Technical Report: A Case Study in Apple

Marius Helten

2025-02-10

https://github.com/Ari-manius/Statistical_Learning_Exam

Introduction

Food and perishable goods are critical products in societies worldwide and provide great challenges for industry and consumers alike. The quality of the goods is constantly changing during the lifetime of the products, and many precautions have to be taken in order to control and maintain the quality; this includes proper harvest and storage and other measures. Therefore a quick and easy way of classifying the quality of individual items is a major concern, a few rotten apples can spoil the bunch.

In the following report, we present a comparative study of a supervised learning classification problem with an approach from deep learning and a comparative approach from "traditional" machine learning. The central question is if we can create neural network models that can reliably predict the quality of an apple based on its features.

For the training and evaluation of the models, we will use a dataset that was generously provided by an anonymous American agriculture company[1] with 4000 observations and the following variables, Unique ID, Size, Weight, Sweetness, Crunchiness, Juiciness, Ripeness, Acidity and Quality.

Analysis

Exploratory Data Analysis

Single Variables and their Distributions

We will start off with an exploratory analysis of the dataset. A model can only be as good as the data. It is very interesting that our dataset contains a sample that is distributed almost perfectly 50/50 into good and bad apples in a uniform distribution, so it seems to be more than just a few bad apples. This at least hints at a stratified sample in the selection of our data. We also do not know how the category was constructed and why apples were classified either way. This will have a big influence on our further analysis, though, since we are trying to classify for exactly these two categories.

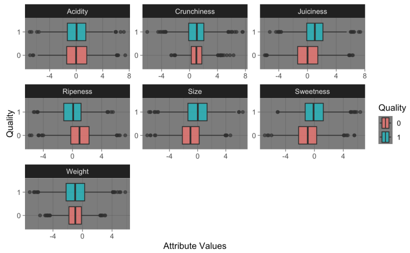
Statistical measures (1.a) give us a first impression as to what we are dealing with. The A_id do not provide us with any useful information and will be excluded from here on. The other features have some variation, and so might provide us with useful variance.

The box plots (1.b) confirm the first impression. We can see that there are differences between the centers of the distribution of the attributes for each of our quality categories. The variables are all uni-modal normal distributions, which is pretty natural for variables such as size and weight. Also, some outliers are part of the distributions (points in the box plots); these were kept due to a limited amount of data.

[1] https://www.kaggle.com/datasets/nelgiriyewithana/apple-quality

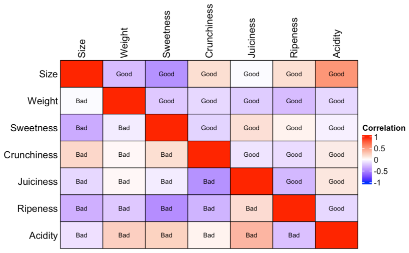
Variable	Mean	SD
A_id	1999.50	1154.84
Size	-0.50	1.93
Weight	-0.99	1.60
Sweetness	-0.47	1.94
Crunchiness	0.99	1.40
Juiciness	0.51	1.93
Ripeness	0.50	1.87
Acidity	0.08	2.11
Quality	0.50	0.50

(a) Summary Statistics for Dataset

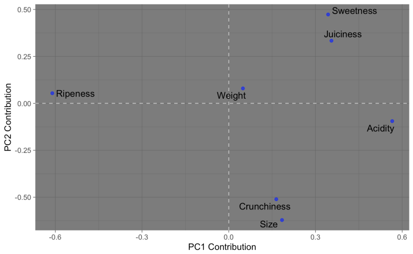


(b) Boxplot for Single Distributions

Figure 1: Single Distributions and Summary Statistics



(a) Correlation Heatmap



(b) PCA Loadings for PC1 and PC2

Figure 2: Single Distributions and Pairwise Correlations for Features

Attribute Correlation Heatmap

The next step after looking at single attributes is to look at more attributes at the same time. (2.a) The attributes on the diagonal are perfectly correlated with each other. The other blue and red fields provide attribute combinations that increase together or go in opposite directions. Also fascinating are pairs where the bad and good quality have an opposite signed correlation because it suggests that the relationship between those features will flip depending on the quality category.

PCA - Principal Component Analysis

A good example for a method that is able to deal with high-dimensional data is the PCA, a technique that generates a set of principal components ordered by their explanatory strength. These components capture the maximum variance in the data. The PCA Loadings (2.b) shows how much influence each attribute has and also infers something about relationships between them; for example, ripeness and acidity seem to have an opposing relation, and sweetness and juiciness, as well as crunchiness and size, seem to be similar.

Feature Selection and Data Preprocessing

The dataset contains only limited amounts of data; therefore, all features were kept, except for the unique ID's, since they provide no relevant information and might compromise the training of the models. The numerical features were centered and scaled through z-standardization, and the target variable of quality was turned into a dummy variable (0: Bad / 1: Good).

Methods

Neural Network

Architecture

The neural network is a feed-forward neural network designed for a binary classification task. For most on the neural net, we use the very efficient (ReLU) Rectified Linear Unit Activation Function. In the output layer the Sigmoid function gives us logarithmic odds, which we can transform into a binary classification. The dropout rate prevents over-fitting by knocking out random neurons on each pass. The model architecture for the hidden layers was chosen based on trial and error in the construction phase.

Table 1: Neural Network Model Architecture

Layer:	Input:	Output:	Activation Function:	Dropout Rate:
Input	7	1024	ReLU	Yes (10%)
Hidden 1	1024	512	ReLU	Yes (10%)
Hidden 2	512	256	ReLU	Yes (10%)
Hidden 3	256	128	ReLU	No
Output	128	1	Sigmoid	No

Training and Learning

For the training of the neural network model we use the binary-classification error loss function and the Adam-optimizer with a fixed learning rate(0.001) and weight decay (0.01). We trained the model with batch of sizes of 32 observations for efficiency and stability. The model Quality is tested with a 5-fold cross validation. Note that k-fold cross-validation was used to evaluate the model design, not a particular training by re-train the model of the same design. After a certain period without improvement, the learning is stopped.

Random Forest

As comparison for our model we used a Random Forest classifier it is an ensemble learning method. It builds multiple decision trees during training and merges their outputs to improve the overall performance and robustness of the model. The models was validated with the 5-fold cross validation.

Results

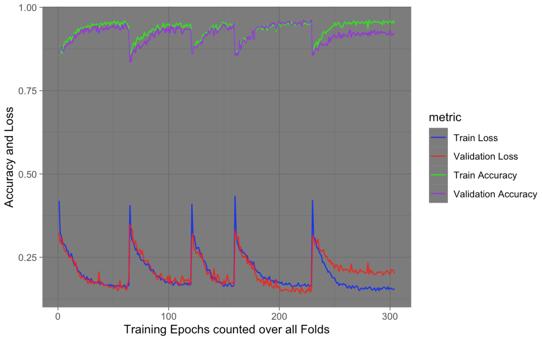
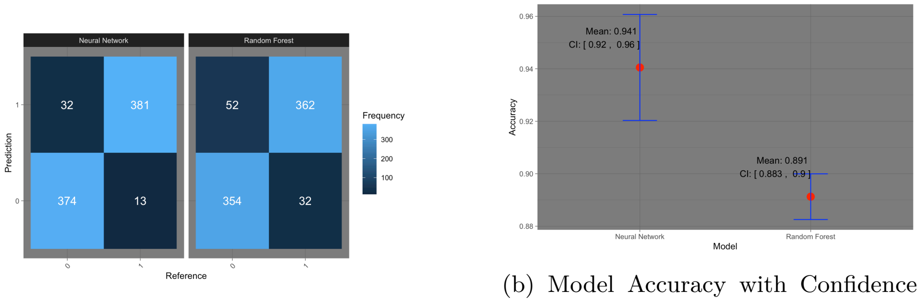


Figure 3: Loss and Accuracy of Neural Network during Training

The first graph (3) shows very well that as training progresses, the loss of the model through false predictions gets smaller, while the accuracy rises through correct predictions. The minimal divergence between training and validation results for loss and accuracy is a good sign, as this shows our model is not over-fitting to the training data. The longer training continues, the more we run the risk of over-fitting; this is why we stop early. We can also see the cross-fold validation in the resets, since in each iteration the model is trained again from scratch.

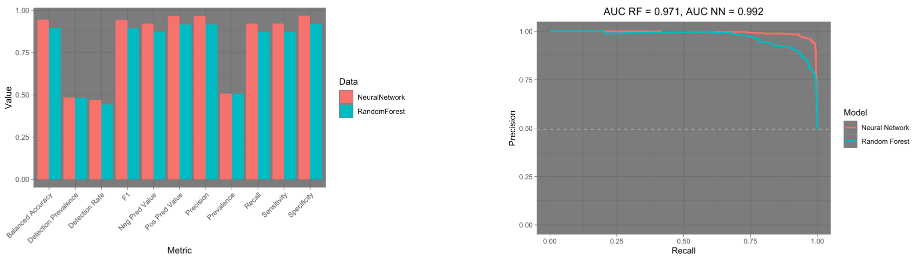


(a) Confusion Matrices for Predictions

(b) Model Accuracy with Confidence Intervall

Figure 4: Model Evaluation and Comparison I

Below (4.a), we can see all the different fields of the confusion matrix. If we wanted to optimize the average quality of our sold apples without caring about waste, we would choose a model with as few false positives as possible, while tolerating some bad apples to increase throughput would lower false negatives, so it heavily depends on the use case, we maximized accuracy.



(a) Evaluative Metrics

(b) Precision-Recall Curve with AUC

Figure 5: Model Evaluation and Comparison II

The neighboring graph (4.b) shows the 95% confidence interval, the CI was calculated by k-fold cross-validation. We can see that the neural network clearly outperforms the other approaches, with an accuracy of around 94% and up to 96%. Taking a look at other evaluative metrics (5.a) the neural network is the stronger across the board for all metrics. Including Recall, F1, Balanced Accuracy, Specificity and Precision.

The following graph (5.b) shows the precision-recall curve. In our case, the curve only starts to dip at the very end, which is good because it means we are not losing F1 score while moving the threshold between precision and recall. The closer the AUC is to 1, the better the model distinguishes between positive and negative cases.

Reflection

The deep learning approach using a neural network proved to be fruitful and practical. While the other methods achieved respectable outcomes, the neural network's performance was superior. The computational costs were very limited during the analysis, but might prove problematic during scale-up. A more comprehensive for tuning the hyperparameters would probably give better results. Also for validation and training more data would lead to a more robust evaluation and training process.

A key drawback of neural networks is the lack of transparency; unlike methods such as PCA or decision trees, we cannot ascertain the relative importance of features. Concrete implementation goals and setting are essential when selecting the best approach and model for a task. For instance, assessing if an apple is poisonous or edible vs. how it looks, is some cases might tolerate higher error rates in our classification than in others. Nonetheless, this example highlights the potential of data analysis and deep learning methods to address complex problems and the practicality of obtaining results with them. Comparing Apples and Oranges is possible.