Quality White Wine Machine Learning Algorithm

Ajay Bhatia
ajbhatia@ucsc.edu

Introduction

The data set that I was given consists of physicochemical properties of a selection of Portuguese Vinho Verde wine, with columns to represent the volatile acidity, alcohol content, citric acid, free sulfur dioxide, and pH of each wine. Additionally, the data indicated whether or not the wine is a white wine of high quality. The goal of this project is to create a machine learning algorithm that is able to identify whether the wine is a quality white wine based on the other data points. My data was set up to be slightly skewed, with a large amount of not high-quality white wines, meaning that there only a few data points that are label as quality white wines. This creates an issue in our algorithm, as it allows for points with similar features to be classified similarly, even if one is a quality white and one is not. My data was set up as volatile acidity, alcohol content, citric acid, and free sulfur dioxide being objects, pH being float64 values, and quality white being a Boolean value. This meant that I had to change the data types into ones that are more easily readable for my models. The models I used for this project were a linear regression model, with a log-loss function, a k-Nearest Neighbors model, and a Decision Tree model.

Data Cleaning

My data included many different errata that I had to change or remove in order for the data to be ready for my models. Initially, I remove all rows that contained any kind of missing value. For example, if a wine did not have a reported volatile acidity, that wine would then be removed from the data set. This ensured that I would avoid iterating over missing data that could mess up the model. Once that was done, I had to change the data types of volatile acidity, alcohol content, citric acid, and free sulfur dioxide columns, as they were all objects that contained text. Text is not iterable by any kind of model, so I extracted the numbers from the column and removed any text that was attached to the number. This allowed me to change the data type of these columns into float64 values, which are easily iterable by models. With that, all of the columns except for quality white were float64s, so to change Boolean values to be an int64 value, I used one-hot encoding, where all True values were replaced with 1s and all False values were replaced by 0s. Through these processes, all values in my data frame were float64 values and int64 values, ensuring that there would be no issues when using such values in models. Lastly, my data contained an outlier, whose row needed to be removed entirely so that it would avoid influencing the model output. To do this, I found the z-scores of each item in the data frame compared to the mean of its column. The z-score represents the possible variation of the mean, so if the z-score is above three, then that indicates that the number is outside of 99.7% of possible values. If a number is outside of this percentile, it is highly likely that it is an outlier, and thus should be dropped from the data set. Using this method, I was able to detect one outlier in my data set and remove it entirely. While going through this process I did have to make some choices about what

data type my information should be in so that it worked well with the models. Originally, I wanted to keep my quality white data in Boolean form, but I would have needed to go through specific processes while cleaning my data to ensure that it worked properly. For example, I would have needed to make my outlier checker skip the quality white column, otherwise, it would have had an error for iterating over a Boolean. Another choice that I had to make was if I should have checked the data for duplicate entries. While trying to implement things to do this, I found how difficult it was for this to work properly since none of the data entries were given names. This meant that with standard tools to find duplication, values that had the same number as another value somewhere else in the set were considered duplicates and it was difficult to tell if they were duplicate entries or just a different, but similar, wine. Other than that, there were not many choices that I had to make for this data set in order to clean it. There were small adjustments here and there in order to make sure my cleaning worked properly, but it was just standard data cleaning that ensured there were no missing values, no values of the wrong type, no outliers, and that the data was standardized and normalized.

Data Visualization

In order to find relationships between two variables, my feature values and my label, I created different correlation plots that depicted relationships between the different values in the sets. Initially, I created a correlation matrix that helped me find the correlations between each column in respect to the other columns. This correlation Matrix is depicted in Figure 1. Figure 1 shows us, in color, which columns are most correlated with other columns, depicting any and all relationships that are found within my data set. The model shows us a few relationships, such as pH's correlation with volatile acidity or the negative correlation between citric acid and volatile acidity. The main correlations that I focused on were those that were tied to quality white. Alcohol and free sulfur dioxide had the highest correlations with a wine being a quality white wine. As a note, it is expected that the yellow line goes through the middle as each column should be correlated with itself entirely, and yellow dictates the highest possible correlation.

Figure 1

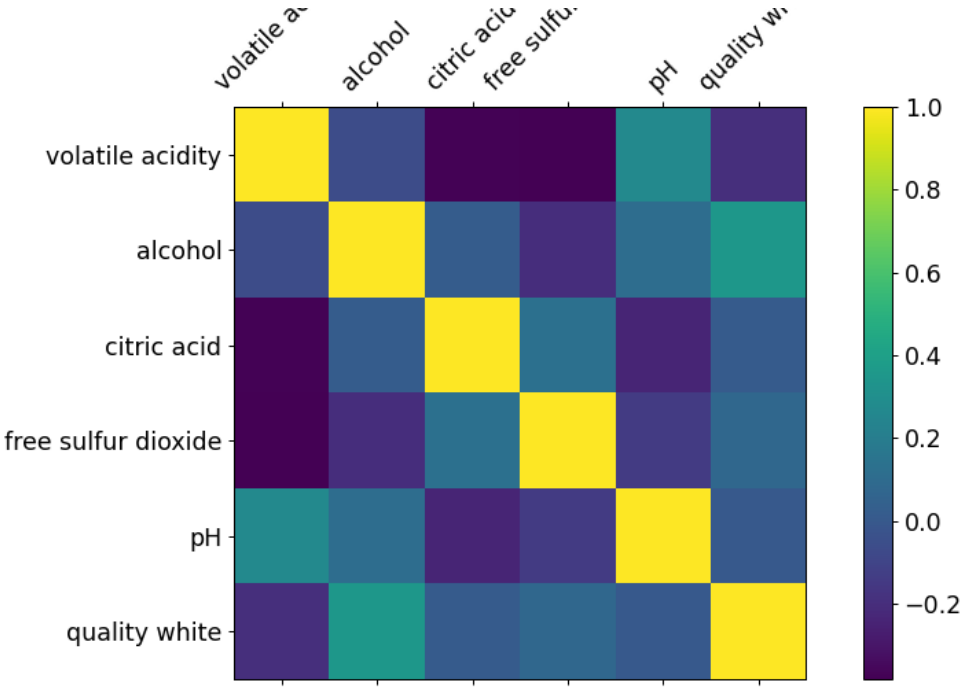

Figure 1: Correlation matrix for each column in the data frame

The next thing I did to ensure the correlations were correct and made sense, was to create a scatterplot that plotted a feature value against whether or not the wine is a quality white wine. Plotting something of this sort would make it extremely easy to tell which correlations were the most positive correlations and which features I should be focusing on for my model. For me, this was incredibly difficult to do. With a binary label, all scatter plots were extremely hard to graph, as all values were either at y = 1 or y = 0. This created graphs similar to Figure 2, where the correlation between the feature values and the label are difficult to see. Figure 2 shows use the scatter plot of citric acid against the quality white label, but it is hard to see if this graph actually indicates anything, as the Y values are all tied to either 1 or 0 and are unable to depict a clear correlation.

Figure 2

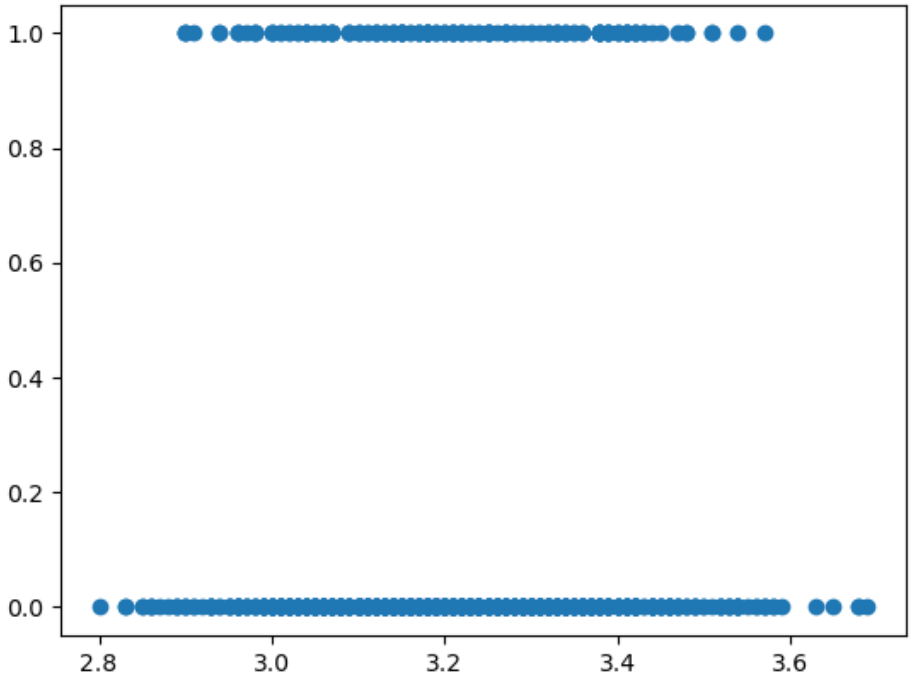

Figure 2: Feature value plotted against the binary label, quality white

In order to get around an issue such as this, I instead created a line of best fit for each scatterplot, which helped me understand the correlations. This is displayed in Figure 3, which shows the correlation line between a feature value and the binary label. Instead of plotting the scatterplot, I used the information from the scatterplot in order to create lines of best fit for each feature value. This then helped showed me which correlations were negative, which were positive, and which were the greatest in relation to quality white wine. The first graph depicts the negative correlation between volatile acidity and quality white, with a value of around -0.19. The next graph shows us the relationship between alcohol and quality white, which has a positive correlation and a value around 0.3. Next, citric acid is plotted against quality white and shows us the positive correlation of value 0.01. Next was free sulfur dioxide plotted against quality white, which showed a positive correlation and a value of 0.07. Lastly, pH showed us a value of 0.002 when plotted against quality white, but still with a positive correlation. This means that the two feature values with the highest correlation to quality white were alcohol and free sulfur dioxide. In order to tell the exact values of the correlation with each feature value, I used the df.corrwith function, but the graph depicts each item in a similar way, by using the y-axis to show how large the correlation is between the two items. Those with the largest y-axes were alcohol and free sulfur dioxide. As a note, I was unable to form the axes like I wanted to with matplotlib, as only the bottom graph had altered axes when using the plt.ylim function. Despite this, we are still able

to see clear correlations between the feature values and our binary label, and which feature values had the largest correlations.

Figure 3

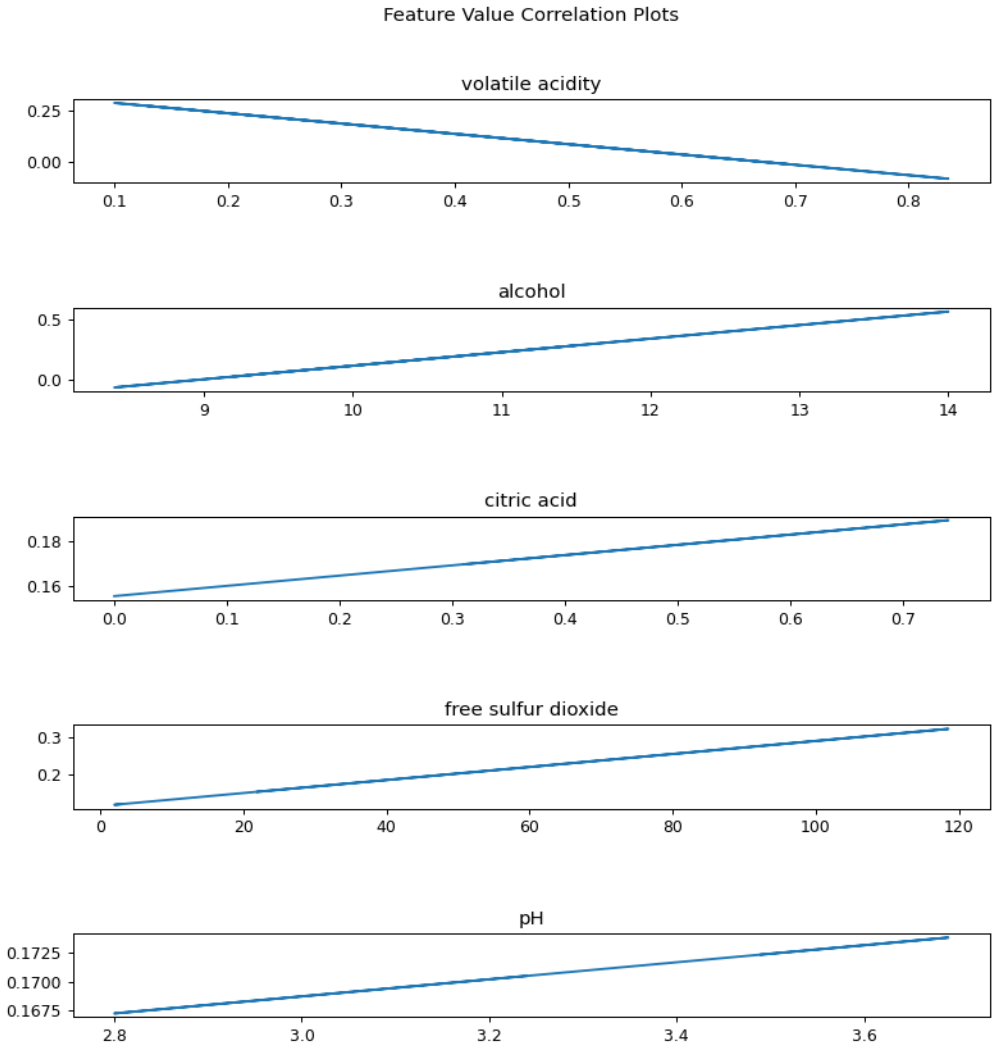

Figure 3: Correlation plots of feature values against the binary label, quality white

Using the information from the correlation matrix, depicted in Figure 1, and the correlation plots, depicted in Figure 3, I was able to find the two feature values that are more correlated with our binary label, alcohol and free sulfur dioxide. Thus, I dropped all feature value columns in order to ensure that I only trained my models on the two feature values that had the highest correlation to my label.

Modeling

For my models, I chose to use a Linear Regression model, with a log-loss function, a K-nearest neighbors model, and a Decision tree model. The results of each model were highly successful, as shown through k-fold validation. My linear regression model had a mean accuracy of 88.9%, my k-Nearest Neighbors model had a mean accuracy of 89.4%, and my Decision Tree model had a mean accuracy of 89.2%. All accuracies were incredibly high for my models. In order to make sure that my models were statistically significant, I used calculated a p-value to find the probability that the F1 scores would be the same for each model. These scored proved that my models were correct, as none of the percentages were under 5%, and thus they are all statistically significant. To find my max_depth for my Decision Tree model and n_neighbors for my k-Nearest Neighbors model, I had to test a lot of different points to find which best fit the data. I eventually settled on using max_depth = 5 and n_neighbors = 8 because they resulted in the highest k-fold validation scores. Additionally, I had many issues with specific values that I had set for max_depth that caused my Decision Tree model to fail completely and output no information. Finally, I was able to conclude that the k-Nearest Neighbors model returned the highest mean accuracy for my data compared to other models, but the Decision Tree was very similar in terms of mean accuracy. Additionally, it is incredibly likely that the two models will have the same exact F1 score when looking at data, so the difference between the two is minimal.

Table 1: Models and their accuracy

K = 5	Mean Accuracy in %	The probability that the F1 Scores are the same as other models in %
Linear Regression	88.9	Linear and Tree: 78.0 Linear and kNN: 78.9
k-Nearest Neighbors	89.4	kNN and Linear: 78.9 kNN and Tree: 91.5
Decision Tree	89.2	Tree and Linear: 78.0 Tree and kNN: 91.5

Analysis

The main relationships that I noticed were the correlations between quality white and alcohol and free sulfur dioxide. This was shown explicitly through the models I used and their results on the test data. In order to maximize the impact of these relationships, I dropped all other feature variables so that the only columns that were taken into account by the model were alcohol and free sulfur dioxide and how they apply to the label, quality white. I believed that removing these other features helped better my model accuracy.

I applied many different concepts from this class to this project in order to complete it. Mainly, I used the eight steps to clean data, importing data, merging sets, rebuilding missing data,

standardization, normalization, deduplication, verification, and exporting the data. I focused on a few of these eight steps to ensure that my data was cleaned properly. Mainly, removing data, standardization, and normalization. Additionally, when training my models, the concept of overfitting was clear to me so that I could ensure my model would not be incredibly complex. This motivated me to find the middle ground between a complex model that would overfit my training data and a model that would be complex enough to accurately represent my data with high accuracy. Lastly, the talk of using P-values in order to determine if our models were statistically significant is something that I was able to apply to this project. Since P-values depict whether or not we should reject or accept a null hypothesis, I used this metric to see if my models were performing the way they were depending on random chance. I ended up finding that the opposite was true, and the models were statistically significant because of how they compared to my other models. None of my p-values were below 0.05, so there was not enough evidence to reject any null hypotheses.

Changing the max_depth variable resulted in a very interesting error with how the model displayed the data. When choosing a higher max_depth, the model would sometimes display nothing. I think this occurred because when we have a high max_depth size, our model becomes increasingly complex. This can cause overfitting of our model and make it so that the test accuracy of the model is incredibly low. To avoid this concept of overfitting and ensure that model runs correctly, I chose a smaller max_depth size so that the model was much simpler and properly represented the data.

Similar to the max_depth variable, the n_neighbors counter needed to be limited in order to ensure that my model was not incredibly complex, and thus avoiding overfitting. But, when changing the value, I found that sometimes a lower value resulted in a less complicated model and, in turn, resulted in lower mean accuracy. To balance this, I found a middle ground between a complicated model and a simple model, as to have the largest possible mean accuracy, while still avoiding overfitting.

If I had more time to work on this data set, I would try to create an algorithm based on something other than a categorical label. I have rarely seen models without categorical labels, so I think it would be interesting to try to find something without much variation, like the pH, depending on the citric acid and alcohol. Additionally, I would like to explore how to use more than two feature values to help classify the label. By using three or more feature values I would be able to compare the F1 scores of the models to the ones I used for this project and see if it would be better or worse. Also, with three or more values, I would be able to see how it affects the complexity of the models and if it makes it more likely for my models to overfit the training data. There many different things that I would be able to do with this data set if I had more time, but the relationships that I found between my feature values and label value were incredible to see, especially because of how well my models worked on identifying the proper classification.

Conclusion

Based on the data set, I was able to find that the model that was the most successful the k-Nearest Neighbors model, but the Decision Tree model was also incredibly accurate. All of my models were proved to be statistically significant through a paired t-test, which proved the two models were very similar in the results, almost to the point that they would predict the same accuracy each time. The relationships that were the clearest and proved to help the model the most, were the correlations between alcohol and quality white and free sulfur dioxide and quality white. While these two relationships were the most prevalent, it is possible that other relationships between feature values and quality white could reveal a more successful model. This means that with further investigation, our model could become more efficient and successful when looking at test data.

References

[1] Pandas Documentation. Visualization.
https://pandas.pydata.org/pandas-docs/stable/user_guide/visualization.html

[2] Stack Overflow. Plot correlation matrix using pandas
https://stackoverflow.com/questions/29432629/plot-correlation-matrix-using-pandas