Technical Report

Introduction

The quality of apples is important for customers. Apples require quality control in order to meet customer expectations. Traditional quality control takes a lot of time by using models to classify apples as good or bad could solve this problem. This could enhance overall product quality delivered to consumers.

Analysis

The data set, found here, contains the following features

- **Size**
- **Weight**
- **Sweetness**
- **Crunchiness**
- **Juiciness**
- **Ripeness**
- **Acidity**
- **Quality**

The metrics used to evaluate our models were: Accuracy, Mean Absolute Error (MAE), Mean Squared Error (MSE)

The target variable is **Quality.** This was the only categorical variable and it classified the quality as "Good" or "Bad" so it was label-encoded to convert "Good" and "Bad" labels into 1 and 0

Features were standardized using standard scaler

I split the data 80/20 for training/testing respectively

Methods

I made a logistic regression and a neural network model.

Logistic Regression Model

The Logistic Regression model was made in order to demonstrate the effectiveness of a linear model on this problem.

The features were used to train the model without tuning. I focused on the default to make a baseline.

Neural Network Model

This model was made in order to capture the non linear relationships between features that the log reg would miss

Architecture:

- **Input Layer**: Seven standardized features.
- **Hidden Layers**:
    - **First Hidden Layer**: 250 neurons with ReLU activation and L2 regularization (λ=0.001) to prevent overfitting.
    - **Dropout Layer**: Applied a dropout rate of 50% to reduce overfitting by randomly disabling neurons during training.
    - **Second Hidden Layer**: 100 neurons with ReLU activation and L2 regularization.
    - **Subsequent Hidden Layers**: Layers with 50, 30, 20, and 10 neurons respectively, each utilizing ReLU activation and L2 regularization.
- **Output Layer**: A single neuron with a sigmoid activation function to output probabilities for binary classification.

Training Configuration:

- **Loss Function**: Binary cross-entropy, appropriate for binary classification tasks.
- **Optimizer**: Adam optimizer with a learning rate of 0.001, offering adaptive learning rates for efficient convergence.
- **Batch Size**: 16 samples per batch, balancing training speed and stability.
- **Epochs**: Trained over 50 epochs to allow sufficient learning iterations.
- **Validation**: Model performance was monitored on the test set after each epoch to detect potential overfitting.

Results

Neural Network Model Performance**Training Metrics**:

    - **Accuracy**: 95.13%
    - **MAE**: 7.43%
    - **MSE**: 3.75%
- **Testing Metrics**:
    - **Accuracy**: 93.62%
    - **MAE**: 8.46%
    - **MSE**: 4.93%

Logistic Regression Model Performance

- **Training Metrics**:
    - **Accuracy**: 74.53%
    - **MAE**: 25.47%
    - **MSE**: 25.47%
- **Testing Metrics**:
    - **Accuracy**: 75.38%
    - **MAE**: 24.62%
    - **MSE**: 24.62%

The neural network outperformed the logistic regression model. It had a much higher accuracy and lower error rates.

The gap between the neural network and logistic regression models suggests that the relationships between the apple attributes and their quality are non-linear.

Logistic regression is limited. The neural network's non-linear activation functions enable it to learn and model the patterns

93% accuracy of the neural network demonstrates the neural network was necessary. The non-linear connections of features like sweetness, acidity, and ripeness contribute to the perception of quality.

Reflection

This assignment demonstrated the ability of the neural network to model the nonlinear connections between features. Logistic regression was picked because it was simple and could establish a baseline to determine if a neural network was necessary. The Logistic Regression had a fairly poor accuracy of 75%. This demonstrated the need for a more complex model.

The hardest part of this was making the architecture for the neural network. Using L2 regularization and dropout were necessary to make the model's capable. Using these techniques made the model go from roughly 40% percent accuracy to around 93% accuracy.

In hindsight, further optimization could include:

- **Hyperparameter Tuning**: Systematically adjusting parameters like the number of neurons, learning rate, and regularization strength to improve performance.
- **Cross-Validation**: Employing k-fold cross-validation to ensure the model's robustness across different data subsets.
- **Feature Engineering**: Creating new features or combining existing ones to provide additional predictive power.