Capstone Project: Wine Quality

Isaiah Lemons

12/31/2019

Introduction

This project is part of the HarvardX:PH125.9x Data Science Capstone project. There are two datasets used that provide multiple physiochemical tests based on red and white wine samples that came from northern Portugal. The goal of this project is to develop machine learning algorithms based on all the physiochemical test results provided, in attempt to predict if a certain wine will be of high quality.

The datasets used can be found at the link below. (https://archive.ics.uci.edu/ml/machine-learning-databases/wine-quality/).

Data Wrangling

```
# Install any neccessary libraries
if(!require(caret)) install.packages("caret", repos = "http://cran.us.r-project.org")
if(!require(tidyverse)) install.packages("tidyverse", repos = "http://cran.us.r-project.org")
```

The following code was used to import the data and split it into test and training sets for later models.

```
# Import the Red and White datasets
url_red <- "https://archive.ics.uci.edu/ml/machine-learning-databases/wine-quality/winequality-red.csv"
url_white <- "https://archive.ics.uci.edu/ml/machine-learning-databases/wine-quality/winequality-white.

red_data <- read.csv(url_red, sep=';')
white_data <- read.csv(url_white, sep=';')

# Merge the two separate red and white wine datasets into one dataset
wine <- rbind(red_data, white_data)

# Adding a column to classify an excellent wine quality
table(wine$quality)
```

```
##
##    3    4    5    6    7    8    9
##   30  216 2138 2836 1079  193    5
```

```
wine <- wine %>% mutate(Excellent = ifelse(quality > 6, 1, 0))
wine$Excellent <- as.factor(wine$Excellent)

# remove files no longer necessary
rm(url_red, url_white, red_data, white_data)

# Splitting data into test and train sets 80/20 split
set.seed(42)
#set.seed(1, sample.kind="Rounding") #if using R 3.5 or later
```

```r
test_index <- createDataPartition(wine$Excellent, times = 1, p = 0.2, list = FALSE)
train_set <- wine[-test_index,]
test_set <- wine[test_index,]
```

Exploratory Analysis

Once the data is available initial analysis and research the dataset can begin. Using the code below we can see that this dataset is in tidy format, and it contains 6497 rows and 13 columns.

```r
# data is in tidy format
wine %>% as.tibble()
```

```
## # A tibble: 6,497 x 13
##    fixed.acidity volatile.acidity citric.acid residual.sugar chlorides
##            <dbl>            <dbl>       <dbl>          <dbl>     <dbl>
##  1           7.4            0.7          0               1.9     0.076
##  2           7.8            0.88         0               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            0.7          0               1.9     0.076
##  6           7.4            0.66         0               1.8     0.075
##  7           7.9            0.6          0.06            1.6     0.069
##  8           7.3            0.65         0               1.2     0.065
##  9           7.8            0.580        0.02            2       0.073
## 10           7.5            0.5          0.36            6.1     0.071
## # ... with 6,487 more rows, and 8 more variables: free.sulfur.dioxide <dbl>,
## #   total.sulfur.dioxide <dbl>, density <dbl>, pH <dbl>, sulphates <dbl>,
## #   alcohol <dbl>, quality <int>, Excellent <fct>
```

```r
# checking the structure of the data
str(wine)
```

```
## 'data.frame':    6497 obs. of  13 variables:
##  $ fixed.acidity       : num  7.4 7.8 7.8 11.2 7.4 7.4 7.9 7.3 7.8 7.5 ...
##  $ volatile.acidity    : num  0.7 0.88 0.76 0.28 0.7 0.66 0.6 0.65 0.58 0.5 ...
##  $ citric.acid         : num  0 0 0.04 0.56 0 0 0.06 0 0.02 0.36 ...
##  $ residual.sugar      : num  1.9 2.6 2.3 1.9 1.9 1.8 1.6 1.2 2 6.1 ...
##  $ chlorides           : num  0.076 0.098 0.092 0.075 0.076 0.075 0.069 0.065 0.073 0.071 ...
##  $ free.sulfur.dioxide : num  11 25 15 17 11 13 15 15 9 17 ...
##  $ total.sulfur.dioxide: num  34 67 54 60 34 40 59 21 18 102 ...
##  $ density             : num  0.998 0.997 0.997 0.998 0.998 ...
##  $ pH                  : num  3.51 3.2 3.26 3.16 3.51 3.51 3.3 3.39 3.36 3.35 ...
##  $ sulphates           : num  0.56 0.68 0.65 0.58 0.56 0.56 0.46 0.47 0.57 0.8 ...
##  $ alcohol             : num  9.4 9.8 9.8 9.8 9.4 9.4 9.4 10 9.5 10.5 ...
##  $ quality             : int  5 5 5 6 5 5 5 7 7 5 ...
##  $ Excellent           : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 2 2 1 ...
```

```r
# checking basic summary statistics
summary(wine)
```

```
##  fixed.acidity    volatile.acidity  citric.acid     residual.sugar
```

```
##  Min.   : 3.800   Min.   :0.0800   Min.   :0.0000   Min.   : 0.600
##  1st Qu.: 6.400   1st Qu.:0.2300   1st Qu.:0.2500   1st Qu.: 1.800
##  Median : 7.000   Median :0.2900   Median :0.3100   Median : 3.000
##  Mean   : 7.215   Mean   :0.3397   Mean   :0.3186   Mean   : 5.443
##  3rd Qu.: 7.700   3rd Qu.:0.4000   3rd Qu.:0.3900   3rd Qu.: 8.100
##  Max.   :15.900   Max.   :1.5800   Max.   :1.6600   Max.   :65.800
##    chlorides       free.sulfur.dioxide total.sulfur.dioxide    density
##  Min.   :0.00900   Min.   :  1.00      Min.   :  6.0        Min.   :0.9871
##  1st Qu.:0.03800   1st Qu.: 17.00      1st Qu.: 77.0        1st Qu.:0.9923
##  Median :0.04700   Median : 29.00      Median :118.0        Median :0.9949
##  Mean   :0.05603   Mean   : 30.53      Mean   :115.7        Mean   :0.9947
##  3rd Qu.:0.06500   3rd Qu.: 41.00      3rd Qu.:156.0        3rd Qu.:0.9970
##  Max.   :0.61100   Max.   :289.00      Max.   :440.0        Max.   :1.0390
##        pH          sulphates         alcohol         quality      Excellent
##  Min.   :2.720   Min.   :0.2200   Min.   : 8.00   Min.   :3.000   0:5220
##  1st Qu.:3.110   1st Qu.:0.4300   1st Qu.: 9.50   1st Qu.:5.000   1:1277
##  Median :3.210   Median :0.5100   Median :10.30   Median :6.000
##  Mean   :3.219   Mean   :0.5313   Mean   :10.49   Mean   :5.818
##  3rd Qu.:3.320   3rd Qu.:0.6000   3rd Qu.:11.30   3rd Qu.:6.000
##  Max.   :4.010   Max.   :2.0000   Max.   :14.90   Max.   :9.000
```

```r
# Number of rows and columns
nrow(wine)
```

```
## [1] 6497
```

```r
ncol(wine)
```

```
## [1] 13
```

```r
# Check for missing values
any(is.na(wine))
```

```
## [1] FALSE
```

Installing additional libraries which may be useful for analysis and modeling

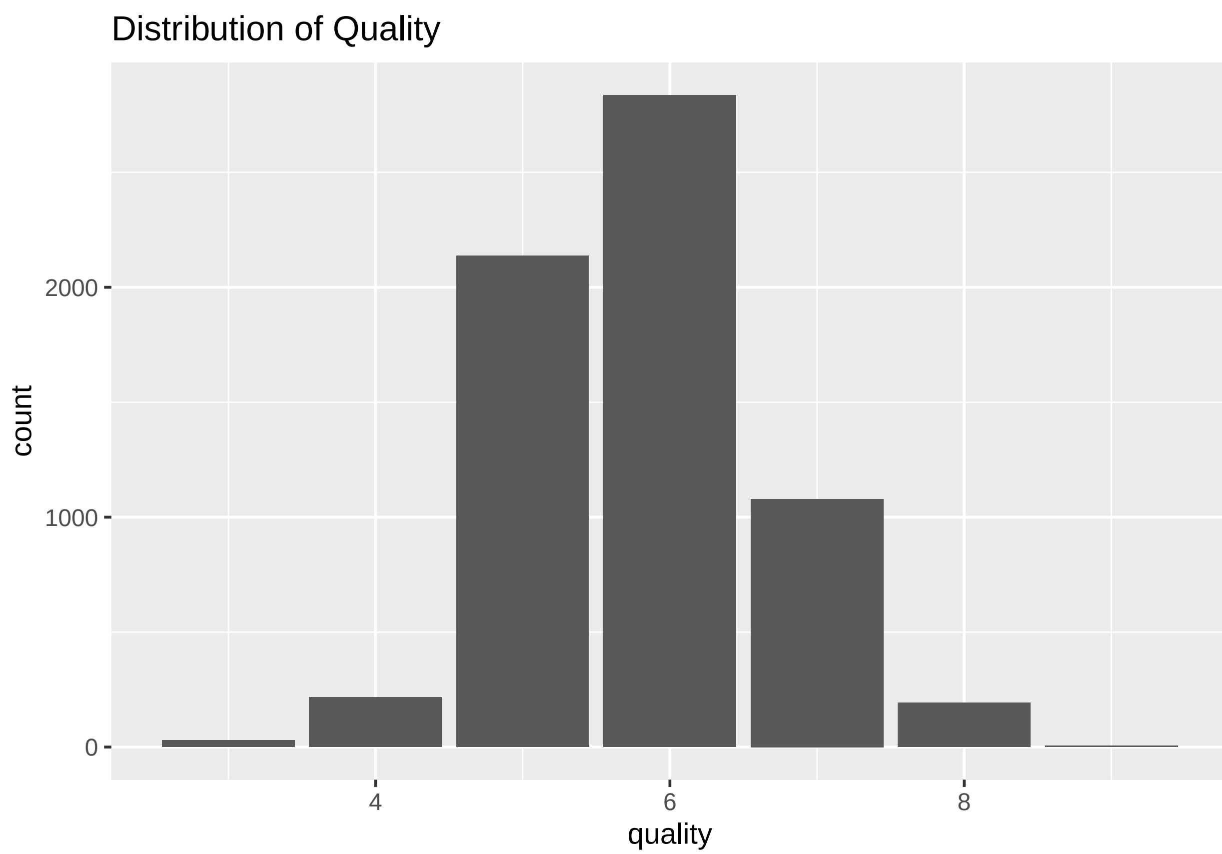
```r
if(!require(ggplot2)) install.packages("ggplot2", repos = "http://cran.us.r-project.org")
if(!require(reshape2)) install.packages("reshape2", repos = "http://cran.us.r-project.org")
if(!require(lubridate)) install.packages("lubridate", repos = "http://cran.us.r-project.org")
if(!require(randomForest)) install.packages("randomForest", repos = "http://cran.us.r-project.org")
if(!require(rpart)) install.packages("rpart", repos = "http://cran.us.r-project.org")
```

The first observation to check is the overall distribution of wines based on their quality.

```r
# Overall Average Quality
mean(wine$quality)
```

```
## [1] 5.818378
```

```r
# Distribution in Quality
wine %>%
  ggplot(aes(quality)) +
  geom_bar() +
  ggtitle("Distribution of Quality")
```



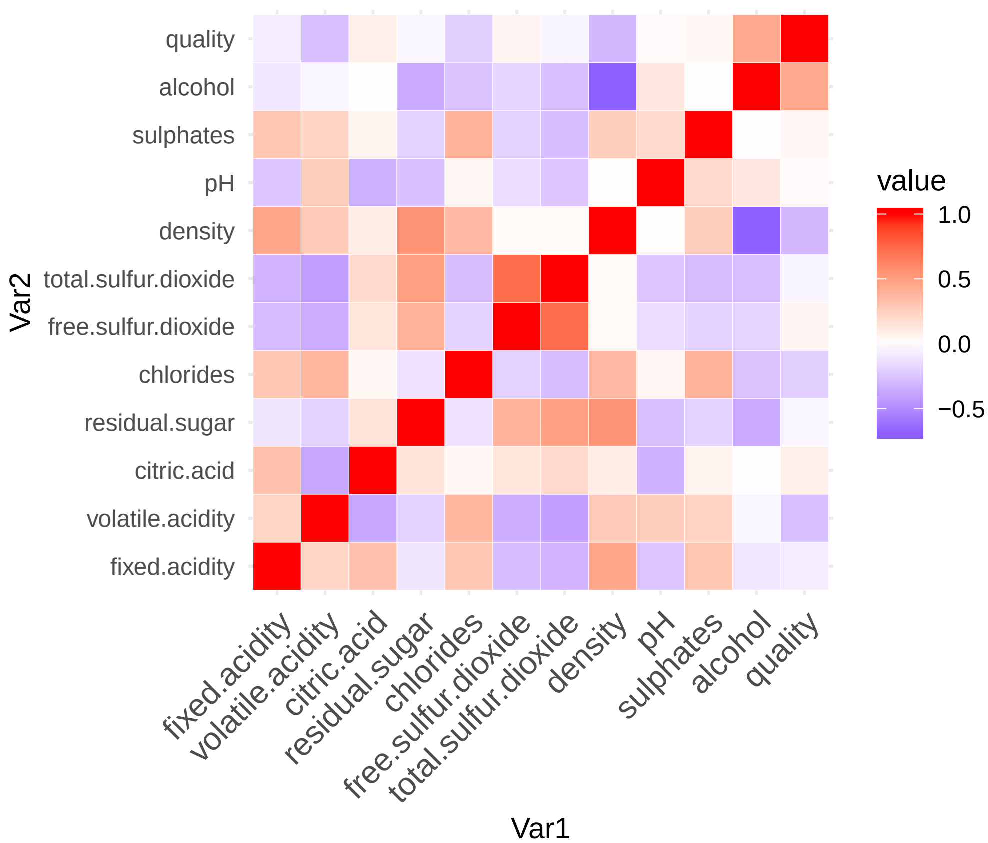

```r
# Percentage of Excellent wines
mean(wine$Excellent == 1)
```

```
## [1] 0.1965523
```

Based on the following code creating a heatmap can allow for pin-pointing a few attributes that have higher correlations than others. These variables may play a bigger part in predictions later on, so it's good to take a further look. (alcohol, total.sulfur.dioxide, free.sulfur.dioxide, residual.sugar, and density)

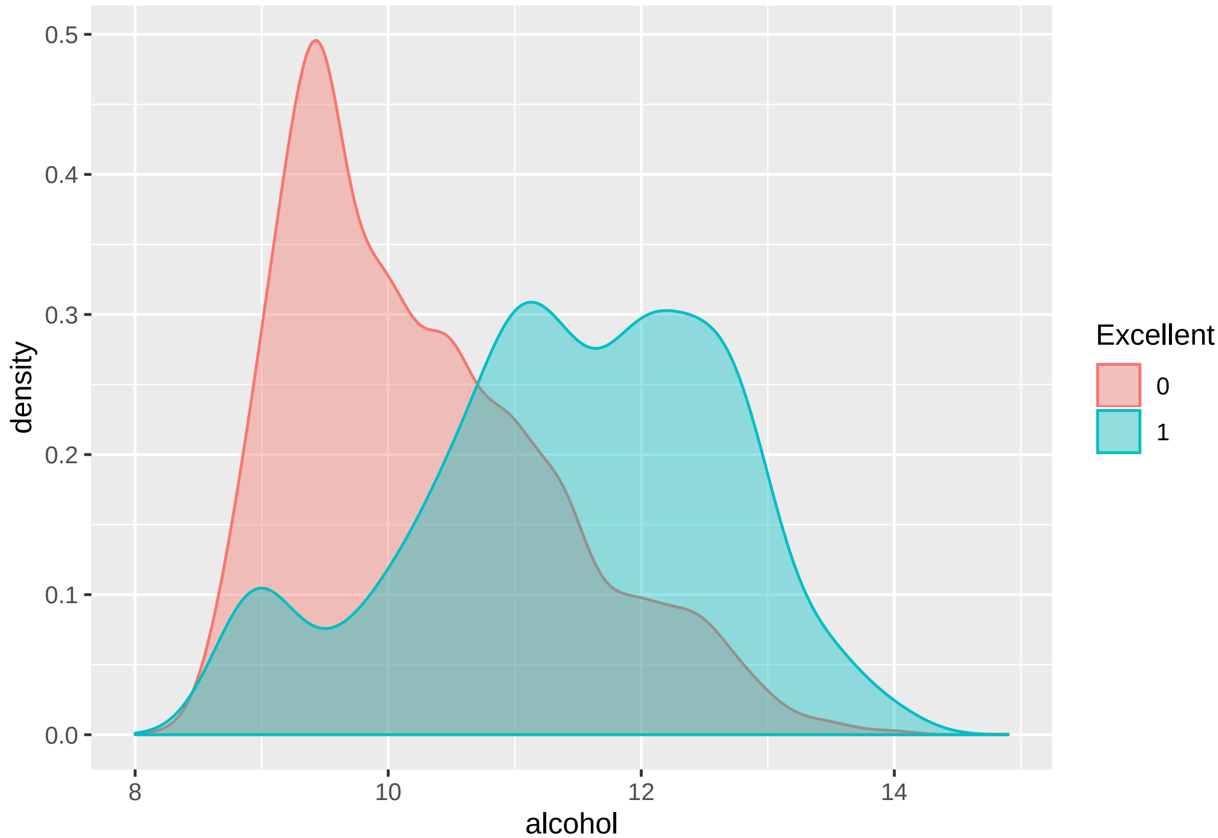
```r
train.cor <- cor(subset(wine, select=-c(Excellent)))

ggplot(melt(train.cor), aes(Var1, Var2, fill=value)) +
  geom_tile(color = "white") + #color white is for border
  scale_fill_gradient2(low="blue", high="red", mid="white") +
  theme_minimal() + # minimal theme
  theme(axis.text.x = element_text(angle = 45, vjust = 1, size = 12, hjust = 1)) +
  coord_fixed()
```

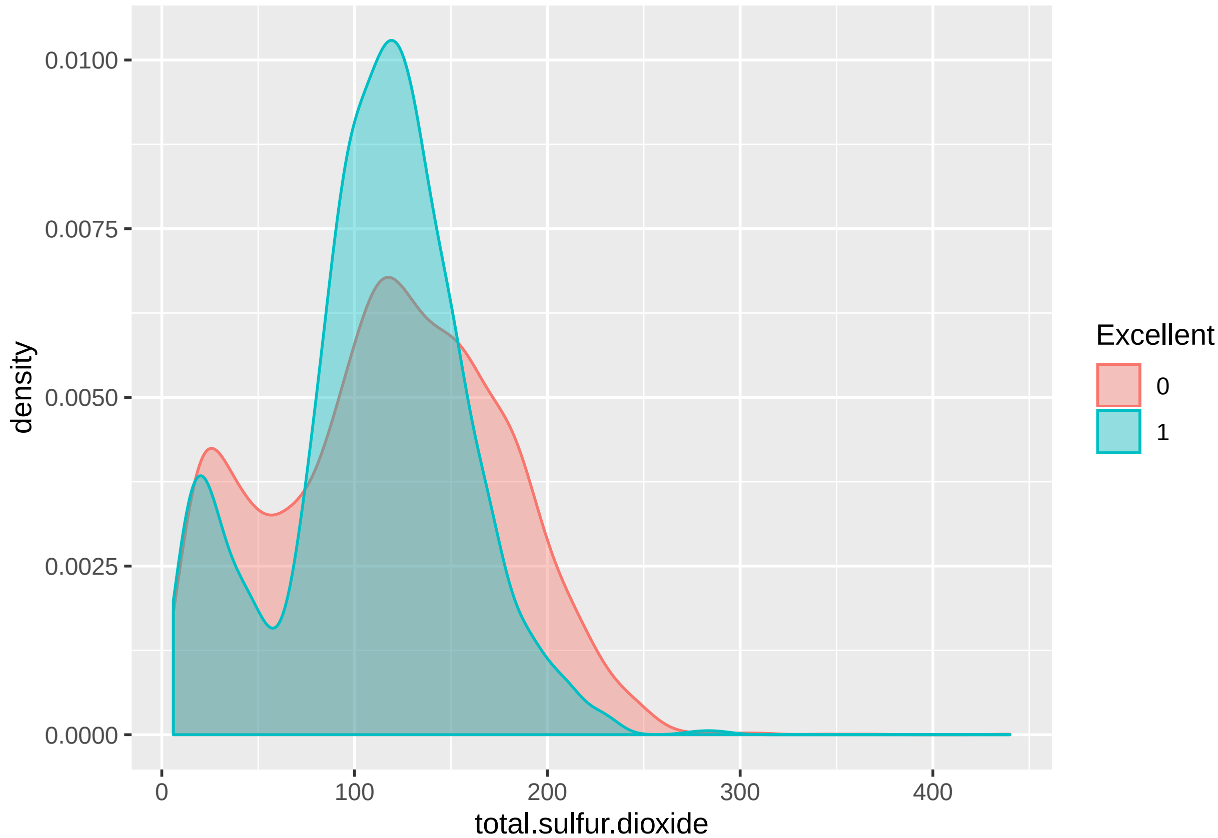


The below density plots check each of the physichemical tests to give a better understanding of their distributions. The flag created during data wrangling portion which specifies if a wine was considered excellent or not can also be added to give further insights.

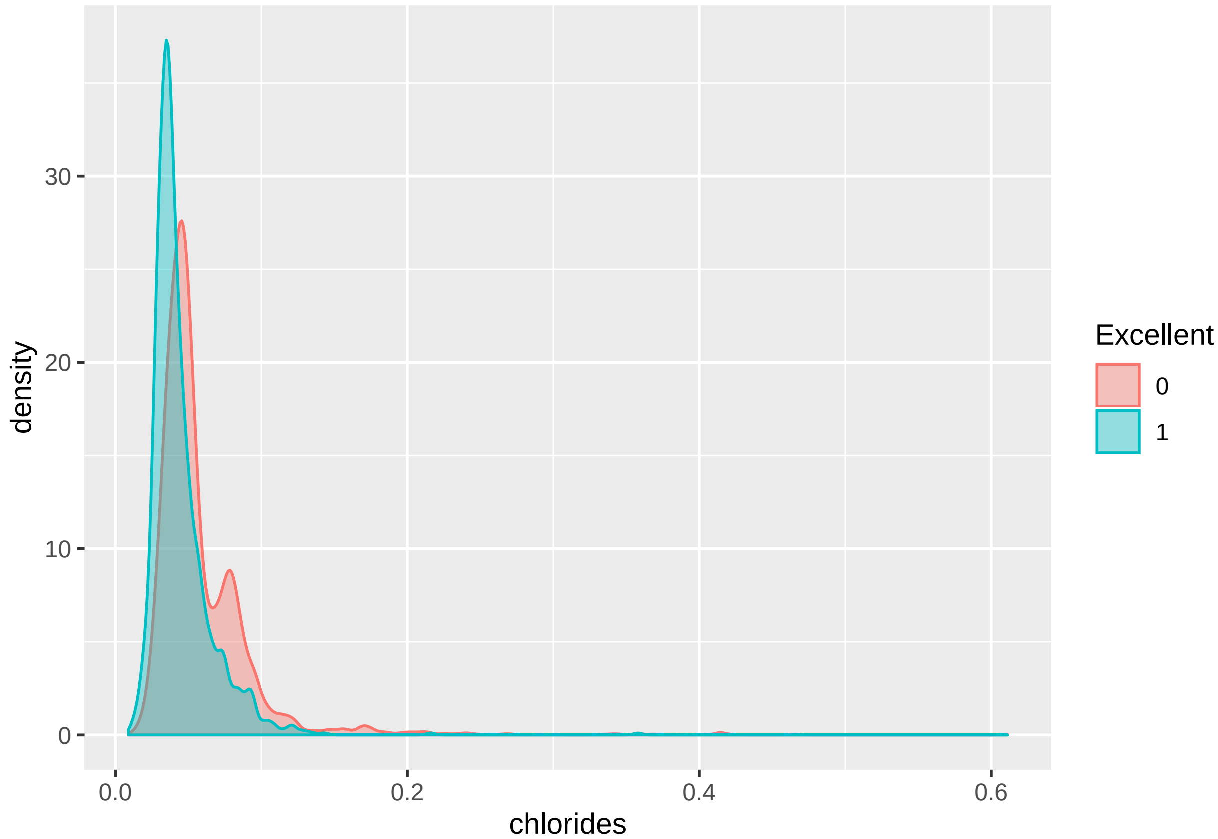
```
ggplot(wine, aes(alcohol, color=Excellent, fill=Excellent)) +
  geom_density(alpha = 0.4)
```



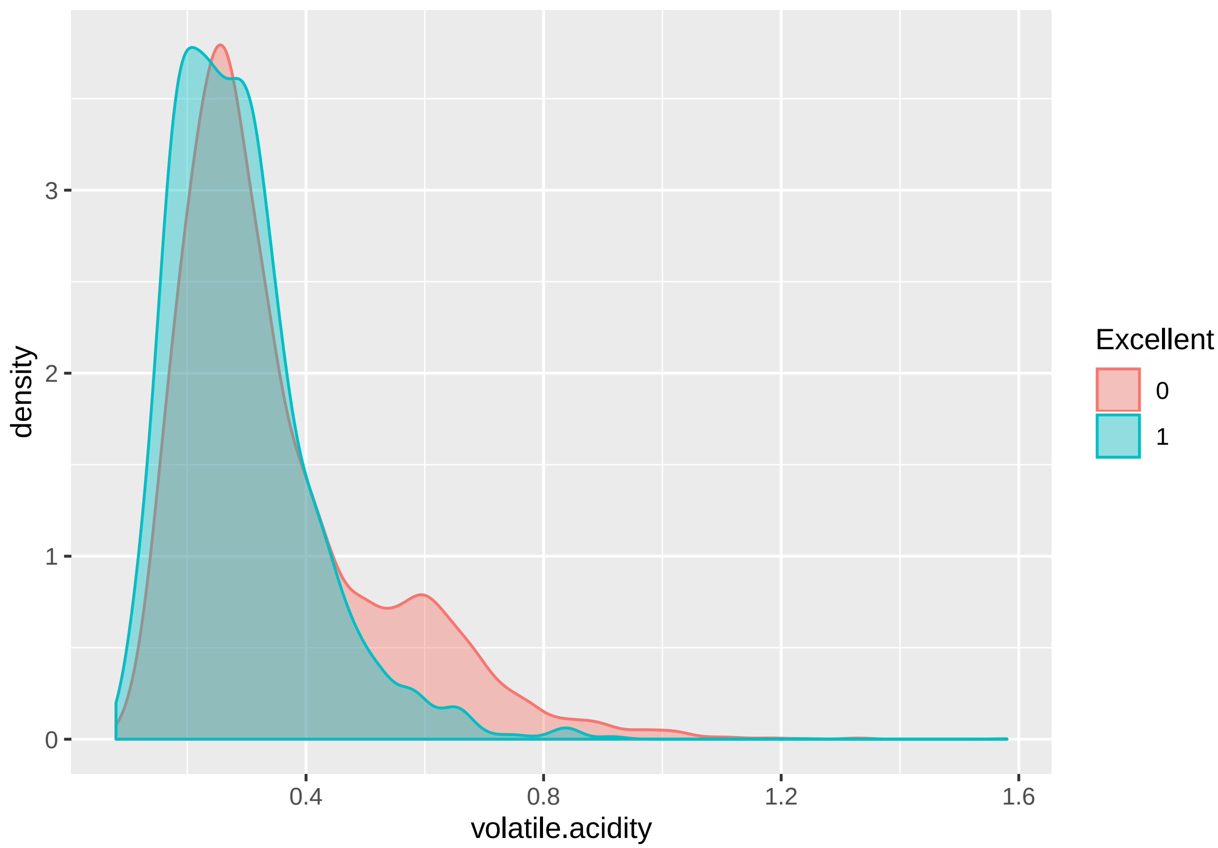

```r
ggplot(wine, aes(total.sulfur.dioxide, color=Excellent, fill=Excellent)) +
  geom_density(alpha = 0.4)
```



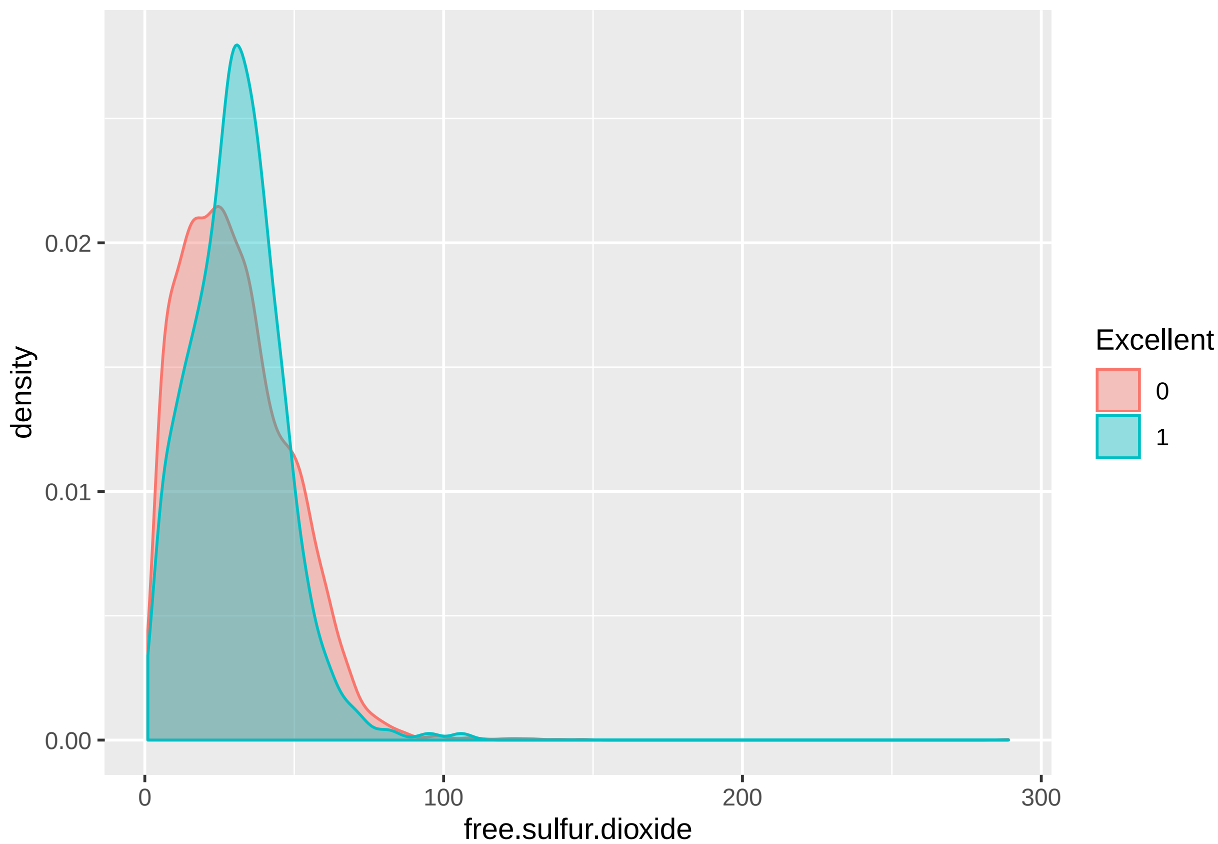

```r
ggplot(wine, aes(chlorides, color=Excellent, fill=Excellent)) +
  geom_density(alpha = 0.4)
```



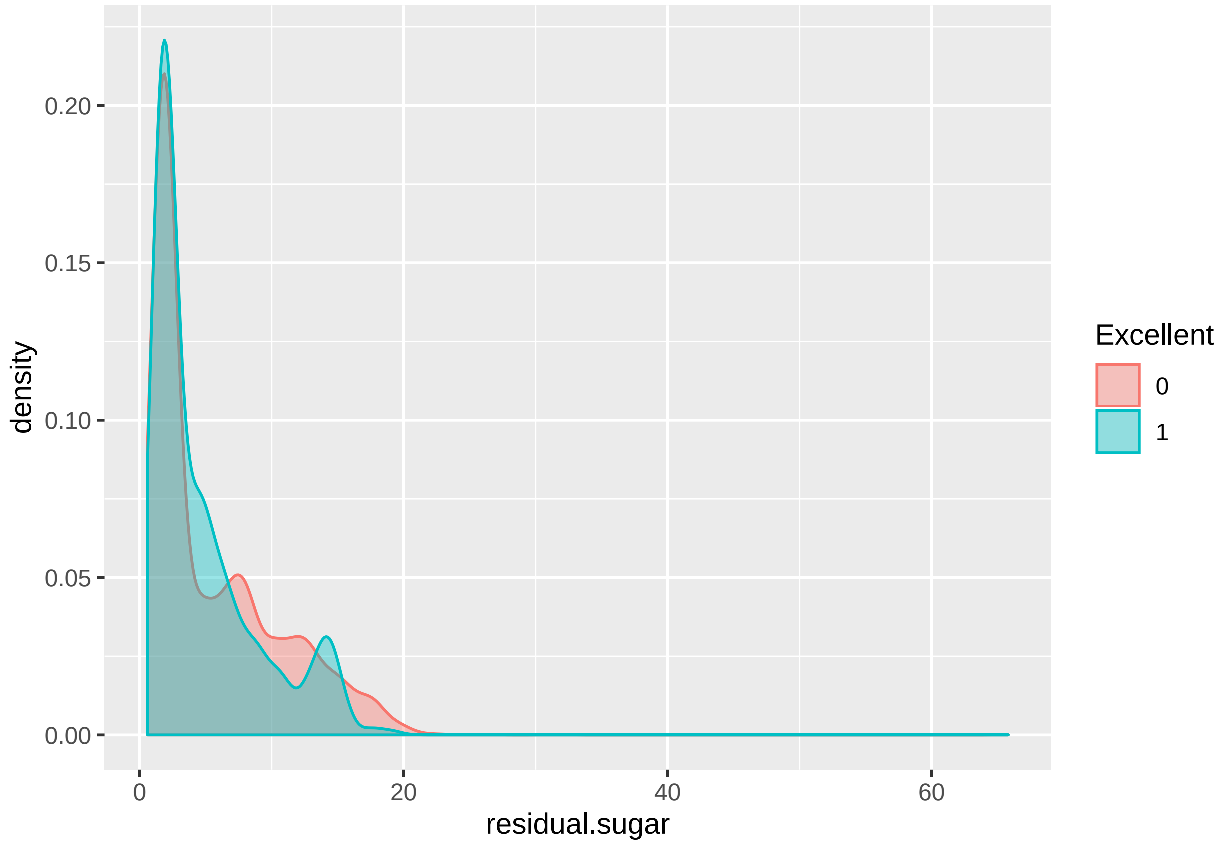

```
ggplot(wine, aes(volatile.acidity, color=Excellent, fill=Excellent)) +
  geom_density(alpha = 0.4)
```



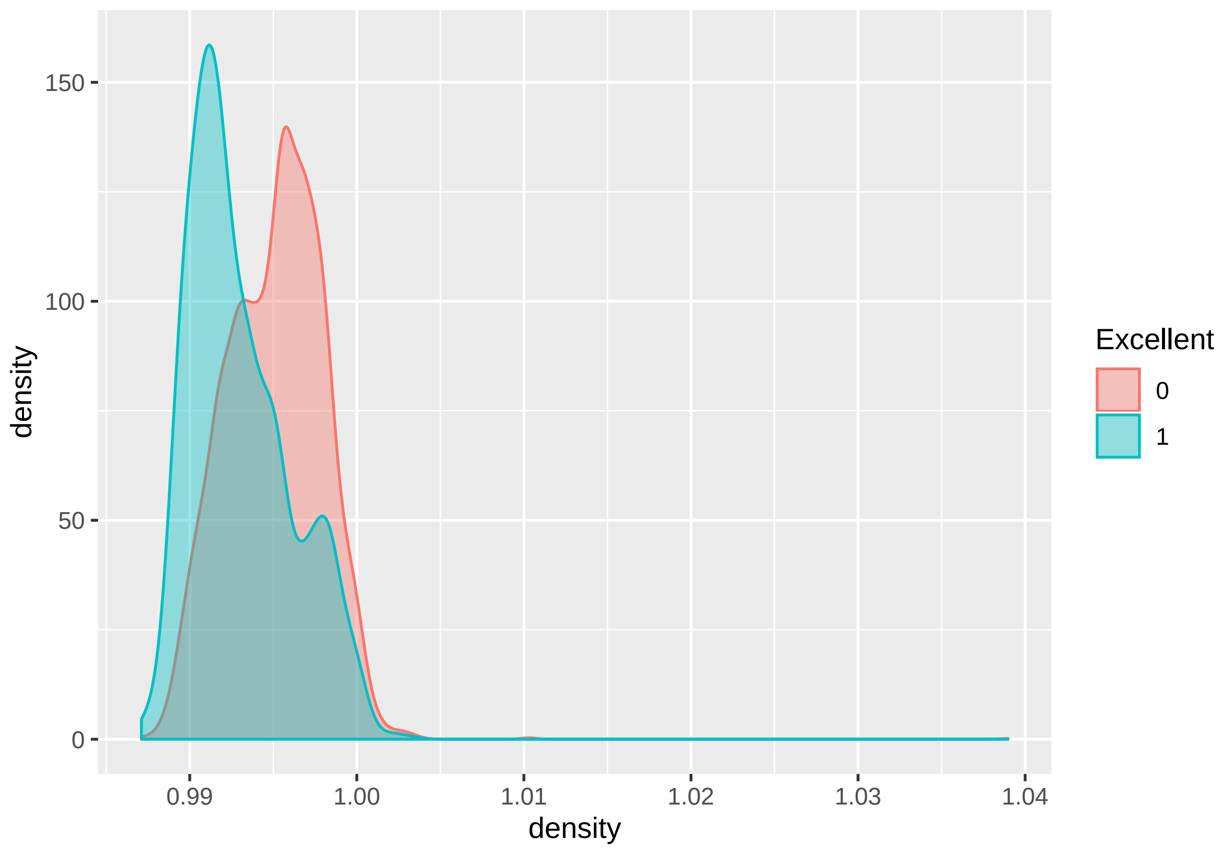

```r
ggplot(wine, aes(free.sulfur.dioxide, color=Excellent, fill=Excellent)) +
  geom_density(alpha = 0.4)
```



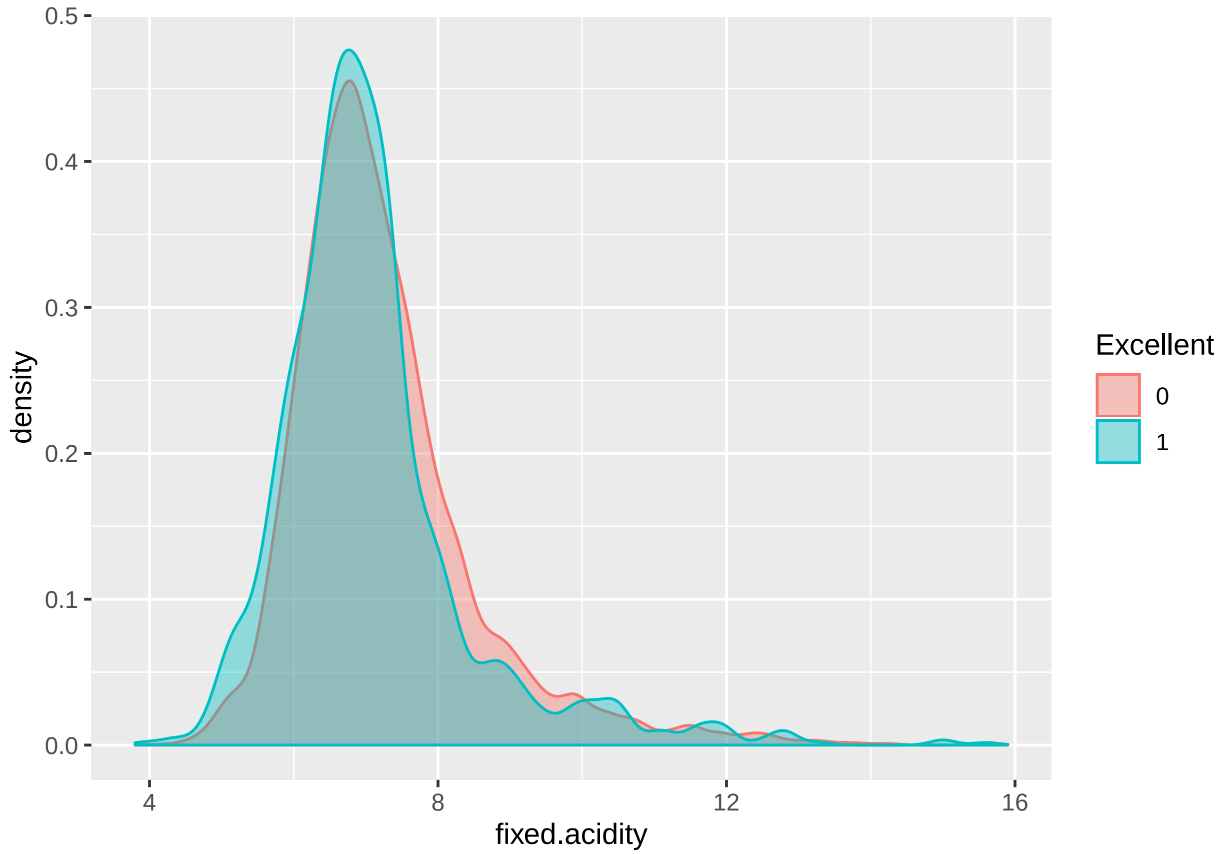

```r
ggplot(wine, aes(residual.sugar, color=Excellent, fill=Excellent)) +
  geom_density(alpha = 0.4)
```



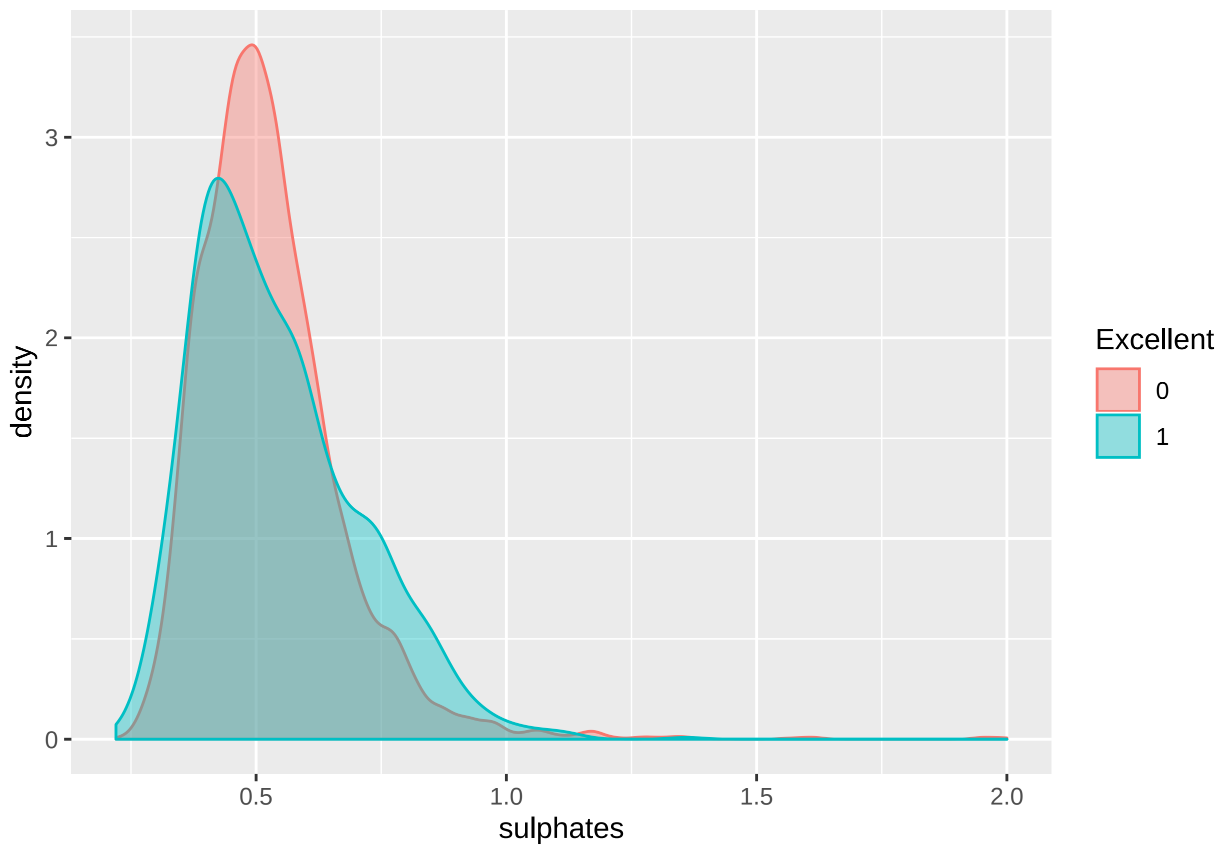

```r
ggplot(wine, aes(density, color=Excellent, fill=Excellent)) +
  geom_density(alpha = 0.4)
```



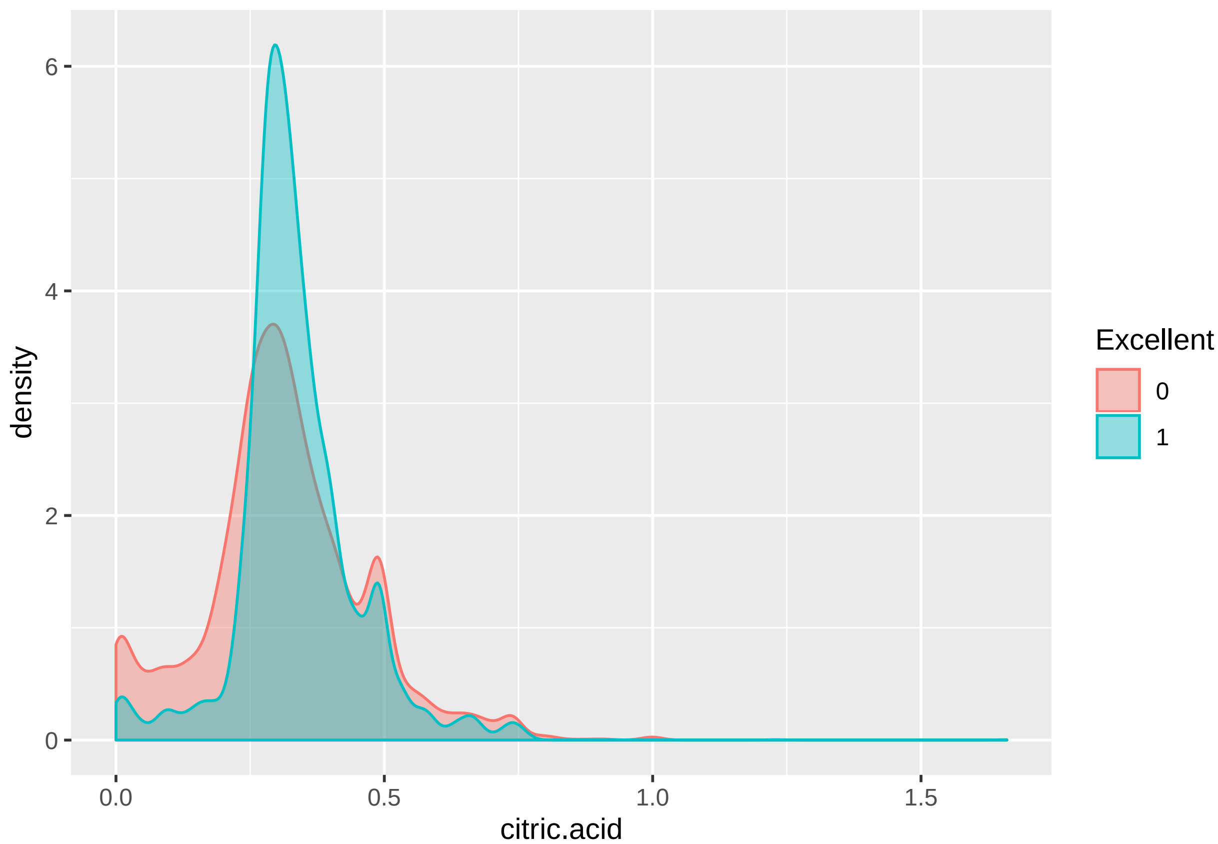

```
ggplot(wine, aes(fixed.acidity, color=Excellent, fill=Excellent)) +
  geom_density(alpha = 0.4)
```



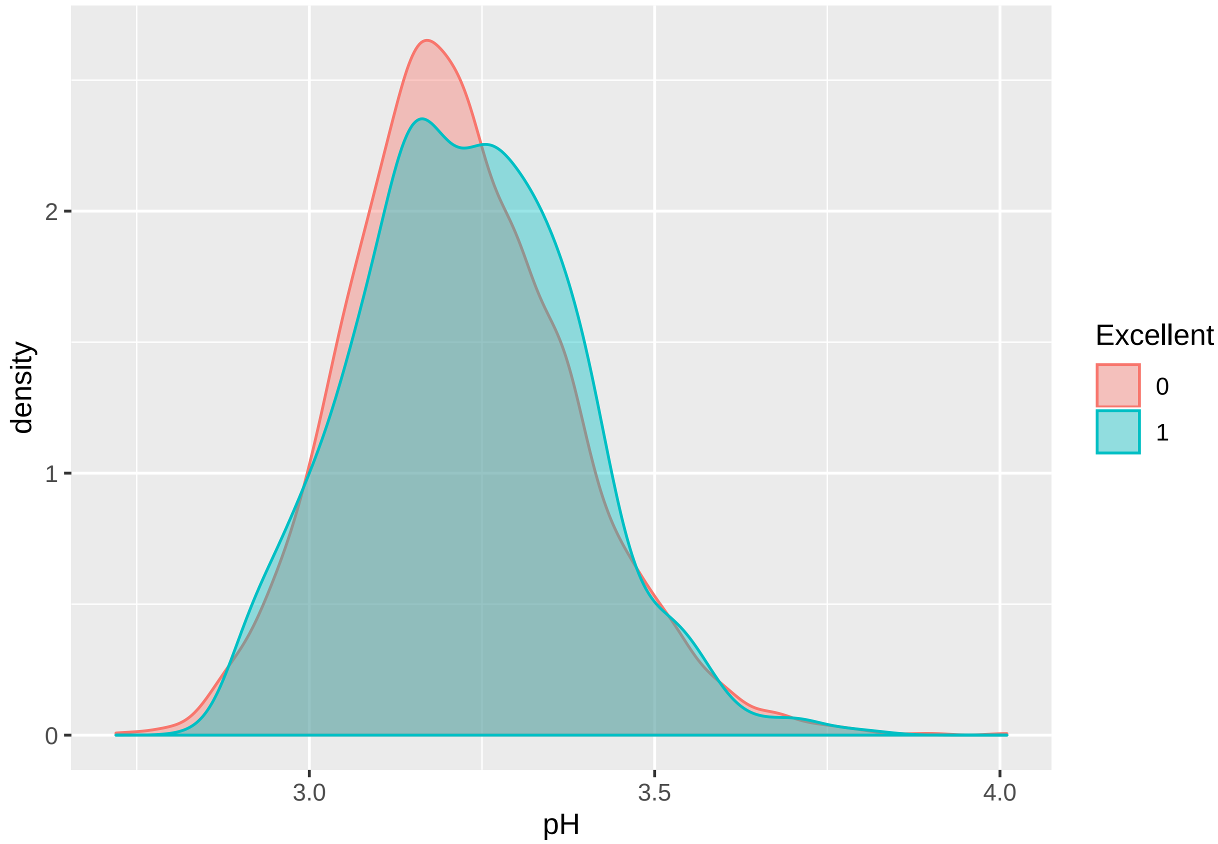

```
ggplot(wine, aes(sulphates, color=Excellent, fill=Excellent)) +
  geom_density(alpha = 0.4)
```




```r
ggplot(wine, aes(citric.acid, color=Excellent, fill=Excellent)) +
  geom_density(alpha = 0.4)
```

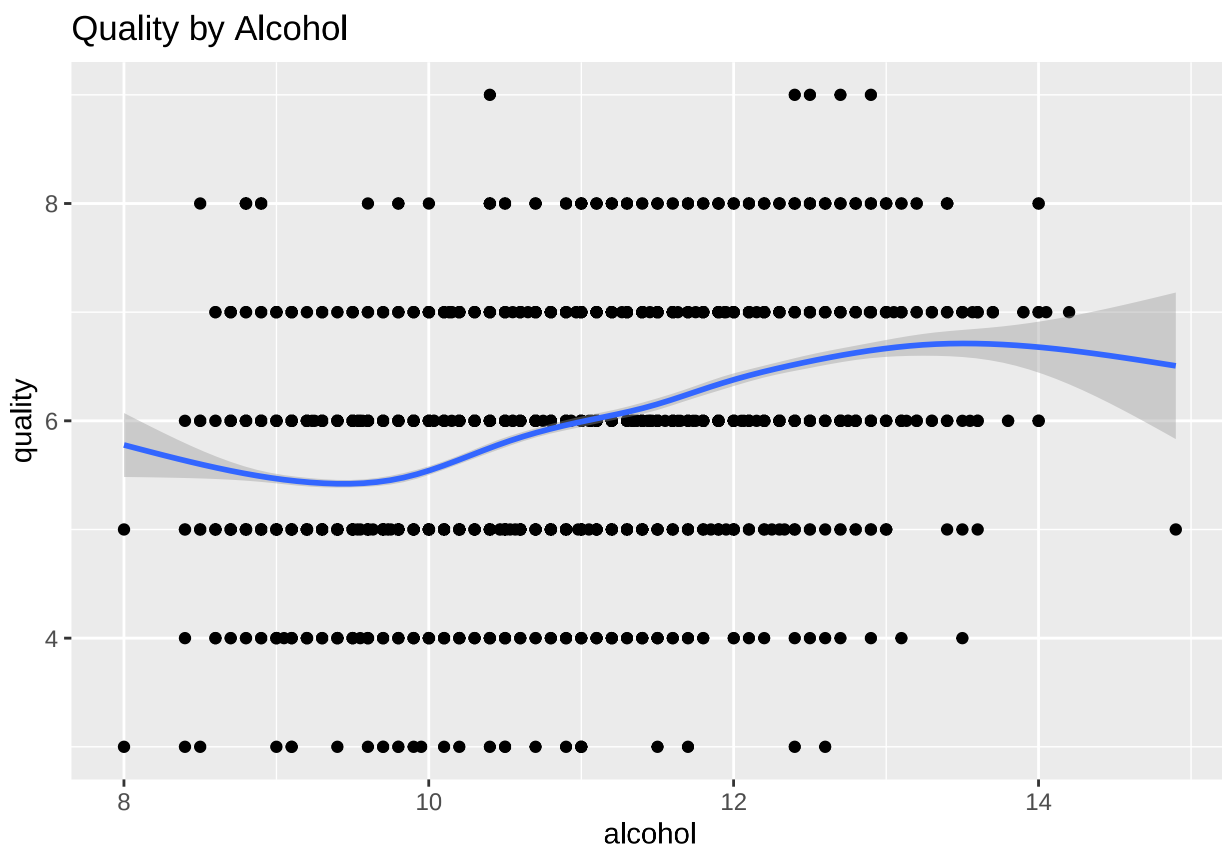



```r
ggplot(wine, aes(pH, color=Excellent, fill=Excellent)) +
  geom_density(alpha = 0.4)
```

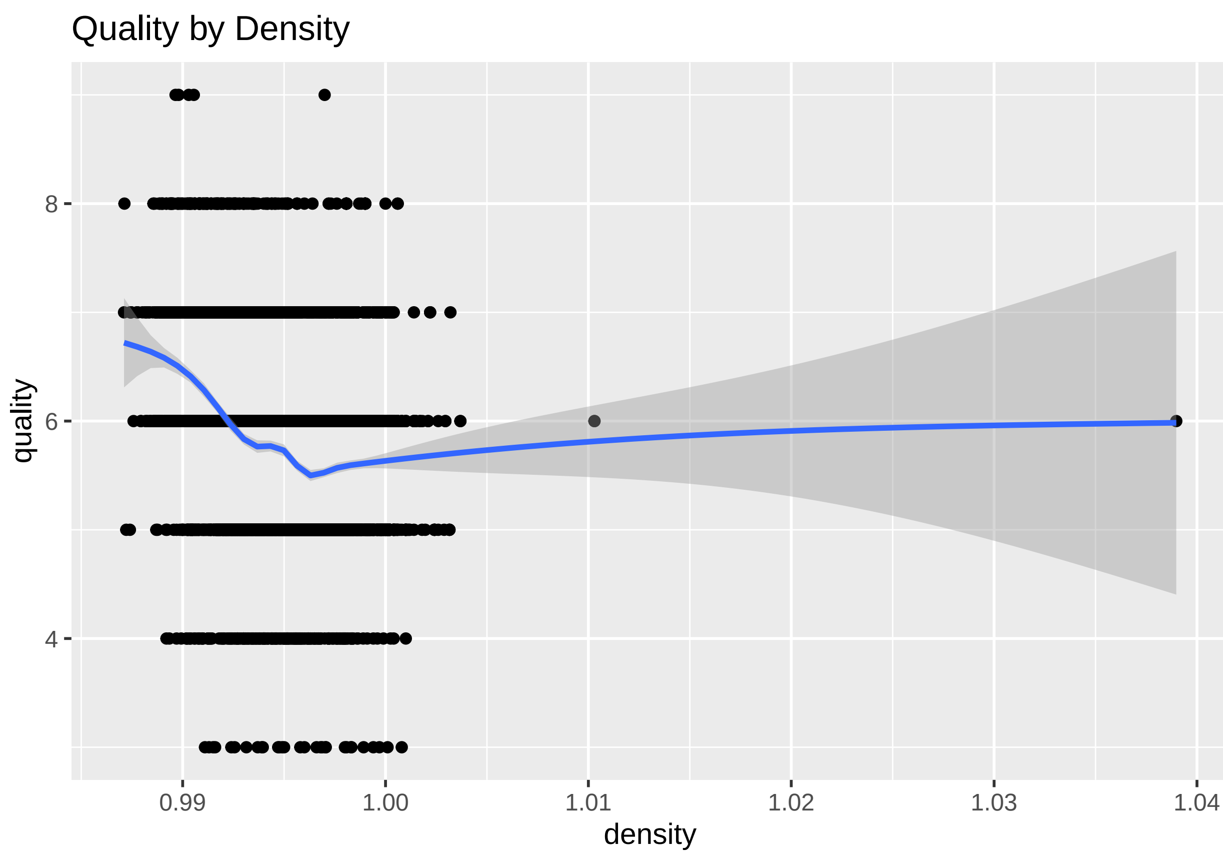


Taking a look at a couple scatterplots may show how certain attributes directly correlate with the quality. As shown in the plots below, when the alcohol content increases the quality generally increases as well. Density however, the quality tends to decrease as density increases.

```
ggplot(wine, aes(alcohol, quality)) +
  geom_point() +
  geom_smooth() +
  ggtitle("Quality by Alcohol")
```

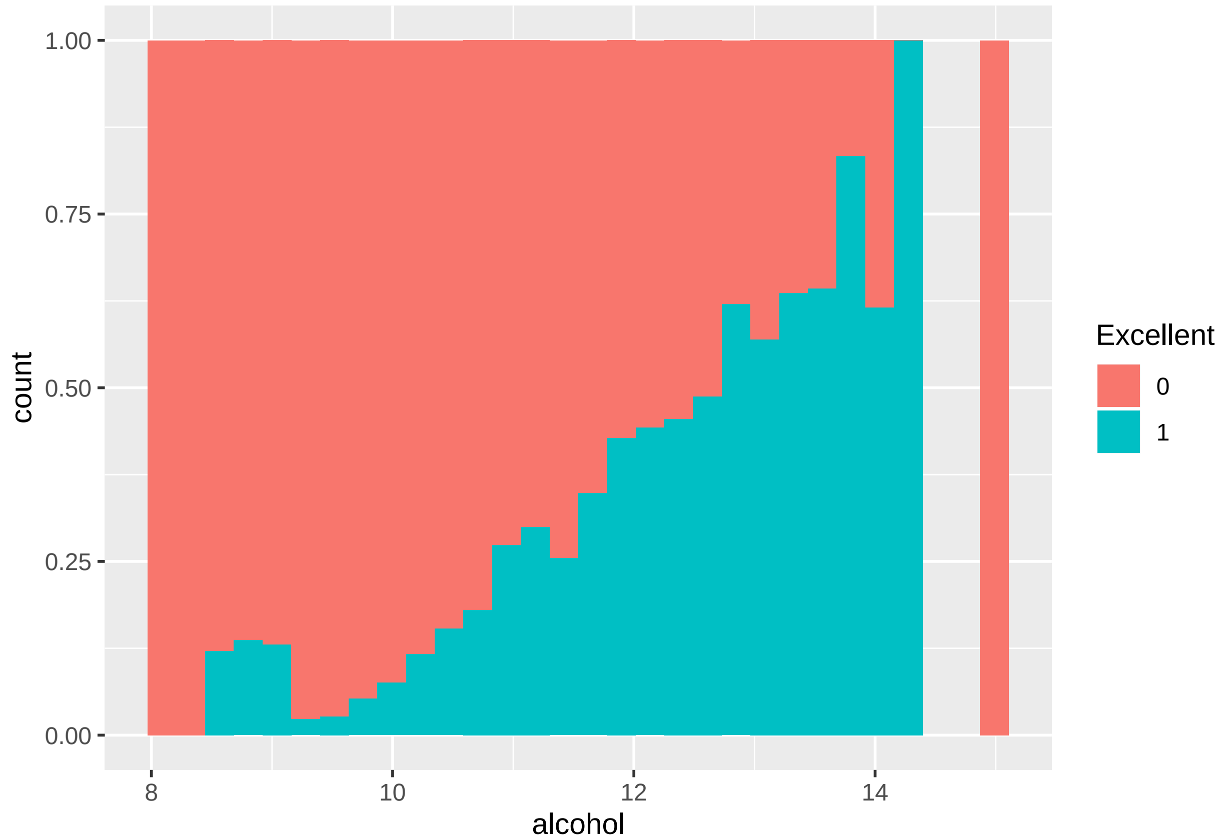



```r
ggplot(wine, aes(density, quality)) +
  geom_point() +
  geom_smooth() +
  ggtitle("Quality by Density")
```

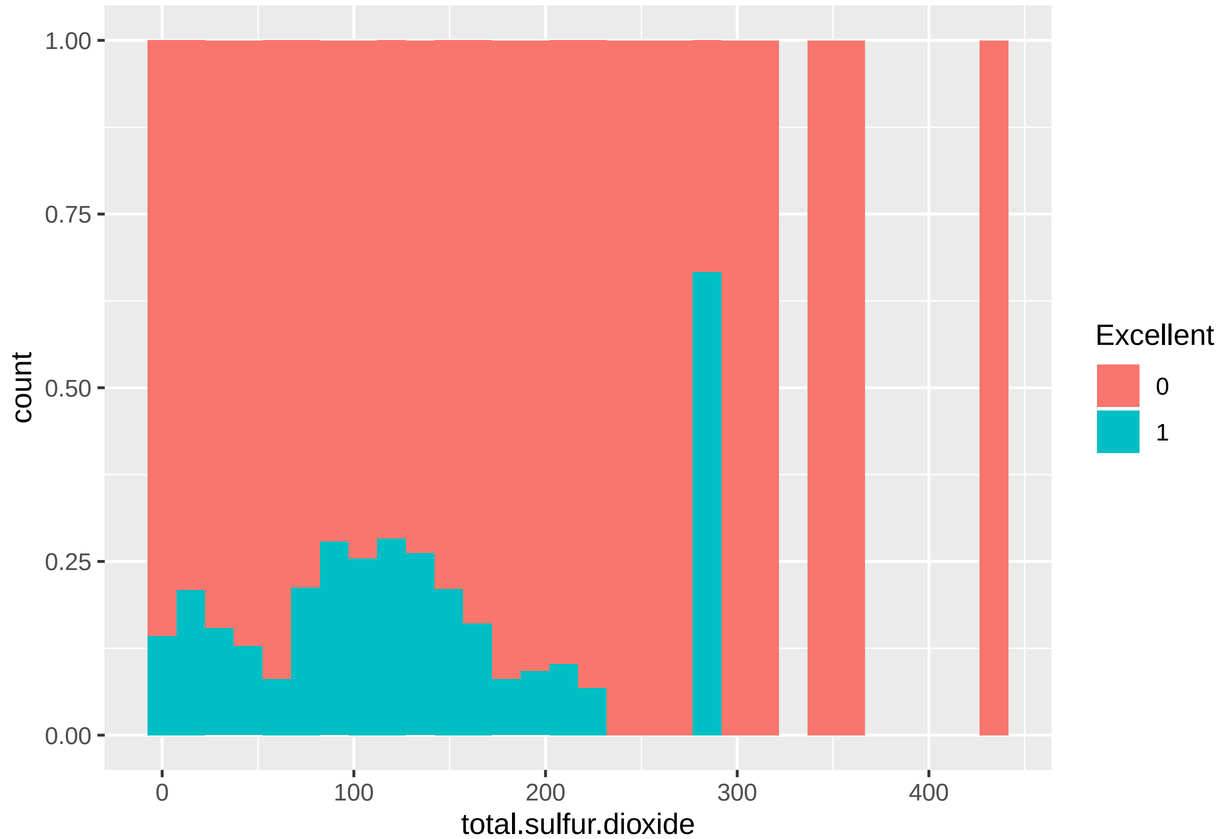


Stacked bar charts to easily show the percentage of excellent wines overall, based on particular variables. The images below continue to show similar results as previously mentioned, where higher quality wines tend to have higher alcohol content and lower densities.

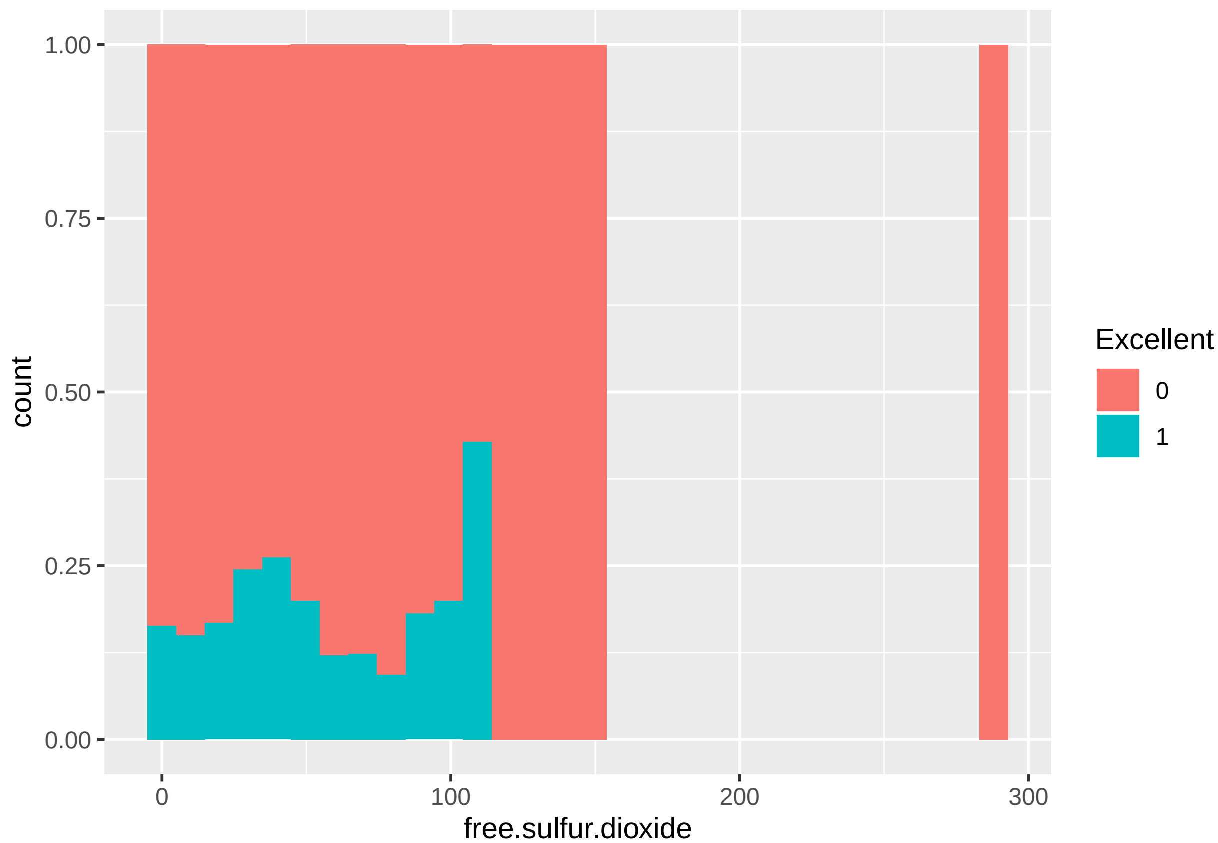
```
ggplot(wine, aes(alcohol, fill=Excellent)) +
  geom_histogram(bins=30, position="fill")
```



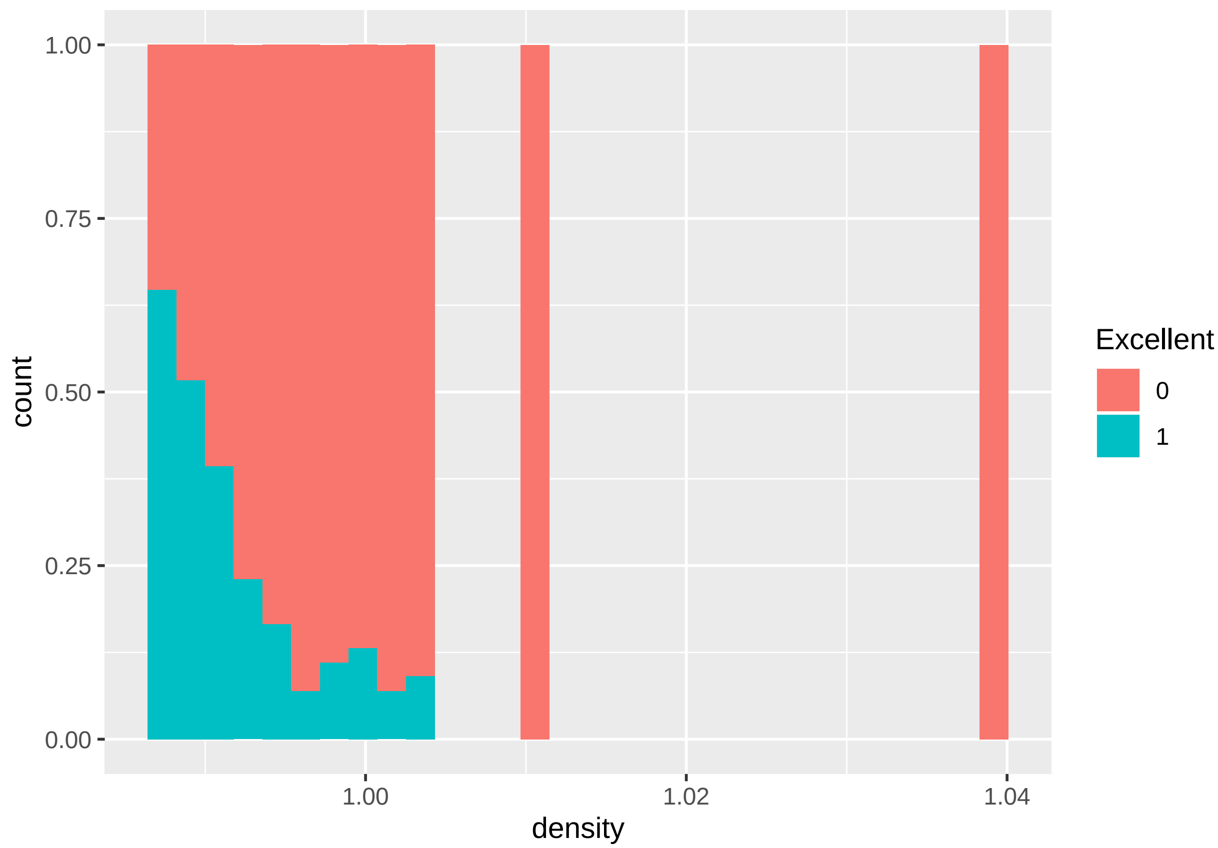

```r
ggplot(wine, aes(total.sulfur.dioxide, fill=Excellent)) +
  geom_histogram(bins=30, position="fill")
```



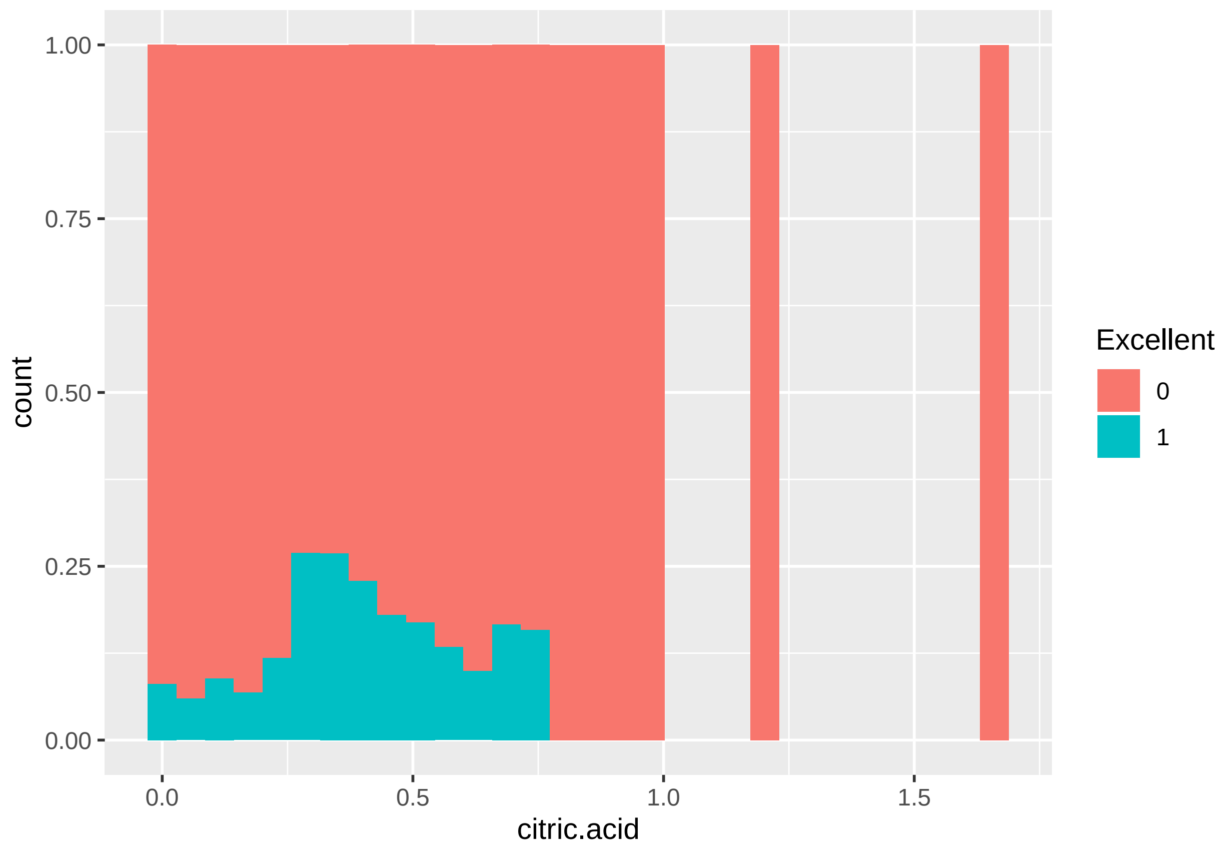

```r
ggplot(wine, aes(free.sulfur.dioxide, fill=Excellent)) +
  geom_histogram(bins=30, position="fill")
```




```
ggplot(wine, aes(density, fill=Excellent)) +
  geom_histogram(bins=30, position="fill")
```




```r
ggplot(wine, aes(citric.acid, fill=Excellent)) +
  geom_histogram(bins=30, position="fill")
```



Modeling

Each of the models below are using all the physiochemical tests variables available to predict if a wine will be considered excellent. The quality column is not included as a predictor of Excellent since it was directly used to create the classification flags.

```
# Logistic Regression Model
set.seed(1)
train_glm <- train(Excellent ~ .-quality, method = "glm", data = train_set)
glm_pred <- predict(train_glm, test_set, type = "raw")
confusionMatrix(glm_pred, test_set$Excellent)$overall[["Accuracy"]]
```

```
## [1] 0.8323077
```

```
# LDA Model
set.seed(1)
train_lda <- train(Excellent ~ .-quality, method = "lda", data = train_set)
lda_pred <- predict(train_lda, test_set)
confusionMatrix(lda_pred, test_set$Excellent)$overall[["Accuracy"]]
```

```
## [1] 0.8323077
```

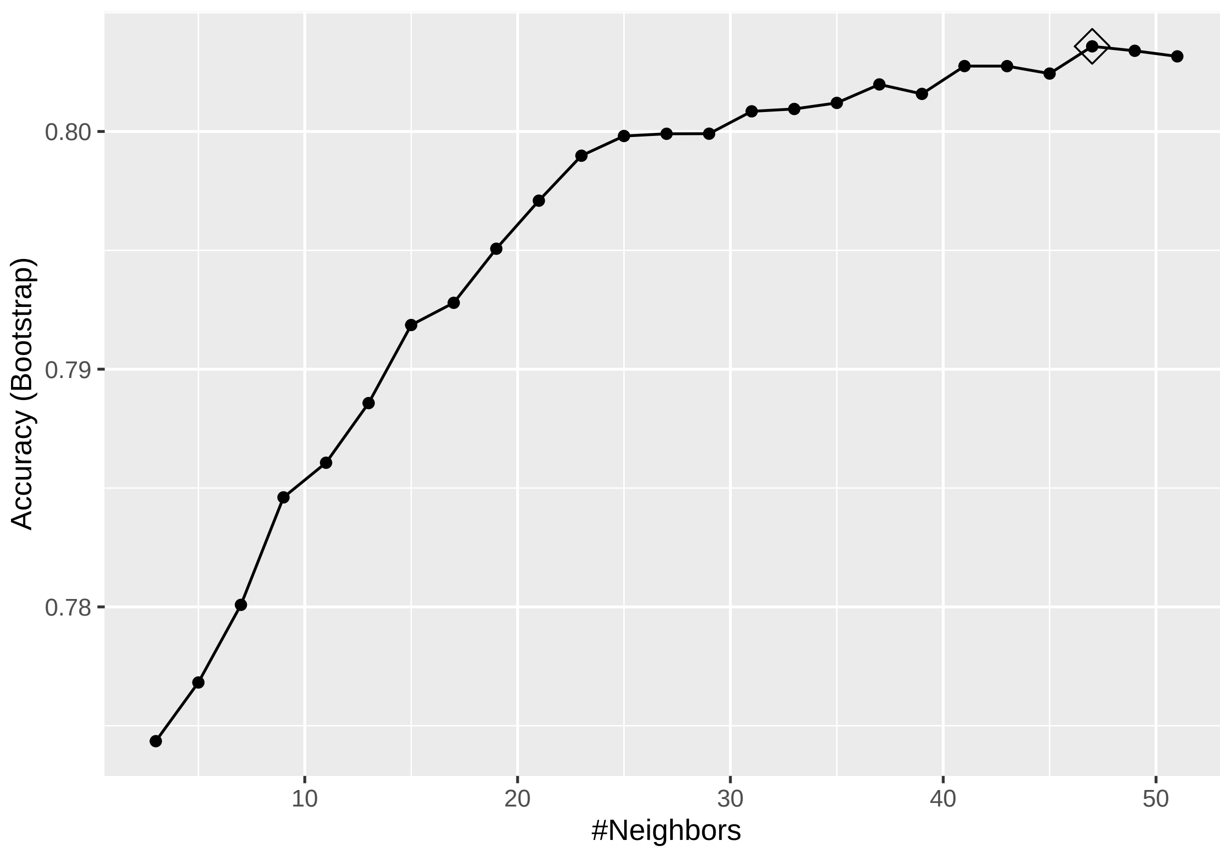
```r
# QDA Model
set.seed(1)
train_lda <- train(Excellent ~ .-quality, method = "qda", data = train_set)
qda_pred <- predict(train_lda, test_set)
mean(lda_pred == test_set$Excellent)
```

```
## [1] 0.8323077
```

```r
# Loess Model
set.seed(1)
train_loess <- train(Excellent ~ .-quality,
                     data = train_set,
                     method = "gamLoess")
loess_pred <- predict(train_loess, test_set)
mean(loess_pred == test_set$Excellent)
```

```
## [1] 0.8376923
```

```r
# K Nearest Neighbors Model
set.seed(1)
train_knn <- train(Excellent ~ .-quality,
                   method = "knn",
                   data = train_set,
                   tuneGrid = data.frame(k = seq(3,51,2)))
ggplot(train_knn, highlight = TRUE)
```



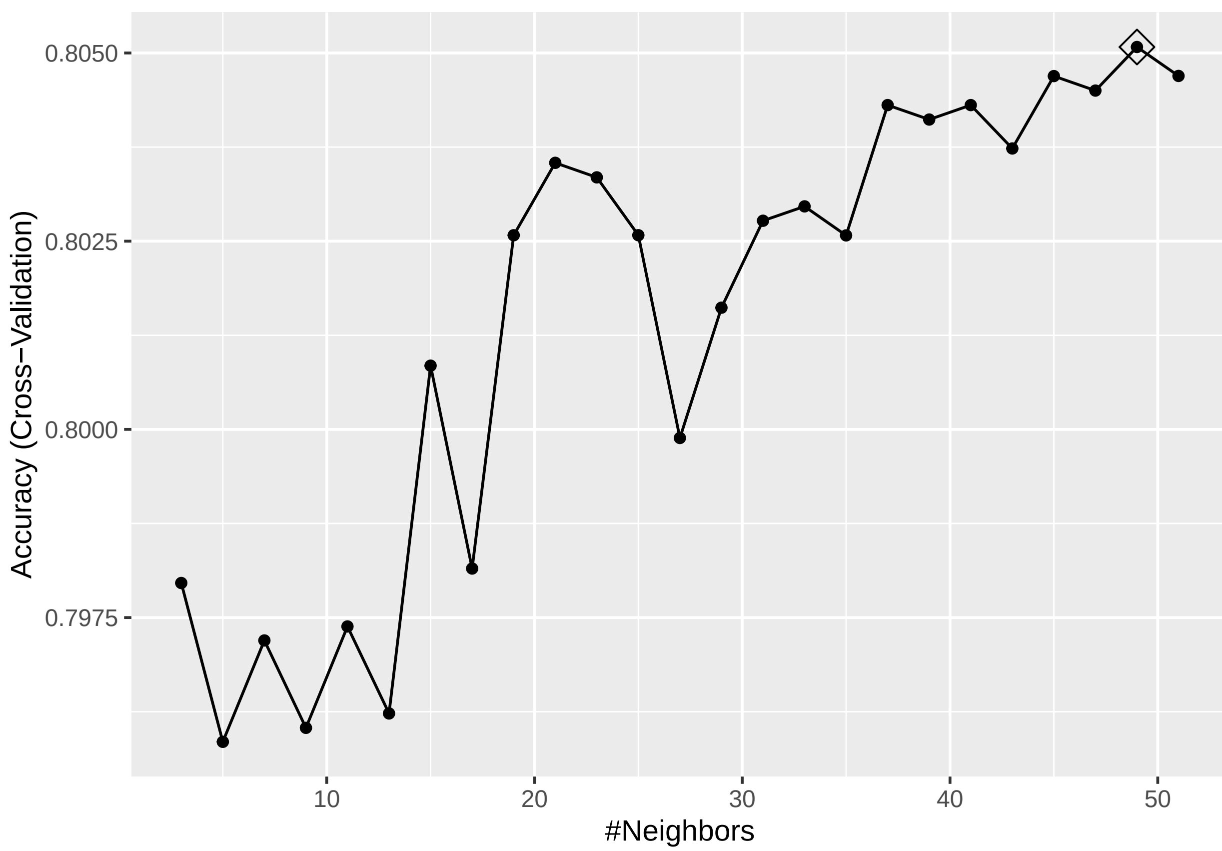
```r
train_knn$bestTune
```

```
##     k
## 23 47
```

```r
knn_pred <- predict(train_knn, test_set, type = "raw")
confusionMatrix(knn_pred, test_set$Excellent)$overall["Accuracy"]
```

```
##  Accuracy
## 0.8030769
```

```r
# Cross Validation Model
set.seed(1)
train_knn_cross <- train(Excellent ~ .-quality,
                         method = "knn",
                         data = train_set,
                         tuneGrid = data.frame(k = seq(3,51,2)),
                         trControl = trainControl(method = "cv", number = 10, p = .9))
ggplot(train_knn_cross, highlight = TRUE)
```



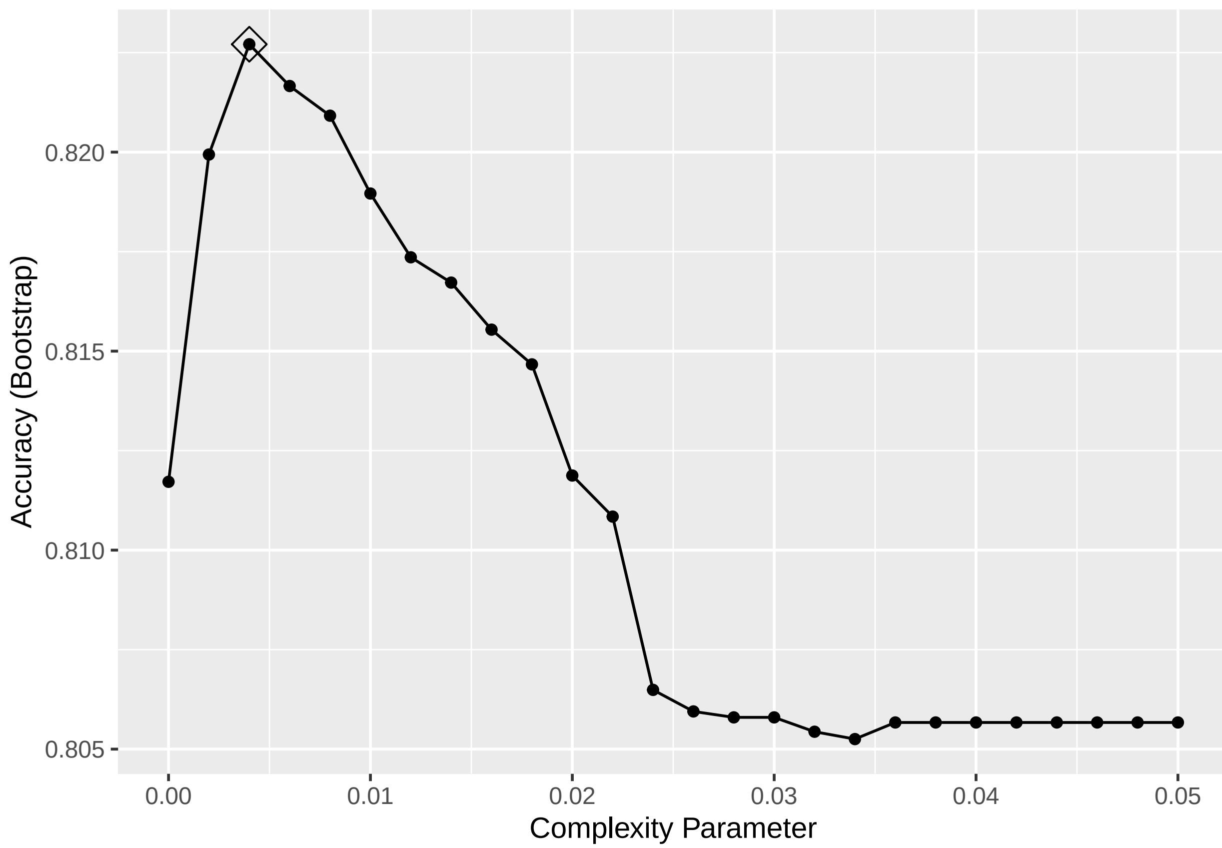

```r
train_knn_cross$bestTune
```

```
##     k
## 24 49
```

```r
knn_cross_pred <- predict(train_knn_cross, test_set, type = "raw")
confusionMatrix(knn_cross_pred, test_set$Excellent)$overall["Accuracy"]
```

```
##  Accuracy
## 0.8038462
```

```r
# Rpart
set.seed(1)
train_tree <- train(Excellent ~ .-quality,
                    method = "rpart",
                    data = train_set,
                    tuneGrid = data.frame(cp = seq(0, 0.05, 0.002)))
ggplot(train_tree, highlight = TRUE)
```




```r
train_tree$bestTune
```

```
##      cp
## 3 0.004
```

```r
rpart_pred <- predict(train_tree, test_set, type = "raw")
confusionMatrix(rpart_pred,test_set$Excellent)$overall["Accuracy"]
```

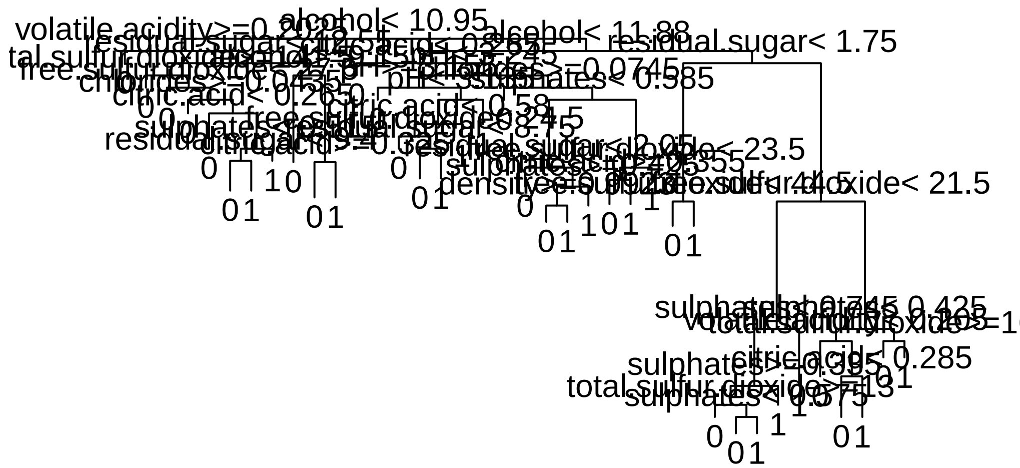
```
##  Accuracy
## 0.8346154
```

```r
train_tree$finalModel
```

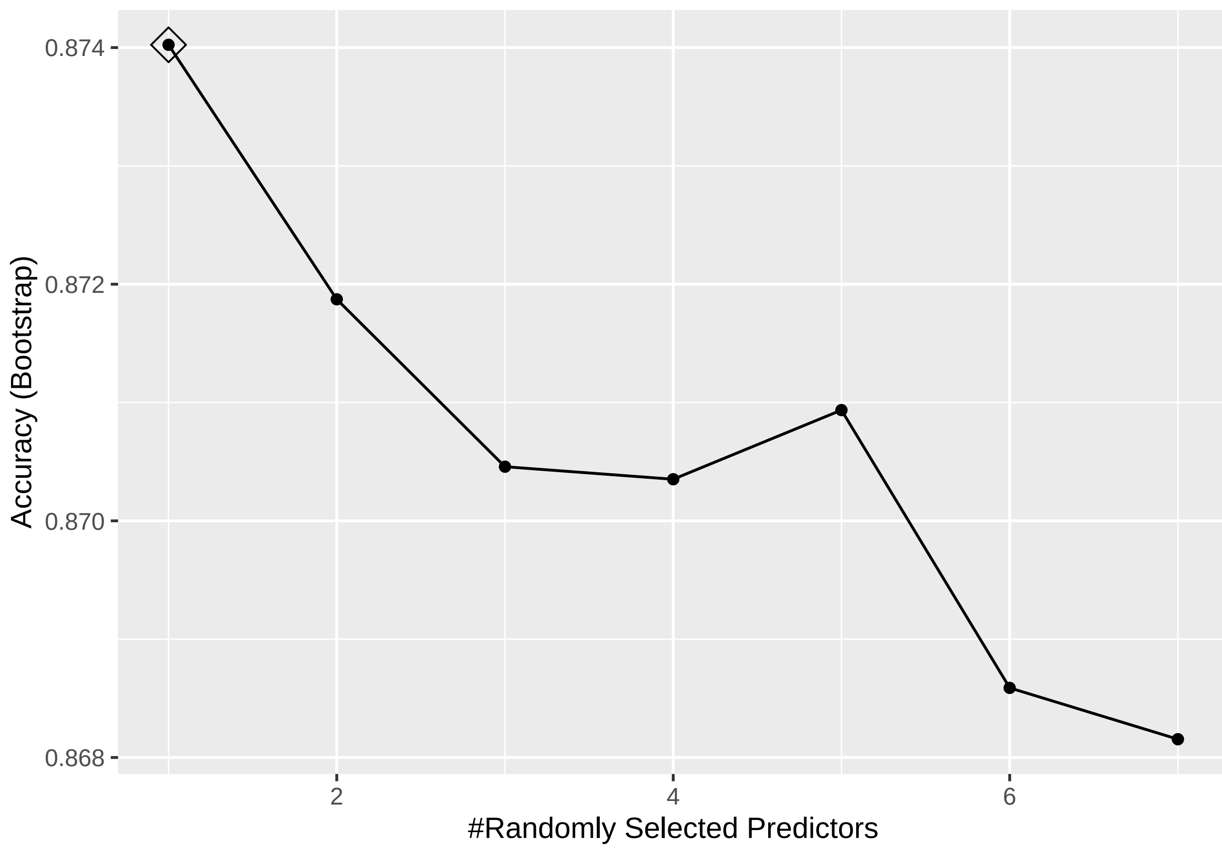
```
## n= 5197
##
## node), split, n, loss, yval, (yprob)
##       * denotes terminal node
##
##   1) root 5197 1021 0 (0.80354050 0.19645950)
##     2) alcohol< 10.95 3453  329 0 (0.90472053 0.09527947)
##       4) volatile.acidity>=0.2025 2838  176 0 (0.93798450 0.06201550) *
##       5) volatile.acidity< 0.2025 615  153 0 (0.75121951 0.24878049)
##        10) residual.sugar< 12.55 486   91 0 (0.81275720 0.18724280)
##          20) total.sulfur.dioxide>=141.5 164    9 0 (0.94512195 0.05487805) *
##          21) total.sulfur.dioxide< 141.5 322   82 0 (0.74534161 0.25465839)
##            42) free.sulfur.dioxide< 27.5 165   25 0 (0.84848485 0.15151515) *
##            43) free.sulfur.dioxide>=27.5 157   57 0 (0.63694268 0.36305732)
##              86) chlorides>=0.0435 74   13 0 (0.82432432 0.17567568) *
##              87) chlorides< 0.0435 83   39 1 (0.46987952 0.53012048)
##               174) citric.acid< 0.265 15    1 0 (0.93333333 0.06666667) *
##               175) citric.acid>=0.265 68   25 1 (0.36764706 0.63235294)
##                 350) sulphates< 0.515 45   22 0 (0.51111111 0.48888889)
##                   700) residual.sugar< 9.4 28    8 0 (0.71428571 0.28571429) *
##                   701) residual.sugar>=9.4 17    3 1 (0.17647059 0.82352941) *
##                 351) sulphates>=0.515 23    2 1 (0.08695652 0.91304348) *
##        11) residual.sugar>=12.55 129   62 0 (0.51937984 0.48062016)
##          22) alcohol>=9.15 57   10 0 (0.82456140 0.17543860) *
##          23) alcohol< 9.15 72   20 1 (0.27777778 0.72222222)
##            46) citric.acid>=0.325 21    4 0 (0.80952381 0.19047619) *
##            47) citric.acid< 0.325 51    3 1 (0.05882353 0.94117647) *
##     3) alcohol>=10.95 1744  692 0 (0.60321101 0.39678899)
##       6) alcohol< 11.875 897  274 0 (0.69453735 0.30546265)
##        12) citric.acid< 0.265 220   35 0 (0.84090909 0.15909091) *
##        13) citric.acid>=0.265 677  239 0 (0.64697194 0.35302806)
##          26) pH< 3.245 413  119 0 (0.71186441 0.28813559)
##            52) pH>=3.155 155   28 0 (0.81935484 0.18064516) *
##            53) pH< 3.155 258   91 0 (0.64728682 0.35271318)
##             106) pH< 3.135 217   66 0 (0.69585253 0.30414747)
##               212) citric.acid< 0.58 209   60 0 (0.71291866 0.28708134)
##                 424) free.sulfur.dioxide< 24.5 79   13 0 (0.83544304 0.16455696) *
##                 425) free.sulfur.dioxide>=24.5 130   47 0 (0.63846154 0.36153846)
##                   850) residual.sugar< 8.75 109   31 0 (0.71559633 0.28440367) *
##                   851) residual.sugar>=8.75 21    5 1 (0.23809524 0.76190476) *
##               213) citric.acid>=0.58 8    2 1 (0.25000000 0.75000000) *
##             107) pH>=3.135 41   16 1 (0.39024390 0.60975610) *
##          27) pH>=3.245 264  120 0 (0.54545455 0.45454545)
##            54) chlorides>=0.0745 37    5 0 (0.86486486 0.13513514) *
##            55) chlorides< 0.0745 227  112 1 (0.49339207 0.50660793)
##             110) sulphates< 0.585 151   63 0 (0.58278146 0.41721854)
##               220) residual.sugar< 2.05 76   22 0 (0.71052632 0.28947368) *
##               221) residual.sugar>=2.05 75   34 1 (0.45333333 0.54666667)
##                 442) sulphates>=0.425 47   20 0 (0.57446809 0.42553191)
##                   884) density>=0.9923 29    8 0 (0.72413793 0.27586207) *
##                   885) density< 0.9923 18    6 1 (0.33333333 0.66666667) *
```

```
##                 443) sulphates< 0.425 28    7 1 (0.25000000 0.75000000) *
##             111) sulphates>=0.585 76   24 1 (0.31578947 0.68421053)
##               222) free.sulfur.dioxide< 23.5 31   15 0 (0.51612903 0.48387097)
##                 444) citric.acid>=0.355 18    4 0 (0.77777778 0.22222222) *
##                 445) citric.acid< 0.355 13    2 1 (0.15384615 0.84615385) *
##               223) free.sulfur.dioxide>=23.5 45    8 1 (0.17777778 0.82222222) *
##       7) alcohol>=11.875 847  418 0 (0.50649351 0.49350649)
##        14) residual.sugar< 1.75 198   68 0 (0.65656566 0.34343434)
##          28) free.sulfur.dioxide< 44.5 183   56 0 (0.69398907 0.30601093) *
##          29) free.sulfur.dioxide>=44.5 15    3 1 (0.20000000 0.80000000) *
##        15) residual.sugar>=1.75 649  299 1 (0.46070878 0.53929122)
##          30) free.sulfur.dioxide< 21.5 239   96 0 (0.59832636 0.40167364)
##            60) sulphates< 0.745 196   64 0 (0.67346939 0.32653061)
##             120) sulphates>=0.335 187   56 0 (0.70053476 0.29946524)
##               240) total.sulfur.dioxide>=13 164   43 0 (0.73780488 0.26219512) *
##               241) total.sulfur.dioxide< 13 23   10 1 (0.43478261 0.56521739)
##                 482) sulphates< 0.575 10    2 0 (0.80000000 0.20000000) *
##                 483) sulphates>=0.575 13    2 1 (0.15384615 0.84615385) *
##             121) sulphates< 0.335 9    1 1 (0.11111111 0.88888889) *
##            61) sulphates>=0.745 43   11 1 (0.25581395 0.74418605) *
##          31) free.sulfur.dioxide>=21.5 410  156 1 (0.38048780 0.61951220)
##            62) sulphates< 0.425 168   82 1 (0.48809524 0.51190476)
##             124) volatile.acidity< 0.205 19    3 0 (0.84210526 0.15789474) *
##             125) volatile.acidity>=0.205 149   66 1 (0.44295302 0.55704698)
##               250) citric.acid< 0.285 45   15 0 (0.66666667 0.33333333) *
##               251) citric.acid>=0.285 104   36 1 (0.34615385 0.65384615) *
##            63) sulphates>=0.425 242   74 1 (0.30578512 0.69421488)
##             126) total.sulfur.dioxide>=168.5 16    5 0 (0.68750000 0.31250000) *
##             127) total.sulfur.dioxide< 168.5 226   63 1 (0.27876106 0.72123894) *
```

```r
plot(train_tree$finalModel, margin = 0.1)
text(train_tree$finalModel)
```




```r
# Random Forest Model
set.seed(1)
train_rf <- train(Excellent ~ .-quality,
                  method = "rf",
                  data = train_set,
                  tuneGrid = data.frame(mtry = seq(1:7)),
                  ntree = 100)
ggplot(train_rf, highlight = TRUE)
```




```
train_rf$bestTune
```

```
##   mtry
## 1    1
```

```
rf_pred <- predict(train_rf, test_set, type = "raw")
confusionMatrix(rf_pred,test_set$Excellent)$overall["Accuracy"]
```

```
##  Accuracy
## 0.9053846
```

```
#importance of variables
varImp(train_rf)
```

```
## rf variable importance
##
##                      Overall
## alcohol              100.000
## density               76.208
## chlorides             39.376
## volatile.acidity      19.975
## total.sulfur.dioxide  17.175
## sulphates             14.729
## residual.sugar        13.334
```

```
## free.sulfur.dioxide   12.734
## citric.acid           11.799
## pH                     9.679
## fixed.acidity          0.000
```

Results

```r
models <- c("Logistic Regression", "LDA", "QDA", "Loess",
            "K nearest neighbors", "Cross Validation", "Rpart", "Random forest")
accuracy <- c(mean(glm_pred == test_set$Excellent),
              mean(lda_pred == test_set$Excellent),
              mean(qda_pred == test_set$Excellent),
              mean(loess_pred == test_set$Excellent),
              mean(knn_pred == test_set$Excellent),
              mean(knn_cross_pred == test_set$Excellent),
              mean(rpart_pred == test_set$Excellent),
              mean(rf_pred == test_set$Excellent))
Model_Results <- data.frame(Model = models, Accuracy = accuracy)
Model_Results
```

```
##                 Model  Accuracy
## 1 Logistic Regression 0.8323077
## 2                 LDA 0.8323077
## 3                 QDA 0.7792308
## 4               Loess 0.8376923
## 5 K nearest neighbors 0.8030769
## 6    Cross Validation 0.8038462
## 7               Rpart 0.8346154
## 8       Random forest 0.9053846
```

When comparing the results from all the different models performance it is clear that the highest performing model was the Random Forest which had a 90.5% accuracy. This is large improvement from the lowest performing model which was QDA at 77.9%.

Conclusion

Through our analysis we were able to interpret the visualizations and determine a few variables that play a higher role in predicting the outcome. These variables were later confirmed with the random forest model which showed both alcohol and density as having the highest importance for predictions. Since this data didn't contain every aspect of wine data, things like brand name, age, sell price could all potentially improve these models and make it easier to classify an excellent wine from a mediocre.