Choose Your Own Project: Capstone

Marisa Acierno

10/18/2019

Introduction

This report represents the Choose Your Own Project Capstone for the Harvard edX Data Science course, a culmination of eight previous courses in R coding, data wrangling, data visualization and machine learning. Students are tasked with sourcing a data set, preparing the data for analysis, and developing a machine learning model. For my report, I have chosen the wine dataset available on the The UCI Machine Learning Repository[1]. I propose to build a model that can take a given wine's chemical characteristics and predict that wine's quality on a zero to ten scale. After cleaning and preparing the data, I explore the variables, build and test four different models, report the results, and consider opportunities for future work.

Methodology and Analysis

Data Wrangling

Before embarking on the modeling exercise, the data must be prepared for use. The UCI Machine Learning Repository, where the data was sourced, saved observations for red and white wines in separate files. These two files ("reds" and "whites") were downloaded and merged, with a new column added to the final, merged "wines" dataset indicating whether the observation was for a red or white wine. Using the caret packet, the wines dataset was then split 50/50 into a training set and a test set. The odd number of observations in the wines dataset (n = 6,497) resulted in a training set with 3,248 observations and a test set with 3,249. The data is now ready for modeling.

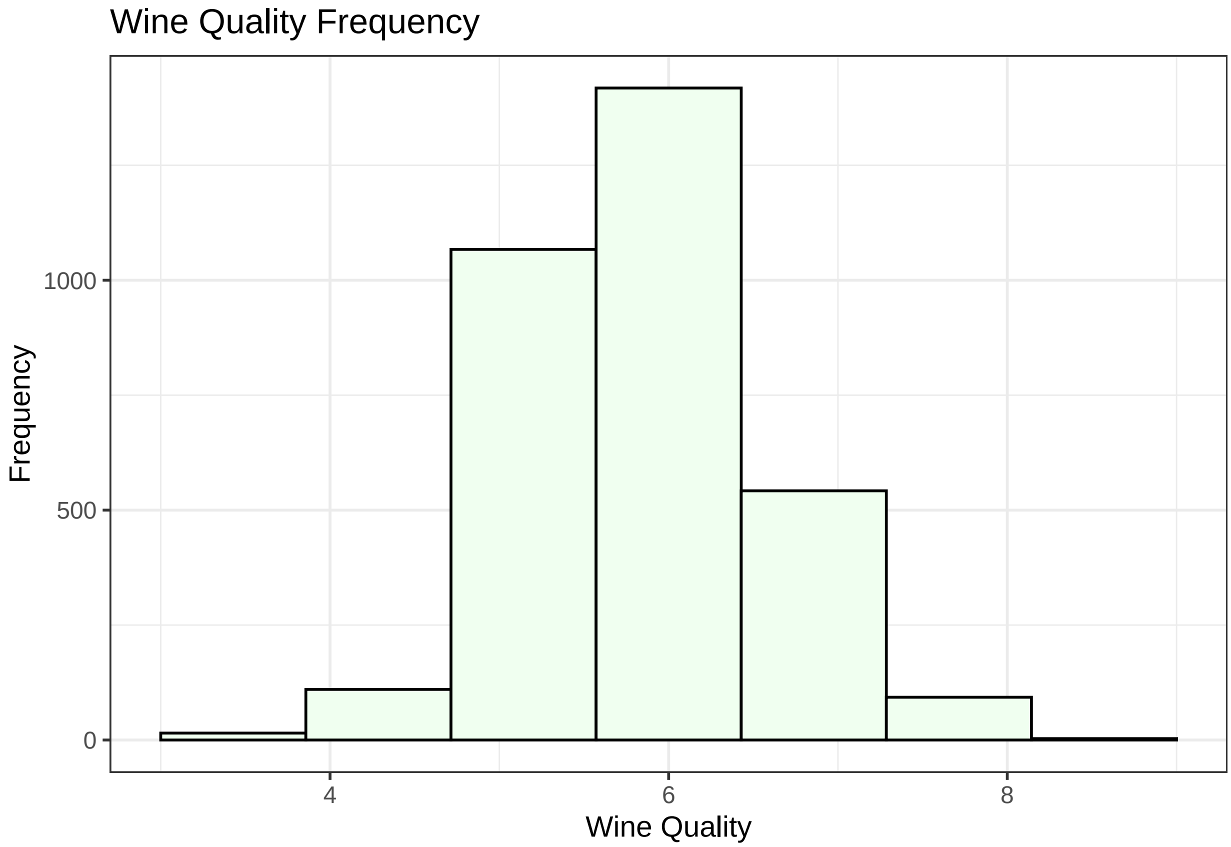
Data Exploration and Visualization

As seen when splitting the data into training and test set, this is a moderately sized data set. There are only 3,248 observations of 13 variables to build a model. All of the variables, except for "type", are continuous. Those 13 variables are:

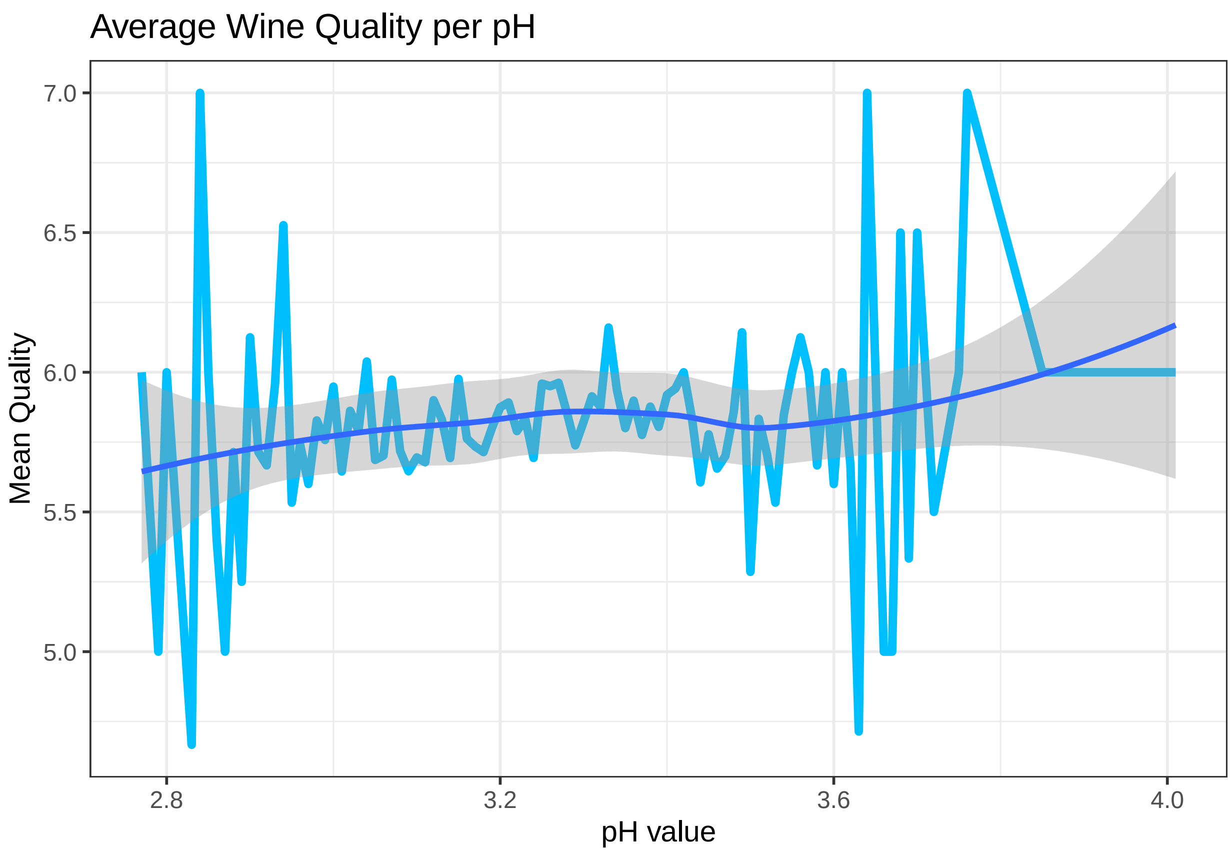
Table 1: Variable Names

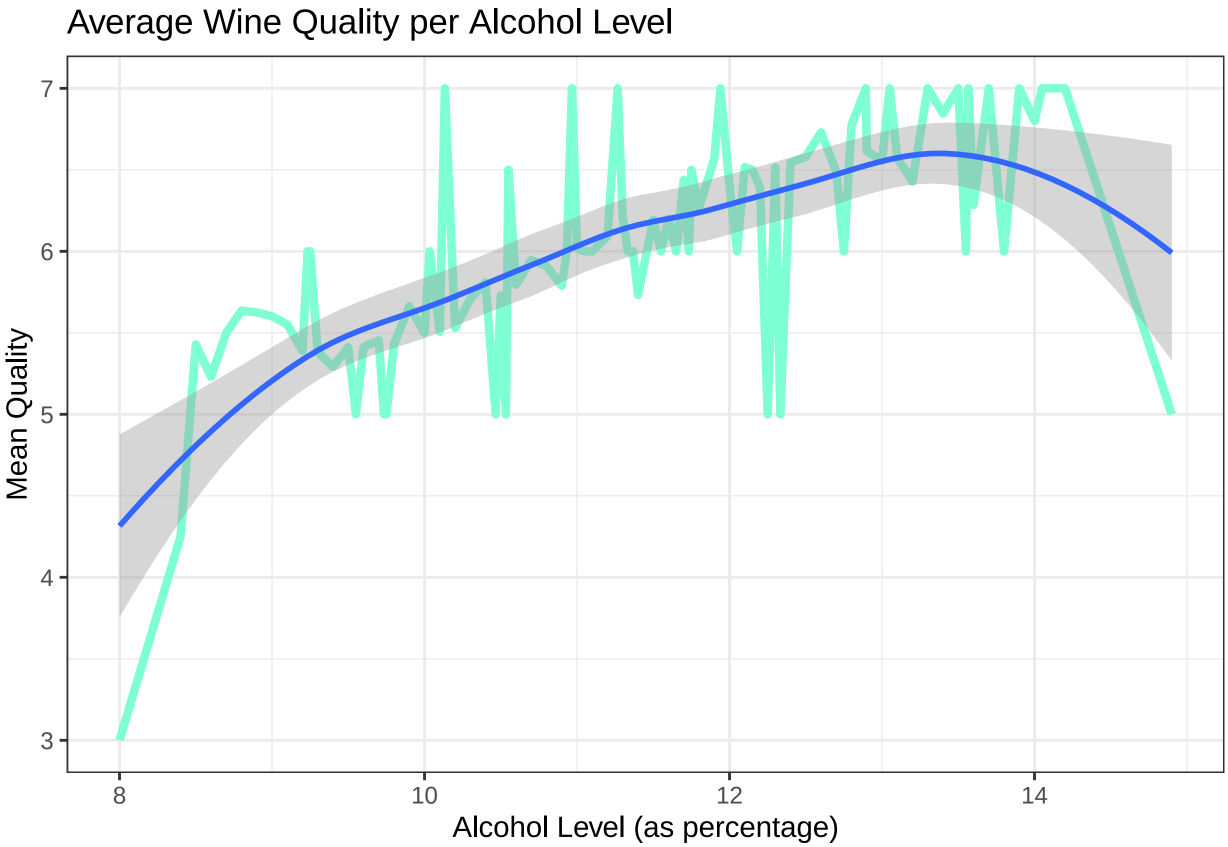
x
fixed acidity
volatile acidity
citric acid
residual sugar
chlorides
free sulfur dioxide
total sulfur dioxide
density
pH
sulphates
alcohol
quality
type

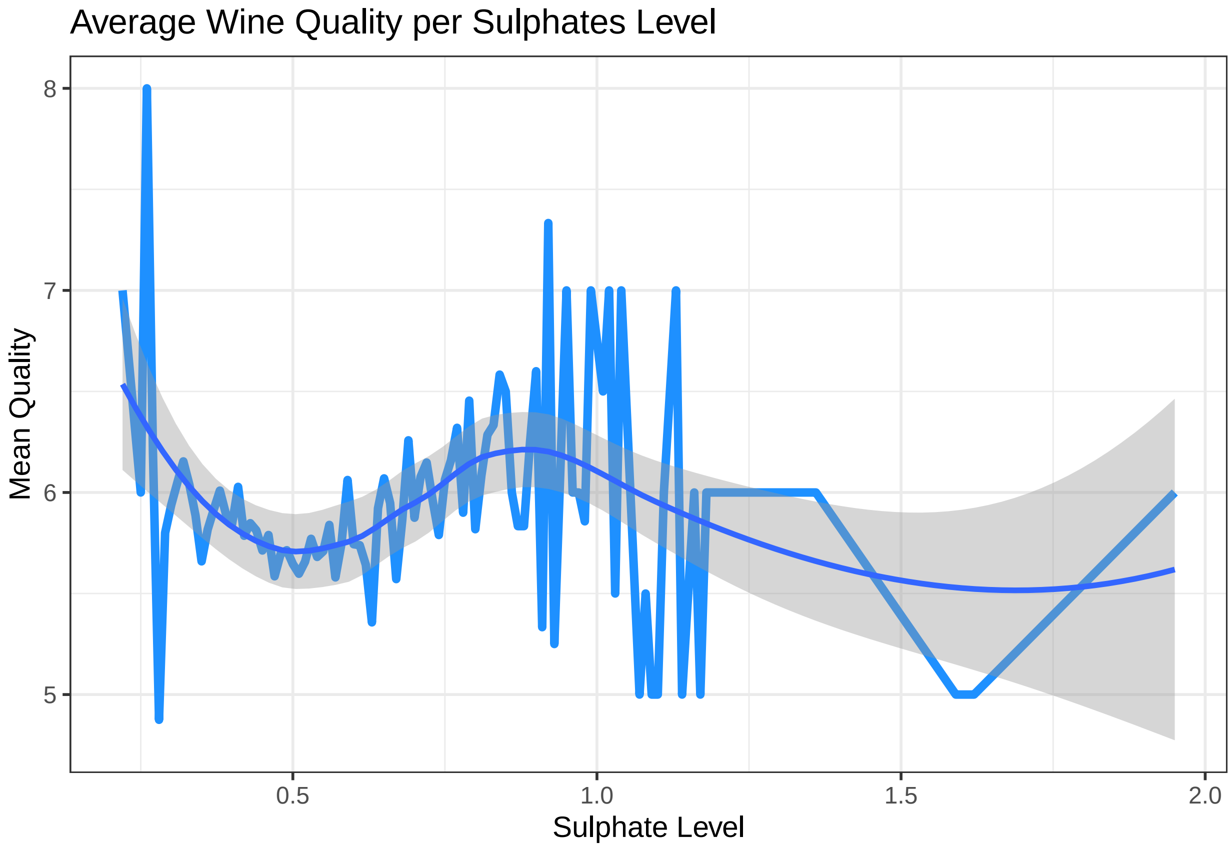
Data visualization is an opportunity to explore and better understand the data. I first look at the distribution of wine quality in the training set. The chart below shows that the majority of wines have ratings of 5 or 6 and significantly fewer wines have ratings above or below those values.



I next select a few variables (pH, alcohol, and sulphates) and graph the average wine quality for each value of the given variable. Those graphs are below.


Average Wine Quality per pH
Mean Quality
7.0
6.5
6.0
5.5
5.0
2.8
3.2
3.6
4.0
pH value


Average Wine Quality per Alcohol Level
Mean Quality
3
4
5
6
7
8
10
12
14
Alcohol Level (as percentage)



While the graphs suggest some general trends and possible relationships between these variables and wine quality, the data trends are fairly irratic. This is particularly the case when looking at pH and wine quality. As mentioned briefly above, the training set is fairly small. There are not many observations for each possible pH value across the continuous scale, which produces the irregularity seen in the graphs. It is thus unlikely that a single, or even several, variables will produce a reliable model. For this reason, the models tested in the next section use all the variables to predict wine quality.

Model 1: Just the average

Wine quality is relatively clustered around the mean, as illustrated in the Wine Quality Frenquency graph above and seen by the standard deviation of 0.873. A model that predicts the mean quality for all observations will provide a benchline to measure accuracy of more complex models.

We determine the average quality score of the training set to be:

```
## [1] 5.81681
```

However, quality scores in the data are all whole numbers. Rounded to the nearest whole number, the average quality score is 6. Using a quality score of 6 for all predictions yields an accuracy of 0.436. That is, a model using the mean for all predictions gets the quality score of the test set correct 43.6% of the time.

Another measure of model accuracy is the root mean square error, or RMSE. RMSE is the standard deviation of the residuals, or prediction errors. It is a measure of how far the values predicted by the model are from the observed values. In this case, the RMSE measures how close the predicted quality scores are to the true quality scores found in the test set. The RMSE of the model using the average quality score for all predictions is in the table below.

Table 2: Accuracy of Model 1: Just the Mean

RMSE	Percent.Correct
0.8919806	0.436442

Model 2: Linear regression

The linear regression model predicts wine quality based on all other variables available, which are:

value
fixed acidity
volatile acidity
citric acid
residual sugar
chlorides
free sulfur dioxide
total sulfur dioxide
density
pH
sulphates
alcohol
type

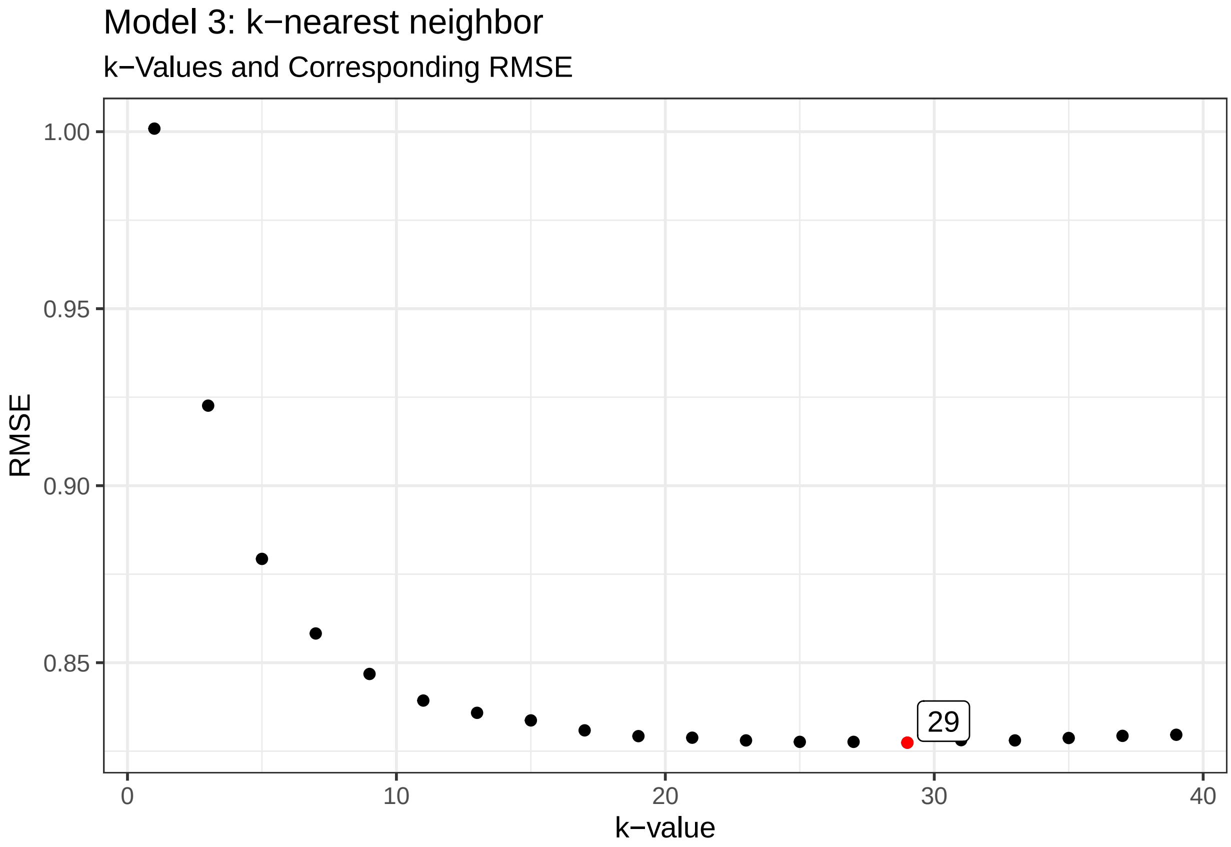
The linear regression model below is correct 53.3% of the time with an RMSE of 0.7922. In both measures, this is an improvement over Model 1.

Table 4: Accuracy of Model 2: Linear Regression

RMSE	Percent.Correct
0.792198	0.5327793

Model 3: k-nearest neighbor

For every observation, the k-nearest neighbors, or knn, model takes k-number of the nearest observations across multiple dimensions and computes the average wine quality of those points. Larger k values take more of the data set into account while smaller k values take a smaller set of neighboring observations. The first knn model applies a range of k values into the formula. The resulting RMSE generated by each value of k is seen in the graph below:



As seen in the graph, a k value of 29 produces the lowest RMSE value.

We then make predictions using a knn formula where k = 29. This model produces disappointing results:

Table 5: Accuracy of Model 3: Tuned knn Model

Percent.Correct	RMSE
0.4669129	0.8655366

The knn model is less accurate than the linear regression approach used in Model 2. We put this model aside and move on.

Model 4: random forest

Random forests are the aggregation of multiple, randomly constructed decision trees. This play between "decision trees" and "random forest" is also evidence that statisticians have a sense of humor. Compared to a single decision tree, a random forest enjoys increased stability as it is the average of many trees. The following code builds the random forest and generates a set of predictions.

```
fit_rf <- train(quality ~ ., method = "rf", data = train_set)
predict_rf <- predict(fit_rf, test_set) %>% round(., digits = 0)
```

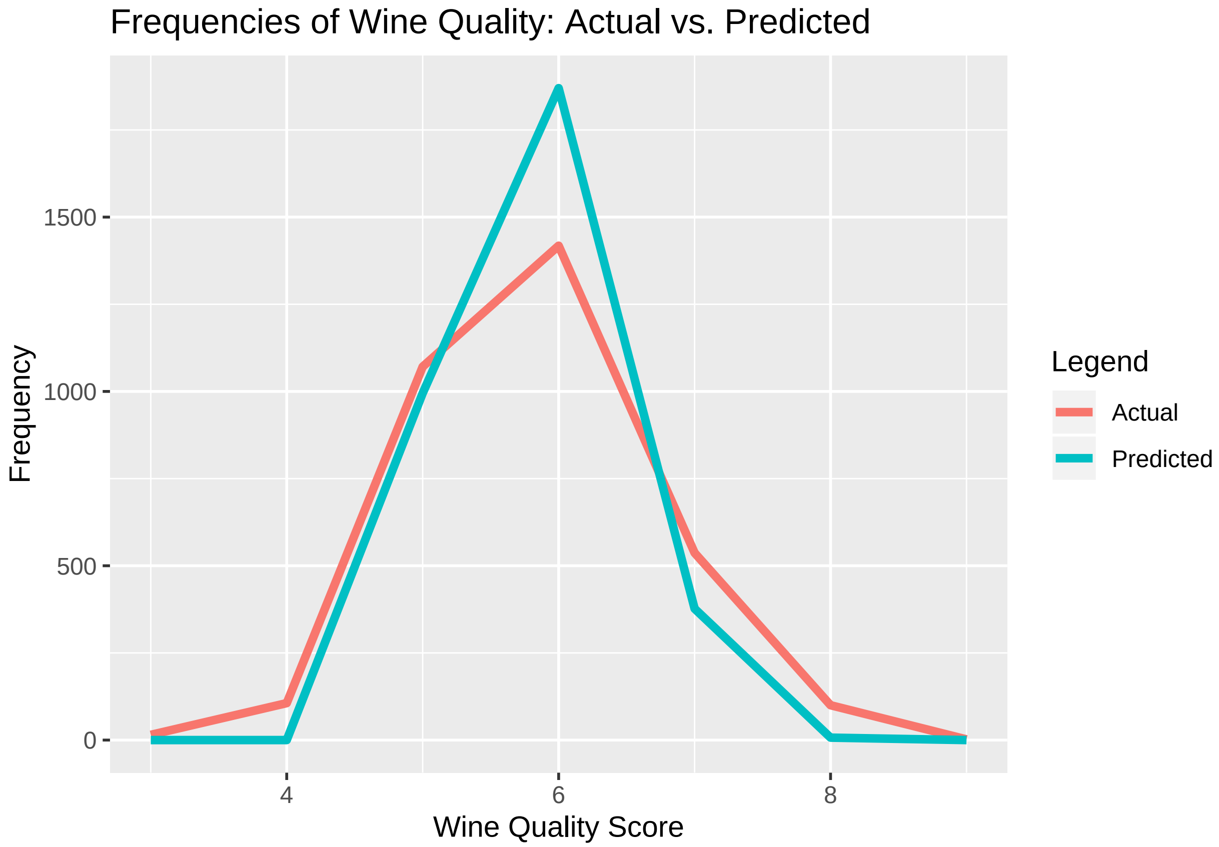
We see that the random forest model is by far the best, as seen in the following accuracy and RMSE values:

Table 6: Accuracy of Model 4: Random Forest

Percent.Correct	RMSE
0.6441982	0.6783374

Results

While the random forest approach generated the most accurate predictions of the four models tested, it was only correct 64% of the time. Graphing the distribution of quality scores in the test set compared to the predictions generated by the random forest model illustrates that the model compresses the predictions too close to the center score.



Indeed, while the mean quality score for the test set and the predictions are nearly identical, the standard deviation of the predicted values is smaller than it should be.

Table 7: Predicted vs. Actual Mean

Predicted	Actual
5.814	5.82

Table 8: Predicted vs. Actual Standard Deviation

Predicted	Actual
0.63	0.874

Predicted	Actual

However, if we widen our definition of “accurate” to mean that the predicted quality is within one point of the true quality, then, unsurprisingly, our model’s accuracy increases dramatically. Indeed, the random forest model predictions are within one point of the true quality nearly 97% of the time. Given how personal wine tastings and perceptions of quality are, I would consider this a success.

Conclusion

After building a variety of models, including one based on the average quality score, a linear regression model, a tuned k-nearest neighbors model, and a random forest model, the latter proved the most accurate based on both percentage correct and RMSE. Additional testing with other model types could be done to find an approach that better mimics the distribution and standard deviation of the test set. Alternatively, because there are so few observations for each unique value of x, where x is any of the twelve predictor variables, transforming those variables from continuous into factors through binning may be worth exploring to determine if the drawbacks of binning are outweighed by the larger samples sizes they generate.

Footnotes

[1] https://archive.ics.uci.edu/ml/datasets.php