Wine Quality Analysis

Ryan Reid

2021

1. Introduction

I enjoy drinking red wine and was interested to discover which properties of red wines contributed most to wine quality. Fortunately, the UCI machine learning repository offers a clean dataset of red wine properties and the corresponding quality score - perfect! The dataset is related to red variants of the Portuguese “Vinho Verde” wine. Unfortunately there is no data about grape types, wine brand, wine selling price, etc.

The following report will look at the properties of wine and examine how they impact overall quality. With that in mind, I will develop a machine learning model to predict wine quality given a known set of wine properties.

2. Initial Analysis

2.1 Preparing the Dataset

Citation for dataset used: P. Cortez, A. Cerdeira, F. Almeida, T. Matos and J. Reis. Modeling wine preferences by data mining from physicochemical properties. In Decision Support Systems, Elsevier, 47(4):547-553, 2009.

```
#Loading dataset

wine <- read.csv("https://raw.githubusercontent.com/ryanreid-code/capstone/main/winequality-red.csv")
```

```
## Loading required package: dplyr
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
## Loading required package: ggplot2
```

```
## Loading required package: randomForest
```

```
## randomForest 4.6-14
```

```
## Type rfNews() to see new features/changes/bug fixes.
```

```
##
## Attaching package: 'randomForest'
```

```
## The following object is masked from 'package:ggplot2':
##
##     margin
```

```
## The following object is masked from 'package:dplyr':
##
##     combine
```

```
## Loading required package: class
```

```
## Loading required package: caret
```

```
## Loading required package: lattice
```

```
## Loading required package: corrplot
```

```
## corrplot 0.90 loaded
```

2.2 Exploring the Dataset

The dataset is quite clean, with 1,599 red wines assigned a quality rating based on 11 features (chemical properties). Those 11 features are:

Inputs (based on physicochemical tests):

1. fixed acidity
2. volatile acidity
3. citric acid
4. residual sugar
5. chlorides
6. free sulfur dioxide
7. total sulfur dioxide
8. density
9. pH
10. sulphates
11. alcohol

Outputs (based on sensory data):

- quality (score between 0 and 10)

```
head(wine)
```

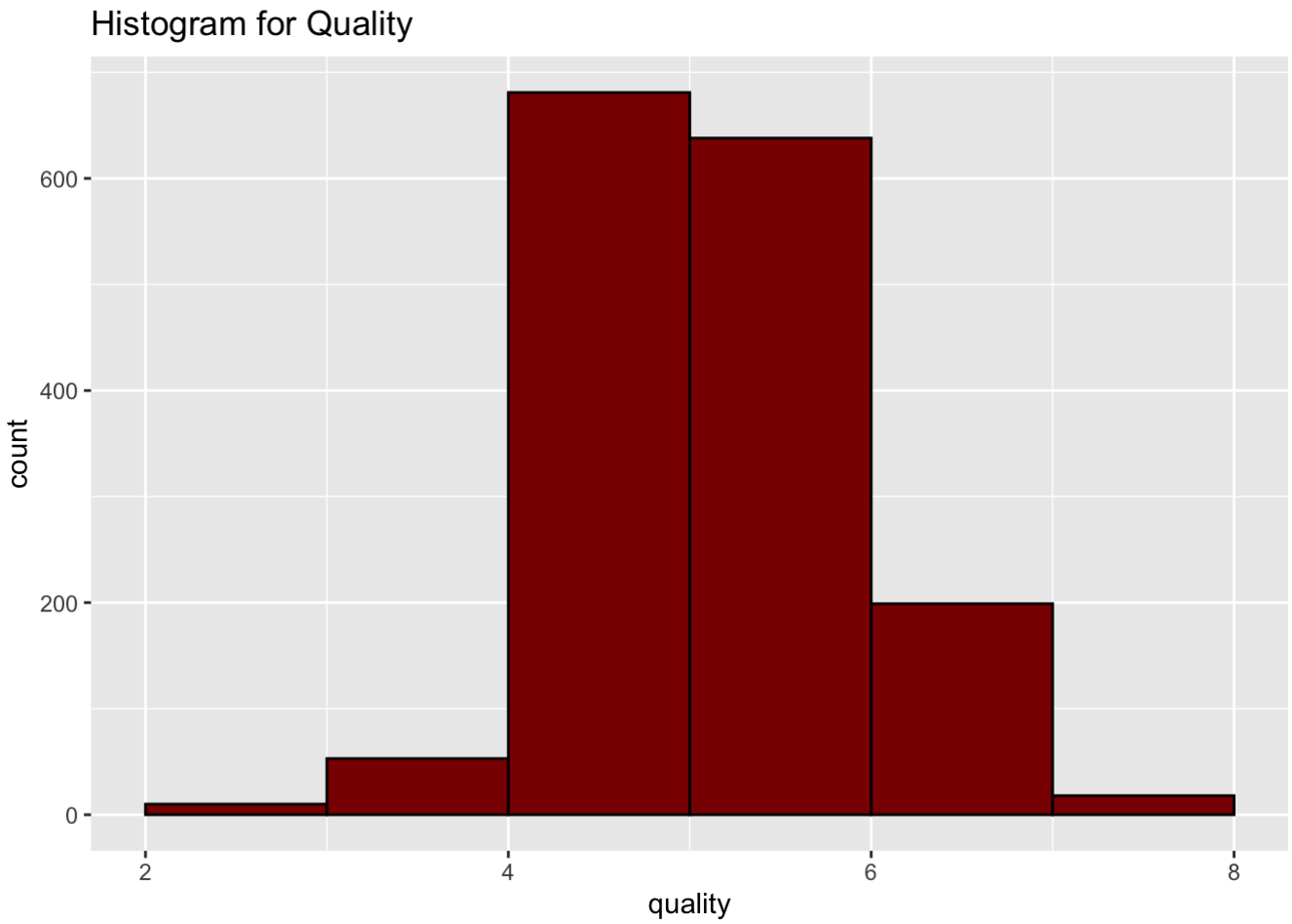
```
##   fixed.acidity volatile.acidity citric.acid residual.sugar chlorides
## 1           7.4             0.70        0.00            1.9     0.076
## 2           7.8             0.88        0.00            2.6     0.098
## 3           7.8             0.76        0.04            2.3     0.092
## 4          11.2             0.28        0.56            1.9     0.075
## 5           7.4             0.70        0.00            1.9     0.076
## 6           7.4             0.66        0.00            1.8     0.075
##   free.sulfur.dioxide total.sulfur.dioxide density   pH sulphates alcohol
## 1                  11                   34  0.9978 3.51      0.56     9.4
## 2                  25                   67  0.9968 3.20      0.68     9.8
## 3                  15                   54  0.9970 3.26      0.65     9.8
## 4                  17                   60  0.9980 3.16      0.58     9.8
## 5                  11                   34  0.9978 3.51      0.56     9.4
## 6                  13                   40  0.9978 3.51      0.56     9.4
##   quality
## 1       5
## 2       5
## 3       5
## 4       6
## 5       5
## 6       5
```

```
summary(wine)
```

```
##  fixed.acidity   volatile.acidity  citric.acid    residual.sugar
##  Min.   : 4.60   Min.   :0.1200   Min.   :0.000   Min.   : 0.900
##  1st Qu.: 7.10   1st Qu.:0.3900   1st Qu.:0.090   1st Qu.: 1.900
##  Median : 7.90   Median :0.5200   Median :0.260   Median : 2.200
##  Mean   : 8.32   Mean   :0.5278   Mean   :0.271   Mean   : 2.539
##  3rd Qu.: 9.20   3rd Qu.:0.6400   3rd Qu.:0.420   3rd Qu.: 2.600
##  Max.   :15.90   Max.   :1.5800   Max.   :1.000   Max.   :15.500
##    chlorides       free.sulfur.dioxide total.sulfur.dioxide    density
##  Min.   :0.01200   Min.   : 1.00       Min.   :  6.00       Min.   :0.9901
##  1st Qu.:0.07000   1st Qu.: 7.00       1st Qu.: 22.00       1st Qu.:0.9956
##  Median :0.07900   Median :14.00       Median : 38.00       Median :0.9968
##  Mean   :0.08747   Mean   :15.87       Mean   : 46.47       Mean   :0.9967
##  3rd Qu.:0.09000   3rd Qu.:21.00       3rd Qu.: 62.00       3rd Qu.:0.9978
##  Max.   :0.61100   Max.   :72.00       Max.   :289.00       Max.   :1.0037
##        pH          sulphates         alcohol         quality
##  Min.   :2.740   Min.   :0.3300   Min.   : 8.40   Min.   :3.000
##  1st Qu.:3.210   1st Qu.:0.5500   1st Qu.: 9.50   1st Qu.:5.000
##  Median :3.310   Median :0.6200   Median :10.20   Median :6.000
##  Mean   :3.311   Mean   :0.6581   Mean   :10.42   Mean   :5.636
##  3rd Qu.:3.400   3rd Qu.:0.7300   3rd Qu.:11.10   3rd Qu.:6.000
##  Max.   :4.010   Max.   :2.0000   Max.   :14.90   Max.   :8.000
```

```
mean(wine$quality)
```

```
## [1] 5.636023
```

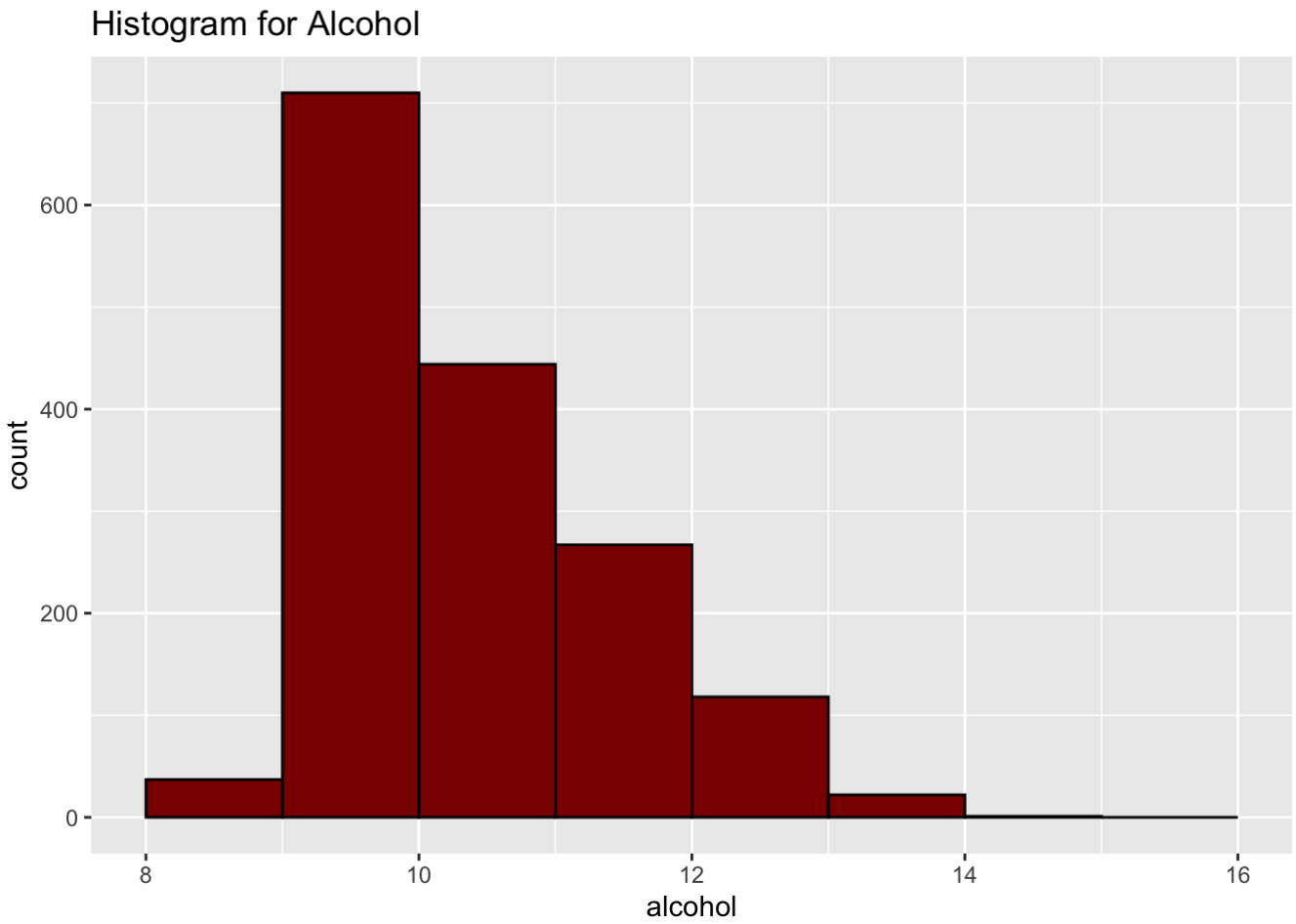


While quality is technically scored on a scale of 0 to 10, the minimum quality score was 3 and the maximum was 8. The average of all scores is 5.636. Let's look at some of the more common characteristics that are talked about by wine sommeliers and printed on the labels of bottles sold in stores; alcohol, sulphates, residual sugar, and acidity.

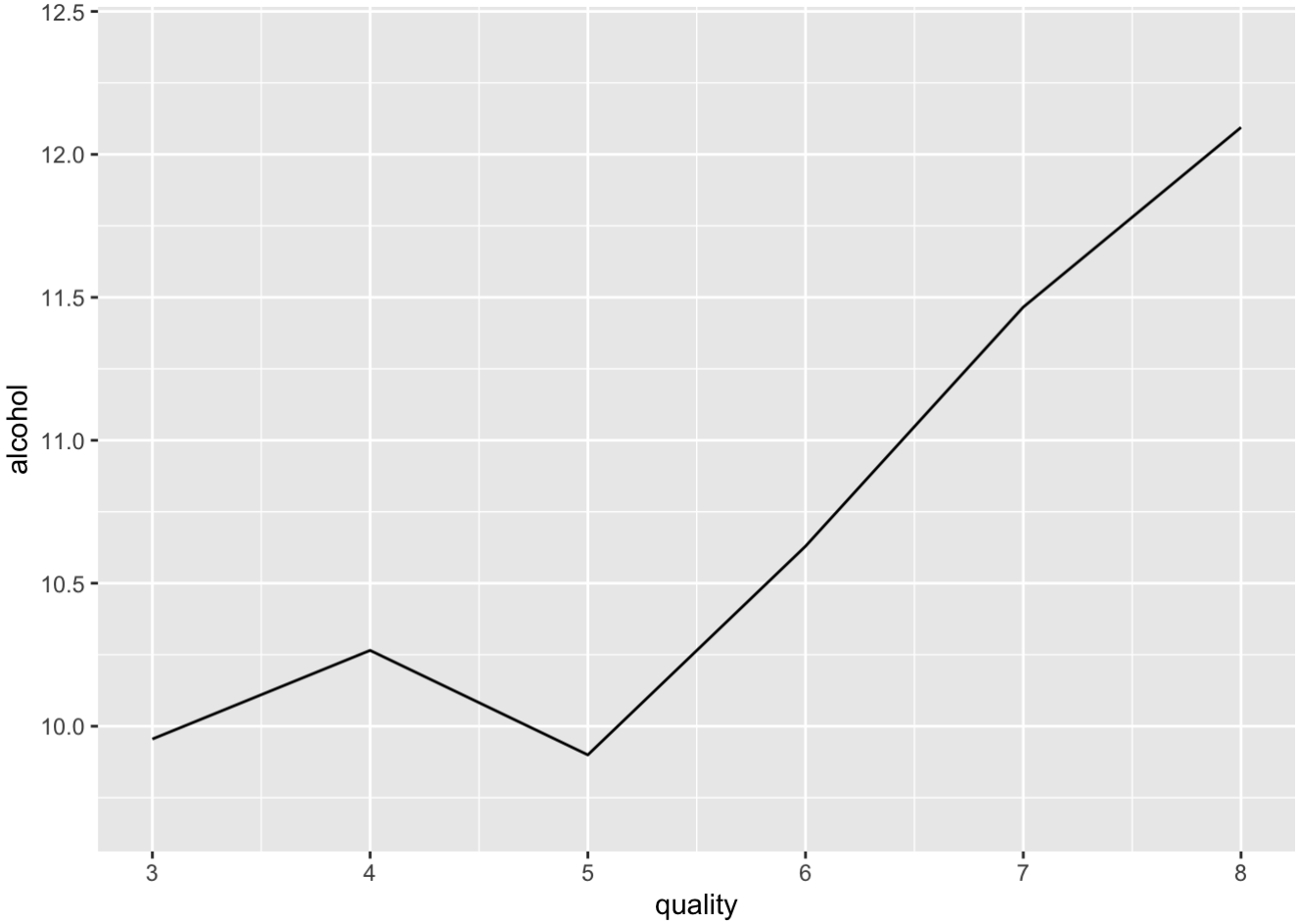
Alcohol

The average alcohol content is 10.4% and higher alcohol content tends to lead to higher quality wine. The majority of wines have at least 9% alcohol, with the quantity of wines decreasing as alcohol goes up. The highest alcohol content is nearly 15%.

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    8.40    9.50   10.20   10.42   11.10   14.90
```



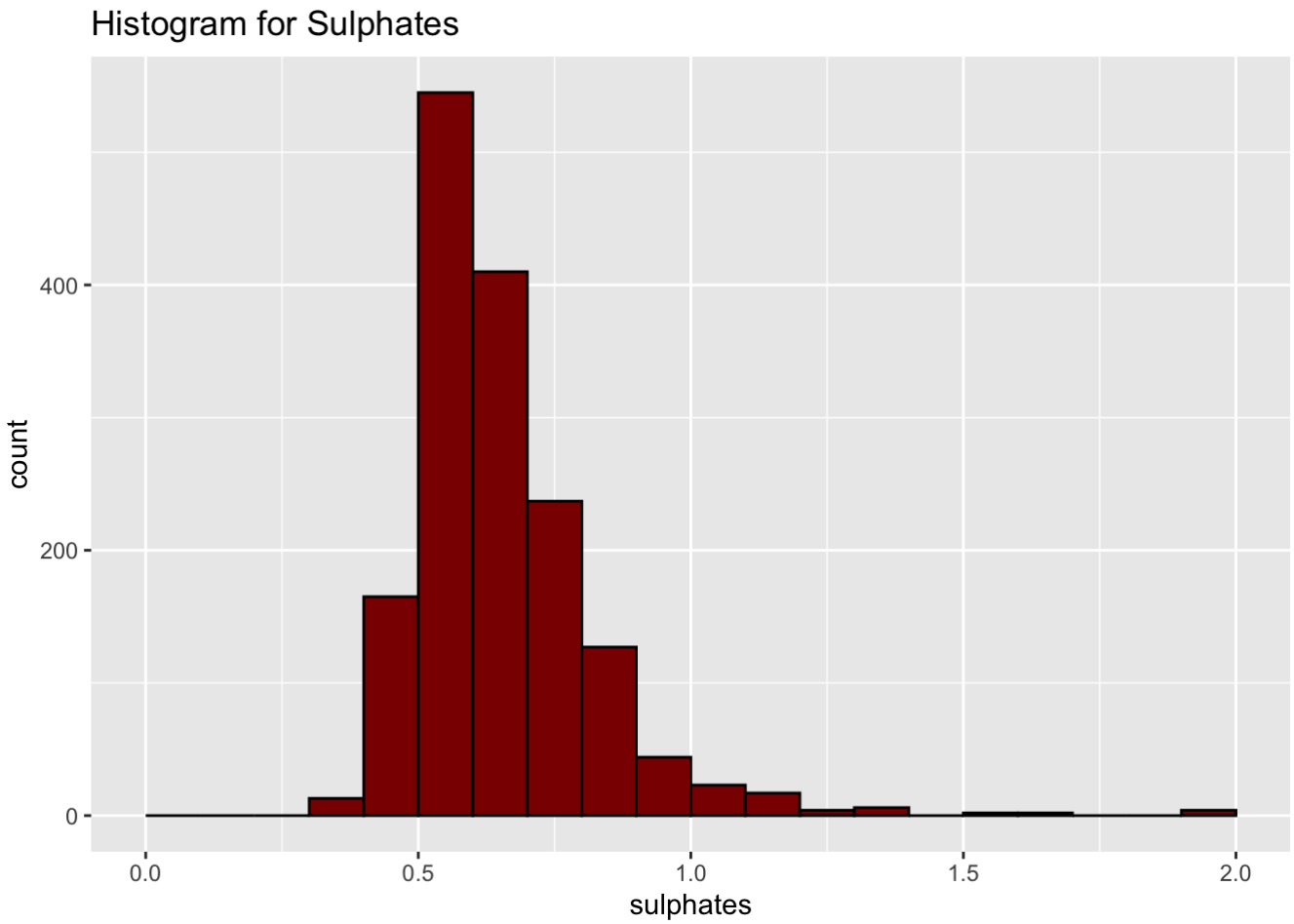

```
## No summary function supplied, defaulting to `mean_se()`
```



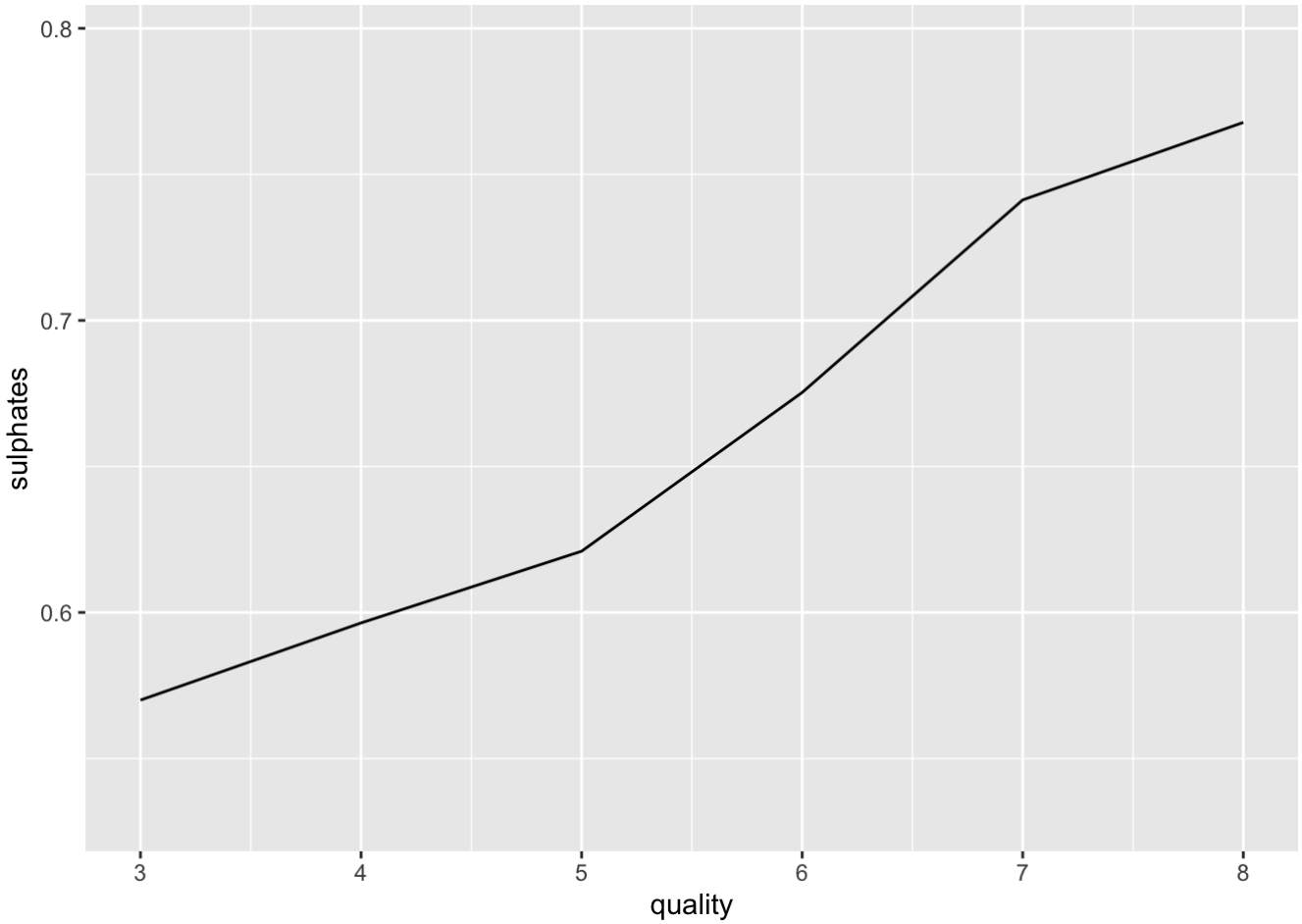
Sulphates

The average sulphate content is 0.658 and higher sulphate content tends to lead to higher quality wine. While some wines have sulphate levels as high as 2, the majority of wines have sulphate levels below 1.

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  0.3300  0.5500  0.6200  0.6581  0.7300  2.0000
```




```
## No summary function supplied, defaulting to `mean_se()`
```

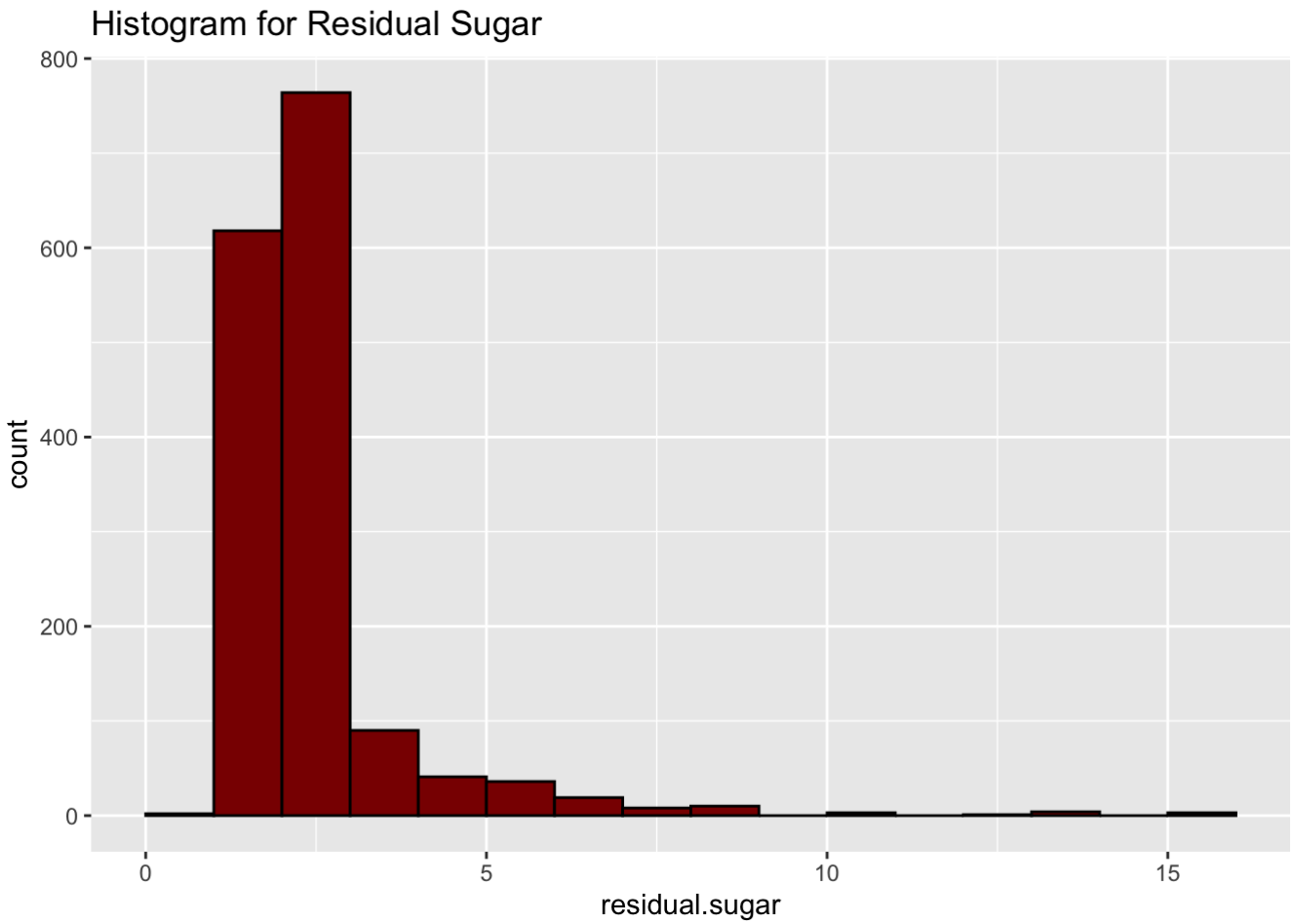


Residual Sugar

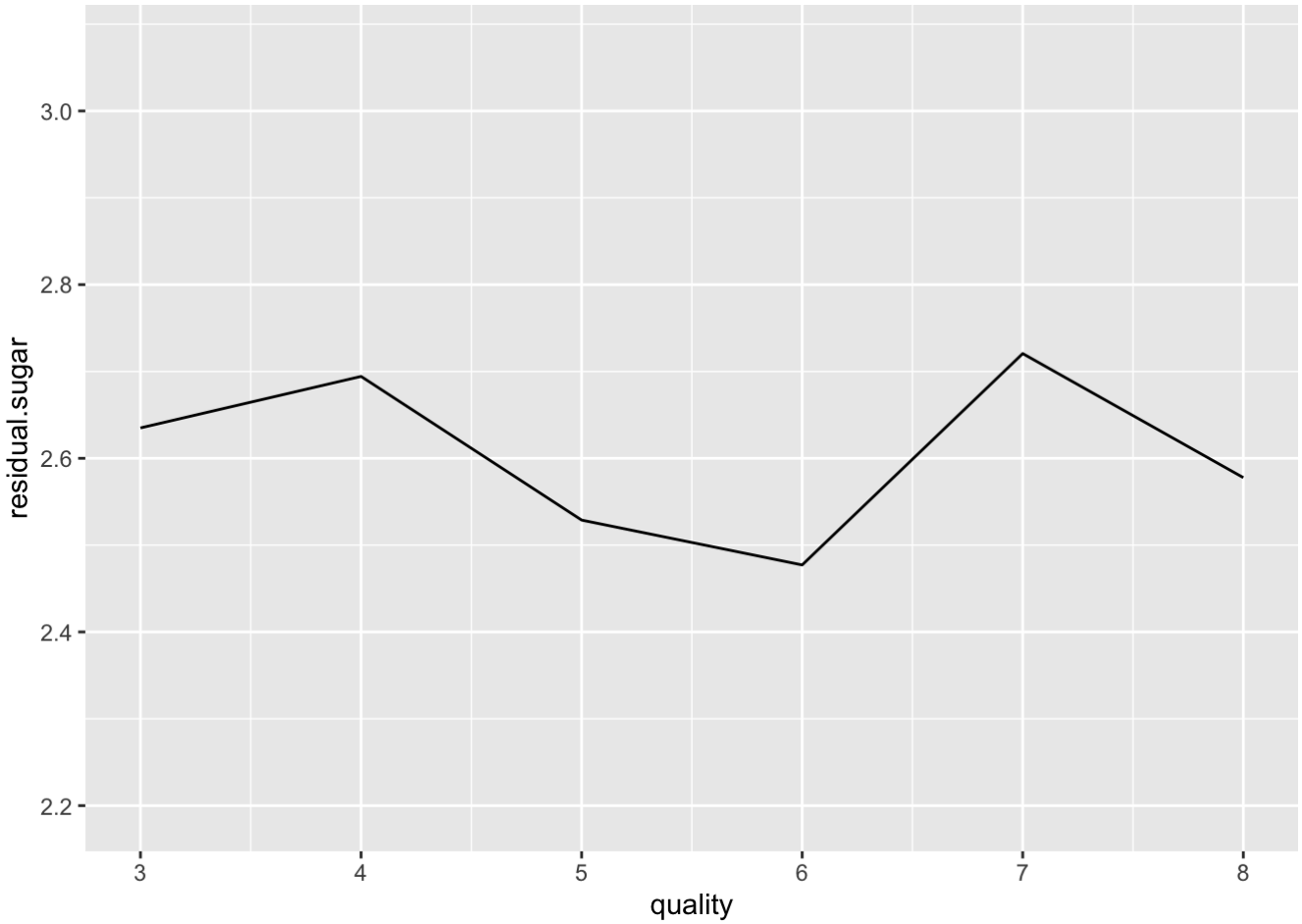
Residual sugar is the amount of sugar remaining after fermentation stops. It's rare to find wines with less than 1 gram/liter of residual sugar, and 75% of all wines have residual sugar below 2.6 grams/liter.

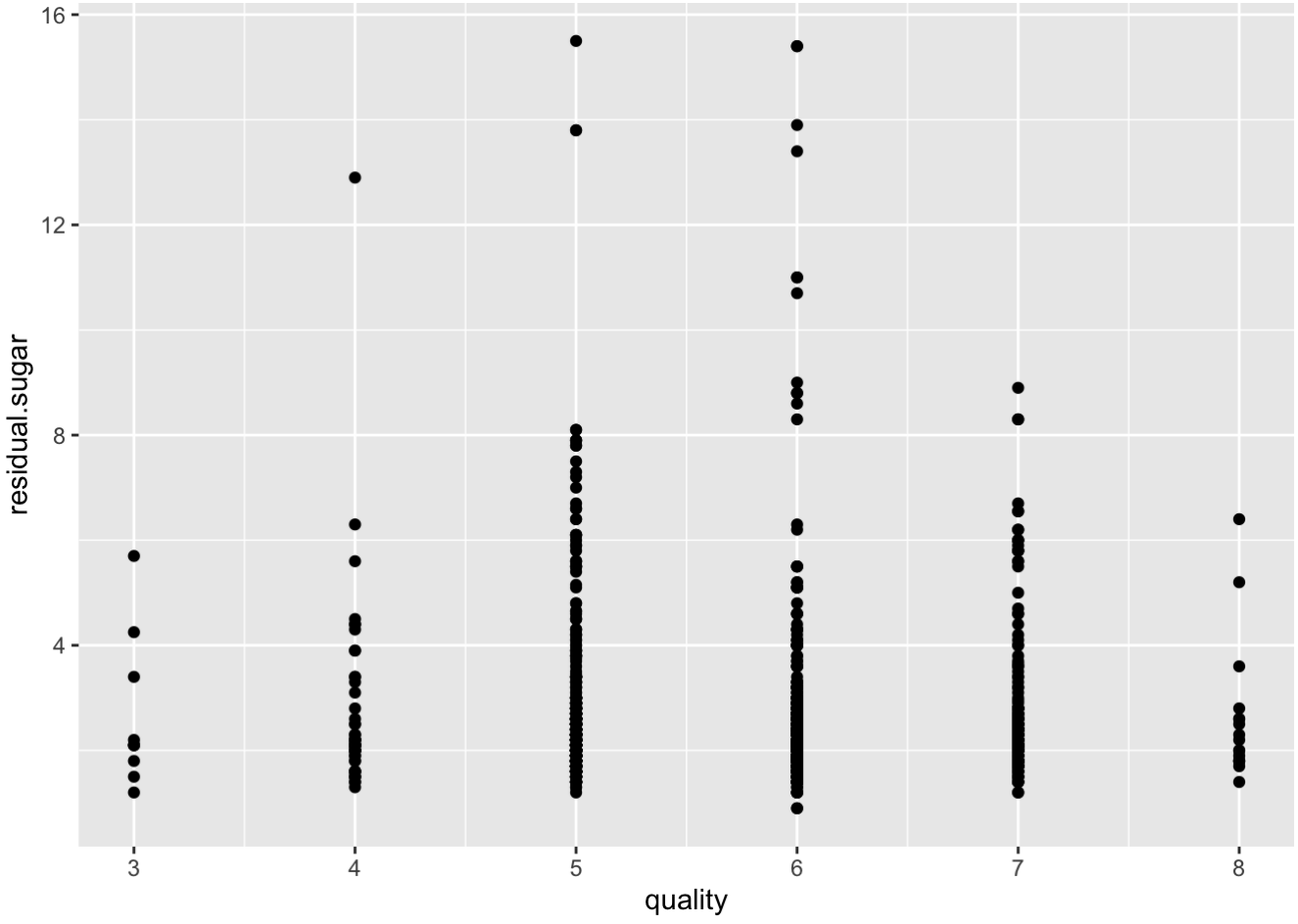
Residual sugar doesn't appear to correlate with quality. There are wines all across the sweetness spectrum at each quality level, however you don't tend to see as many very sweet wines with high quality scores. The highest residual sugar level is nearly 15 grams/liter, however wines sweeter than 10 grams/liter are quite rare.

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   0.900   1.900   2.200   2.539   2.600  15.500
```




```
## No summary function supplied, defaulting to `mean_se()`
```

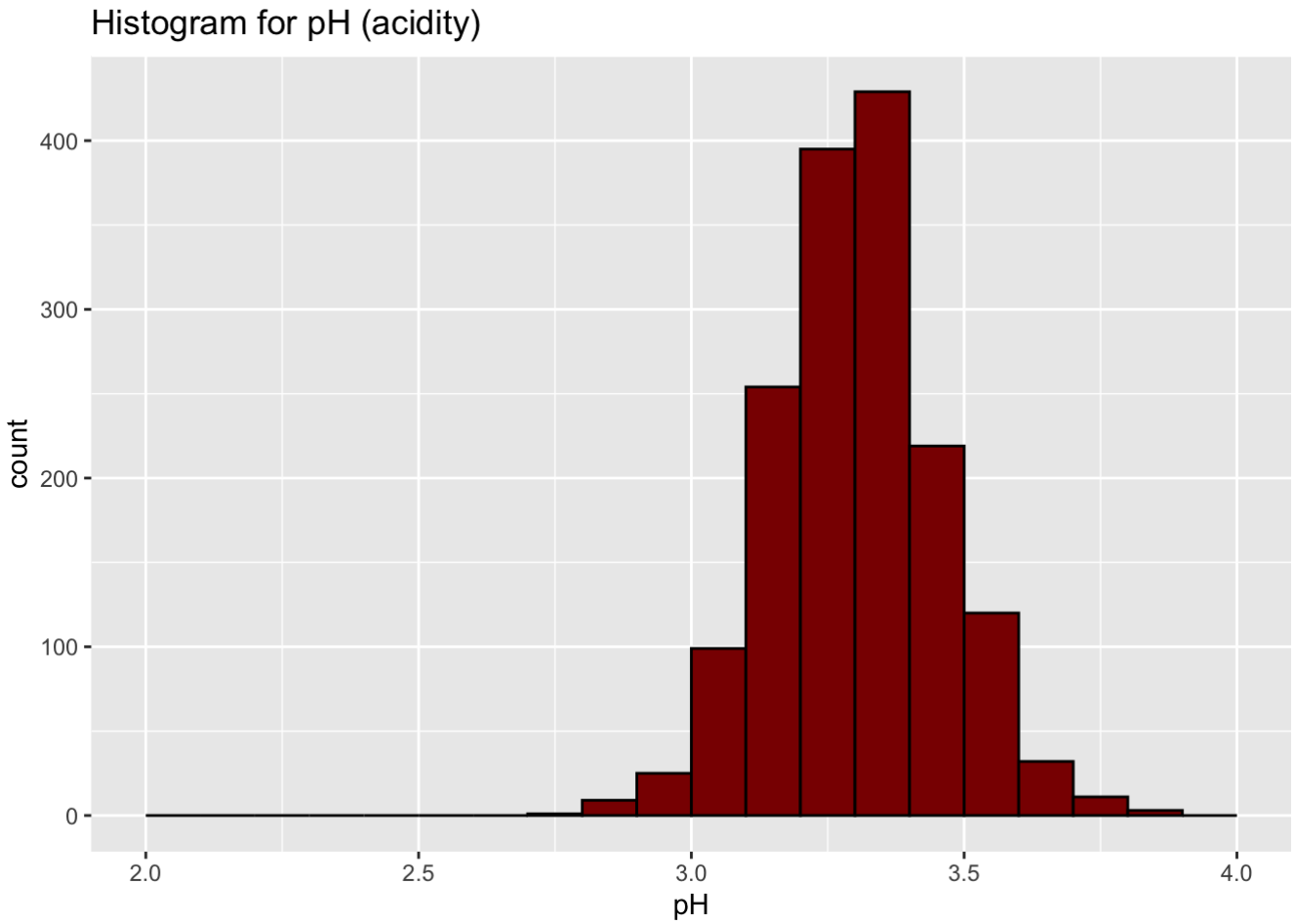

3.0
2.8
2.6
2.4
2.2
residual.sugar
3
4
5
6
7
8
quality


16
12
8
4
residual.sugar
3
4
5
6
7
8
quality

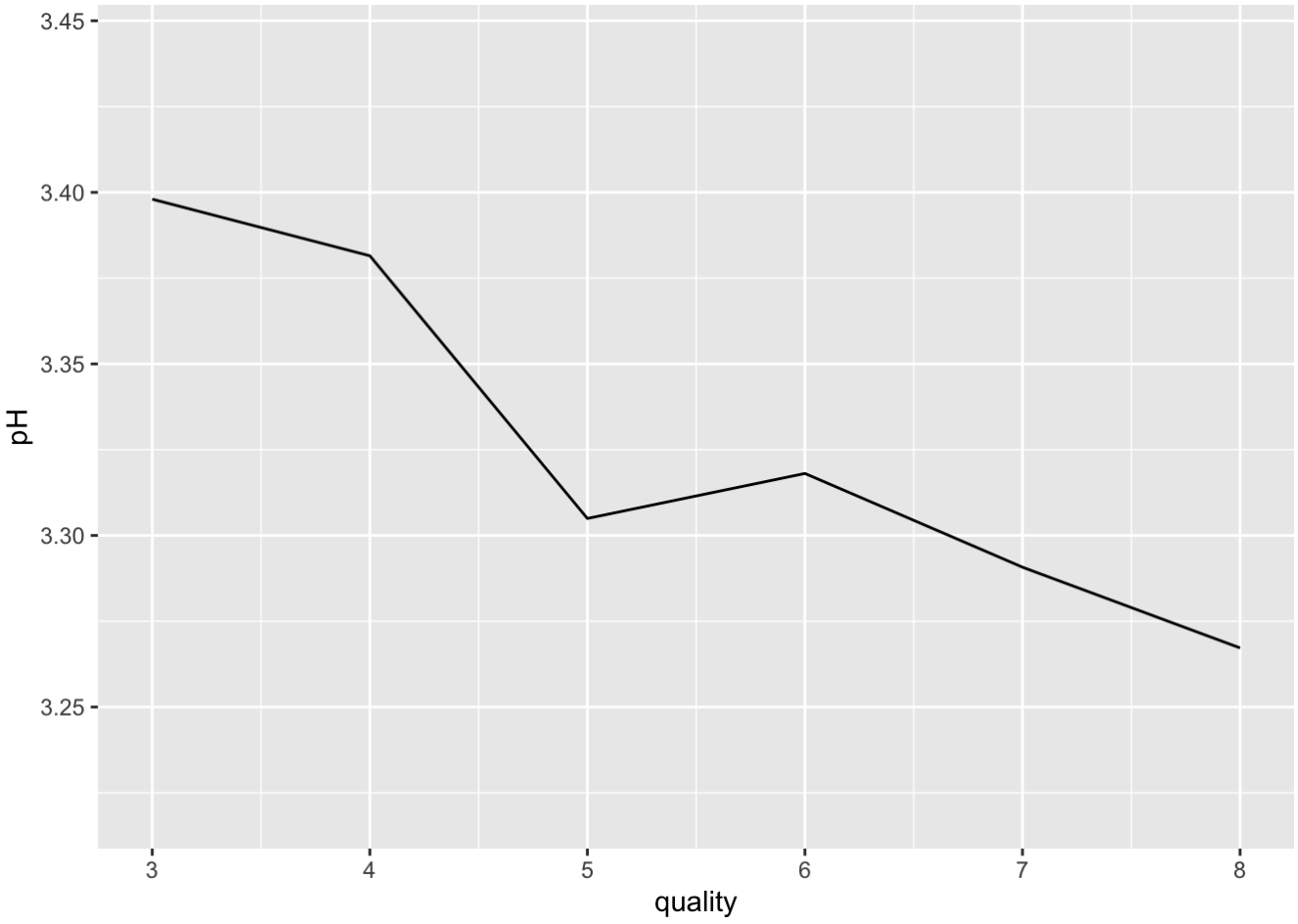
pH

The average pH level is 3.3 and a lower pH level (more acidity) tends to lead to higher quality wine. The pH levels of these red wines appear to be normally distributed around the mean, with the vast majority falling between 3.0 and 3.6 pH.

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   2.740   3.210   3.310   3.311   3.400   4.010
```



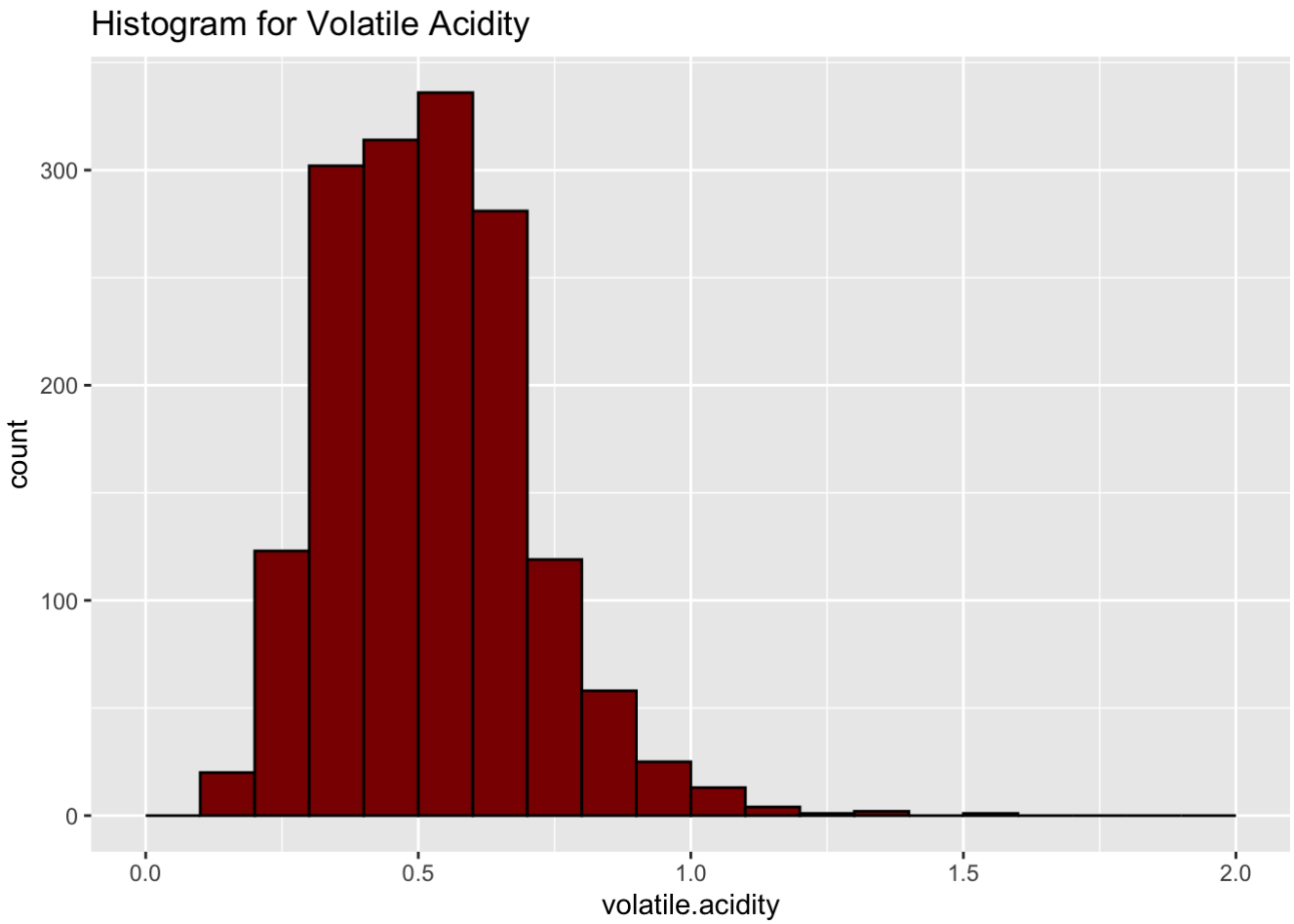

```
## No summary function supplied, defaulting to `mean_se()`
```



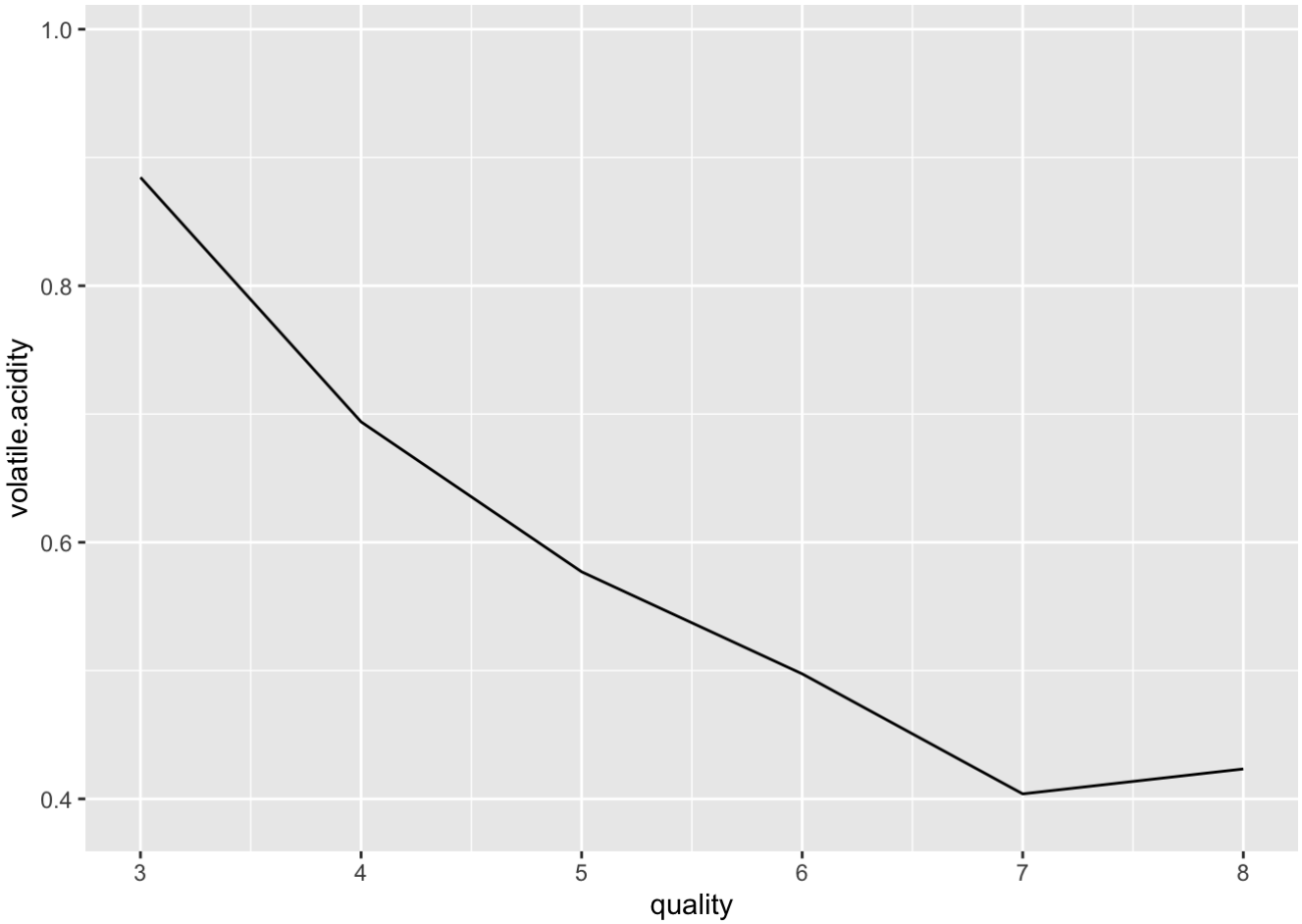
Volatile Acidity

We've seen that more acidic wines tend to receive higher quality scores, however volatile acidity is clearly undesirable in wine. Volatile acidity measures the amount of acetic acid in wine, which at too high of levels can lead to an unpleasant, vinegar taste.

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  0.1200  0.3900  0.5200  0.5278  0.6400  1.5800
```

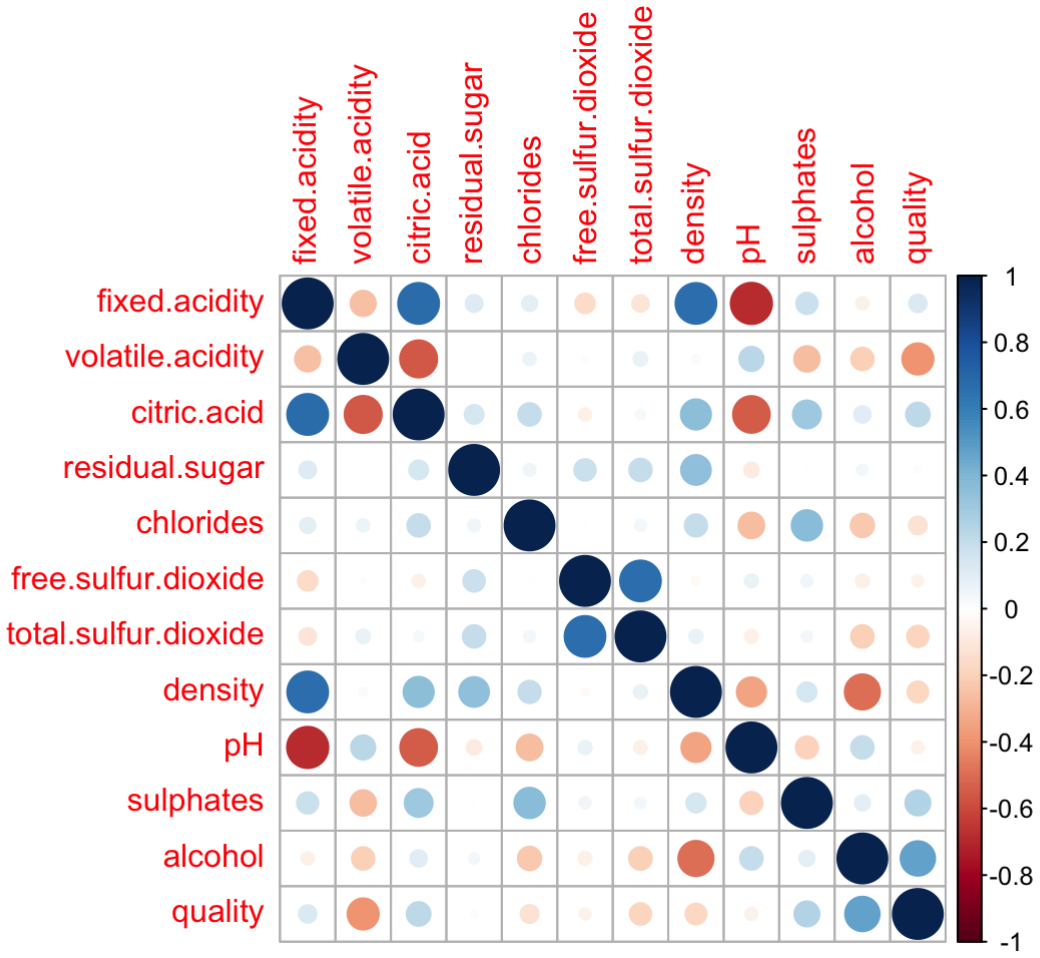



```
## No summary function supplied, defaulting to `mean_se()`
```



How are the features of wine related?

The corrplot shows that alcohol, sulphates, and citric acid have positive correlation with quality, while density, sulfur dioxide, and volatile acidity have negative correlation with quality.



3 Modeling Approach and Results

The data was split into train and test sets (80/20) for modeling. This problem is well suited to KNN and Random Forest modeling techniques. Let's see how these two models compare using the red wine dataset.

```
## Warning in set.seed(123, sample.kind = "Rounding"): non-uniform 'Rounding'
## sampler used
```

KNN

The k-nearest neighbors algorithm (KNN) will predict wine quality based on the quality score of wines with similar properties. We can train a model using the six most highly correlated features: alcohol, sulphates, citric acid, density, sulfur dioxide, and volatile acidity.

```
# KNN using 6 key features

set.seed(123, sample.kind ="Rounding")
```

```
## Warning in set.seed(123, sample.kind = "Rounding"): non-uniform 'Rounding'
## sampler used
```

```
train_x <- subset(train_wines, select= c(alcohol, sulphates, total.sulfur.dioxide, volat
ile.acidity, density, citric.acid))
test_x <- subset(test_wines, select= c(alcohol, sulphates, total.sulfur.dioxide, volatil
e.acidity, density, citric.acid))
train_y <- train_wines$quality
test_y <- test_wines$quality

knn3_pred <- knn(train_x, test_x, train_y, k=3)
knn3_cm <- table(knn3_pred, test_y)

knn1_pred <- knn(train_x, test_x, train_y, k=1)
knn1_cm <- table(knn1_pred, test_y)

knn3_cmstat <- confusionMatrix(knn3_cm)
knn1_cmstat <- confusionMatrix(knn1_cm)

knn3_cmstat
```

```
## Confusion Matrix and Statistics
##
##          test_y
## knn3_pred  3  4  5  6  7  8
##         3  0  0  1  1  0  0
##         4  0  0  2  0  0  0
##         5  0  7 97 51  6  1
##         6  2  7 27 61 12  2
##         7  0  0  6 14 22  1
##         8  0  0  0  1  0  0
##
## Overall Statistics
##
##                Accuracy : 0.5607
##                  95% CI : (0.5045, 0.6158)
##     No Information Rate : 0.4143
##     P-Value [Acc > NIR] : 8.963e-08
##
##                   Kappa : 0.3093
##
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: 3 Class: 4 Class: 5 Class: 6 Class: 7 Class: 8
## Sensitivity          0.000000 0.000000   0.7293   0.4766  0.55000 0.000000
## Specificity          0.993730 0.993485   0.6543   0.7409  0.92527 0.996845
## Pos Pred Value       0.000000 0.000000   0.5988   0.5495  0.51163 0.000000
## Neg Pred Value       0.993730 0.956113   0.7736   0.6810  0.93525 0.987500
## Prevalence           0.006231 0.043614   0.4143   0.3988  0.12461 0.012461
## Detection Rate       0.000000 0.000000   0.3022   0.1900  0.06854 0.000000
## Detection Prevalence 0.006231 0.006231   0.5047   0.3458  0.13396 0.003115
## Balanced Accuracy    0.496865 0.496743   0.6918   0.6087  0.73763 0.498423
```

```
knn1_cmstat
```

```
## Confusion Matrix and Statistics
##
##          test_y
## knn1_pred  3  4  5  6  7  8
##         3  0  0  1  0  0  0
##         4  0  1  1  1  0  0
##         5  1  5 99 39  3  1
##         6  1  8 28 71 14  1
##         7  0  0  4 15 23  2
##         8  0  0  0  2  0  0
##
## Overall Statistics
##
##                Accuracy : 0.6044
##                  95% CI : (0.5485, 0.6582)
##     No Information Rate : 0.4143
##     P-Value [Acc > NIR] : 5.636e-12
##
##                   Kappa : 0.3805
##
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: 3 Class: 4 Class: 5 Class: 6 Class: 7 Class: 8
## Sensitivity          0.000000 0.071429   0.7444   0.5547  0.57500 0.000000
## Specificity          0.996865 0.993485   0.7394   0.7306  0.92527 0.993691
## Pos Pred Value       0.000000 0.333333   0.6689   0.5772  0.52273 0.000000
## Neg Pred Value       0.993750 0.959119   0.8035   0.7121  0.93863 0.987461
## Prevalence           0.006231 0.043614   0.4143   0.3988  0.12461 0.012461
## Detection Rate       0.000000 0.003115   0.3084   0.2212  0.07165 0.000000
## Detection Prevalence 0.003115 0.009346   0.4611   0.3832  0.13707 0.006231
## Balanced Accuracy    0.498433 0.532457   0.7419   0.6426  0.75013 0.496845
```

KNN with 6 features is around 60% accurate at predicting the correct wine quality score out of 10 (KNN1 60%, KNN3 56%). The 6 features didn't include residual sugar or pH. How does adding 2 more features (for a total of 8 features) impact model accuracy?

```
# KNN with 8 features. Added residual sugar and pH

set.seed(123, sample.kind ="Rounding")
```

```
## Warning in set.seed(123, sample.kind = "Rounding"): non-uniform 'Rounding'
## sampler used
```

```
train8_x <- subset(train_wines, select= c(alcohol, sulphates, total.sulfur.dioxide, vola
tile.acidity, density, citric.acid, residual.sugar, pH))
test8_x <- subset(test_wines, select= c(alcohol, sulphates, total.sulfur.dioxide, volati
le.acidity, density, citric.acid, residual.sugar, pH))
train_y <- train_wines$quality
test_y <- test_wines$quality

knn3_pred <- knn(train8_x, test8_x, train_y, k=3)
knn3_cm <- table(knn3_pred, test_y)

knn1_pred <- knn(train8_x, test8_x, train_y, k=1)
knn1_cm <- table(knn1_pred, test_y)

knn3_cmstat <- confusionMatrix(knn3_cm)
knn1_cmstat <- confusionMatrix(knn1_cm)

knn3_cmstat
```

```
## Confusion Matrix and Statistics
##
##          test_y
## knn3_pred  3  4  5  6  7  8
##         3  0  0  1  0  0  0
##         4  0  1  1  0  2  0
##         5  0  6 89 43  5  1
##         6  1  4 33 72 17  0
##         7  1  3  9 13 15  3
##         8  0  0  0  0  1  0
##
## Overall Statistics
##
##                Accuracy : 0.5514
##                  95% CI : (0.4952, 0.6067)
##     No Information Rate : 0.4143
##     P-Value [Acc > NIR] : 5.196e-07
##
##                   Kappa : 0.2976
##
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: 3 Class: 4 Class: 5 Class: 6 Class: 7 Class: 8
## Sensitivity          0.000000 0.071429   0.6692   0.5625  0.37500 0.000000
## Specificity          0.996865 0.990228   0.7074   0.7150  0.89680 0.996845
## Pos Pred Value       0.000000 0.250000   0.6181   0.5669  0.34091 0.000000
## Neg Pred Value       0.993750 0.958991   0.7514   0.7113  0.90975 0.987500
## Prevalence           0.006231 0.043614   0.4143   0.3988  0.12461 0.012461
## Detection Rate       0.000000 0.003115   0.2773   0.2243  0.04673 0.000000
## Detection Prevalence 0.003115 0.012461   0.4486   0.3956  0.13707 0.003115
## Balanced Accuracy    0.498433 0.530828   0.6883   0.6388  0.63590 0.498423
```

```
knn1_cmstat
```

```
## Confusion Matrix and Statistics
##
##          test_y
## knn1_pred  3  4  5  6  7  8
##         3  1  0  0  0  0  0
##         4  0  1  2  2  0  0
##         5  0  6 93 40  5  0
##         6  1  5 33 69 12  1
##         7  0  2  5 12 22  3
##         8  0  0  0  5  1  0
##
## Overall Statistics
##
##                Accuracy : 0.5794
##                  95% CI : (0.5234, 0.634)
##     No Information Rate : 0.4143
##     P-Value [Acc > NIR] : 1.914e-09
##
##                   Kappa : 0.3488
##
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: 3 Class: 4 Class: 5 Class: 6 Class: 7 Class: 8
## Sensitivity          0.500000 0.071429   0.6992   0.5391  0.55000  0.00000
## Specificity          1.000000 0.986971   0.7287   0.7306  0.92171  0.98107
## Pos Pred Value       1.000000 0.200000   0.6458   0.5702  0.50000  0.00000
## Neg Pred Value       0.996875 0.958861   0.7740   0.7050  0.93502  0.98730
## Prevalence           0.006231 0.043614   0.4143   0.3988  0.12461  0.01246
## Detection Rate       0.003115 0.003115   0.2897   0.2150  0.06854  0.00000
## Detection Prevalence 0.003115 0.015576   0.4486   0.3769  0.13707  0.01869
## Balanced Accuracy    0.750000 0.529200   0.7140   0.6348  0.73585  0.49054
```

With 8 features, KNN1 is 58% accurate and KNN3 is 55% accurate. Doesn't look like adding the extra features contributes much. Likely better to stick with 6 for model simplicity.

Now let's see if we can do even better with the random forest algorithm.

Random Forests

The random forest algorithm predicts wine quality by merging a large number of decision trees in to a single model.

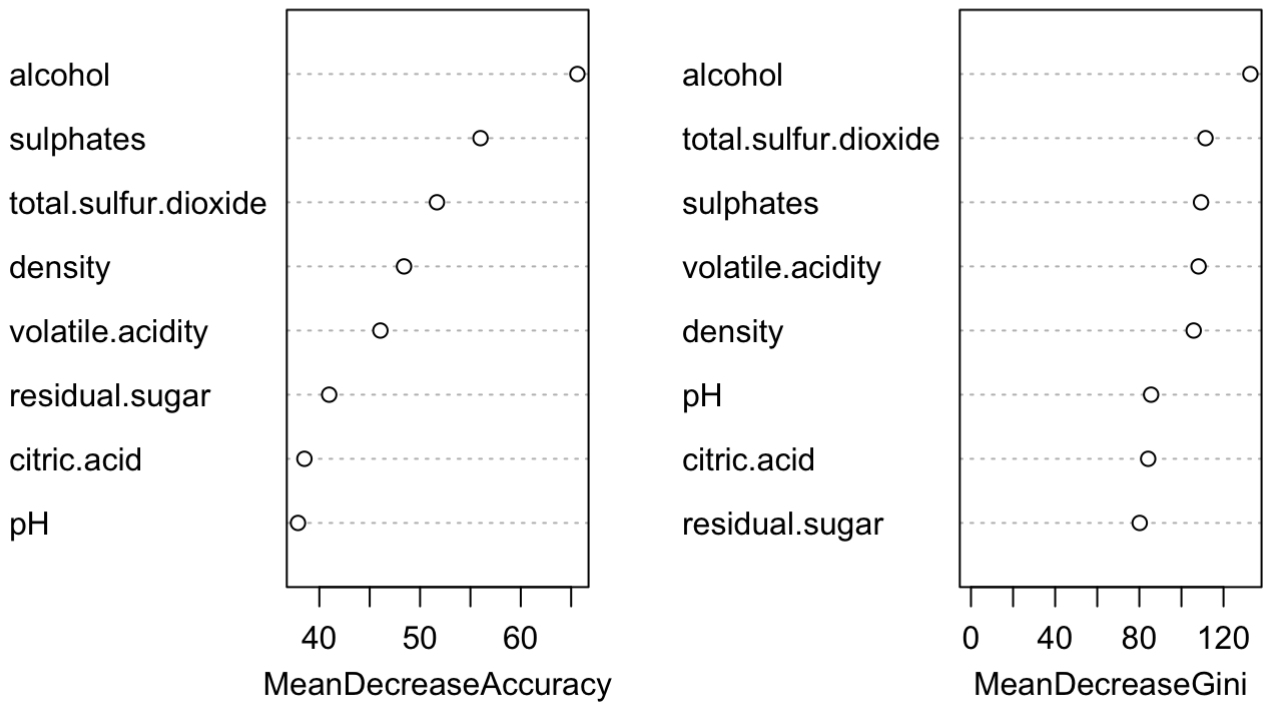
```
set.seed(123, sample.kind ="Rounding")
```

```
## Warning in set.seed(123, sample.kind = "Rounding"): non-uniform 'Rounding'
## sampler used
```

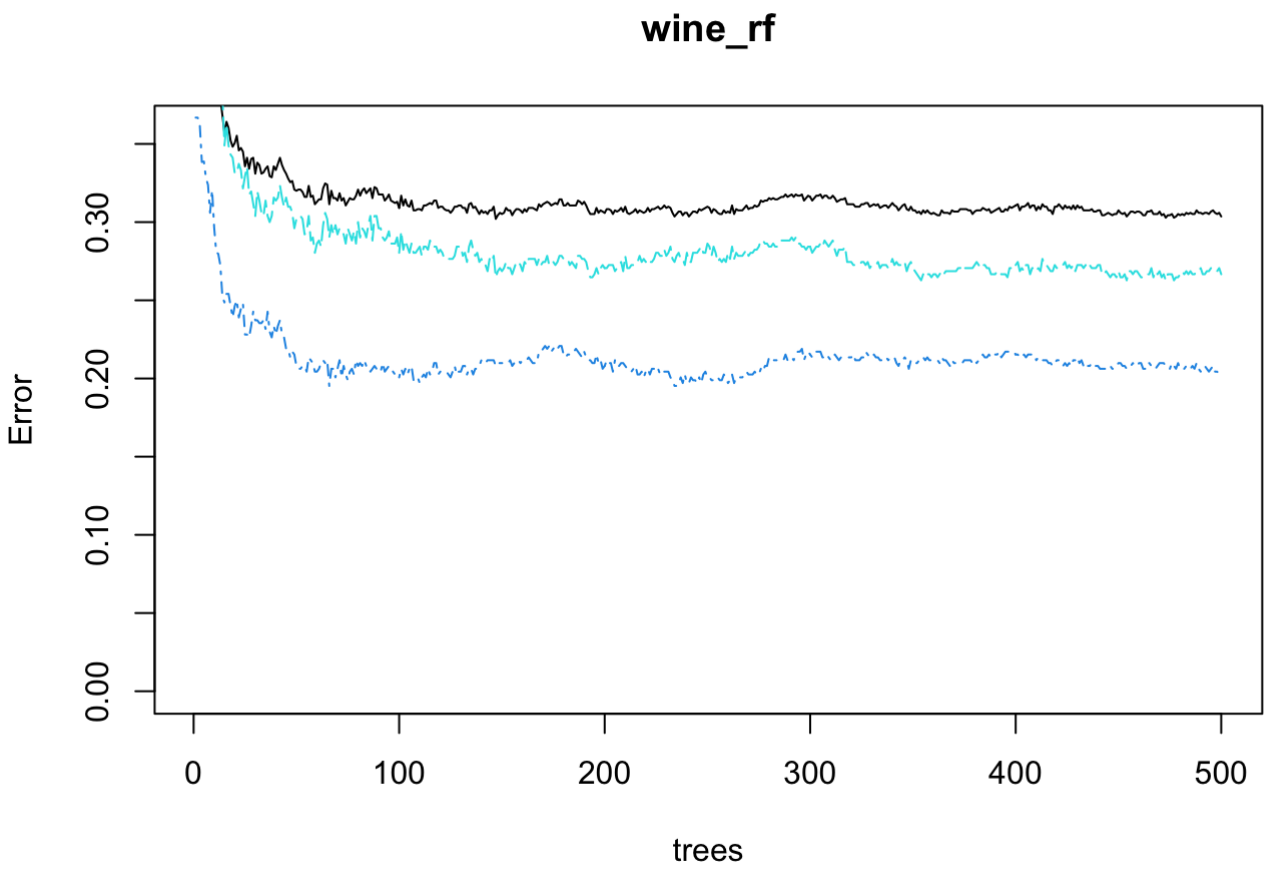
```
wine_rf <- randomForest(as.factor(quality)~alcohol+ sulphates+ total.sulfur.dioxide+ vol
atile.acidity+ density+ citric.acid+ residual.sugar+ pH, data=train_wines, importance=TR
UE, ntree = 500)
wine_rf
```

```
##
## Call:
##  randomForest(formula = as.factor(quality) ~ alcohol + sulphates +      total.sulfur.
dioxide + volatile.acidity + density + citric.acid +      residual.sugar + pH, data = tr
ain_wines, importance = TRUE,      ntree = 500)
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 2
##
##         OOB estimate of  error rate: 30.36%
## Confusion matrix:
##   3 4   5   6  7 8 class.error
## 3 0 0   6   2  0 0   1.0000000
## 4 0 0  30   8  1 0   1.0000000
## 5 1 0 437 106  4 0   0.2025547
## 6 0 0 109 374 27 0   0.2666667
## 7 0 0   8  72 77 2   0.5157233
## 8 0 0   0   6  6 2   0.8571429
```

```
varImpPlot(wine_rf)
```




```
plot(wine_rf, ylim=c(0,0.36))
```




```
wine_rf$confusion
```

```
##    3 4   5   6   7 8 class.error
## 3  0 0   6   2   0 0   1.0000000
## 4  0 0  30   8   1 0   1.0000000
## 5  1 0 437 106   4 0   0.2025547
## 6  0 0 109 374  27 0   0.2666667
## 7  0 0   8  72  77 2   0.5157233
## 8  0 0   0   6   6 2   0.8571429
```

The error rate declines as more trees are added to the forest (greater ntree), however the error rate is minimized with fewer than 100 trees. The most important variables are: alcohol, sulphates, sulfur dioxide, and volatile acidity.

The Random Forest model is quite effective, with an error rate of 30%. Let's see how this model performs on the test data.

Outcome and Conclusion

```
set.seed(123, sample.kind ="Rounding")
```

```
## Warning in set.seed(123, sample.kind = "Rounding"): non-uniform 'Rounding'
## sampler used
```

```
final_prediction <- predict(wine_rf, test_wines)

confMat <- table(actual = test_wines$quality, predicted = final_prediction)
confMat
```

```
##       predicted
## actual   3   4   5   6   7   8
##      3   0   0   1   1   0   0
##      4   1   0   7   6   0   0
##      5   0   0 106  25   2   0
##      6   0   1  26  93   7   1
##      7   0   0   2  21  16   1
##      8   0   0   0   2   2   0
```

```
accuracy <- sum(diag(confMat))/sum(confMat)
accuracy
```

```
## [1] 0.6697819
```

The Random Forest model is 67% accurate, which is an improvement over KNN (KNN1 with 6 features - 60%). The model is quite effective at predicting wine quality - I definitely couldn't "taste" a wine (or see a list of it's properties) and guess the the correct quality score on a scale from one to ten 67% of the time!

Here are a few takeaways to keep in mind next time you're shopping for wine:

- Stay away from wines with a low alcohol percentage. Higher alcohol is almost always better.
- Sulphates aren't a bad thing. Sulphates are natural compounds produced during fermentation (more can be added to preserve freshness)
- Volatile acidity (acetic acid) will spoil a wine. Avoid wines that have been poorly stored or exposed to air.
- There are good wines across all sweetness levels (residual sugars) - drink whatever suits your taste!

Thanks for reading,

Ryan

Data Source

P. Cortez, A. Cerdeira, F. Almeida, T. Matos and J. Reis. Modeling wine preferences by data mining from physicochemical properties. In Decision Support Systems, Elsevier, 47(4):547-553, 2009.