Statistical Learning Final Project

Predicting the quality of apples with logistic regression and neural networks

Jan Thiele (1484203) & Katrin Sauter (1485922)

Introduction

Ensuring the quality of apples is critical for supermarkets and producers to meet customer expectations and optimize production. This work addresses these needs by developing a predictive model for apple quality classification. We analyze a dataset of 4,000 apples, each characterized by seven attributes: size, weight, sweetness, crunchiness, juiciness, ripeness, and acidity. Our goal is to predict whether an apple is of good or bad quality. To approach this binary classification problem, we compare a logistic regression model to a simple neural network, to determine which method provides more accurate and reliable predictions, while also considering their computational costs.

Analysis

The dataset comprises 4,000 observations with seven continuous predictor variables, an ID-variable and the binary target variable "Quality". Table 1 shows the summary statistics for all predictors. We cleaned the data by converting character to numeric variables and one-hot encoded the binary target variable, as well as excluding one missing value.

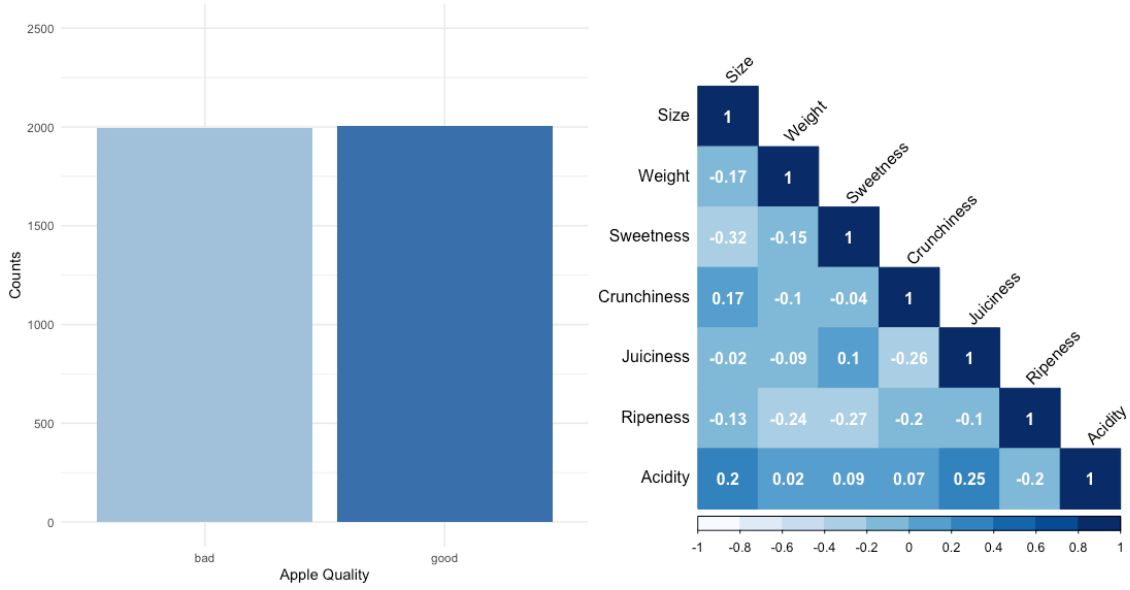
Table 1: *Features Summary Statistics*

	Size	Weight	Sweetness	Crunchiness	Juciness	Ripeness	Acidity
mean	-0.5030	-0.9895	-0.4705	0.9855	0.5121	0.4983	0.0769
sd	1.9281	1.6025	1.9434	1.4028	1.9303	1.8744	2.1103
min	-7.1517	-7.1498	-6.8945	-6.0551	-5.9619	-5.8646	-7.0105
max	6.4064	5.7907	6.3749	7.6199	7.3644	7.2378	7.4047

The target variable is equally balanced between quality classes, with n=1996 for "bad" quality apples and n= 2004 for "good" quality, eliminating concerns about class imbalances impacting

model performance. The correlation matrix of features revealed no substantial correlations, indicating that multicollinearity is not a concern for our modeling approaches either.

We split the dataset into training (80%) and test (20%) sets, resulting in 3200 and 800 observations respectively. We removed outliers defined as values exceeding three standard deviations from the mean. While neural networks are generally robust to outliers, logistic regression can be sensitive to extreme values. For the neural network model, we standardized all predictor variables to have mean zero and unit variance. Note that we applied standardization using only training set statistics to both train and test data to prevent data leakage.



(a) Target Distribution

(b) Feature Correlation Matrix

Figure 1: *Exploratory Analysis of target and feature variables.*

Methods

We constructed a feedforward neural network, whose architecture is summarized in table 2. The three hidden layers contained 128, 64, and 32 neurons respectively. The input layer dimension matches our seven predictor variables, while the output layer contains a single neuron. We used ReLU activation functions in the hidden layers and a sigmoid function in the output layer matching our binary classification task.The model was trained with a batch size of 128 for up to 100 epochs, using the Adam optimizer with the default learning rate of 0.001 and binary cross-entropy loss as the loss function. To prevent overfitting and improve training efficiency, we implemented early stopping with a patience of 20 epochs and learning rate reduction by a factor of 0.05 when validation loss plateaued. We incorporated dropout regularization at a rate

of 20% between all layers to randomly deactivate neurons during training, reducing the risk of overfitting. Additionally, we applied batch normalization after each dense layer to stabilize gradient propagation and accelerate convergence.

We arrived at this configuration after initially starting of with a smaller architecture comprising three hidden layers (32, 16, and 8 neurons). Here, training loss fell below validation loss while both continued to decrease, indicating potential overfitting. Even after introducing dropout regularization to prevent this, early stopping was never triggered and both curves kept improving, suggesting insufficient model capacity. We therefore expanded to the configuration shown in Table 2. Since convergence remained slow, we added batch normalization while keeping other hyperparameters constant, which accelerated convergence substantially. We retained batch normalization in our final architecture.

Table 2: Model Architecture

Element	Choice
Number of hidden layers	3 (with 128, 64, 32 neurons)
Size of input layer	7
Size of output layer	1
Dropout	0.2
Activation Function hidden layers	ReLu
Activation Function output layer	Sigmoid
Loss	Binary-cross-entropy

Figure 2 shows training and validation loss curves of the model. We evaluated the model training on 20% of the training data (validation set), by tracking loss, accuracy, and learning rate over time. While validation accuracy exceeded training accuracy throughout, validation loss fell below training loss after approximately epoch 12 and stayed below until end of training. This inverted pattern, where the model performs better on validation than training data, is characteristic of dropout regularization, which restricts the model in training by deactivating neurons but not during validation. Overall the observed behavior indicated effective regularization and successful generalization. Early stopping triggered around 95 epochs, with a drop in learning rate indicating a performance plateau and subsequently no further improvement in accuracy or loss.

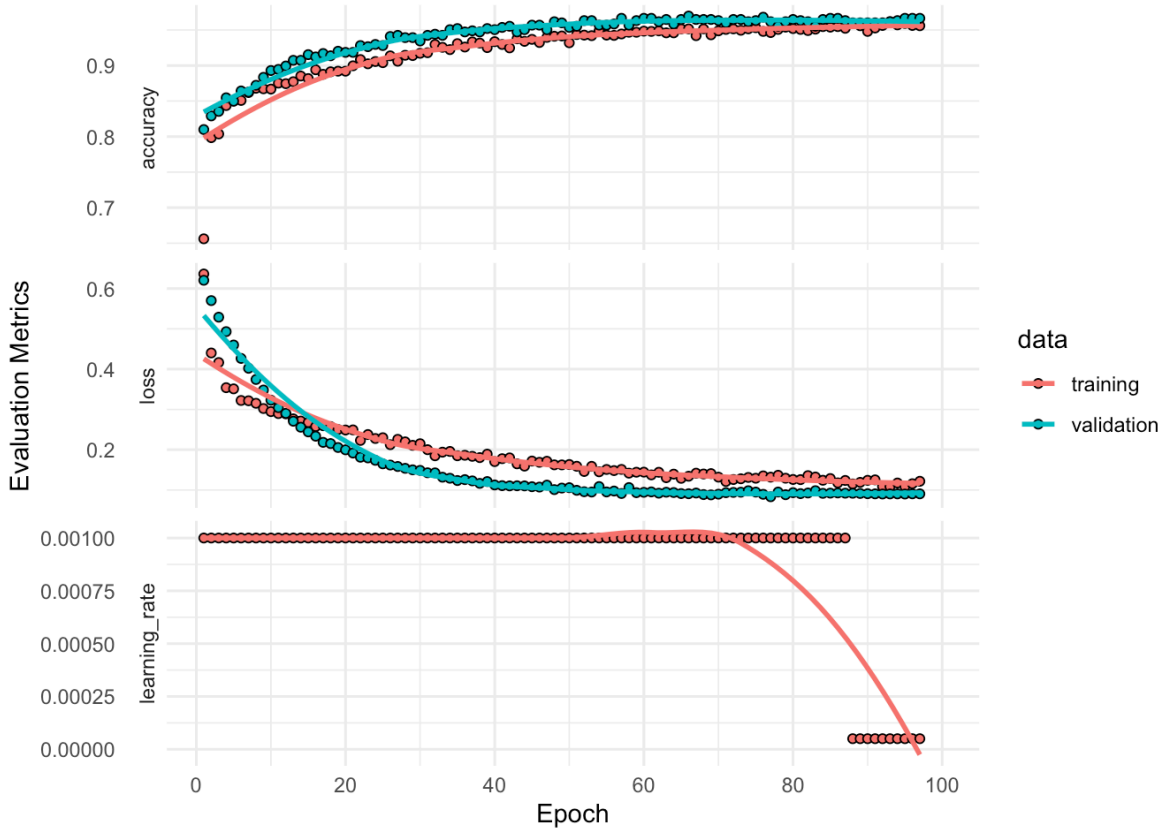


Figure 2: *Neural Network training curves on training and validation set*

Results

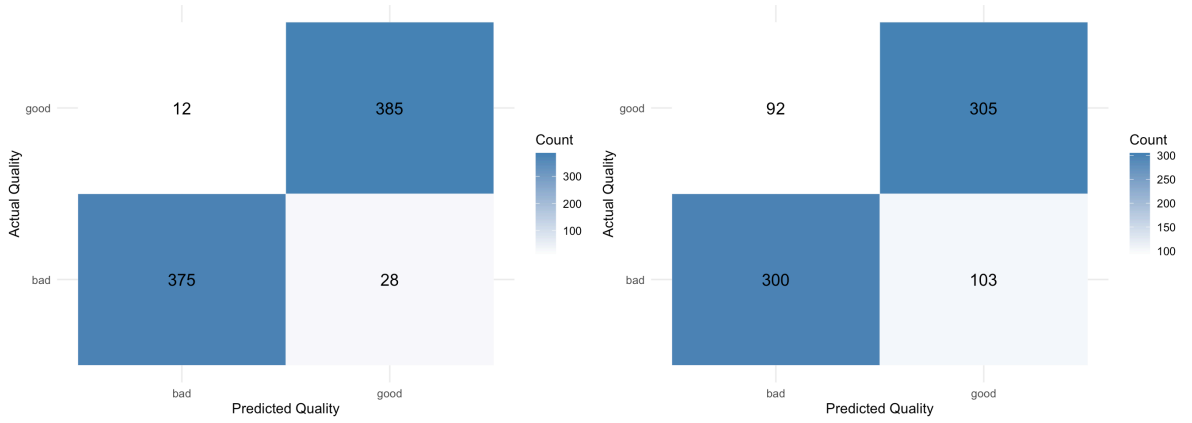
We evaluated both the neural network and logistic regression models on the test set. Figure 3 shows the confusion matrices for the neural network and the logistic regression baseline model. The neural network achieved near-perfect classification with minimal misclassifications across both classes, while logistic regression missclassified approximately one quarter of observations in each class. Both approaches display a slight asymmetry in their error patterns, producing marginally more false negatives than false positives. However, this tendency is minor in both models and has limited practical consequences for the application context of quality control in retail or waste reduction.

Table 3 displays the model performances in comparison. The neural network outperforms the logistic regression baseline across all metrics, achieving 95% accuracy compared to 76%. As expected based on the confusion matrices, both models exhibit balanced precision and recall, further reflected in their F1-scores. The clear reduction in log-loss between the models

however indicates more reliable probability predictions and greater model confidence for the neural network compared to the logistic regression baseline.

Table 3: Model Performance Comparison

Model	Accuracy	Precision	Recall	F1-Score	Log-Loss
Neural Network	0.950	0.932	0.970	0.951	0.163
Logistic Regression (Baseline)	0.756	0.748	0.768	0.758	0.511



(a) Neural Network

(b) Logistic Regression

Figure 3: *Confusion Matrices for Neural Network and Logistic Regression on Test Set*

This performance gap suggests the relationship between fruit characteristics and quality might be non-linear. Introducing non-linearity through the neural network proved beneficial, as logistic regression is constrained by its linear decision boundary and cannot capture such relationships. However, whether a deep architecture with three hidden layers was necessary remains questionable. While non-linearity was clearly essential to reduce bias, a shallower neural network or tree-based model might achieve comparable performance with lower complexity. Given the principle of parsimony (preferring the simplest effective model) a less complex architecture could represent a better bias-variance trade-off. Additionally, tree-based models offer the advantages of greater interpretability and lower computational cost.

Reflection

The most valuable insight from this project was recognizing that neural networks, while powerful, are not always necessary. Despite the neural network's superior performance, we suspect a shallower network or tree-based model could have achieved comparable results with significantly lower complexity. Going forward, we will approach such problems with simpler models

and add complexity incrementally rather than defaulting to deep architectures. Hyperparameter tuning presented another challenge. Manual experimentation, was time-consuming and we potentially missed better configurations. For future projects, we will consider systematic grid search to hypertune more efficiently.