Classification of Apple Quality using R

Data Mining and Knowledge Discovery

by

Amrithya Balaji

May 1, 2024

Classification of Apple Quality using R

Data Mining and Knowledge Discovery

Abstract

A round, edible fruit produced by apple trees is called an apple. In order to validate the proverb, "An apple a day keeps the doctor away," which highlights the purported health advantages of the fruit, we must also pay attention to the quality of the apples that are eaten.

Farmers are forced to adapt to changing conditions by implementing new controlled methods that consider a variety of factors, such as temperature, sunlight, and chlorophyll concentration. This quantitative control not only improves apple quality but also helps farmers reduce waste by discarding bad or low-quality apples early on.

What actually decides the quality of an apple? To answer this question, we look at a variety of fruit attributes, including size, weight, sweetness, crunchiness, juiciness, ripeness, and acidity. We apply machine learning through R with the help of clustering and other models such as decision tree and random forest to study the impacts of these attributes.

Keywords: Apple quality, Clustering, Decision tree and Random forest

1. Introduction and Dataset Exploration

An apple tree (Malus spp., among them the domestic or orchard apple; Malus domestica) yields round, edible fruits that are known as apples. Apple trees are the most commonly grown species in the genus Malus and are grown all over the world. Apples come in over 7,500 different cultivars.Different cultivars are bred for various tastes and uses, including cooking, eating raw, and cider or apple juice production. Apple production reached 93 million tonnes worldwide in 2021.[Wik]

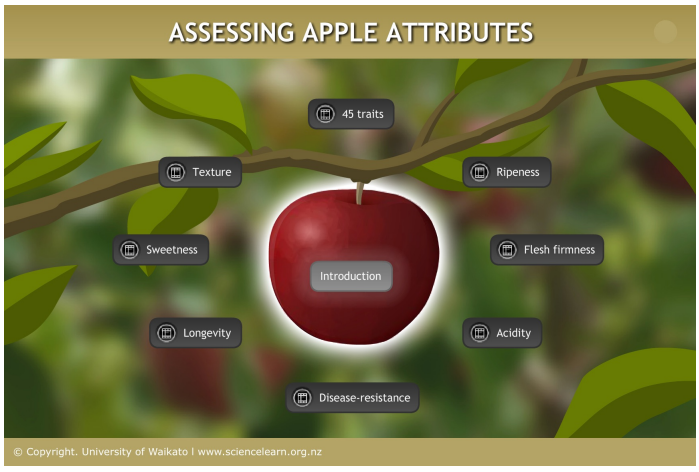


Figure 1: Assessing quality of apple[sci]

Apple quality is important for a variety of reasons. It has a direct impact on consumer satisfaction by meeting expectations for taste, texture, and appearance. Superior quality control reduces waste and improves supply chain efficiency.

Quality apples retain more nutritional value, providing greater health benefits, and are more durable for extended storage and processing into other products. Furthermore, focusing on apple quality can promote more sustainable agricultural practices, such as better pest management and fertilization techniques. This comprehensive importance

spans economic, environmental, and consumer aspects, making quality a critical focus in apple production and consumption.

Attribute	Methodology	Tools/Standards Used	Key Points
Ripeness	Starch pattern index test; spraying apple flesh with iodine solution to identify starch presence	Iodine solution	Less starch indicates a riper apple as starch breaks down into sugars.
Cruniness	Measure of flesh firmness using a penetrometer which records resistance	Penetrometer, computer	Penetrometer probe measures resistance at a set size, speed, and distance.
Sweetness	Measurement of soluble solids concentration using a refractometer	Refractometer	Soluble solids mostly comprise sugars; refractometer measures light refraction in apple juice.
Acidity	Acid concentration in apple juice measured via titration technique	Titration equipment, juicer	Major acid is malic acid; acidity measured by neutralizing juice sample with a base solution.
Juiciness	Assessed through human sensory evaluation	Human assessors, standards (carrot, celery, watermelon)	Assessors rate attributes on a 0-9 scale, calibrated with standards like carrot and celery.

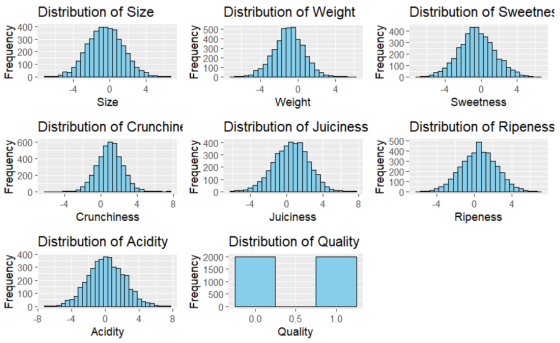
Table 1: Summary of Apple Quality Measurements[sci]

The Apple Quality Dataset is an open-source data set taken from Kaggle[ELG]. There are 4,001 entries in the Apple Quality Data set, and each one describes a different feature of a single apple. This data set is organized to make it easier to create machine learning models that can evaluate and forecast apple quality based on both chemical and physical traits. The following characteristics allow each entry in the data set to be uniquely identified and quantitatively described:.

Column Name	Description	Data Type	Potential Use
A_id	Identifier for each apple	Numerical	Indexing apples
Size	Represents the size of the apples	Numerical	Analyzing size-related quality factors
Weight	Indicates the weight of the apples	Numerical	Analyzing weight-related quality factors
Sweetness	Scores reflecting the sweetness of the apples	Numerical	Correlating sweetness with consumer preference
Crunchiness	Scores indicating the crunchiness of the apples	Numerical	Studying texture preferences
Juiciness	Scores assessing the juiciness of the apples	Numerical	Evaluating juiciness in quality assessment
Ripeness	Related to the ripeness of the apples	Numerical	Determining optimal harvest time
Acidity	Values representing the acidity level of the apples	Numerical	Balancing taste and preservability
Quality	Overall quality of the apples ('good' or 'bad')	Categorical	Outcome variable for quality models

Table 2: Summary of Apple Quality Dataset

The histograms (in Figure 2) show the distribution of different apple quality attributes from the dataset. Each attribute—Size, Weight, Sweetness, Crunchiness, Juiciness, Ripeness, and Acidity—has a bell-shaped curve, indicating that their values are normally distributed around the mean. This suggests that the majority of apples have average characteristics, with fewer apples displaying extreme values in any attribute. However, the Quality attribute is displayed as a categorical variable with two levels, representing 'good' and 'bad', and shows the number of apples in each category. The frequencies indicate a balanced distribution between the two quality levels.



(a) Distribution of each features

Ripeness	-0.2	-0.13	-0.2	-0.26	-0.1	-0.27	-0.24
-0.2	Crunchiness	0.17	0.07	-0.01	-0.26	-0.04	-0.1
-0.13	0.17	Size	0.2	0.24	-0.02	-0.32	-0.17
-0.2	0.07	0.2	Acidity	-0.01	0.25	0.09	0.02
-0.26	-0.01	0.24	-0.01	Quality	0.26	0.25	0
-0.1	-0.26	-0.02	0.25	0.26	Juiciness	0.1	-0.09
-0.27	-0.04	-0.32	0.09	0.25	0.1	Sweetness	0.15
-0.24	-0.1	-0.17	0.02	0	-0.09	-0.15	Weight

(b) Correlation Matrix

Figure 2: Feature distribution and Correlation Matrix of the Dataset

2. Data Pre-processing

Several quality checks were performed on the data to guarantee its dependability for further analysis.

2.1. Data Cleaning

To eliminate potential bias in the analysis, and to maintain data integrity the following steps were taken:

- **Null Value Check:** Confirmed the dataset's absence of null values.
- **Outliers:** Used box plots(in Figure 3) to find the outliers and removed outliers using Z_Scores

Algorithm 1 Remove Outliers Using Z-Scores

Input: *apple_quality* (Data frame of apple quality attributes)
Output: *apple_quality* (Data frame with outliers removed)

apple_quality ← as.data.frame(*apple_quality*)
outlier_indices ← list()
for all *col_name* in names(*apple_quality*) **do**
    z_scores ← scale(*apple_quality*[[*col_name*]])
    threshold ← 3
    outliers ← which(abs(*z_scores*) > *threshold*)
    outlier_indices[[*col_name*]] ← *outliers*
end for
all_outliers ← unique(unlist(*outlier_indices*))
apple_quality ← *apple_quality*[−*all_outliers*,]

- **Feature Scaling:** The data set was already scaled. So no further scaling was done.
- **Feature Encoding:** The categorical feature Quality was converted into binary, where (Good = 1) and (Bad = 0).
- **Class Balance:** From The histograms (in Figure 2) show the frequencies indicate a balanced distribution between the two quality levels.

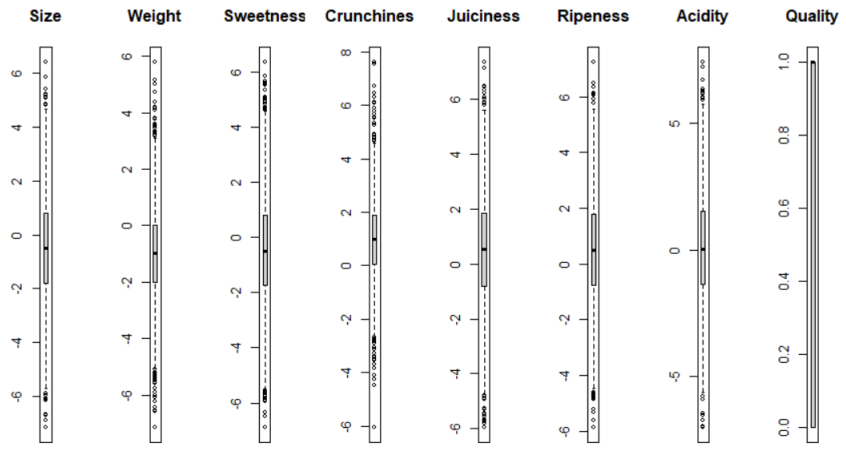


Figure 3: Box plot for outliers

3. KMeans

K-means is a distance- or centroid-based algorithm that uses distance measurements to determine how to assign a point to a cluster. Each cluster in K-Means has a centroid attached to it. Although KMeans is typically applied to unsupervised data, we can cluster our data points to see if they are easily distinguishable from one another. We apply KMeans to the entire data set. We also apply PCA through fviz_cluster to visualize scatter plots. We use the silhouette test to see our optimal clusters. We get K = 2.

Initially, PCA is applied to the training data, which has been centered and scaled, to identify principal components that capture significant variance, with a focus on those that explain at least 80% of the variance. These principal components are then applied to reduce the dimensionality of both the training and testing datasets. The script includes a clustering function that assigns each data point in the reduced-dimension test dataset to the nearest cluster based on Euclidean distance to the k-means cluster centers[Pro]. These clusters are then used to predict quality categories from the test dataset. The accuracy of these predictions is calculated by comparing predicted quality to actual values, which provides a quantitative measure of the model's performance.

Algorithm 2 PCA and k-Means Clustering for Feature Reduction and Classification

Input: $train\_data, test\_data$
Output: $test\_data$ with predicted clusters and accuracy of prediction
$feature\_train\_data \leftarrow train\_data[, !\text{names}(train\_data) \text{ in } "Quality"]$
$pca\_result \leftarrow \text{prcomp}(feature\_train\_data, \text{center} = TRUE, \text{scale.} = TRUE)$
$explained\_variance \leftarrow \text{summary}(pca\_result)\$importance[2,]$
$cum\_explained\_variance \leftarrow \text{cumsum}(explained\_variance)$
$num\_components \leftarrow \text{which}(cum\_explained\_variance \geq 0.8)[1]$
$pca\_scores \leftarrow pca\_result\$x[, 1 : num\_components]$
$feature\_test\_data \leftarrow test\_data[, !\text{names}(test\_data) \text{ in } "target\_variable"]$
$test\_data\_pca \leftarrow \text{predict}(pca\_result, \text{newdata} = feature\_test\_data)$
$test\_data\_pca \leftarrow test\_data\_pca[, 1 : num\_components]$
function GET_CLUSTER_ASSIGNMENT($point, centers$)
    $distances \leftarrow \text{apply}(centers, 1, \text{function}(center)\text{sum}((point - center)^2))$
    return which.min($distances$)
end function
$test\_clusters \leftarrow \text{apply}(test\_data\_pca, 1, \text{get\_cluster\_assignment}, centers = kmeans\_result\$centers)$
$test\_data\$predicted\_cluster \leftarrow test\_clusters$
$test\_data\$Quality\_predicted \leftarrow \text{ifelse}(test\_data\$predicted\_cluster == 1, 1, 0)$
$true\_clusters \leftarrow test\_data\$cluster$
$accuracy \leftarrow \text{mean}(test\_data\$Quality == test\_data\$Quality\_predicted)$

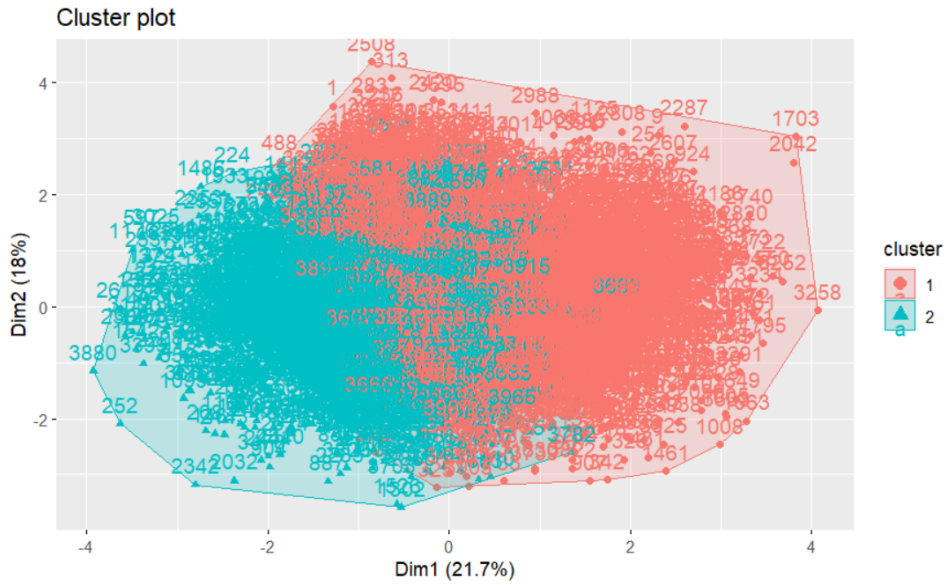
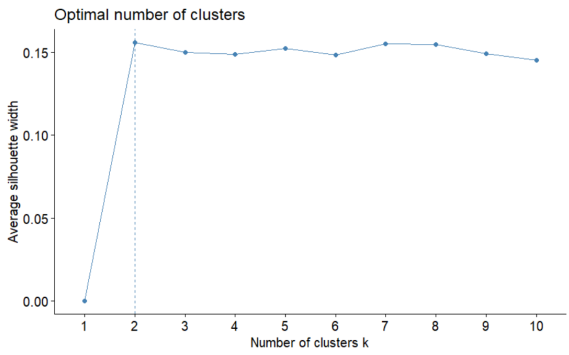
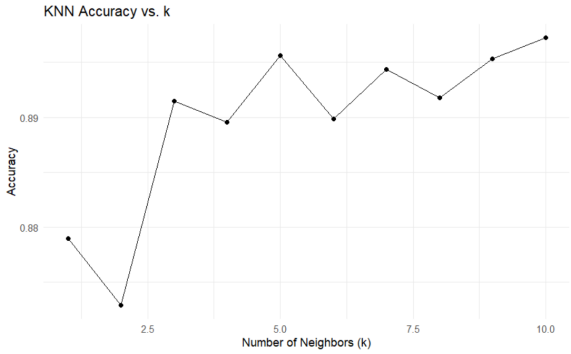


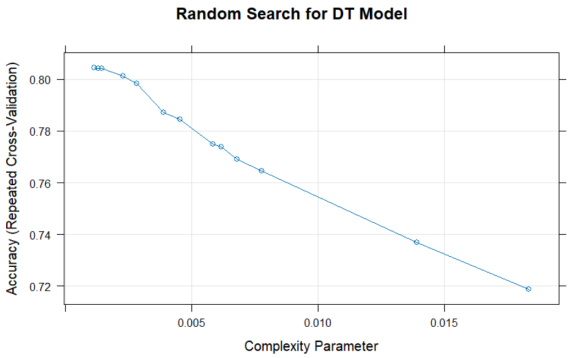
Figure 4: Clusters using Kmeans



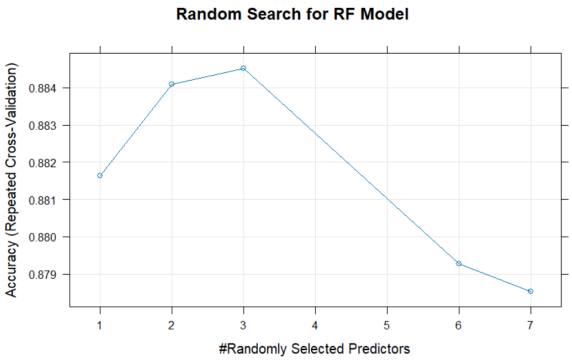
(a) Optimal cluster



(b) Optimal K value for KNN



(c) Optimal CP for DT



(d) Optimal Ntree for RF

Figure 5: Optimal hyper-parameter

4. KNN

KNN is a simple algorithm which classifies or predicts how a single data point will be grouped based on proximity. We do hyper-parameter tuning to see which K value performs well. We take values as k = 1:10. We can see the accuracies are stable with the values of k=6,7,8. So we can take the optimal value of k=7 (in Figure 5b).

Algorithm 3 Train and Evaluate a kNN Model with 10-Fold Cross-Validation

Input: $train\_data, test\_data$
Output: $accuracy$
train_control ← trainControl(method = "cv", number = 10) ▷ Setup for 10-fold cross-validation
knn_model ← train(Quality ., data = $train\_data$, method = "knn",
    trControl = $train\_control$, preProcess = "scale", tuneGrid = data.frame($k = 7$)) ▷ Train kNN model with k = 7
predictions ← predict(knn_model, newdata = $test\_data$) ▷ Predict using the trained model
accuracy ← sum(predictions == $test\_data\$Quality$)/nrow($test_data$) ▷ Calculate accuracy

5. Decision tree and Random Forest

We also evaluate our test data using decision tree and random forest. For hyper parameter tuning, I have used random search[Bro].

Algorithm 4 Train and Evaluate Decision Tree and Random Forest Models

Common Setup:
control ← trainControl(method = "repeatedcv", number = 10, repeats = 3, search = "random")
metric ← "Accuracy"
mtry ← sqrt(ncol($train\_data$))

Decision Tree Model Training and Evaluation
dt_random ← train(Quality ., data = $train\_data$, method = "rpart",
    metric = $metric$, tuneLength = 15, trControl = $control$)
test_predictions ← predict(dt_random, newdata = $test\_data$, type = "raw", cp = 0.001611863)
accuracy_dt ← mean($test\_predictions$ == $test\_data\$Quality$)
print "Decision Tree Model Accuracy: "$accuracy\_dt$

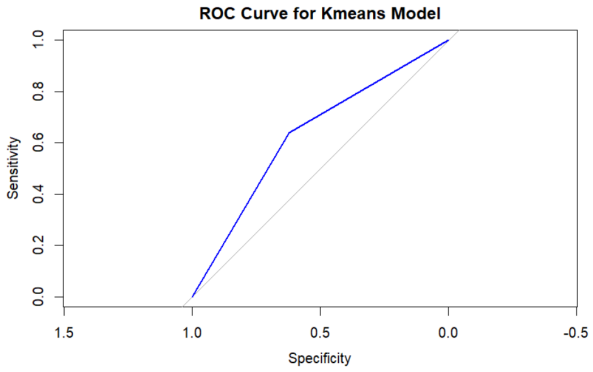
Random Forest Model Training and Evaluation
rf_random ← train(Quality ., data = $train\_data$, method = "rf",
    metric = $metric$, tuneLength = 15, trControl = $control$)
test_predictions ← predict(rf_random, newdata = $test\_data$, type = "raw", ntree = 4)
accuracy_rf ← mean($test\_predictions$ == $test\_data\$Quality$)
print "Random Forest Model Accuracy: "$accuracy\_rf$

6. Results

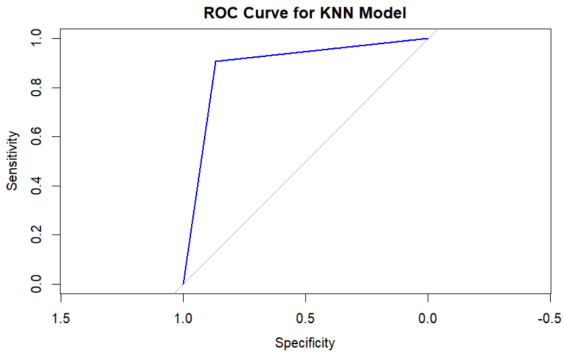
The performance of four different machine learning models—Random Forest, Decision Tree, KNN (k-Nearest Neighbors), and K-Means clustering—using two metrics: Accuracy and F1 Score were analyzed. Accuracy measures the overall correctness of the model, while the F1 Score is a harmonic mean of precision and recall, providing a balance between them, especially useful when the class distribution is uneven. KNN provides better results than other model.

Table 3: Performance Summary of Machine Learning Models

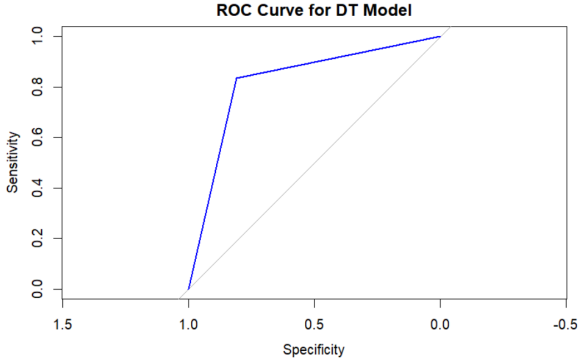
Model	Accuracy	F1 Score
Random Forest	0.874	0.874
Decision Tree	0.804	0.809
KNN	0.895	0.894
K-Means	0.653	0.630



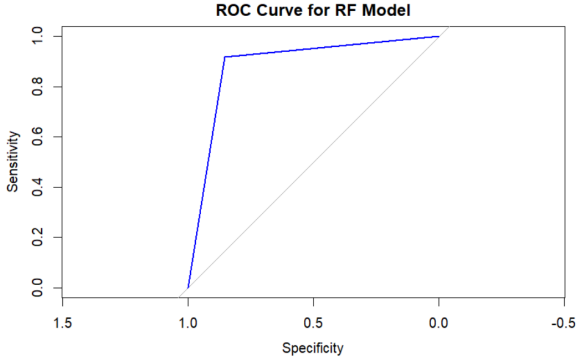
(a) ROC curve for KMeans



(b) ROC curve for KNN



(c) ROC curve for DT



(d) ROC curve for RF

Figure 6: ROC curves for all models

7. Acknowledgements

I would like to express our sincere gratitude to Fabrice Muhlenbach for their supervision throughout this project. I would also like to thank to Professor Jeudy Baptiste for his teaching of the theoretical part of the course. Furthermore, we would like to acknowledge my MLDM (Machine Learning and Data Mining) classmates who generously shared their knowledge and provided assistance when needed.

References

[Bro] Jason Brownlee. *tune-machine-learning-algorithms-in-r*. machinelearningmastery.com. URL: https://machinelearningmastery.com/tune-machine-learning-algorithms-in-r.

[ELG] NIDULA ELGIRIYEWITHANA. *apple-quality-dataset*. www.kaggle.com. URL: https://www.kaggle.com/datasets/nelgiriyewithana/apple-quality.

[Pro] UC R Programming. *kmeans_clustering*. https://uc-r.github.io/. URL: https://uc-r.github.io/kmeans_clustering.

[sci] sciencelearn.org. *Assessing-apple-attributes*. sciencelearn.org. URL: https://www.sciencelearn.org.nz/image_maps/38-assessing-apple-attributes.

[Wik] Wikipedia. *Apple*. wikipedia.org. URL: https://en.wikipedia.org/wiki/Apple.