Melbourne Institute of Technology

MDA512 – Data Science

Assignment Report on

Accuracy Prediction of Apples

Ankit Dhakal MIT23354

Submitted to: Dr. Deepani Guruge

Acknowledgement

I like to offer our heartfelt appreciation to Dr Deepani Guruge for her important assistance and advice throughout the completion of this assignment and the lectures. Her informed feedback and support have greatly influenced the quality and direction of this work.

In addition, we'd like to thank Melbourne Institute of Technology for providing resources that helped us conduct research and acquire essential material for this assignment.

Contents

Introduction:

Customers enjoy apples because of their high dietary fiber, mineral, and multivitamin content, which offers them anti-aging, immunity-boosting, and antioxidant benefits. When eating apples, the most fresh food is devoured, and the fruit's internal quality influences its flavor and texture, which in turn affects consumer willingness and consumption levels. While some consumers may rely on the performance of parameters that can represent a particular aspect of an apple's internal quality to meet their personalized consumption needs, most consumers would rather choose apples that exhibit the best overall performance across all internal quality-related parameters. Stated differently, the overwhelming majority of consumers are more inclined to buy apples that have a higher "internal comprehensive quality." The benefits and drawbacks of internal comprehensive quality serve as a representation of Apple's total internal quality situation. Researchers have always focused more on the subject of whether it is possible to evaluate an apple's internal quality in a thorough and accurate manner.[1]

Problem Statement

The accurate assessment of apple quality is a multifaceted challenge influenced by a variety of features such as size, color, firmness, sugar content, and taste. And because of the impact of multiple factors, including apple variety, growing circumstances, and storage techniques, maintaining consistent quality throughout the apple supply chain presents significant hurdles.[2] All of these factors work together to affect the overall quality of apples, which makes the environment difficult to anticipate and maintain uniform standards in. Our suggested approach to addressing this problem entails using machine learning methods for predictive analysis. In a variety of dynamic supply chain scenarios, the goal is to better forecast and maintain apple quality by utilizing these technologies and that utilizes various attributes to assess apple quality and provides an accuracy score for the predictions.

Resources

Data sets were extracted from:
https://www.kaggle.com/datasets/nelgiriyewithana/apple-quality

This dataset contains information about various attributes of a set of fruits, providing insights into their characteristics. The dataset includes details such as fruit ID, size, weight, sweetness, crunchiness, juiciness, ripeness, acidity, and quality. Sourced from an esteemed American agriculture company, the dataset has been meticulously scaled and cleaned for ease of use.[3]

Attributes of the data set:

- **A_id**: *Unique identifier for each fruit*
- **Size**: *Size of the fruit (type: float)*
- **Weight**: *Weight of the fruit (type: float)*
- **Sweetness**: *Degree of sweetness of the fruit (type: float)*
- **Crunchiness**: *Texture indicating the crunchiness of the fruit (type: float)*
- **Juiciness**: *Level of juiciness of the fruit (type: float)*
- **Ripeness**: *Stage of ripeness of the fruit (type: float)*
- **Acidity**: *Acidity level of the fruit (type: float)*
- **Quality**: *Overall quality of the fruit (type: integer, categorized into good and bad)*

Business Model

Important Elements Affecting Apple Quality:

- What might be the major factors that affect the overall quality of fruits?
- Is it possible to predict apple's quality using various characteristics like: size, weight, sweetness, ripeness, quality etc.
- What might be the main factors affecting the fruits in the dataset in terms of overall quality?
- To what extent do the size and weight of apples impact their overall quality?

Objectives:

- Determining and comprehend the characteristics that most strongly influence the quality of the fruit.
- Using a selection of specified criteria, developing a predictive model to evaluate and forecast fruit quality.
- Compare and analyse the quality attributes of different fruit varieties to understand variations in quality.
- Determining the relevance of the link between fruit quality, weight, and size.

Challenges:

- The prime challenge was to collect data sets that matched our system which included algorithms like decision tree and k means. Since, decision tree requires one of the Boolean attributes in their datasets.
- Accurately capturing the combined impact of several variables on fruit quality is difficult due to their complex interactions and linkages.
- Since, the data sets had negative values in the attributes. It needed cleaning to prepare data.
- The model's ability to generalize to new scenarios may be impacted by changing environmental circumstances that have an impact on fruit quality.

Data Preparation

- Initially we extracted the datasets and checked the null values from both excel and python.

- Even though the data set was quite clear and filtered, to reduce errors and confusions. We filtered and prepared data using systematic sampling. We have around 4001 dataset for several months including date and time, and based on this estimation, we calculated an ideal sample size of 101. Our sampling interval nth, hence, equals 4001/101 = 39.61, which ultimately round to 40^{th} value.
- Removed negative values from the datasets.

Usability/Outcomes:

- Important Elements Affecting Apple's Quality
- Ideal Circumstances for Fruit Quality
- Fruit Quality Comparison by Varieties
- More financial success
- We can also analyze the storage time of apples from the outcomes.

Methodologies/ algorithms used in the project:

1. **Decision Tree:** Models of decision trees show us how machine learning can improve our ability to make decisions. Based on prior experience, a Decision Tree, which is a Flow Chart, can assist in making judgments. A decision tree algorithm is a type of machine learning algorithm that predicts using a decision tree. It uses a model of decisions and their potential outcomes that resembles a tree. Recursively dividing the data into subsets according to the most important attribute at each tree node is how the method operates.[4]
2. **K means for clustering:** Clustering method for large data analysis and graphical representation of corporate data. A large class of methods for identifying observational subgroups within a data collection is called clustering. the act of assembling items or humans into groups so that those in one group resemble one other more than those in other groupings. Since this approach is unsupervised, it looks for connections between the n observations without using a response variable as a training tool. The simplest and most popular clustering technique for dividing a dataset into a set of k groups is called K-means clustering.

Toolkits/Libraries used:

1. **Scikit-learn :** It's a highly reputable and extensively utilized general-purpose Python machine learning toolkit. Along with facilities for data transformation, data loading, model persistence, and model selection and evaluation, it includes a wide range of common supervised and unsupervised machine learning techniques. Classification, clustering, prediction, and other typical tasks can be performed with these models. sklearn, sklearn.tree, sklearn.model etc are used in the project.
2. **Pandas:** Pandas makes it simple to do many of the time consuming, repetitive tasks associated with working with data, including: Data cleansing, Data fill, Data normalization, Merges and joins, Data visualization, Statistical analysis, Data inspection, Loading and saving data, And much more. In fact, with Pandas, you can do everything that makes world-leading data scientists vote Pandas as the best data analysis and manipulation tool available.
3. **Matplotlib:** Matplotlib is a fantastic Python visualization package for two-dimensional array charts. Based on NumPy arrays, Matplotlib is a multi-platform data visualization

package intended to be used with the larger SciPy stack. In the year 2002, John Hunter made its debut. The ability to visually access vast volumes of data in a format that is simple to understand is one of visualization's biggest advantages. Matplotlib consists of several plots like line, bar, scatter, histogram, etc.In our project we have used matplotlib.pyplot , matplotlib.image for the visualization part.

4. **Seaborn:** A fantastic visualization library for Python statistical graphics plotting is called Seaborn. It offers lovely color schemes and default styles to enhance the visual appeal of statistical plots. It is based on the matplotlib library and has tight integration with pandas' data structures.Seaborn wants to make data exploration and comprehension revolve around visualization. For a better understanding of the dataset, it offers dataset-oriented APIs that allow us to flip between various visual representations for the same variables. [5]
5. **Pydotplus :** PyDotPlus is an improved version of the old pydot project that provides a Python Interface to Graphviz's Dot language.

Data Analysis

We predicted the quality of the apples with decision tree and k means algorithm.

Initially when the test size was about 30%, we got the accuracy of 0.61

```
In [48]: #Now splitting the dataset into training set and test set by using function train_test_splilit
         #we need to pass the 3 parameters
         x_test, x_train, y_test, y_train = train_test_split(x, y, test_size=0.3, random_state=1)
```

```
In [53]: print("Accuracy:",metrics.accuracy_score(y_test, y_pred))
         Accuracy: 0.6142857142857143
```

Whereas, when we decrease the test size to 20%, we get the accuracy of 1%. Even though the accuracy is not possible to get 1 as an accuracy. So, we pre-processed the data and removed negative values from the attributes. Yet, we got the same value.

```
In [54]: x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.2, random_state=1)

In [55]: print("Training split input -",  x_train.shape)
         print("Testing split input-", x_test.shape)
         print("Testing split output-", y_test.shape)
         Training split input - (80, 7)
         Testing split input- (20, 7)
         Testing split output- (20,)

In [56]: y_pred = clf.predict(x_test)

In [57]: print(y_pred)
         [1 0 1 1 1 1 1 0 1 0 1 0 0 1 0 0 1 0 0 0]

In [58]: print("Accuracy:",metrics.accuracy_score(y_test, y_pred))
         Accuracy: 1.0
```

Various Data analytics tools or python libraries were used: Sci-kit learn, Pandas, Matplotlib, seaborn, pydotplus. For prediction we used decision tree and k means clustering.

Visualization

For the visualization part we chose the following (figures are present in the results/appendices part):

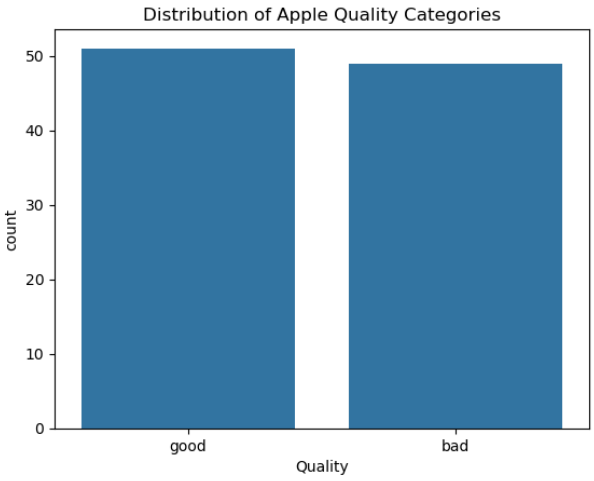
1. Scatterplot: We used scatterplot in our project because of the relationship between two quantitative variables measured for the same individuals is displayed in a scatterplot. One variable's values are displayed on the horizontal axis, and the other variable's values are displayed on the vertical axis. On the graph, every individual in the data is represented as a point. In our case, the attributes are compared with the quality of the apple which is also an attribute of the dataset.
2. Boxplot: A box-and-whisker plot, also called a boxplot, is a graph that summarizes a collection of data. The boxplot's shape displays both the distribution of the data and any outliers. With the ability to create multiple boxplots per graph, it is a helpful method for comparing various data sets.
3. Pairplot: Use the pairplot() function to plot multiple pairwise bivariate distributions within a dataset. The relationship for the (n, 2) combination of variables in a Data Frame is shown as a matrix of plots in the diagonal plots, which are the univariate plots. Because of its best visualizations, we used it for better comparisons of the attributes.
4. Correlation Heatmap: A graphical tool called a correlation heatmap shows the correlation as a colour-coded matrix between several variables. It resembles a colour chart that illustrates the degree of relationship between various variables of the apples.
5. Countplot: The countplot figure is used for the difference between good and bad apples.

Results/Appendices

Distribution of quality:

```
In [40]: # Distribution of quality
         sns.countplot(x='Quality', data=df)
         plt.xlabel('Quality')
         plt.title('Distribution of Apple Quality Categories')
```

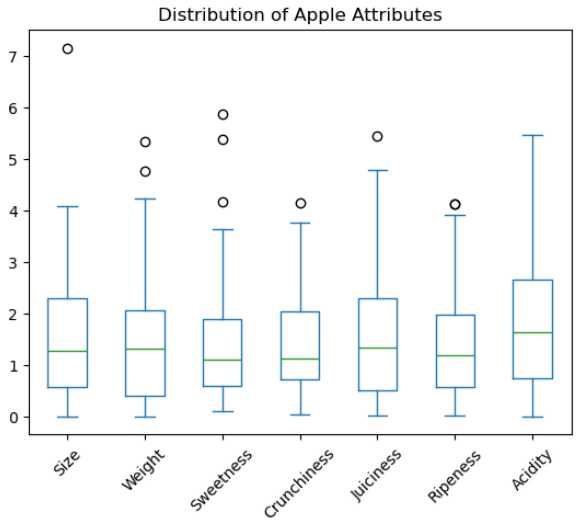
```
Out[40]: Text(0.5, 1.0, 'Distribution of Apple Quality Categories')
```



Boxplot:

```
In [41]: # Boxplots for continuous variables
         df.drop('A_id', axis=1).plot(kind='box')
         plt.title('Distribution of Apple Attributes')
         plt.xticks(rotation=45)
```

```
Out[41]: (array([1, 2, 3, 4, 5, 6, 7]),
          [Text(1, 0, 'Size'),
           Text(2, 0, 'Weight'),
           Text(3, 0, 'Sweetness'),
           Text(4, 0, 'Crunchiness'),
           Text(5, 0, 'Juiciness'),
           Text(6, 0, 'Ripeness'),
           Text(7, 0, 'Acidity')])
```

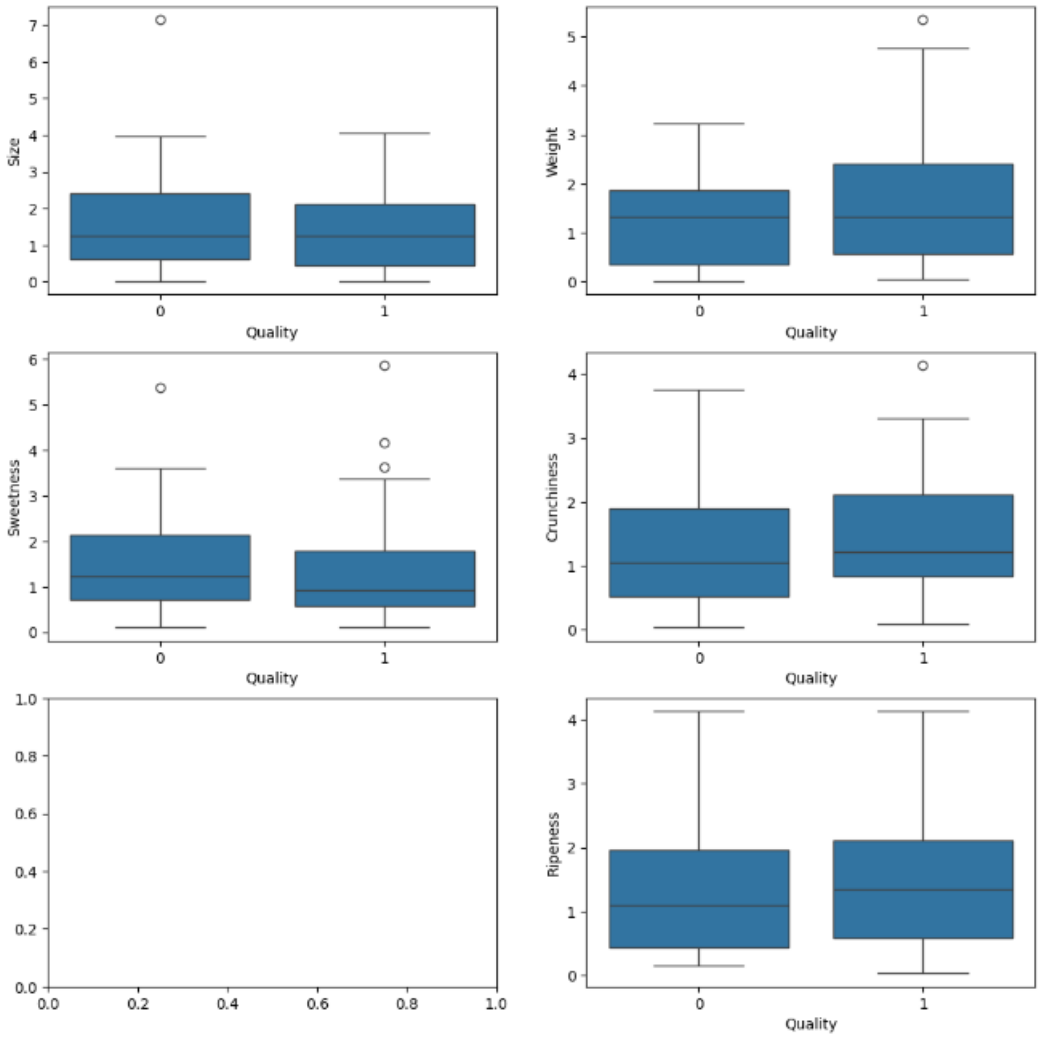


In [45]:
```
# Compare bad vs good apples
fig, ax = plt.subplots(3,2, figsize=(12,12))
sns.boxplot(x="Quality", y="Size", data=df, ax=ax[0,0])
sns.boxplot(x="Quality", y="Weight", data=df, ax=ax[0,1])

sns.boxplot(x="Quality", y="Sweetness", data=df, ax=ax[1,0])
sns.boxplot(x="Quality", y="Crunchiness", data=df, ax=ax[1,1])

sns.boxplot(x="Quality", y="Ripeness", data=df, ax=ax[2,1])
```

Out[45]: <Axes: xlabel='Quality', ylabel='Ripeness'>



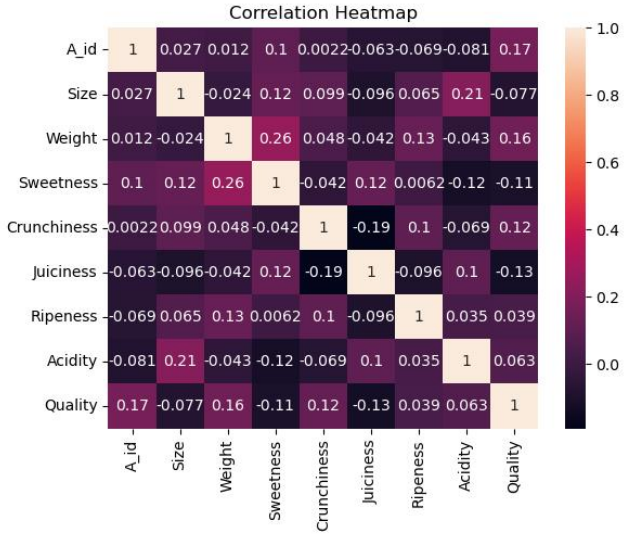
Correlation Heatmap:

Here, Good apple is denoted as 1 and Bad apple is denoted as 0.

```
In [42]: # Encode Quality column
         df['Quality'] = df['Quality'].map({'good':1, 'bad':0})

         # Heatmap with encoded quality
         sns.heatmap(df.corr(), annot=True)
         plt.title('Correlation Heatmap')
```

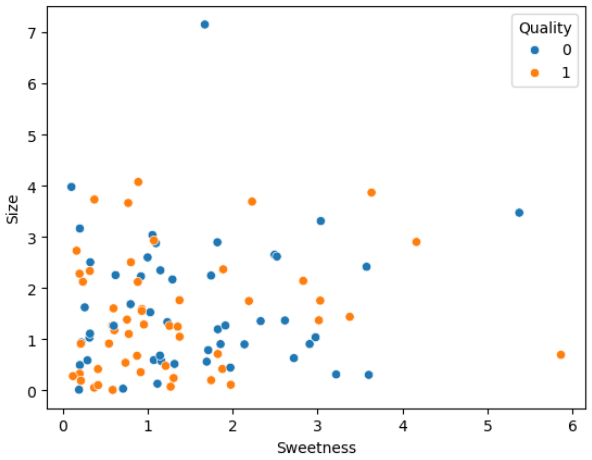
```
Out[42]: Text(0.5, 1.0, 'Correlation Heatmap')
```



Scatterplot:

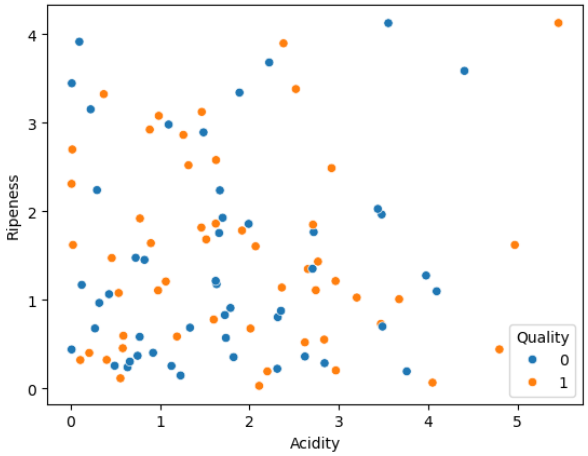
```
In [43]: # Sweetness vs Size
         sns.scatterplot(x='Sweetness', y='Size', hue='Quality',
                         data=df)
```

```
Out[43]: <Axes: xlabel='Sweetness', ylabel='Size'>
```



```
In [44]: # Acidity vs Ripeness
         sns.scatterplot(x='Acidity', y='Ripeness',
                         hue='Quality', data=df)
```

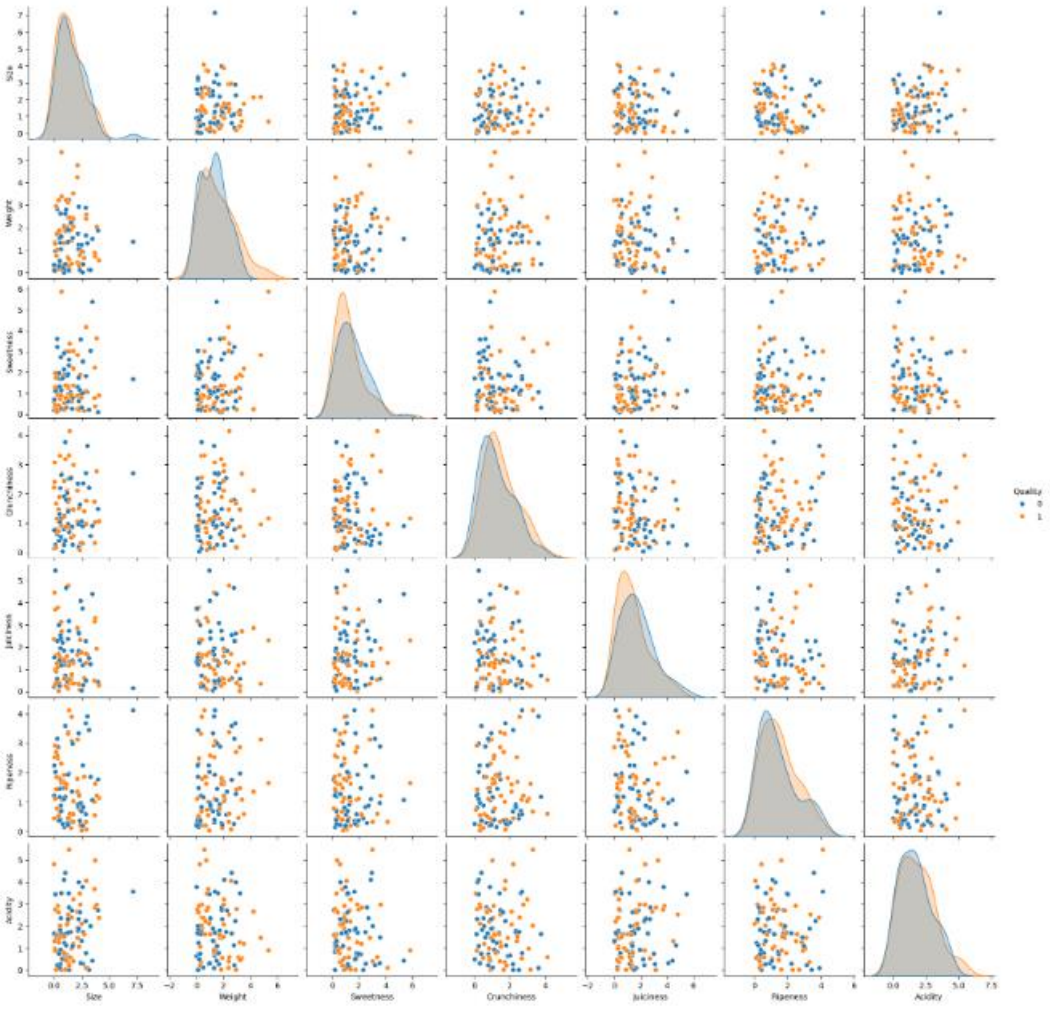
```
Out[44]: <Axes: xlabel='Acidity', ylabel='Ripeness'>
```



Pair plot:

```
In [46]: sns.pairplot(df.drop('A_id', axis=1), hue='Quality')
```

```
Out[46]: <seaborn.axisgrid.PairGrid at 0x1c6a4c652d0>
```



Splitting Dataset in features and target variables and exporting decision tree to the text representation:

All the other attributes are added to feature in x variable and quality as y variable.

In [47]:
```
#splitting dataset in features and target variables
feature_cols=['Size','Weight','Sweetness','Crunchiness','Juiciness','Ripeness','Acidity']
x=df[feature_cols] #all features
y=df.Quality
```

In [48]:
```
#Now splitting the dataset into training set and test set by using function train_test_splilit
#we need to pass the 3 parameters
x_test, x_train, y_test, y_train = train_test_split(x, y, test_size=0.3, random_state=1)
```

In [49]:
```
print("Training split input -",  x_train.shape)
print("Testing split input-", x_test.shape)
```

```
Training split input - (30, 7)
Testing split input- (70, 7)
```

In [17]:
```
#creating decision tree classifier object
clf= DecisionTreeClassifier()
```

In [18]:
```
#Train Decision Tree Classifier
clf = clf.fit(x_train, y_train)
```

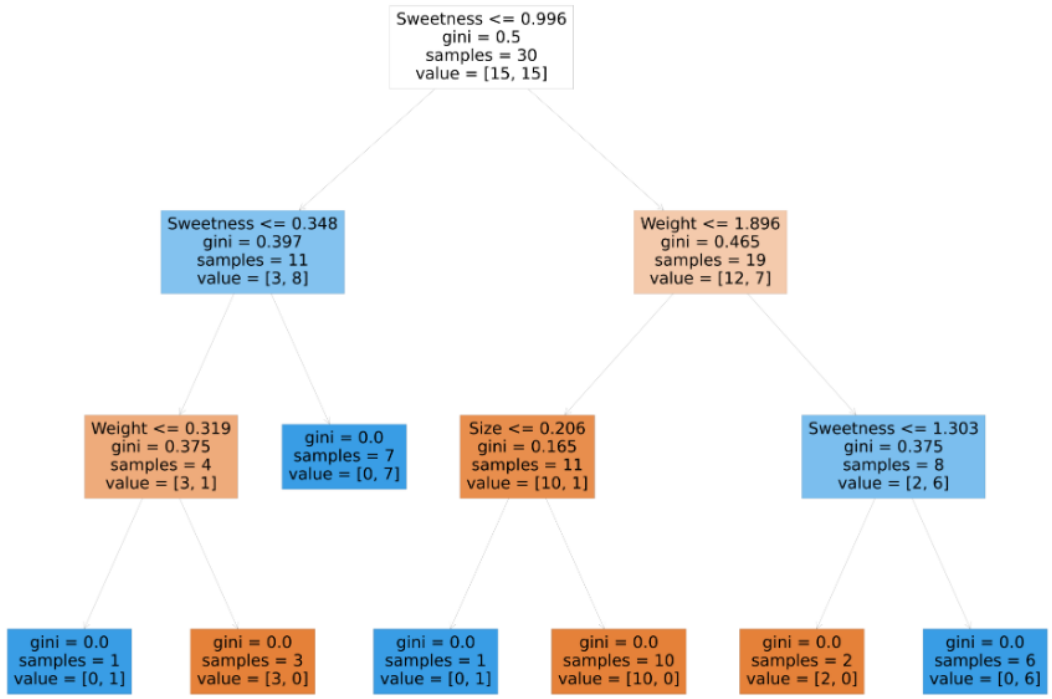
In [19]:
```
#Print Text Representation. Exporting Decision Tree to the text representation can be useful
test_representation = tree.export_text(clf)
print(test_representation)
```

```
|--- feature_2 <= 1.00
|   |--- feature_2 <= 0.35
|   |   |--- feature_1 <= 0.32
|   |   |   |--- class: 1
|   |   |--- feature_1 >  0.32
|   |   |   |--- class: 0
|   |--- feature_2 >  0.35
|   |   |--- class: 1
|--- feature_2 >  1.00
|   |--- feature_1 <= 1.90
|   |   |--- feature_0 <= 0.21
|   |   |   |--- class: 1
|   |   |--- feature_0 >  0.21
|   |   |   |--- class: 0
|   |--- feature_1 >  1.90
|   |   |--- feature_2 <= 1.30
|   |   |   |--- class: 0
|   |   |--- feature_2 >  1.30
|   |   |   |--- class: 1
```

Plotting the decision tree:

```
In [59]: #plotting the tree

fig = plt.figure(figsize=(200,150))
_ = tree.plot_tree(clf, feature_names = feature_cols, filled= True)
```



Predicting Accuracy :

In [50]:
```
print("Training split input -",  x_train.shape)
print("Testing split input-", x_test.shape)
print("Testing split output-", y_test.shape)
```

```
Training split input - (30, 7)
Testing split input- (70, 7)
Testing split output- (70,)
```

In [51]:
```
y_pred = clf.predict(x_test)
```

In [52]:
```
print(y_pred)
```

```
[0 0 0 1 0 1 1 1 1 0 1 0 0 1 0 1 0 0 1 0 1 0 1 0 1 0 0 1 1 0 0 1 0 0 1 0 1
 1 1 0 1 0 1 0 1 0 1 1 1 1 0 0 1 0 0 0 0 0 0 1 0 1 0 0 0 0 1 0 0 1]
```

In [53]:
```
print("Accuracy:",metrics.accuracy_score(y_test, y_pred))
```

```
Accuracy: 0.6142857142857143
```

In [54]:
```
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.2, random_state=1)
```

In [55]:
```
print("Training split input -",  x_train.shape)
print("Testing split input-", x_test.shape)
print("Testing split output-", y_test.shape)
```

```
Training split input - (80, 7)
Testing split input- (20, 7)
Testing split output- (20,)
```

In [56]:
```
y_pred = clf.predict(x_test)
```

In [57]:
```
print(y_pred)
```

```
[1 0 1 1 1 1 1 0 1 0 1 0 0 1 0 0 1 0 0 0]
```

In [58]:
```
print("Accuracy:",metrics.accuracy_score(y_test, y_pred))
```

```
Accuracy: 1.0
```

Video Link for the Video Demonstration:

Video is uploaded in the Moodle. Also, it is also uploaded in the drive which is given in the link below:

https://drive.google.com/drive/folders/1OJ1oOKDWTtUalCAt2qHkGi5XjSF0N7-L?usp=sharing

References

[1] Y. Hu *et al.*, "Building models to evaluate internal comprehensive quality of apples and predict storage time," *Infrared Phys Technol*, vol. 136, p. 105043, Jan. 2024, doi: 10.1016/J.INFRARED.2023.105043.

[2] "Fruit Quality Control Solution for Apples by Clarifruit." Accessed: Feb. 03, 2024. [Online]. Available: https://www.clarifruit.com/knowledge-base/fresh-produce-categories/apples/

[3] "Apple Quality." Accessed: Feb. 03, 2024. [Online]. Available: https://www.kaggle.com/datasets/nelgiriyewithana/apple-quality/data

[4] "Decision Tree - A Step-by-Step Guide." Accessed: Feb. 03, 2024. [Online]. Available: https://www.analyticsvidhya.com/blog/2021/08/decision-tree-algorithm/

[5] "Introduction to Seaborn - Python - GeeksforGeeks." Accessed: Feb. 03, 2024. [Online]. Available: https://www.geeksforgeeks.org/introduction-to-seaborn-python/