Report

1. Table of Contents
2. Table of Tables
3. Table of Figures
4. Problem Statement
5. Literature Review
6. Methodology
7. Results and Discussion
8. References

# Problem Statement

This project focuses on creating classification machine learning models to predict fruit quality (good or bad) from a dataset which has seven features: size, weight, sweetness, crunchiness, juiciness, ripeness, and acidity. Each attribute can be described as:

Features

* Size: Size of the fruit
* Weight: Weight of the fruit
* Sweetness: Degree of sweetness of the fruit
* Crunchiness: Texture indicating the crunchiness of the fruit
* Juiciness: Level of juiciness of the fruit
* Ripeness: Stage of ripeness of the fruit
* Acidity: Acidity level of the fruit

Target Variable: Quality – 1 if the fruit quality is good, 0 if the fruit quality is bad.

After getting model which can accurately determine the quality of given fruit, we can potentially utilize the model for these benefits:

* Improving fruit grading
* Reducing food waste
* Enhancing customer experience

# Literature Review

**2. Libraries:**

* Pandas and NumPy: These fundamental libraries form the foundation for data manipulation, analysis, and numerical computations.
* TensorFlow and scikit-learn (sklearn): These libraries provide powerful tools for building and training machine learning models. TensorFlow is particularly adept at deep learning algorithms, while scikit-learn offers a comprehensive suite of traditional machine learning algorithms like Logistic Regression and SVMs.
* Matplotlib and Seaborn: These libraries allow for the creation of informative and visually appealing data visualizations, aiding in data exploration and model evaluation.
* Optuna: This library assists in efficient hyperparameter tuning, a crucial step in optimizing the performance of machine learning models.
* Machine Learning
* Classification Problem
* Logistics Regression
* SVM
* Neural Network
* Elbow method
* StandardScaler
* K-fold cross validation
* Data imbalance problem
* Correlation matrix

# 3. Methodology

## 3.1 Exploratory Data Analysis (EDA)

This project will follow a methodology to develop and evaluate machine learning models for fruit quality prediction. The EDA stage:

1. Data Import and Loading: The process begins by importing the provided fruit quality dataset. Pandas library will be used to read the data into a DataFrame.
2. Data Inspection: The DataFrame will be examined to understand its dimensions (number of rows and columns). Then, the data types of each column will be verified to ensure proper handling during analysis.

A screenshot of a computer code

Description automatically generated

Figure 1. Data dimension

A screenshot of a computer

Description automatically generated

Figure 2. Data type of each column

1. Descriptive Statistics: Descriptive statistics will be computed to summarize the central tendency (mean, median, Q1, Q2, Q3) and standard deviation of all features.

A screenshot of a computer

Description automatically generated

Figure 3. Descriptive Statistics of dataset

1. Missing Value Analysis: The presence of missing values (NA values) will be investigated. Depending on the extent and distribution of missing data, appropriate imputation techniques may be applied to address them.

A screenshot of a computer

Description automatically generated

Figure 4. Number of missing data in each column

1. Target Variable Analysis: The value counts for the target variable (fruit quality) will be calculated. This analysis provides insights into the class imbalance (ratio of good vs. bad quality fruits) which can influence the choice of classification models and evaluation metrics.

A graph of a fruit quality

Description automatically generated

Figure 5. Each fruit quality count

Figure 6 shows that from all the data collected, 2004 was good quality and 1996 was bad. It is safe to say that this data will not face imbalance classification problem.

1. Data Visualization: Data visualization plays a crucial role in understanding the relationships between features. Histograms will be employed to visualize the distribution of each numerical feature, revealing potential skewness or outliers. Boxplots will be used to compare the distribution of features across the two quality classes (good and bad).

A blue and black graph

Description automatically generated with medium confidenceA blue and black graph

Description automatically generated with medium confidence

Figure 6. Histogram and boxplot of features

After the step of EDA, it turns out that from the histogram, all features (except crunchiness) lie around practically the same range of value. Which means the data might be cleansed and normalized already. And from box plot, for some features of bad quality fruit have smaller range compared to the good quality one. But the census of it is not that much different.

1. Correlation Analysis: Correlation Matrix is used to evaluate the correlation between each feature. This matrix helps identify the correlation and detect the correlation problem which will cause the issues when creating model.

A screenshot of a graph

Description automatically generated

Figure 7. Correlation Matrix of Features

From figure 7, it shows that there is at most weak correlation between features, if not no correlation at all.

## 3.2 Creating machine learning model

In this project, aiming to develop machine learning models to predict the quality of fruit (good or bad) based on a dataset containing various measurable attributes. While traditional methods rely on visual inspection, this project explores a more objective and automated approach.

The choice of models incorporates Logistic Regression, Support Vector Machines (SVM), and Neural Networks. Logistic Regression provides a solid foundation for binary classification tasks like this one. SVM is adept at handling non-linear relationships between features, which might exist in the fruit quality data. Finally, Neural Networks offer exceptional flexibility in modeling complex relationships, potentially capturing intricate patterns within the data that simpler models might miss. By evaluating these diverse models, we can determine the most effective approach for accurate fruit quality prediction.

Accuracy is the initial choice for evaluating the Logistic Regression model's performance. This metric simply reflects the proportion of fruit quality predictions (good or bad) that align with the actual labels. And since the dataset contains roughly equal numbers of good and bad quality fruit samples, accuracy directly reflects the model's ability to correctly classify both categories. In balanced data, a high accuracy score indicates the model performs well in identifying both good and bad quality fruits.

### 3.2.1 Logistic Regression

The baseline Logistic Regression model employs a baseline configuration where all features are included with their default hyperparameter values. This serves as a starting point for evaluating model performance.

1st model - Baseline Model:

* All features are included in the model.
* Default values are used for all hyperparameters. This serves as a starting point for further optimization.

2nd model - Grid Search for Initial Hyperparameters:

A grid search is performed to find a good initial set of hyperparameters for the logistic regression model. The grid search explores different values for the following hyperparameters:

* 'C': This is the regularization parameter that controls the trade-off between model complexity and fitting the training data. The grid search considers a range of C values (0.001, 0.01, 0.1, 1, 10, 100)
* 'solver': This specifies the algorithm used to solve the logistic regression problem. We consider both 'lbfgs' and 'liblinear' to be evaluated in the grid search.

Polynomial Feature Selection: A loop iterates through polynomial degrees from 1 to 4.

Inside the loop:

* The 'liblinear' solver is used for logistic regression with polynomial features of the current degree.
* The mean squared error (MSE) is calculated on a 5-fold cross-validation.
* The calculated MSE is stored in a list for later analysis.

After the loop is completed, a plot is created to visualize the relationship between polynomial degree and the average MSE across cross-validation folds. This helps identify the polynomial degree that leads to the lowest average error.

3rd model - Final Model Tuning is based on the results from step 3, the optimal polynomial degree is chosen. A new grid search is performed using the chosen polynomial degree and the 'liblinear' solver (since it only has L2 regularization).

* 'C': The regularization parameter with a wider range from 0.001 to 10000 for more precise tuning.
* 'max\_iter': The maximum number of iterations for the solver, exploring values of 100, 500, and 1000 to ensure convergence.
* The accuracy is calculated using 5-fold cross-validation for each hyperparameter combination.

The hyperparameter combination that leads to the highest average accuracy is selected as the final model.

### 3.2.2 Support Vector Machine (SVM)

This script outlines the process of using SVM with grid search for hyperparameter tuning and exploring polynomial features.

Data Preprocessing: Standardize the data using a standard scaler. This ensures all features are on a similar scale, improving the performance of some kernel functions.

1st model - Baseline model: SVM model which hyperparameter is not tuned. All the hyperparameter are default value.

2nd model - Initial Hyperparameter Tuning: A grid search is performed to find a good starting point for the SVM hyperparameters. The grid search explores different values for the following hyperparameters:

* 'C': The regularization parameter that controls the complexity of the model. We try values of 0.1, 1, and 10 to cover a range of regularization strengths.
* 'kernel': This specifies the type of kernel function used by the SVM. We explore four options: 'linear', 'rbf', 'poly', and 'sigmoid'. Depending on the chosen kernel, additional hyperparameters might be relevant.
* 'gamma': This hyperparameter controls the influence of training points in the RBF kernel. We try values of 0.01, 0.1, and 1 to explore different levels of influence.

The accuracy of each hyperparameter combination is evaluated using 5-fold cross-validation. The combination with the highest average accuracy across folds is selected as the initial model.

Polynomial Feature Exploration: A loop iterates through polynomial degrees from 1 to 4.

Inside the loop:

* The SVM model is trained with a polynomial kernel of the current degree.
* The accuracy is calculated using 5-fold cross-validation.
* The average accuracy across folds is stored for later analysis.

After the loop is completed, a plot is created to visualize the relationship between polynomial degree and the average accuracy. This helps identify the polynomial degree that leads to the highest average accuracy.

3rd model - Final Model Tuning:

Based on the results from step 3, the optimal polynomial degree (if applicable) is chosen.

A new grid search is performed using the chosen kernel function (including polynomial if applicable) and the following hyperparameters:

* ‘C’: The regularization parameter with the same range as before (0.1, 1, 10).
* ‘kernel': The chosen kernel function from step 2 or 3.
* 'gamma': The hyperparameter for RBF kernel with the same range as before (0.01, 0.1, 1).

The accuracy is calculated using 5-fold cross-validation for each hyperparameter combination. The hyperparameter combination that leads to the highest average accuracy is selected as the final SVM model.

### 3.2.3 Neural networks

This script outlines the process of using neural network with Optuna for hyperparameter tuning.

1st model - Baseline Model:

* A simple neural network with one hidden layer is created as a baseline.
* The model is trained with 50 epochs and a batch size of 32 to get a preliminary understanding of the achievable accuracy.

2ndmodel - Hyperparameter Optimization with Optuna:

* Optuna is used to optimize the hyperparameters of the neural network for better performance. The following hyperparameters are tuned within specified ranges:
  + 'units': The number of neurons in the hidden layer. This is searched as integers from 16 to 192 with a step size of 16.
  + 'dropout\_rate': The dropout rate for regularization. This is explored as floats between 0.0 and 0.5 with a step size of 0.1.
  + 'learning\_rate': The learning rate for the optimizer. Currently, it is in range of 1e-2, 1e-3, and 1e-4.
  + 'num\_layers': The number of hidden layers in the network. This is searched as integers from 1 to 5.
* Noted that the number of hidden layers, units, number of layers are keeping low to prevent overfitting due to its nature of this model is prone to be overfit.

3rd model - Evaluation of Optimized Model:

* After Optuna finds the best hyperparameter configuration, the neural network is retrained with those parameters.
* The accuracy of the optimized model is evaluated using accuracy to assess its performance.

## 3.3 Metric assessment on testing dataset

Following the evaluation of individual models (Logistic Regression, SVM, and Neural Network), their performance will be compared using accuracy as the primary metric. Accuracy reflects the overall proportion of correctly predicted fruit quality labels (good or bad) on the held-out testing set. The model that achieves the highest accuracy on the testing data will be deemed the most effective for fruit quality prediction in this context.

This approach prioritizes overall classification accuracy, ensuring the chosen model excels at correctly identifying both good and bad quality fruits. By evaluating on the unseen testing set, we obtain a more reliable assessment of the model's generalizability to real-world scenarios beyond the training data. The model with the strongest performance on this final test becomes our champion, ready for potential deployment in a fruit quality assessment system.

# 4. Results and Discussion

## 4.1 Results from model

### 4.1.1 Logistic Regression result

To start, baseline model starts with all default value in term of hyperparameters, so it can be determined the expected performance of this model. Random state is set to 42 to make the result consistent. Also, the accuracy of the models developed is calculated in terms of average accuracy from K-fold cross validation on the data (K=5).

Table 1. Results from Logistics Regression Model

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Model | Solver | Penalty | C  (penalty strength) | Max iteration | Degree of polynomial | Model accuracy |
| 1st Baseline model (default) | lbfgs | l2 | 1.0 | 100 | 1 | 0.7262 |
| 2nd Grid search for hyperparameter  tuning | lbfgs | l2 | 0.01 | 100 | 1 | 0.7312 |
| 3rd Final model | liblinear | l1 | 0.1 | 100 | 3 | 0.8850 |

From the 2nd model to 3rd model, learning curve (which will be shown in Figure 8) is utilized to determine the most appropriate degree of polynomial which should be used in the model. The elbow method can point out that the number of 3 is the optimal point due to it being the elbow point, which has the lowest loss out of four degree of polynomial which are tested in this method. Loss in Figure 8 is calculated from mean squared error (MSE).

A graph with a line going up

Description automatically generated

Figure 8. Learning Curve of Logistic Regression Model

After checking model accuracy, the result shows that the final model has the most accuracy, which is 0.8850 of the trained model. Then after checking accuracy of k-fold cross validation when k=5, the model yields average accuracy 0.8942, which is objectively improved when compared to the accuracy to the baseline model (accuracy = 0.7262).

### 4.1.2 Support Vector Machine

To start, this model, SVM is recommended to normalize the data before doing it furthermore. Standard scaler is used to normalize the data, then putting it in model with the random state = 42 so it can generate consistent result. The hyperparameters are not touched in this phase of baseline model. The accuracy of this model is calculated average accuracy from K-fold cross validation on the data (K=5).

Table 2. Result from SVM

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Model | Kernel | C | Gamma | Degree of polynomial | Accuracy |
| 1st Baseline model | rbf | 1.0 | \*default\* | 1 | 0.8854 |
| 2nd Grid search for hyperparameter  tuning | rbf | 1.0 | 1.0 | 1 | 0.8992 |
| 3rd Final Model | rbf | 1.0 | 1.0 | 3 | 0.9037 |

Between the 2nd and 3rd model, learning curve is used to determine the degree of polynomial that is propriate for the mode. Figure 9 shows that from the elbow method, degree of polynomial of 3, with the lowest MSE. Thus, the final model will use 3 as the degree of polynomial.

A graph with a line and a line

Description automatically generated

Figure 9. Learning Curve of Support Vector Machine

After checking model accuracy, the result shows that the final model has the most accuracy, which is 0.9037 of the trained model. Then after checking accuracy of k-fold cross validation when k=5, the model yields average accuracy 0.8992, which is objectively improved when compared to the accuracy to the baseline model (accuracy = 0.8854).

### 4.1.3 Neural Network

Start with baseline model, the model with input layer, output layer, and one hidden layer with drop out layer to prevent overfitting. The optimizer of model is ‘adam’, loss is binary cross entropy and metric to be optimized is accuracy, epochs is 50, and batch size is 32. The accuracy from this model is 0.8788 with 0.2826 loss. Noted that activation function is all relu and for the output layer, it is sigmoid.

Table 3. Results from Neural Network

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Model | Epochs | Batch Size | Units | Dropout rate | Learning rate | Number of hidden layer | Accuracy |
| 1st Baseline model | 50 | 32 | 16 | 0.2 | 0.01 | 1 | 0.8788 |
| 2nd Model with early stopping | 150 | 32 | 64 | 0.2 | 0.01 | 1 | 0.9237 |
| 3rd Model tuned from optuna | 50 | 32 | 144 | 0.5 | 0.01 | 2 | 0.9512 |

Seeing that the model after tuning with optuna has the best result, which its accuracy is 0.9521 compared to baseline model with the accuracy of 0.8788. After that, the last evaluation is to utilize this hyperparameters into the final model. The hyperparameters were units = 144, dropout rate = 0.5, learning rate = 0.01, and number of hidden layers = 2.

## 4.2 Comparing between model

Table 4. Table comparing accuracy

|  |  |
| --- | --- |
| Model | Accuracy |
| Logistics Regression | 0.8946 |
| SVM | 0.9037 |
| Neural Network | 0.9512 |

From all the best of each model, it can be compared with accuracy, which is the metric that is chosen from the start of methodology. Result is tuned neural network model has the best accuracy of 0.9512 from the validate data.

The last step is to calculate the accuracy using the test dataset. The accuracy of the tuned neural network on test datasets is 0.9462. Which is usable due to its negligible difference on accuracy.

## 4.3 Discussion

As highlighted in Table 4, the tuned Neural Network model emerged as the champion with the highest accuracy (0.9512) on the validation data. This indicates its superior ability to correctly classify fruit quality (good or bad) compared to Logistic Regression (0.8946) and SVM (0.9037). The Neural Network's success can be attributed to its flexibility in capturing complex relationships within the data, potentially due to its non-linear activation functions and ability to learn intricate feature representations.

The final evaluation involved testing the tuned Neural Network model on unseen data from the test set. Despite a slight decrease in accuracy (0.9462 compared to 0.9512 on the validation set), the performance remains highly promising. This minor difference suggests the model generalizes well to unseen data, making it a strong candidate for real-world fruit quality prediction tasks.

While this project achieved promising results, some limitations offer avenues for future exploration. The evaluation relied primarily on accuracy, and incorporating additional metrics like precision, recall, and F1-score could provide a more comprehensive understanding of model performance. Additionally, while the data underwent cleaning procedures, the absolute values used in certain features remain unknown. Furthermore, limited knowledge regarding the data's background and the specific methods used to measure the features could potentially influence model generalizability. Future work can address these limitations by exploring a broader range of evaluation metrics, potentially delving deeper into the data's origins and measurement techniques.

# 5. References