, all features have some importance, with alcohol, free sulfur dioxide, and volatile acidity being the most important features. However, as the dataset is not very large (n = 10,000), removing features based on their relative importance will result in the decision tree not being well-optimized as the interaction effect becomes over-generalized. As such, we decided to leave the features untouched.

As shown in *Figure 4.4.A.2*, that the decision tree is overfitted. Comparing accuracy scores between training features and training predictions shows that a 100% accuracy, further validating this fact. In order to correct the issue of overfitting, the use of cost-complexity pruning in the post-pruning process can help reduce the training accuracy while reducing testing accuracy to a minimum.

Cost-complexity pruning uses the complexity parameter , where is the cost-complexity measure, and is the number of leaf nodes. In general, the best is one that minimizes the cost-complexity measure . Hence, the optimal pruning aggressiveness occurs at the minimization problem of . As can be seen from *Figure 4.4.A.3*, the total impurity of leaves decreases as the effective alpha increases.

The effect of increasing the alpha can be seen in *Figure 4.4.A.4*. As the pruning becomes more aggressive, the training accuracy falls and eventually converges with the testing accuracy. Further pruning past this path of convergence is not optimal because it merely decreases the accuracy of our predictions without providing any significant benefit to our overfitting issue. The pruned decision tree based on a cost-complexity alpha of 0.03 is shown in *Figure 4.4.A.5.*

## 4.4.B Random Forest

Next, the random forest algorithm is used to improve upon decision trees. As mentioned in the previous section, we have opted not to remove features manually due to concerns that interaction effects may be under-represented as a consequence of low sample size. Random forest adequately addresses this issue, as the bagging principle allows the algorithm to randomly select features to create the best outcome for each individual decision tree, before going through a voting process to make a final prediction. This addresses the curse of dimensionality without adversely impacting our predictions.

Similar to the case of decision trees, we first rank the feature importance to understand the predictions from a random forest. As shown in *figure 4.4.B.1*, Differences were expected, such as how density ranks third in random forest, but ranks last in decision tree.

The use of RandomSearchCV was opted in deciding the hyperparameters of random forests, due to its cost-effective and better generalization nature compared to GridSearchCV.

The final tree based on the optimized hyperparameters is shown in *Figure 4.4.B.2*. As the random forest technique uses a combination of decision trees, multiple trees can be visualized by correcting the index in rf\_best.estimators[0]. No pruning needs to be done because the decision trees in every estimate has been automatically pruned by the RandomForestClassifier() algorithm, and therefore there is no need to correct for overfitting.

## 4.5. Neural Networks

The dataset, final\_wine, is copied and renamed to nn\_wine to be used for neural network to prevent overwriting the original dataset. It then takes the last column as the labels and the rest as the parameters. Then it is split to 10% test and 90% train dataset using train\_test\_split with random\_state to ensure deterministic split for every run. The labels are converted to one-hot encoding using tf.keras.utils.to\_categorical. RandomSearch is used for hyperparameter search by using kerastuner.tuners.RandomSearch.

The hypermodel build function, build\_model(), takes an argument keras\_tuner.HyperParameters() to variably define the hyperparameters of the model architecture. Inside the build\_model(hp) function, tensorflow.keras.Sequential() is used to construct linear stack of layers, tensorflow.keras.layers is used to construct the Dense, BatchNormalization, and activation layers. Xavier initialization and zero initialization are used to initialize the weights and biases in each layer and are implemented using tensorflow.keras.initializers and passed in as the argument value of kernel\_initializer and bias\_initializer of layers.Dense inside the hypermodel build function, build\_function(hp).

During compilation of keras.Sequential(), Adam optimizer is used as the optimizer, Categorical crossentropy loss is used as the loss function, “accuracy” and “mse” are used as the metrics to monitor the performance. Hyperparameters that are defined as variables using keras\_tuner.HyperParameters() includes the number of hidden layers (1 to 3 inclusive), number of neurons in each layers (32 to 512 inclusive with 32 increments), activation function (ReLU, sigmoid, and tanh), learning rate (1e-5 to 1e-2 inclusive and sampled from a logarithmic space to ensure small values are sampled equally as large values). 20 different combinations of hyperparameters (max\_trials) are used with 2 training runs for each combination (executions\_per\_trial), and 20 training iterations (epochs) is done for each training runs.

The hyperparameter search is done on 20% of the training dataset (validation\_split=0.2) and its objective is to minimize the validation loss. Custom callback (WeightMonitorCallback) is implemented to monitor any issues for the weights and biases, e.g. vanishing/exploding gradients that can lead weights to all zero or nan. The best set of hyperparameters are trained again on the training set and validated on 20% of the training set with epoch from (1, 500). The best epoch is then chosen that minimize the validation loss. This epoch (14) is then used to evaluate the model on the test dataset. Summary of the model is as follows: 3 hidden layers, 192-64-480 neurons, ReLU-ReLU-tanh activation. SoftMax activation is used at the output layer. Learning rate = 4.65e-4.

# 5. Results & Analysis

## 5.1 Linear Regression

As shown in *figure 5.1.1*, Linear Regression model has an x-intercept of -0.1238 with the coefficient value according to the above graph. From the graph, we observe that:

* ‘Residual sugar’ exhibits strongest positive relationship
* ‘Density’ shows strongest negative relationship

From all three results *(figure 5.1.2)*, we saw that there is consistent performance on all datasets. (considering similarities in MSE and R²) From here, we conclude that the linear regression model have a low variance and bias. There is also no signs of overfitting on the testing and overall dataset, further proving that the model is well generalized to unseen data.

Despite that, we found the MSE and R² value far from ideal, showing that there in fact, have no goodness of fit for our linear regression. Therefore, we perform additional steps to try improving the model, namely Standardization using standardscaler and regularization using ridge regression with system derived hyperparameter.

As shown in *figure 5.1.3,* there is only an insignificant improve in the model, from 0.374 to 0.3727 for MSE and 39.13% to 39.49% for R² value.

As shown in *figure 5.1.2,* after the improvement, outliers have a lighter shade of blue. This is due to standardization making the anomalies less significant in the model. Overall, Linear Regression is still a subpar model in predicting wine quality because of its inability to capture complex pattern of data.

## 5.2 K-Means Clustering

The comparison of the K-Means model results for k = 3 and k = 6 *(figure 5.2.1 & 5.2.2)* shows a slight improvement in the clustering, as evidenced by the decrease in mean square errors from 0.3907 to 0.2009, and the average distance between centroids had increased from 1.6791 to 0.4326. This indicates that increasing the value of k improves the model’s performance as the clusters are more compact and better separation. However, excessively increasing the value of k to achieve better results could potentially lead to overfitting.

Besides, the overall purity has also increased from 53.75% to 56.26%. However, a purity of 56.26% is still considered relatively low, suggesting that K-Means might not be effective for our dataset. One of the reasons could be the selection of only “alcohol” and “chloride” as parameters for the model. This low-dimensional dataset might not provide sufficient information for K-Means clustering to draw insightful results.

Additionally, as mentioned previously, we have attempted to use a higher dimensional dataset by trying all the possible combinations of parameters, resulting in a worse outcome. One of the possible reasons is that in higher dimensional, our dataset tends to exhibit non-linear clustering shapes, which K-Means is unable to capture. Advanced clustering models such as SNN Clustering and DBSCAN might be preferred.

## 5.3. Decision Tree & Random Forest Classifiers

The accuracy of decision trees and random forest were 60.13% and 62.36% respectively, showcasing the advantage of random forest algorithms over decision trees. These are further validated by higher cross-validation scores (0.4963 versus 0.5332), F1 scores (0.6006 versus 0.6189), and MSE (0.5122 versus 0.4432). While the accuracy substantially improves in decision trees and random forest algorithms compared to the previously mentioned two, the classifications between “low”, “medium”, and “high” may have caused the interaction effects to become more pronounced. Due to this, both classifiers encounter difficulties in predicting medium and high qualities. The confusion matrix can be seen from Figures 5.3.1 and 5.3.2 in the appendix.

## 5.4. Neural Networks

The result obtained are as follows: Accuracy = 69.04%, MSE = 0.1465. The suggestion from RandomSearch to use 3 hidden layers indicate that there are complex relationships between the parameters. This might also suggest that there might be dependency between the input parameters. It can also be seen from the confusion matrix in *figure 5.4.1* that the model have difficulty in differentiating “Low” and “High” quality. Prior to using categorical cross entropy loss as the loss function, MSE is used and when validated on validation dataset, the accuracy is somewhat similar, but the MSE obtained is always twice of that when using categorical cross entropy loss. Hence, categorical cross entropy loss is used as the loss function both for hyperparameter search and for the final model. This results in better accuracy, however, the False Positives on the further classification, e.g. “Medium” and “High” are equally penalized when the truth label is “Low.”

## 5.5. Comparison Schemes *(shown in figure 5.5)*

Lowest predictive accuracy: Linear Regression (39.49% accuracy with 0.3727 MSE)

Highest predictive accuracy: Neural Network (69.04% accuracy with 0.1465 MSE)

# 6. Advantages and Limitations

## 6.1 Linear Regression

|  |  |
| --- | --- |
| **Advantages** | **Limitations** |
| **High Interpretability**  Coefficients associated with variable indicate the strength and direction of their relationship with the target variable. | **Assumption of Linearity**  Assumes relationship between predictor variables and the target variable is linear. If the true relationship is nonlinear, linear regression may provide biased or inaccurate results. |
| **Simplicity & Computational Efficiency**  Linear Regression is capable of handling datasets with relatively low computational resources and complexity | **Limited Flexibility**  May not perform well when relationship is highly non-linear or when interactions between variables need to be modelled |

## 6.2 K-Means Clustering

|  |  |
| --- | --- |
| **Advantages** | **Limitations** |
| **Simplicity and Speed**  K-Means are simple and direct to implement, which make it fast to run and suitable for large data | **Sensitivity to Initial Conditions**  The initial centroids have direct impact to final outcome, as a different centroid result in different clusters. |
| **Easy to Interpret**  K-Means’ result is easy to understand as it is imaginable in low dimensional, which make it a good start for data analysis | **Not Suitable for Non-linear Clustering**  K-Means do not work well with complex geometrical shaped as it has bias toward globular dataset |

## 6.3. Decision Tree & Random Forest Classifiers

|  |  |
| --- | --- |
| **Advantages** | **Limitations** |
| **Simple interpretation**  Decision trees are simple to interpret due to their Boolean logic. This also allows splits to be drafted in an intuitive manner | **Biased learners**  Decision trees can create biased trees if learning on imbalance data. This causes the predictions to become inaccurate. |
| **Low costs of implementation**  Decision trees are generally efficient, provided that they are given the correct parameters for implementation. | **Issues of nonlinearity, dimensionality, and interactive effects**  Decision trees struggle with nonlinear classifications and high dimensionality. Additionally, we cannot discount “unimportant” features due to the presence of interaction in our wine datasets. |

### 

## 6.4. Neural Networks

|  |  |
| --- | --- |
| **Advantages** | **Limitations** |
| **Complex pattern recognition**  Able to learn complex relationships in data | **Overfitting**  When there are only small datasets, the model is prone to overfitting |
| **Ease of implementation**  A lot of modern frameworks based on the latest research papers are available for free use | **Highly dependent on initialization and hyperparameters**  Require careful hyperparameter tuning specific to the problem |

# 6. Conclusions

## 6.1 Project achievements

The report effectively depicted the employment of all five models: Linear Regression, K-Means clustering, Decision Tree Classifier, Random Forest Classifier and Neuron Networks Classifier to the White Wine dataset, facilitating the assessment of predictive metrics like MSE and accuracy for each model. The conclusion drawn was that Neural Network is the most effective model, demonstrating the highest accuracy and lowest MSE. The study also yielded valuable insights, such as the characteristics of our dataset and a comprehensive exploration of different predictive modeling techniques, with an evaluation of which model is most suitable for application to this type of data structure.

## 6.2 Project limitations

In this project, wine quality ratings 3, 4, and 5 are grouped as "Low," and ratings 7, 8, and 9 are grouped as "High." Because of this, some relationships/patterns between the parameters might be masked over, which will reduce the separation ability of the models. The choice of grouping is also subjective and may not align with domain knowledge.

## 6.3 Directions for improvements

Possible improvements include using a more advanced preprocessing technique to minimize dependency between input variables and using more advanced Machine Learning/Deep Learning technique to improve the model's ability to distinguish between different quality levels. We can also leverage on wine expert’s knowledge for selection of relevant variables and defining more meaningful categorizations for wine quality ratings. Instead of relying solely on the existing dataset, we can align with industry standards and establish criteria for quality assessment that extend beyond numerical ratings.

# 7. References

Gunay, D. (2023, September 14). Random Forest - Deniz Gunay - Medium. *Medium*. https://medium.com/@denizgunay/random-forest-af5bde5d7e1e#:~:text=Random%20Forest%20is%20an%20ensemble%20machine%20learning%20algorithm%20that%20combines,overfitting%20and%20improve%20predictive%20performance.

Koli, S. (2023, February 28). Decision Trees: A complete introduction with examples. *Medium*. https://medium.com/@MrBam44/decision-trees-91f61a42c724

Lim, Y. (2022, March 22). Unsupervised Outlier Detection with Isolation Forest. *Medium*. https://medium.com/@limyenwee\_19946/unsupervised-outlier-detection-with-isolation-forest-eab398c593b2

Maklin, C. (2022, May 21). Synthetic Minority Over-sampling TEChnique (SMOTE) - Cory Maklin - Medium. *Medium*. https://medium.com/@corymaklin/synthetic-minority-over-sampling-technique-smote-7d419696b88c

Saleem, S. (2023, May 15). Neural networks in 10mins. Simply explained! - Sadaf Saleem - medium. *Medium*. https://medium.com/@sadafsaleem5815/neural-networks-in-10mins-simply-explained-9ec2ad9ea815

Thavanani, S. (2021, December 14). K-Means Clustering: Optimizing cost function mathematically. *Medium*. https://medium.com/analytics-vidhya/k-means-clustering-optimizing-cost-function-mathematically-1ccae156299f

Tyagi, N. (2021, December 13). Understanding the Gini index and information gain in decision trees. *Medium*. https://medium.com/analytics-steps/understanding-the-gini-index-and-information-gain-in-decision-trees-ab4720518ba8

# 8. Appendix

A group of blue and white lines

Description automatically generated

***Figure 4.1.1.1 Scatter plot***

A chart with red and blue squares

Description automatically generated

***Figure 4.1.1.2 Heatmap***

A group of blue and white graphs

Description automatically generated

***Figure 4.1.4.1 Histogram***

A chart of different types of substances

Description automatically generated with medium confidence

***Figure 4.1.4.2 Q-Q plots***

A bar chart of quality and quality

Description automatically generated

***Figure 4.1.5.1 Before resampling***

A bar chart of wine quality after smote

Description automatically generated

***Figure 4.1.5.2 After resampling***

A bar chart of wine quality after reclossification

Description automatically generated

***Figure 4.1.6 After reclassifying***

A screenshot of a number

Description automatically generated

***Figure 4.3.1 Result of the parameter combination***

A graph with blue bars

Description automatically generated

***Figure 4.4.A.1 Feature Importance of Wine Quality (Decision Tree)***



***Figure 4.4.A.2 Unpruned Decision Tree***

A graph of an average alpha

Description automatically generated

***Figure 4.4.A.3 Effective Alpha for Cost-Complexity Pruning***

A graph of a graph

Description automatically generated

***Figure 4.4.A.4* *Relationship Between Alpha and Accuracy***

A diagram of a chemical reaction

Description automatically generated

***Figure 4.4.A.5 Pruned Decision Tree***

A graph with blue bars

Description automatically generated

***Figure 4.4.B.1 Feature Importance of Wine Quality (Random Forest)***

A diagram of a person

Description automatically generated with medium confidence

***Figure 4.4.B.2 Random Forest***

A graph with red and green squares

Description automatically generated

***Figure 5.1.1 Coefficient of Linear Regression Model***

A screenshot of a computer error

Description automatically generatedA white background with black text

Description automatically generated

***Figure 5.1.2 Evaluation on training, testing, and overall dataset***

A white background with black text

Description automatically generated

***Figure 5.1.3 Evaluation Linear Regression after standardization and regularization***

*A graph of a function

Description automatically generated with medium confidence*

***Figure 5.1.2 Effects of Standardization & Regularization***

A chart with colorful dots

Description automatically generated with medium confidence

***Figure 5.2.1 Result of K-Means model when k = 3***

A chart with colorful dots

Description automatically generated

***Figure 5.2.2 Result of K-Means model when k = 6***

*A screenshot of a graph

Description automatically generated*

***Figure 5.3.1 Final result of Decision Tree Classifier***

*A screenshot of a graph

Description automatically generated*

***Figure 5.3.2 Final result of Random Forest Classifier***

*A chart of different colors

Description automatically generated*

*A number of numbers and symbols

Description automatically generated with medium confidence*

***Figure 5.4.1 Final result of Neural Network***

A graph of different colored bars

Description automatically generated

***Figure 5.5 Comparison of the result of the model***

**Link to view the code in NBviewer:**

<https://nbviewer.org/github/JONA0031/SC5010/blob/main/Source%20Code.ipynb>

(The actual source code file is included in the zip folder)