# Wine Production Quality and Satisfaction: Supervised Learning Methods

CSCI 5622: Machine Learning

**Author: Samuel Kwon**

Department of Computer Science
University of Colorado Boulder

10/27/22

# Introduction

The varieties of red and white wine are increasing every year. The increased production and sales of wine make it harder for consumers to purchase good quality wine. As a result, this project aims to explore wine datasets and consumer satisfaction through supervised machine learning methods. In this paper, decision trees, naive Bayes, and support vector machines will be employed on different datasets to explore the properties of classification.

Supervised learning requires labeled data that can be divided into training and testing sets. Generally, supervised learning models will be trained using the training set and the training labels. Then, label predictions will be made on the testing set without providing actual labels to compare the model's accuracy. Classification is a method of collecting and organizing data attributes into different classes. Decision tree is a method of visually representing the classes and decision-making related to the labels of the dataset. As the name implies, the plot of a decision tree is similar to a tree with roots, branches, and leaves. Naive Bayes is a classification method based on Bayes theorem. Naive Bayes uses the conditional probability algorithm to predict the labels of a given dataset. Support vector machines classify data points depending on the location of a hyperplane. When a dataset is plotted, a plane can be drawn that splits different data points with a margin. Support vector machines aim to optimize the plane kernel to split the dataset with a better maximum margin.

The Portugal wine dataset will be utilized in decision trees, naive Bayes, and support vector machines. In addition, the wine enthusiast dataset will be utilized in naive Bayes. The wine enthusiast is a text-heavy dataset; therefore, a text-based multinomial naive Bayes will be employed. Lastly, the chemical analysis dataset will be utilized in support vector machines. These methods will give insight into patterns in the dataset and if the set can be classified into different categories.

# Analyses

## Analysis: Decision Trees

Decision tree is a popular classification method because it is easy to read and analyze. In addition, there is some familiarity associated with the tree-like model. Decision trees will take the dataset and create the root, internal, and terminal nodes. Like an actual tree, the terminal nodes are called leaves, and the internal nodes are called branches. Splitting the tree happens recursively until the tree displays all of the information and no impurities are present. Generally, splitting can happen using two criterion based methods: GINI or entropy. Both measure the amount of information gained during each split. The GINI impurity index formula is represented below, where $p_i$ is the probability of being classified into a distinct class.

$$GINI = 1 - \sum_{i=1}^{C}(p_i)^2 \tag{1}$$

GINI measures the frequency of a feature being mislabelled when chosen randomly. The criterion-based methods often describes tree splitting as pure or impure. When the GINI index is zero, the node is pure, indicating that the node contains all of the elements in a single class. The best splitting using GINI occurs when the probability of each class is the same. Generally, a GINI around 0.5 indicates good tree splitting where distinct classes are present. Entropy is the second criterion based method. The entropy formula is represented below, where $p_i$ is the probability of class $i$.

$$Entropy = \sum_{i=1}^{C} -p_i * log_2(p_i) \tag{2}$$

The entropy definition is similar to the Boltzmann entropy equation ($S = klog(W)$), where comparisons of the amount of microstates in a macrostate are made, giving information about disorder and randomness. When splitting a tree using entropy, it is ideal to find the number of splits with less disorder. In order to evaluate the quality of each split, the information gain will quantify the amount of features present after the split. Information gain can be calculated by the difference in entropy or GINI before and after the split. The equation of information gain is presented below.

$$\text{Information Gain} = \text{Entropy(before)} - \sum_{j=1}^{K}\text{Entropy(j,after)} \tag{3}$$

$$\text{Information Gain} = \text{GINI(Before)} - \sum_{j=1}^{K}\text{GINI(j,after)} \tag{4}$$

After generating a tree, pruning the tree is important to improve readability and eliminate unnecessary splits. Examining the theoretical maximum depth of a tree is one way of preventing the overfitting of the dataset and pruning the tree.

This paper will employ decision trees on the Portugal wine dataset. A total of three different trees will be present. The first tree will be generated using R and optimizing the complexity parameter. The second tree will be generated using Python. Entropy will be the main splitting source. The third tree will also be generated using Python and GINI as the main method of splitting. In all three cases, optimization of the trees will be made to produce a more accurate tree.

### Analysis: Naive Bayes

The basis of the naive Bayes classification is the Bayes Theorem. Bayes theorem calculates the probability of event A occurring given information B. This classification method is applied to a dataset to predict the probability of the label type given information about the features. The utilization of Bayes theorem is called "naive" because two assumptions are made. First, the features are independent. Second, the features are equal. This assumption is naive because it is unrealistic to assume that none of the features have a relationship and that all the features contribute equally to the response. Naive Bayes is formulated below, where $X = x_1, x_2, x_3....x_n$ represents the features of the dataset or called predictors. This statement is similar to equation six .

$$P(y|X) = \frac{P(X|y)P(y)}{P(X)} \tag{5}$$

$$P(y|x_1, x_2....x_n) = \frac{P(x_1|y)P(x_2|y)....P(x_n|y)P(y)}{P(x_1)P(x_2)....P(x_n)} \tag{6}$$

Naive Bayes can be divided into three categories: Multinomial, Bernoulli, and Gaussian. Multinomial is generally used on discrete data or assigning text data to classes. The assumption is that the dataset follows a multinomial distribution. Bernoulli is similar to multinomial naive Bayes; however, it only predicts binary classes. Gaussian Naive Bayes assumes a continuous normal distribution. The predictors in a Gaussian distribution are continuous variables and cannot be discrete. One important feature of Naive Bayes is Laplace smoothing. This technique is utilized to avoid a zero probability and improve the fitting. The naive Bayes with Laplace is written below. Note that $C$ is the number of classes. When implementing Laplace smoothing, the probability will never be zero because of the +1 and $C$ terms.

$$P(A_i|C) = \frac{N_{ic} + 1}{N_c + C} \tag{7}$$

This paper will employ naive Bayes using the Portugal wine dataset and wine enthusiast dataset.

### Analysis: Support Vector Machines

Support vector machines transform the dataset into a higher dimensional space, so the features are classifiable by a hyperplane. Different kernels, such as linear, gamma, sigmoidal, radial, or polynomial, can be used to separate the dataset. Hyperplanes will decide the boundaries of the data points that can be classified into different groups. In addition to the hyperplane, a maximum margin is optimized to make predictions with higher accuracy. In other words, the margin is an area that considers the possible error associated with future prediction. Two types of margins can be generated: soft margin hyperplanes or hard margin hyperplanes. Soft margin hyperplanes occur when the constraint on maximizing the margin is less strict. Datasets that have some mixing with two classifications can be managed with a soft margin hyperplane. Hard margin hyperplanes are the opposite, where distinct boundaries are chosen with the margin and hyperplane. The dataset has clear classification boundaries, and no mixing between groups is present. The main goal of support vector machines is maximizing the margin, which is a quadratic convex optimization problem.

This paper will employ support vector machines using the Portugal wine dataset and chemical analysis dataset.

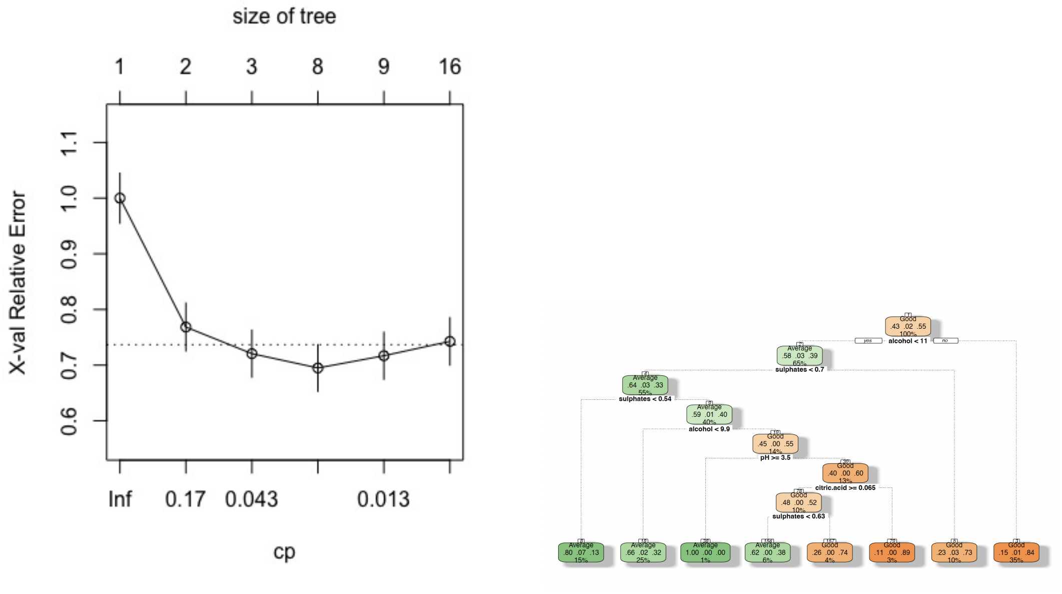
## Results

### Results: Decision Trees

The Portugal wine dataset decision trees were generated using R and Python. The predictors of the dataset are the chemical properties that are fixed acidity, volatile acidity, citric acid, residual sugar, chlorides, total sulfur dioxide, density, pH, sulphates, and alcohol. The label data is going to be the quality. It is important to note that the label data is discretized into "bad", "average", and "good" quality wine. The dataset used in unsupervised learning methods was used in this case. The original decision trees produced using Python and R are relatively large. Therefore, images of those trees will be presented in the appendix section.

Appendix section 1 represents an unpruned tree produced using R. The tree is relatively large, and it is hard to extract information. Furthermore, the label ”bad” is not present in the tree. The lack of the label ”bad” produces an unbalanced tree indicating that there could be some error during future predictions. Furthermore, the unbalanced tree could represent a lack of data. When summing the quality labels, there are 18 ”Bad”, 352 ”Average”, and 434 ”Good” labels. Just examining these values, it is evident that there are more ”Good” and ”Average” data points. Thus, the tree will perform well in predicting future ”Good” and ”Average” wines. However, the lack of data points representing ”Bad” wine provides limited training information for the decision tree algorithm. To mitigate this issue, the dataset can be sampled so that equal proportions of ”bad”, ”good”, and ”average” labels are present. However, this is a hard task since the amount of ”bad” data is very small, which could impact the overall learning process of the tree. As a result, the next time decision tree is performed on this dataset, it is important to set good boundary ranges for discretizing the labels.

Since the decision tree is relatively large, it is important to perform pruning to optimize the size. In R, the tree was pruned using the complexity parameter (CP).



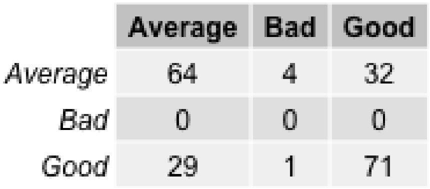
(a)

(b)

Figure 1: Complexity Parameter

Figure 1a represents the CP plot of the overall tree. When optimizing CP, choosing the CP value that produces the lowest error is key. The optimized CP value is 0.014706. After implementing the CP criterion, figure 1b shows the pruned tree. Comparing the original tree and the pruned tree, the size of the pruned tree is significantly smaller. A confusion matrix was calculated to evaluate the classification. The accuracy of the pruned tree is 63.16%, which is higher than the unpruned tree. Figure 2 is the confusion matrix.

Appendix section 3 and 5 represents the original GINI and entropy tree produced by Python. Similar to the results from R, the tree is large and has to be pruned. Pruning in Python was implemented by optimizing the GINI, entropy, and max depth. A for loop was implemented to test the accuracy associated with max depth



|  | Average | Bad | Good |
|---|---|---|---|
| *Average* | 64 | 4 | 32 |
| *Bad* | 0 | 0 | 0 |
| *Good* | 29 | 1 | 71 |

Figure 2: Complexity Parameter Optimization Confusion Matrix

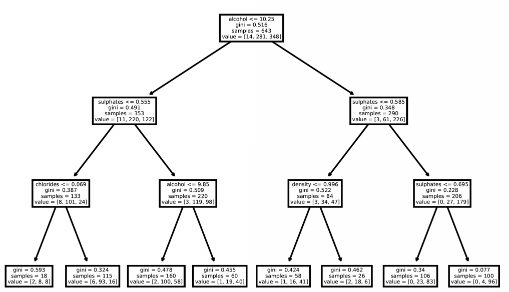
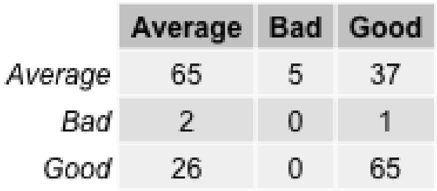


Figure 3: GINI Tree: Max Depth of Three

ranges from one to thirty. The max depth with the highest accuracy was chosen to prune the tree. Figure 3 represents the GINI decision tree with a max depth of three. This significantly reduces the size of the tree with an accuracy of 73.9%. The accuracy of the GINI tree with a max depth of three is higher than the accuracy produced using the complexity parameter. Appendix section 6 represents the optimized entropy tree with a max depth of 27. The highest accuracy when testing the max depth of the entropy tree is 73.2%. The accuracy values produced by entropy and GINI are nearly the same. However, the tree produced by GINI requires a lower max depth which improves interpretation of the dataset.

## Results: Naive Bayes

Naive Bayes was implemented on the Portugal wine dataset and the wine enthusiast dataset. It is important to note that the Portugal wine dataset is mixed data with some numerical and text data. The label in the Portugal wine dataset is quality, which is text data. The quality was discretized similarly to the dataset used for decision tree. Gaussian Naive Bayes was employed on the Portugal wine dataset. In order to prevent overfitting and a zero probability, Laplace smoothing was also included in the computation. The accuracy of this method was 64%. Figure 4 represents the confusion matrix of the Gaussian Naive Bayes. The same simulation was run using Python. However, the quality column was converted to numeric values where 0 represents "bad", 1 represents "average", and 2 represents "good". The accuracy of the Gaussian Naive Bayes in Python is 67.7%, which is nearly the same as the accuracy produced using R. In order to further extend the applications of naive Bayes on this dataset, a multinomial naive Bayes was implemented in Python using the same dataset. The accuracy of this model is 59.6%, which is significantly lower than the accuracy produced using Gaussian Naive Bayes. As a result, when comparing the two naive Bayes methods, the Gaussian Naive Bayes method produced the highest accuracy rate. The Portugal wine dataset is relatively large; therefore, it is safe to assume a normal distribution.



| | Average | Bad | Good |
|---|---|---|---|
| *Average* | 65 | 5 | 37 |
| *Bad* | 2 | 0 | 1 |
| *Good* | 26 | 0 | 65 |

Figure 4: Confusion Matrix for Gaussian Naive Bayes (Portugal Wine Dataset)

Besides the Portugal wine dataset, multinomial naive Bayes was implemented on the wine enthusiast dataset. The wine enthusiast dataset is mixed data; however, most of it is text data. The dataset includes a description of the wine, country of origin, province of origin, price, grape varietal, and wine points. Text-based naive Bayes was run with the description as the predictor and points as the label. The first step in this process is to manage the text data. In this set, the description is presented in chunks of text. As a result, it is important to clean the description by converting text into lowercase, removing numbers, stop words, punctuation, and any extra spaces. After this step is performed, the description is tokenized. The text-based multinomial naive Bayes will review the words produced in each description to predict the wine point. The accuracy of this model is around 43.77%. Since the accuracy is lower than 50%, multinomial naive Bayes is not a good way of classifying this dataset.

### Results: Support Vector Machines

Support vector machines were employed on the chemical analysis and Portugal wine datasets. Similarly to decision trees and naive Bayes, the label for the Portugal wine dataset is quality, which is represented in terms of discrete numerical values. Linear, radial, and the sigmoidal kernels were utilized on both datasets. The support vector machine never converged when running the linear kernel on the Portugal wine dataset. Furthermore, changing the cost variable from 10 to 20 didn't change the convergence pattern. Based on this result, it is safe to assume that the linear kernel is not adequate for classifying the Portugal wine dataset. The radial and sigmoidal kernels converged for the Portugal wine dataset using a cost parameter of 10. The accuracy using the radial kernel is 60.86%, and the sigmoidal kernel is 52.17%. Both kernels do not have a high accuracy rate. When the cost parameter of the radial kernel increased to 20, the accuracy of the model decreases slightly. Furthermore, when the sigmoidal kernel is used with the cost parameter at 20, the accuracy decreases to around 48.44%.

The chemical analysis dataset features are Malic Acid, Ash, Ash Alcanity, Magnesium, Total Phenols, Flavanoids, Nonflavanoid Phenols, Proanthocyanins, Color Intensity, Hue, OD280, and Proline. The label for this dataset is Alcohol. The label set was discretized where if the Alcohol level is greater than or equal to 13.4, it is classified as ”High”. If the alcohol level is less than 13.4, the wine is classified as ”Average”. Linear, sigmoidal, and radial kernels were utilized. The accuracy using the linear kernel with a cost parameter of one is 83.33%. As the cost parameter increases when using the linear kernel, the accuracy of the model decreases. The accuracy using the radial kernel with a cost parameter of 10 is 46.22%. When increasing the cost parameter to 20, the accuracy of the model increases to around 60%. The accuracy using the sigmoidal kernel with a cost parameter of 10 is 44.44%. When the cost parameter is increased to 20 for the sigmoidal kernel, the accuracy of the model jumps to around 50%. Considering this information, the linear kernel does a good job at classifying the chemical analysis dataset since it has the highest accuracy.

## Conclusion

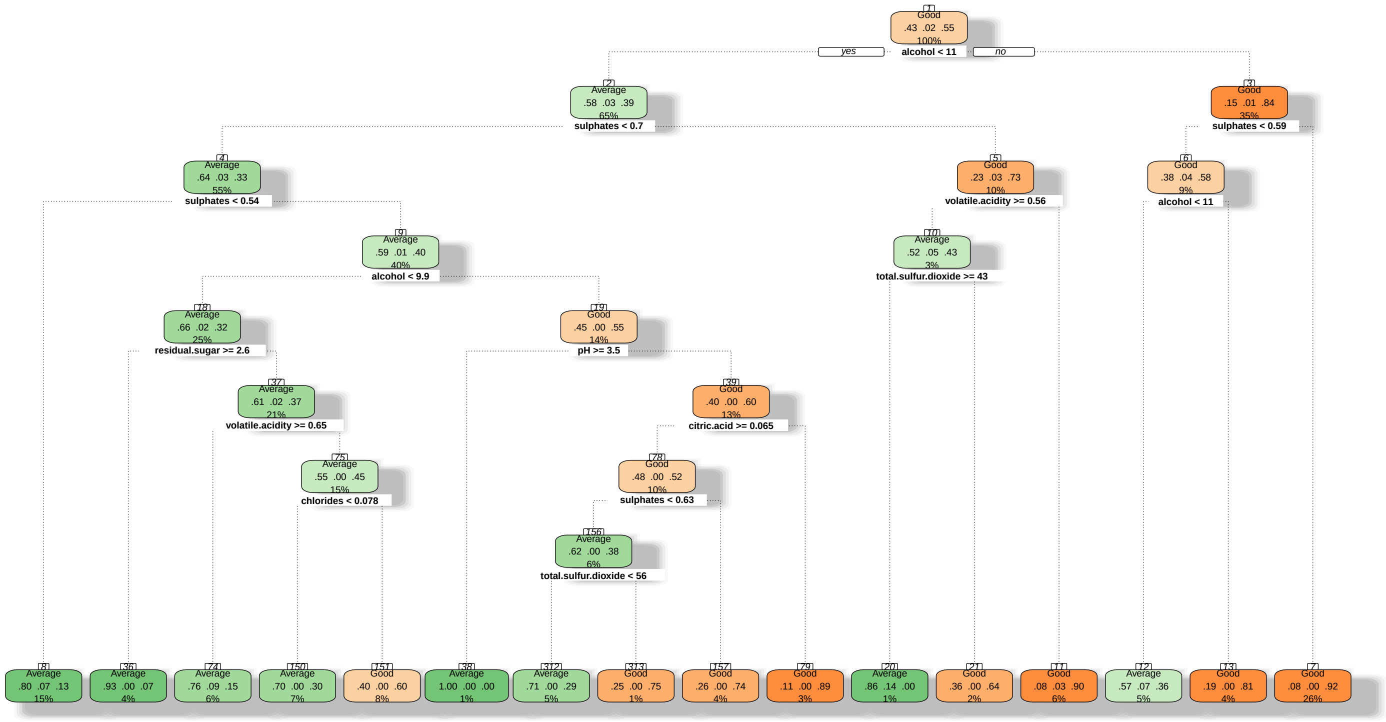
Three supervised learning methods were employed to explore the classification properties in the Portugal wine, chemical analysis, and wine enthusiast datasets. The decision tree for the Portugal wine dataset was produced using R and Python. Furthermore, the trees were pruned using complexity parameter, GINI, and entropy. When analyzing these trees, the optimized GINI tree produced the best results. The size of the optimized GINI tree is relatively small, and the accuracy is greater than the trees produced when optimizing complexity parameter and entropy. The Gaussian Naive Bayes produced the best results for the Portugal wine dataset. The accuracy of the model was much greater than the multinomial naive Bayes. Besides using numerical data for naive Bayes, the text-based multinomial naive Bayes performed on the wine enthusiast dataset has an accuracy lower than 50%. Lastly, when performing support vector machines on the Portugal wine and chemical analysis dataset, it is clear that the sigmoidal kernel performs the worst. Furthermore, the linear kernel performs the best for the chemical analysis data, while the radial kernel works the best for the Portugal wine dataset. It is important to note that the cost parameter changes the model's accuracy. However, the changes are minimal for some models.

Through supervised learning methods, important features regarding wine datasets were discovered. When running the multinomial naive Bayes on the wine enthusiast dataset, the model's accuracy was lower than 50%, indicating that it would perform poorly during future classification. This result highlights that it might take much work to rank wine accurately by points given the description of the wine. Because taste is very biased, finding some relationship between wine description and points is challenging. When looking at the overall accuracy of each supervised model, the values seem to remain around the 60%-80% range. The accuracy of these models is relatively low. This can be managed by finding alternative methods for discretizing the labels.

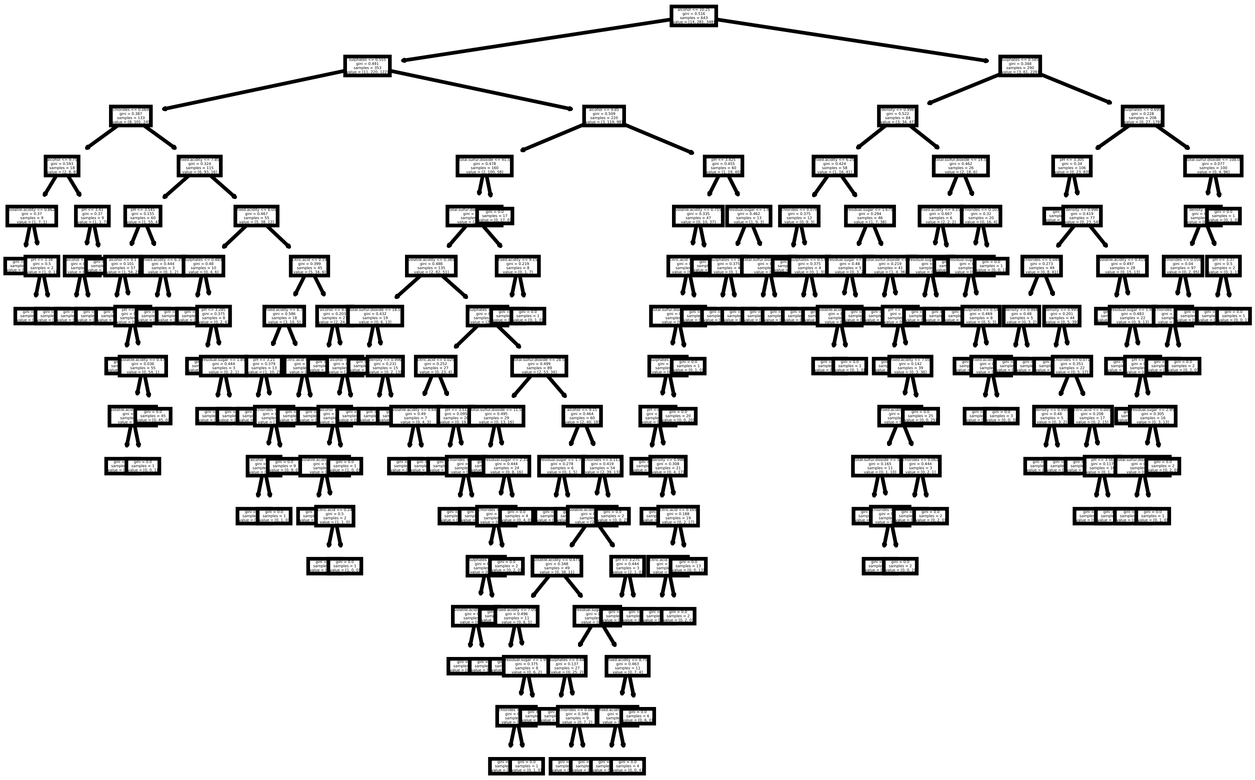
In this paper, three datasets were utilized to perform supervised learning methods. For future research, it would be interesting to see how the results would change if there were a more efficient way to discretize label data. The wine enthusiast dataset includes wine from all over the world. In the United States, wine quality is not regulated. However, many European countries, such as Italy, regulate the wine quality by classifying them at different quality levels. It would be interesting to see how the results from supervised learning would change if text data regarding regulated wine were utilized.

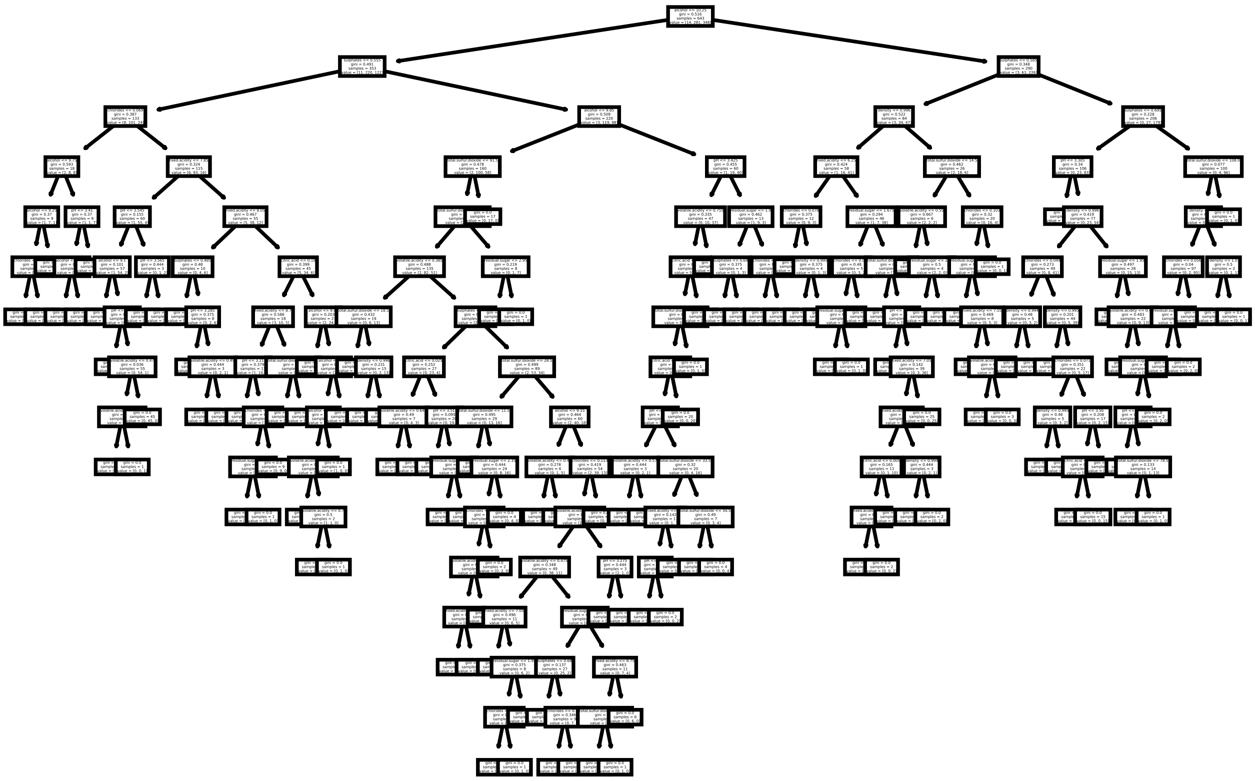
# Appendix

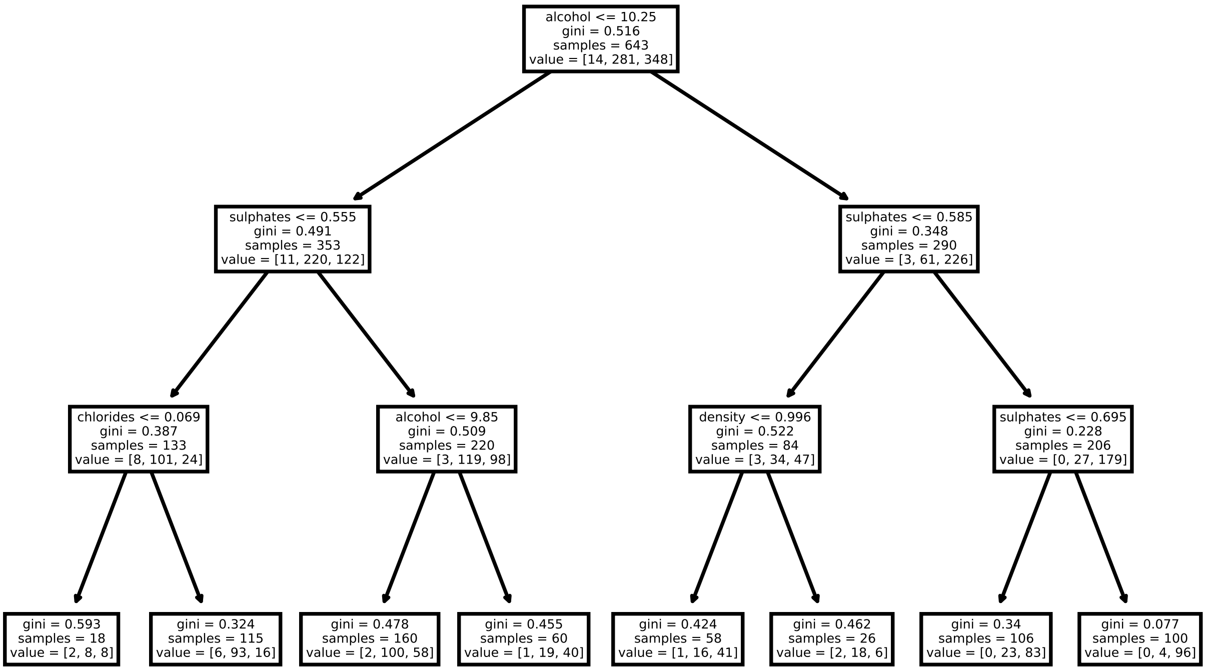
Reference Below for Decision Tree Plots.

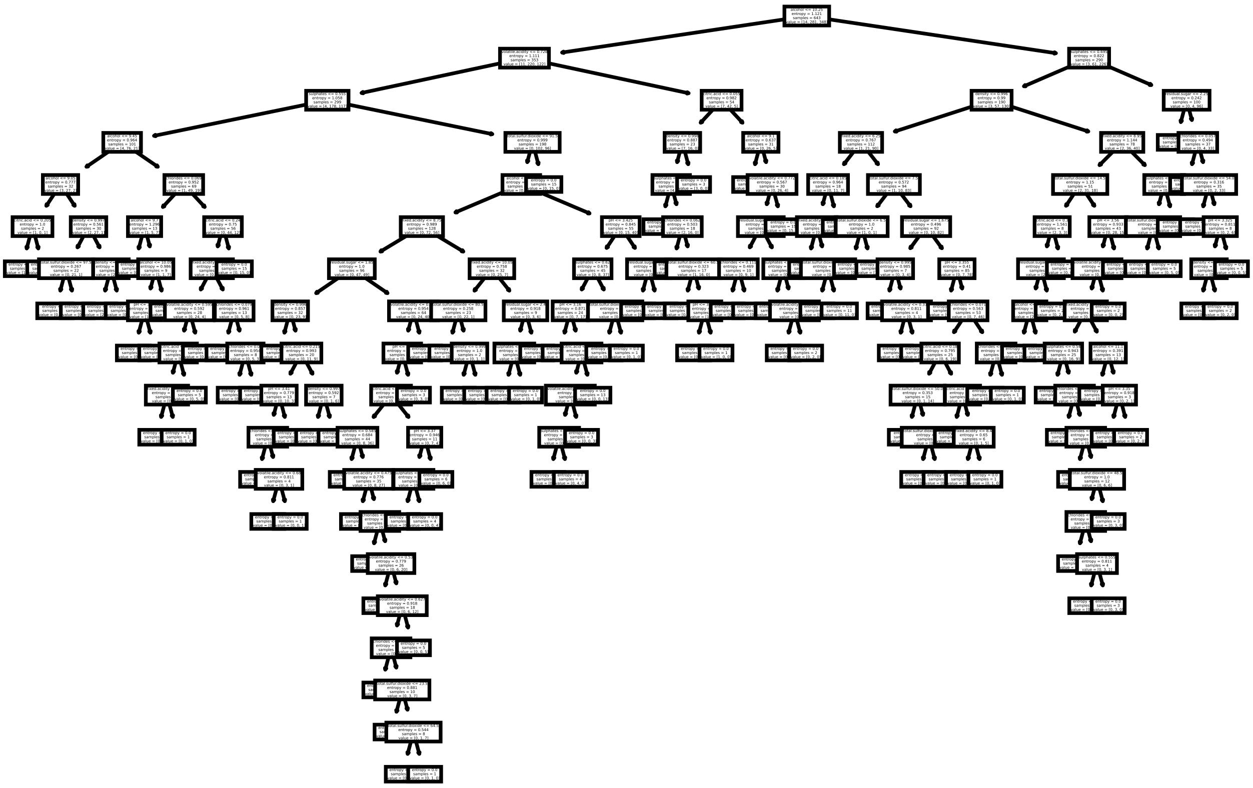


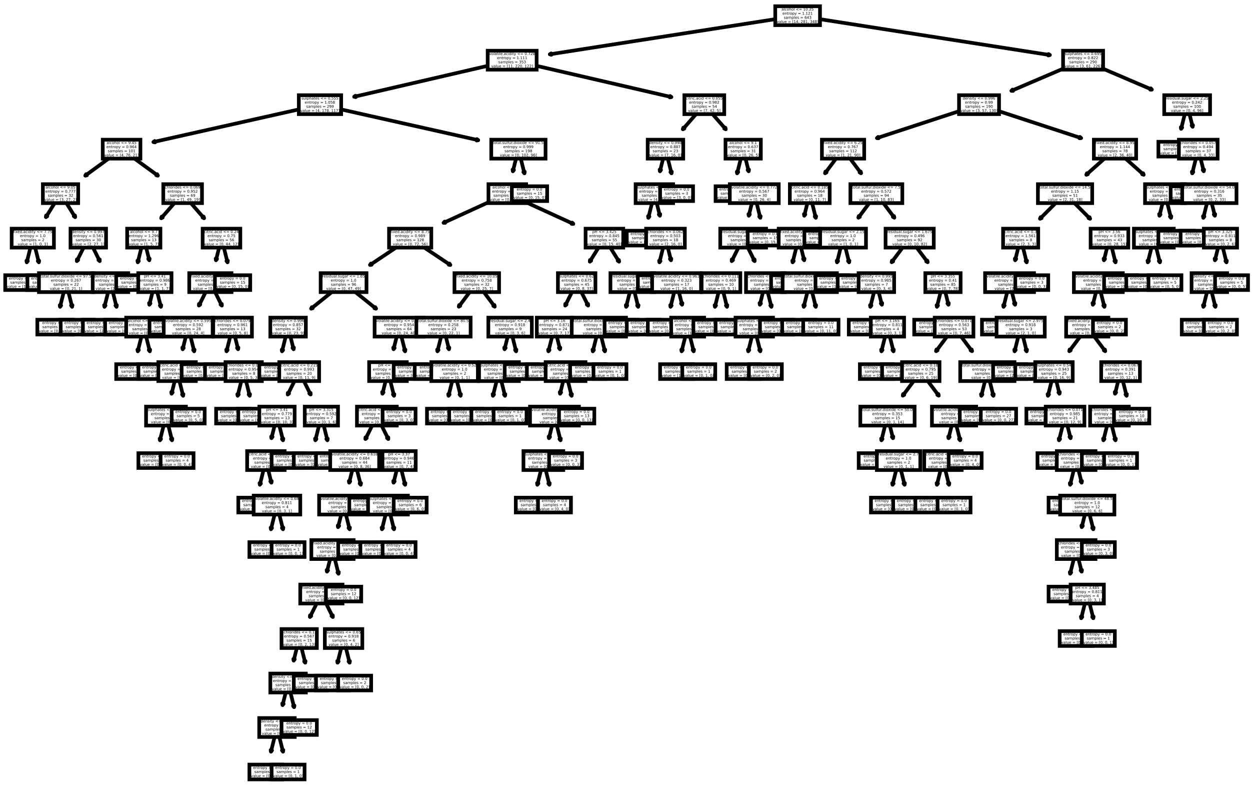
Rattle 2022−Oct−26 11:38:19 samuelkwon


alcohol <= 10.25
gini = 0.516
samples = 643
value = [14, 281, 348]
sulphates <= 0.555
gini = 0.491
samples = 353
value = [11, 220, 122]
sulphates <= 0.585
gini = 0.348
samples = 290
value = [3, 61, 226]
chlorides <= 0.069
gini = 0.387
samples = 133
value = [8, 101, 24]
alcohol <= 9.85
gini = 0.509
samples = 220
value = [3, 119, 98]
density <= 0.996
gini = 0.522
samples = 84
value = [3, 34, 47]
sulphates <= 0.695
gini = 0.228
samples = 206
value = [0, 27, 179]
alcohol <= 9.75
gini = 0.593
samples = 18
value = [2, 8, 8]
fixed.acidity <= 7.85
gini = 0.324
samples = 115
value = [6, 93, 16]
total.sulfur.dioxide <= 91.5
gini = 0.478
samples = 160
value = [2, 100, 58]
pH <= 3.425
gini = 0.455
samples = 60
value = [1, 19, 40]
fixed.acidity <= 6.25
gini = 0.424
samples = 58
value = [1, 16, 41]
total.sulfur.dioxide <= 14.5
gini = 0.462
samples = 26
value = [2, 18, 6]
pH <= 3.305
gini = 0.34
samples = 106
value = [0, 23, 83]
total.sulfur.dioxide <= 108.0
gini = 0.077
samples = 100
value = [0, 4, 96]


alcohol <= 10.25
gini = 0.516
samples = 643
value = [14, 281, 348]
sulphates <= 0.555
gini = 0.491
samples = 353
value = [11, 220, 122]
sulphates <= 0.585
gini = 0.348
samples = 290
value = [3, 61, 226]
chlorides <= 0.069
gini = 0.387
samples = 133
value = [8, 101, 24]
alcohol <= 9.85
gini = 0.509
samples = 220
value = [3, 119, 98]
density <= 0.996
gini = 0.522
samples = 84
value = [3, 34, 47]
sulphates <= 0.695
gini = 0.228
samples = 206
value = [0, 27, 179]
pH <= 3.41
gini = 0.37
samples = 9
value = [1, 1, 7]
pH <= 3.545
gini = 0.155
samples = 60
value = [1, 55, 4]
residual.sugar <= 1.675
gini = 0.294
samples = 46
value = [1, 7, 38]
volatile.acidity <= 0.55
gini = 0.667
samples = 6
value = [2, 2, 2]


alcohol <= 10.25
gini = 0.516
samples = 643
value = [14, 281, 348]
sulphates <= 0.555
gini = 0.491
samples = 353
value = [11, 220, 122]
sulphates <= 0.585
gini = 0.348
samples = 290
value = [3, 61, 226]
chlorides <= 0.069
gini = 0.387
samples = 133
value = [8, 101, 24]
alcohol <= 9.85
gini = 0.509
samples = 220
value = [3, 119, 98]
density <= 0.996
gini = 0.522
samples = 84
value = [3, 34, 47]
sulphates <= 0.695
gini = 0.228
samples = 206
value = [0, 27, 179]
gini = 0.593
samples = 18
value = [2, 8, 8]
gini = 0.324
samples = 115
value = [6, 93, 16]
gini = 0.478
samples = 160
value = [2, 100, 58]
gini = 0.455
samples = 60
value = [1, 19, 40]
gini = 0.424
samples = 58
value = [1, 16, 41]
gini = 0.462
samples = 26
value = [2, 18, 6]
gini = 0.34
samples = 106
value = [0, 23, 83]
gini = 0.077
samples = 100
value = [0, 4, 96]


alcohol <= 10.25
entropy = 1.121
samples = 643
value = [14, 281, 348]
volatile.acidity <= 0.728
entropy = 1.111
samples = 353
value = [11, 220, 122]
sulphates <= 0.695
entropy = 0.822
samples = 290
value = [3, 61, 226]
citric.acid <= 0.055
entropy = 0.982
samples = 54
value = [7, 42, 5]
density <= 0.996
entropy = 0.99
samples = 190
value = [3, 57, 130]
total.sulfur.dioxide <= 91.5
entropy = 0.999
samples = 198
value = [0, 102, 96]
density <= 0.998
entropy = 0.887
samples = 23
value = [7, 16, 0]
alcohol <= 9.1
entropy = 0.637
samples = 31
value = [0, 26, 5]
fixed.acidity <= 6.25
entropy = 0.767
samples = 112
value = [1, 21, 90]
fixed.acidity <= 8.95
entropy = 1.144
samples = 78
value = [2, 36, 40]


alcohol <= 10.25
entropy = 1.121
samples = 643
value = [14, 281, 348]
volatile.acidity <= 0.728
entropy = 1.111
samples = 353
value = [11, 220, 122]
sulphates <= 0.695
entropy = 0.822
samples = 290
value = [3, 61, 226]
total.sulfur.dioxide <= 91.5
entropy = 0.999
samples = 198
value = [0, 102, 96]
fixed.acidity <= 8.95
entropy = 1.144
samples = 78
value = [2, 36, 40]