MTH 369 Final Project

Payton Reaver

2022-12-12

Contents

Overview

In this analysis, I will be using the Random Forest Algorithm to try and predict the quality of red wines based on a number of physical and chemical properties.

Research Question(s)

The question this study aimed to answer was:

- Which properties of wine are best when trying to predict the quality?

Abstract

With this study, it found that alcohol, volatile acidity and sulphate levels have the most predictive power for the quality of wine. Using the Random Forest Algorithm (ntree = 100, mtry = 4) on all of the predictor variables, a success rate of 85.5% was observed.

Data Introduction/Visualization

This data set contains 1599 different variants of the Portuguese red wine "Vinho Verde". Each wine was measured on multiple different properties such as: Acidity, density, pH, alcohol, and many others.

Data Table

Below is an output of the first ten observations with each variable being measured listed along the columns.

```
head(wine, n = 10)
```

```
## # A tibble: 10 x 13
##    fixed~1 volat~2 citri~3 resid~4 chlor~5 free_~6 total~7 density    pH sulph~8
##      <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl> <dbl>   <dbl>
##  1     7.4    0.7     0        1.9   0.076      11      34   0.998  3.51    0.56
##  2     7.8    0.88    0        2.6   0.098      25      67   0.997  3.2     0.68
##  3     7.8    0.76    0.04     2.3   0.092      15      54   0.997  3.26    0.65
##  4    11.2    0.28    0.56     1.9   0.075      17      60   0.998  3.16    0.58
##  5     7.4    0.7     0        1.9   0.076      11      34   0.998  3.51    0.56
##  6     7.4    0.66    0        1.8   0.075      13      40   0.998  3.51    0.56
##  7     7.9    0.6     0.06     1.6   0.069      15      59   0.996  3.3     0.46
##  8     7.3    0.65    0        1.2   0.065      15      21   0.995  3.39    0.47
##  9     7.8    0.58    0.02     2     0.073       9      18   0.997  3.36    0.57
## 10     7.5    0.5     0.36     6.1   0.071      17     102   0.998  3.35    0.8
## # ... with 3 more variables: alcohol <dbl>, rating <dbl>, quality <fct>, and
## #   abbreviated variable names 1: fixed_acidity, 2: volatile_acidity,
## #   3: citric_acid, 4: residual_sugar, 5: chlorides, 6: free_sulfur_dioxide,
## #   7: total_sulfur_dioxide, 8: sulphates
```

With each red wine observed, there exist a rating that ranges from 0-10. The higher the rating, the "better" the wine is said to be. For this study, each observation will be classified into being either "Bad", "Normal", or "Good". This is based on the corresponding rating that each wine received. The column labeled 'quality' will measure which classification each wine falls into.

The classification is as follows:

- rating < 5: Bad
- rating $= 5$ or rating $= 6$: Normal
- rating > 6: Good

This classification may seem random by just reading a table. Below is the distribution of 'rating' from this data set.

```
ggplot(wine, aes(x = rating)) +
  geom_bar(color = "#99E5E5", fill = "#00F7F7") +
  labs(x = "Rating", y = "Count") +
  theme_classic()
```

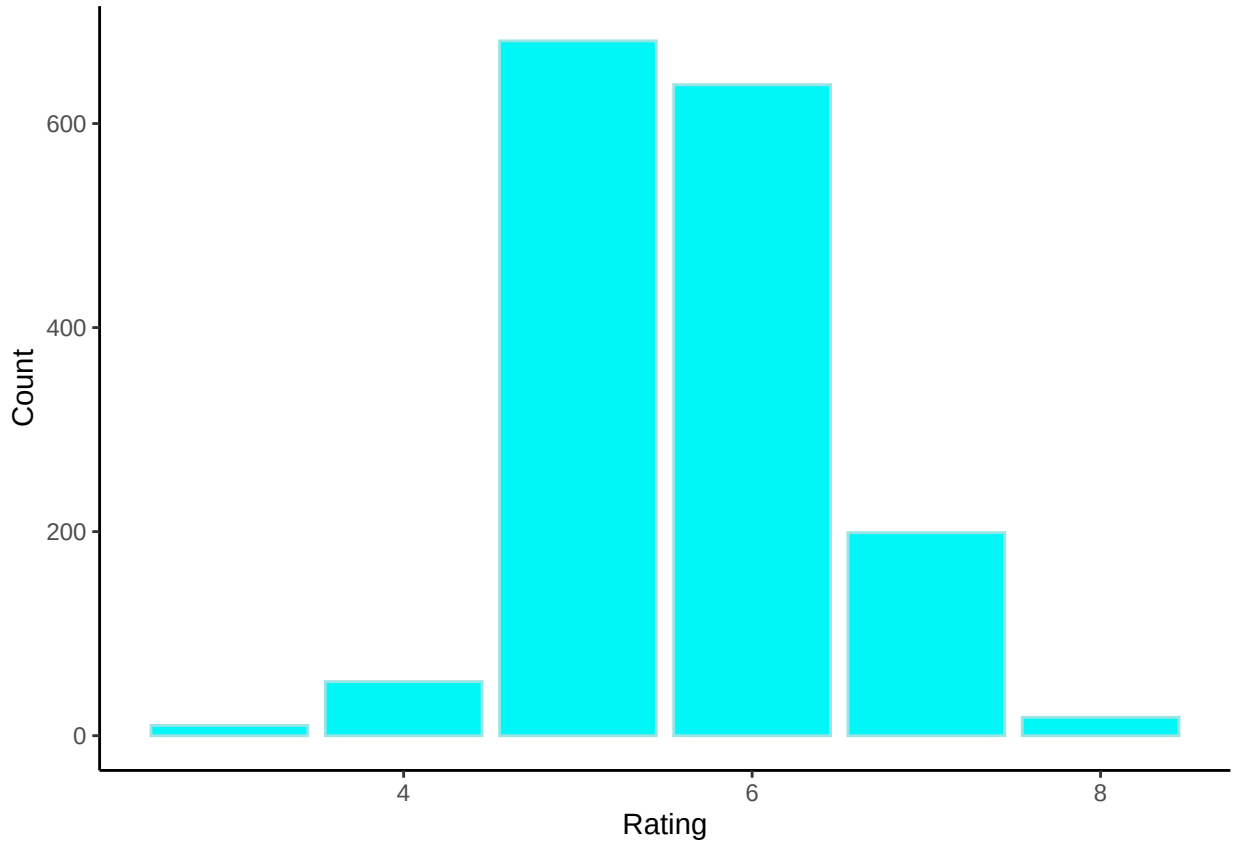

Figure 1: Distribution of Rating

After analyzing the distribution, we can see a very large concentration of observations within the rating interval of 5 and 6. Very few observations lay outside of this interval. So, an inference one could make is that it is rare to receive a rating above 6, and even rarer to receive a rating below 5.

Boxplots

Here we will use graphical summaries to make inferences about which variables we think will be useful when it comes time to make a model.

```
ggplot(wine, aes(x = reorder(quality, volatile_acidity), y = volatile_acidity)) +
  geom_boxplot(color = "#9bf09b", fill = "#c8f7c8") +
  theme_classic() +
  labs(x = "Quality" , y = "Volatile Acidity Level")
```

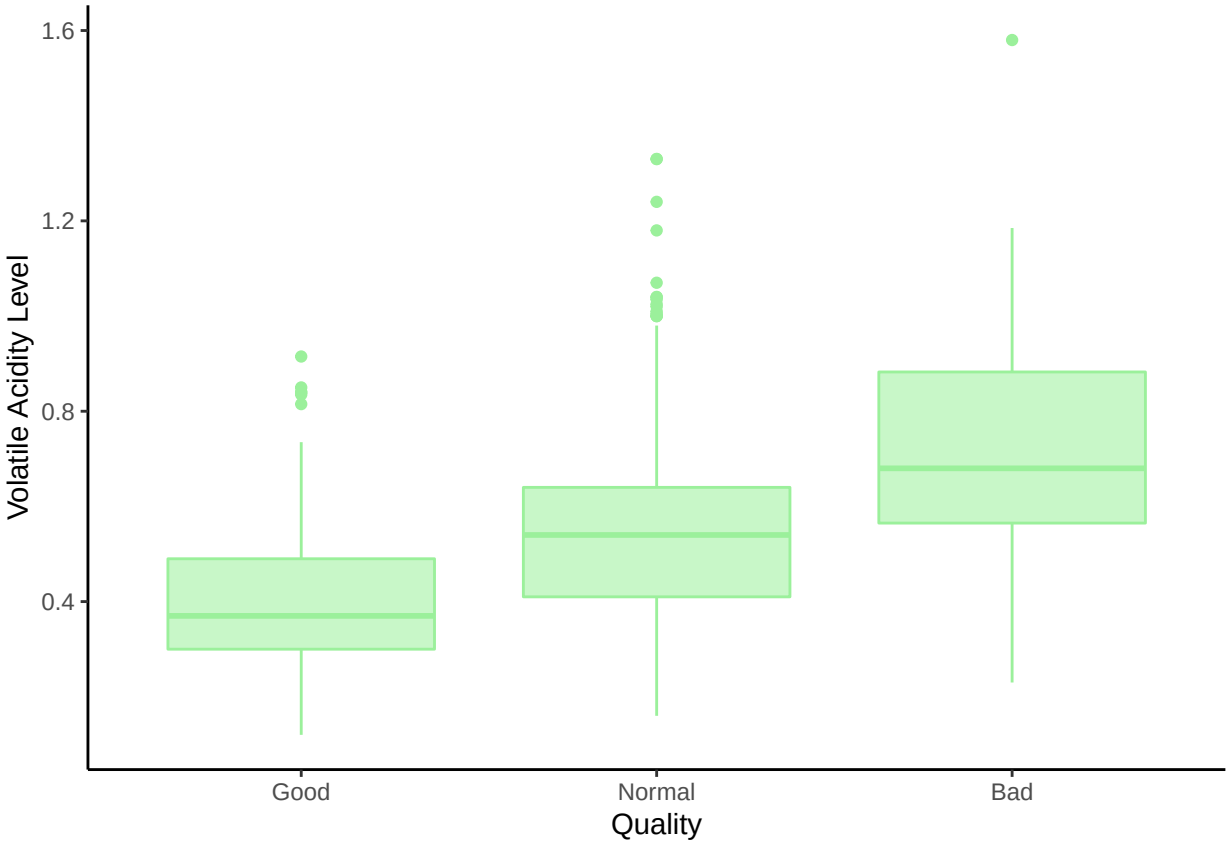

Figure 2: Boxplot Distribution of Volatile Acidity vs. Quality

Based off of the graphical display, it is clear that there is a relationship between the level of volatile acidity and the classification of the wine. It seems the increase of volatile acidity causes the rating to decrease (hence, quality is decreasing towards 'Bad'), thus a negative relationship is displayed.

Further, notice how the two interquartile ranges of the 'Good' and 'Bad' classification are entirely disjoint. This is suggesting that the two group means (μ_1, μ_2) for volatile acidity at each classification 'Good' and 'Bad' are different.

```r
ggplot(wine, aes(x = reorder(quality, sulphates), y = sulphates)) +
  geom_boxplot(color = "#ff7e82", fill = "#ffc8c9") +
  theme_classic() +
  labs(x = "Quality", y = "Sulphate Level")
```

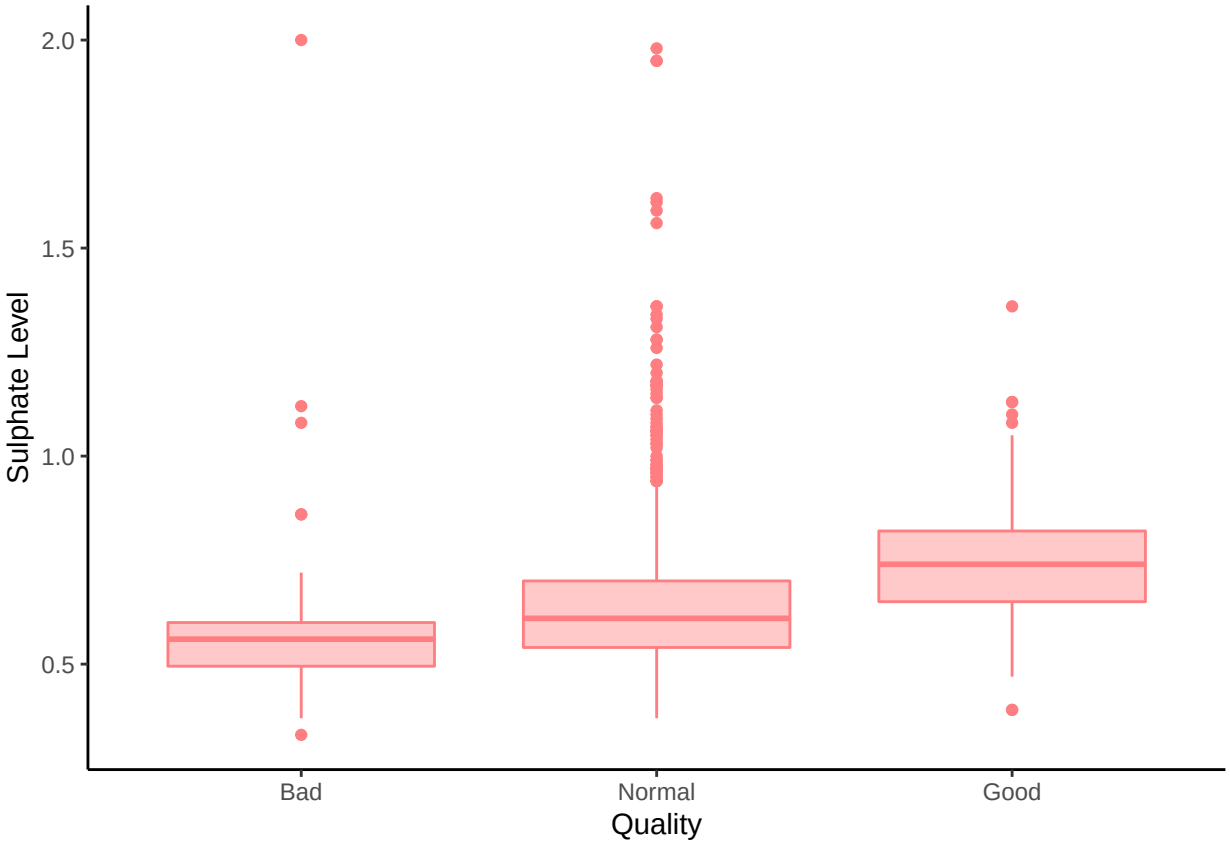

Figure 3: Boxplot Distribution of Sulphate Level vs. Quality

By inspection, the relationship between each wine classification based on the sulphate level behaves oppositely in comparison to Volatile Acidity vs. Quality. The relationship between the sulphate level and the quality is positive.

Another note about this distribution is the number of outliers in this data set. The 'Normal' classification holds the majority of all the outliers, but this can be explained by 'Normal' also having the most number of observations.

```
ggplot(wine, aes(x = reorder(quality, alcohol), y = alcohol)) +
  geom_boxplot(color = "#cc92f8", fill = "#e5c9fb") +
  theme_classic() +
  labs(x = "Quality", y = "Alcohol Level")
```

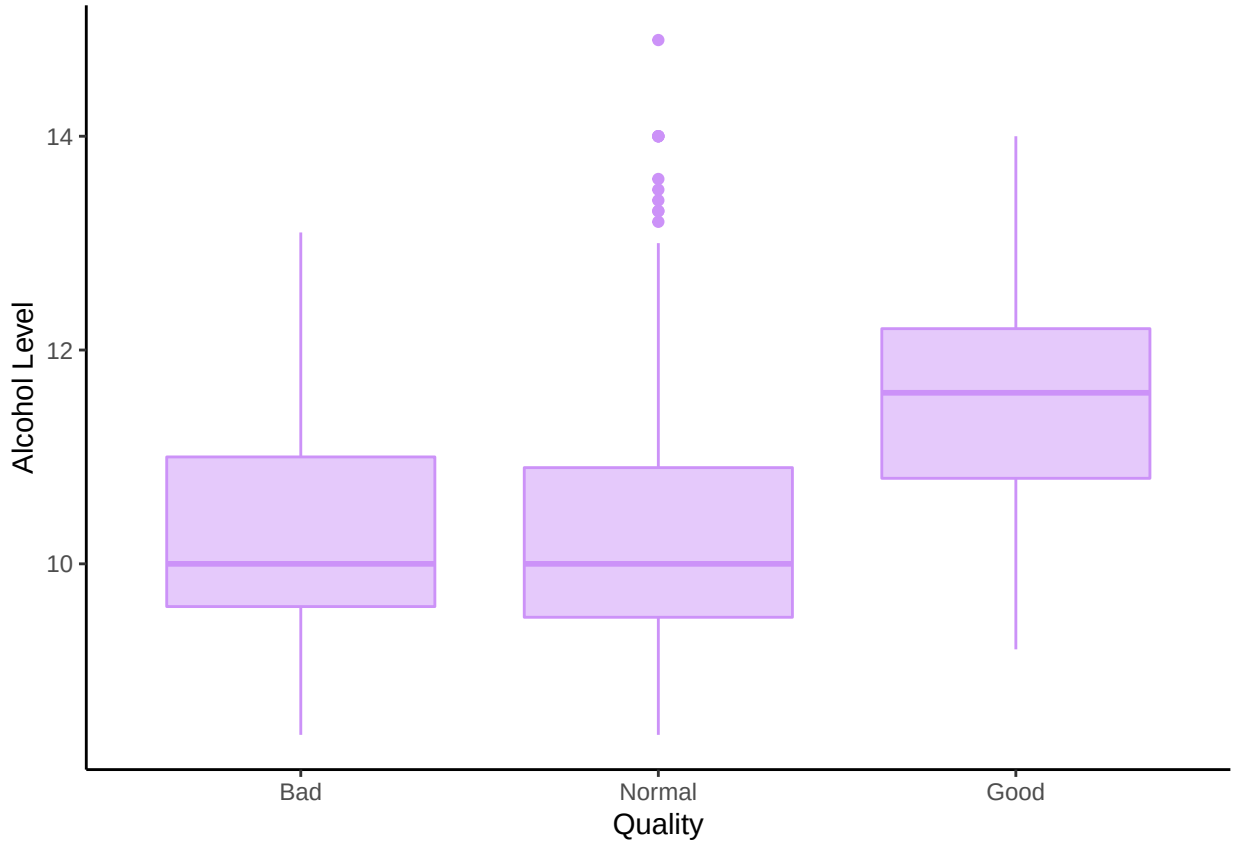

Figure 4: Boxplot Distribution of Alcohol Level vs. Quality

Contrary to Figures 1 and 2, it is not as obvious the kind of relationship alcohol level and the quality share. What we can infer is that there is a difference in group means between 'Good' wines and 'Bad' or 'Normal' wines. The differences between 'Bad' and 'Normal' wines in comparison to the alcohol level is minimal. We can see that the 'Normal' classification has all of the outliers in this distribution.

Random Forest

What is it?

The expectation is that we all know what a forest is, if not; a forest is an area or region that is dominated by trees. The machine learning algorithm Random Forest is defined in a similar manner. However, rather than just regular trees like oak or pine, Random Forest specializes in decision trees.

Decision Trees

Decision trees are vital in the understanding of the Random Forest algorithm. Decision trees take any number of qualitative or quantitative variables from your data set and explores the data in a random fashion that boils down to a question on: Which classification does this observation belong to?

Shown below is an example:

Let $A = \{\sigma_i : i \in P, P \in \mathbb{N}\}$ be your sample space of observations for a given training set of data.

Then a decision tree that randomly selected two variables say volatile acidity and alcohol level would look like this:

Note : Each sample space need not contain $|P|$ unique observations (selected with replacement), it suffices to preserve magnitude. This selection of observations is also random and not a coincidence.

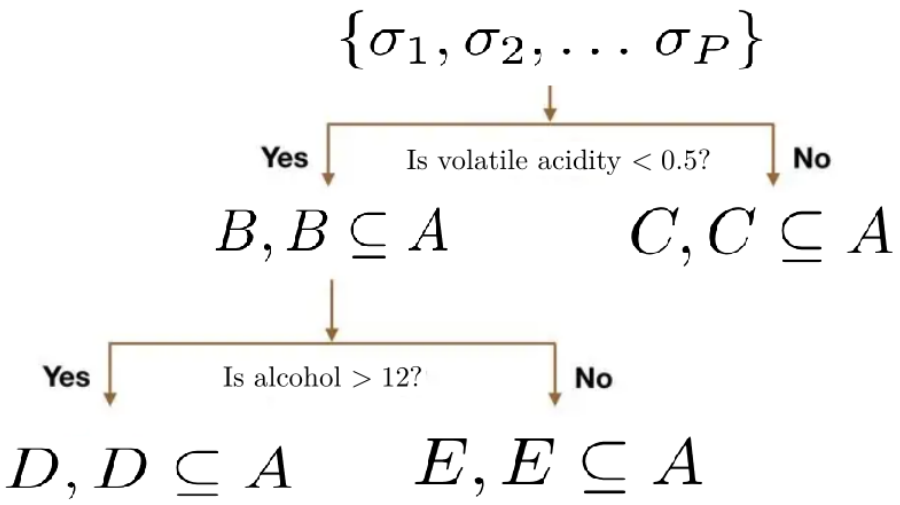

Figure 5: Decision Tree

Such that $|E| + |D| + |C| + |B| = |A|$.

The randomly chosen variables volatile acidity and alcohol represent this decision tree's nodes. The number of nodes per decision tree can vary depending on the goals of each analysis, the only limitation is how many variables you have in your data set.

The classification question we are 'boiling down' to is what quality each wine observed is. This is how a decision tree works within a Random Forest Model, but on a much larger scale. So, as the name entails, a Random Forest is a collection of a bunch of random decision trees.

How does it work?

There are two key ideas to understand that will help motivate/convince why the Random Forest Algorithm works.

- There must be a 'tell' or 'signal' within the variables selected that help models predict better than just arbitrary guessing.
- Each prediction made by each tree must have low correlation with one another.

This first bullet is rather obvious. The outcome of our algorithm will be more favorable the more explanatory power each variables has. Using variables that otherwise generate random noise when compared to some dependent variable will not be as helpful.

So, how do you ensure each decision tree has low correlation with one another? Bagging.

Bagging (Bootstrap Aggregation)

Earlier, it was stated that when it comes to filling the nodes (variables) and sample space (observations), the process is random. Now is where this fact shines.

Because the process in which we are selecting variables and observations is random, each of our decision trees has the capacity to be wildly different (different in shape and outcome). Now, having just one or two decision trees may not capture all of the unique features a data set has; and those two trees have the chance of being identical, which again, does not help us. Therefore, the more decision trees we have, the more features the model is able to capture and in turn the stronger the predictive power becomes. Think of it as having strength in numbers. This process is known as bagging.

This is the driving force in the Random Forest Algorithm and ensures that our trees have little correlation with one another.

In short, the Random Forest Classification Algorithm is made up of many decision trees that use randomness and bagging to come together as a congregated forest of uncorrelated trees. Time to apply what we have learned.

Training a Model

Model Specification

Before we can start training a model, we must first create a training data set.

I will create an index variable defined by a binomial distribution with a probability of success, $p = 0.7$. This will allow the study to continue with as little bias as possible. The result is shown below.

```
head(wine_rf[8:13], n = 10)
```

```
## # A tibble: 10 x 6
##    density    pH sulphates alcohol quality index
##      <dbl> <dbl>     <dbl>   <dbl> <fct>   <int>
##  1   0.998  3.51      0.56     9.4 Normal      1
##  2   0.997  3.2       0.68     9.8 Normal      1
##  3   0.997  3.26      0.65     9.8 Normal      1
##  4   0.998  3.16      0.58     9.8 Normal      1
##  5   0.998  3.51      0.56     9.4 Normal      1
##  6   0.998  3.51      0.56     9.4 Normal      1
```

```
##  7   0.996  3.3       0.46     9.4 Normal      0
##  8   0.995  3.39      0.47    10   Good        0
##  9   0.997  3.36      0.57     9.5 Good        1
## 10   0.998  3.35      0.8     10.5 Normal      1
```

I will now separate this larger data set into training and test data. The training set will contain every observation such that index = 1. Likewise, the test set will contain every observation that satisfies index = 0. So, on average, the split will be 70-30 between the two data sets.

Now we are ready to train our model.

```
rf.model <- randomForest(quality ~ .,
                         ntree = 100, mtry = 4, data = train)
```

Here is the input for our Random Forest model with 100 decision trees each with 4 nodes. The classification is based off quality, and the pool of variables the algorithm is allowed to pull from is denoted with '.', or the entire model. The data set we will be using is the 'train' data frame we just created.

Interpretation of Results

```
rf.model
```

```
##
## Call:
##  randomForest(formula = quality ~ ., data = train, ntree = 100,      mtry = 4)
##                Type of random forest: classification
##                      Number of trees: 100
## No. of variables tried at each split: 4
##
##         OOB estimate of  error rate: 13.06%
## Confusion matrix:
##        Bad Good Normal class.error
## Bad      3    0     40  0.93023256
## Good     0   72     72  0.50000000
## Normal   2   28    870  0.03333333
```

Following the call, we can see the distinguishing features of the model. Below that are two very important characteristics to our Random Forest Model. Specifically the OOB (Out of Bag) estimate of error rate and the Confusion Matrix.

The OOB estimate of error rate corresponds to the estimated error associated with this trained model on unseen data. Or, I should expect about a (100− OOB)% success rate on predicting the quality of wine with my test data set. Or, specifically with this model, we should expect a success rate around 87%.

The confusion matrix is a way of showing how many classifications were misplaced, and if so, where the model incorrectly placed each observation. Hence, confusion matrix. The row classification is what the correct label was, the corresponding column classification displays how many were placed in that label. In other words, the diagonal shows the number of correctly placed observations while all other entries show all incorrectly placed observations.

Note : When our model comes across a wine that is classified as 'Bad', almost every single observation was placed as 'Normal'.

Variable Significance

A very reasonable question to ask is: Which independent variables provide the most predictive power? To that, the Mean Decrease Gini must be studied.

```
round(importance(rf.model), 2)
```

```
##                       MeanDecreaseGini
## fixed_acidity                    24.33
## volatile_acidity                 38.62
## citric_acid                      24.45
## residual_sugar                   24.70
## chlorides                        22.54
## free_sulfur_dioxide              21.00
## total_sulfur_dioxide             27.29
## density                          26.59
## pH                               21.46
## sulphates                        39.88
## alcohol                          51.80
```

Shown above is the Mean Decrease in Gini for all of the variables used in the model. Each number is a measure of how important a variable is for predicting the classification variable across all of the decision trees that make up the forest. The higher the Mean Decrease in Gini implies the greater the importance that variable is.

Shown below is the same information as a scatter plot.

```
varImpPlot(rf.model)
```

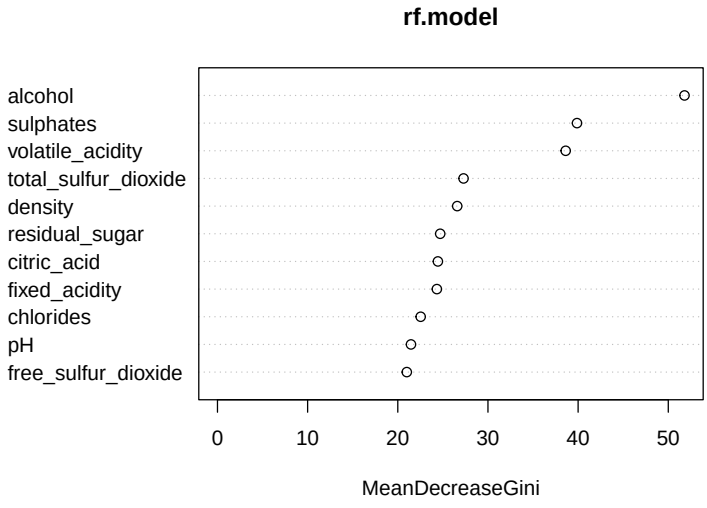

Figure 6: Variable Importance Plot

Here, it is clear to see that alcohol carries the most importance when predicting quality while sulphates and volatile acidity are close seconds.

Testing the Trained Model

Predicting Quality

We have trained our model, and we have a baseline expectation of what kind of results to expect, so we are ready to test our models accuracy.

```
pred <- predict(rf.model, newdata = test)
```

All that needs to be done to start predicting is to feed our Random Forest Model the test data set and record the results.

Accuracy/Precision of Model

Below is the output summary of the model's prediction results.

```
table(pred, test2$quality)
```

```
##
## pred     Bad Good Normal
##   Bad      1    0      2
##   Good     0   34     11
##   Normal  19   39    406
```

We interpret this model the same way we interpret the Confusion Matrix from our training data set. To get our prediction accuracy we simply add up the numbers of the diagonal and divide that by the total sum of all entries, which is just the number of observations in the testing set. Thus, the corresponding success rate from this model is: $\frac{(2+37+399)}{(512)} = \frac{438}{512} = 0.855$.

Conclusion/Results

From this study, we found that alcohol, sulphate, and volatile acidity levels have the most predicting power on the overall quality of wine ($OOB_{alc} = 51.52, OOB_s = 38.67, OOB_{v.a} = 38.02$). Splitting the original data set into a 70-30 split (Training-Test respectively), the trained model was able to correctly predict the quality of wine on the test set at a success rate of 85.5%.

Limitations/Improvement

To further the study, one could analyze a Random Forest Model just based on the top n predictor variables based on the Mean Decrease in Gini score. Then, you could compare the two models (Full scope model vs. Best predictor model).

Another way to improve this study is the presence of cross-validation. Cross-validating the results of the Random Forest Algorithm ensures that what you are actually observing is not by chance. I.e., validate the model further.