224038\_Katie\_Taris\_c7081\_Project

Katie Taris

2022-11-23

**GitHub Repository**

<https://github.com/ketaris/c7081_Project_KT>

**Background**

Wine is famously difficult to describe by a consumer[3]. It can usually be communicated if the wine is pleasing or not, but describing which aspects and characteristics are the deciding factors is more difficult[3]. A wheel of terminology containing standardised mouth-feel terms was implemented in the beer industry before the wine industry[3]. Although one now exists for the wine industry, it often requires experience to know when to use the terms[3]. Despite not being easy to accurately obtain, a wine quality measurement is important for the producers to measure output standards and set prices of their product[1]. Easier parameters to measure in wine include physicochemical tests like pH and density, made simpler without the need for human interpretation[1]. If these laboratory tests could be analysed for their relationship with quality, it would give producers a reliable and reproducible standard in which to rate the quality of their wine[1]. Amongst these tests is the alcohol content of the wine. It has been proposed that climate change is increasing sugar levels, which in turn is increasing wine alcohol levels[4]. This is contrary to the increasing trend for alcoholic drinks with reduced alcohol levels. This need arises from awareness of health related to alcohol consumption, and potential consequences like violating drink driving laws[4]. It is therefore of interest to see how measurable components in the wine also relate to alcohol content, and in the future see if altering the former would change the latter. It is the goal of this analysis to see the potential relationships between physicochemical tests to both wine quality and alcohol content.

**Methods**

The analysis was conducted using R and RStudio. Finding the relationship between physicochemical tests and quality was treated as a classification problem. The potential relationship between physicochemical tests and alcohol content was treated as a regression problem.

## set working directory  
setwd("C:/Users/Katie/Documents/RStudio")  
getwd()

## [1] "C:/Users/Katie/Documents/RStudio"

## Load necessary libraries  
  
library(BART)

## Loading required package: nlme

## Loading required package: nnet

## Loading required package: survival

library(boot)

##   
## Attaching package: 'boot'

## The following object is masked from 'package:survival':  
##   
## aml

library(class)  
library(dplyr)

##   
## Attaching package: 'dplyr'

## The following object is masked from 'package:nlme':  
##   
## collapse

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

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

library(e1071)  
library(gbm)

## Loaded gbm 2.1.8.1

library(ISLR)  
library(ISLR2)

##   
## Attaching package: 'ISLR2'

## The following objects are masked from 'package:ISLR':  
##   
## Auto, Credit

library(leaps)  
library(MASS)

##   
## Attaching package: 'MASS'

## The following object is masked from 'package:ISLR2':  
##   
## Boston

## The following object is masked from 'package:dplyr':  
##   
## select

library(randomForest)

## randomForest 4.7-1.1

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

##   
## Attaching package: 'randomForest'

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

library(tree)  
  
## Import red wine data  
winedata <- read.csv("winequality-red.csv")  
head(winedata)

## 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(winedata)

## 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

Once the initial setup was completed, classification of quality was evaluated with different methods.

cor(winedata)

## fixed.acidity volatile.acidity citric.acid residual.sugar  
## fixed.acidity 1.00000000 -0.256130895 0.67170343 0.114776724  
## volatile.acidity -0.25613089 1.000000000 -0.55249568 0.001917882  
## citric.acid 0.67170343 -0.552495685 1.00000000 0.143577162  
## residual.sugar 0.11477672 0.001917882 0.14357716 1.000000000  
## chlorides 0.09370519 0.061297772 0.20382291 0.055609535  
## free.sulfur.dioxide -0.15379419 -0.010503827 -0.06097813 0.187048995  
## total.sulfur.dioxide -0.11318144 0.076470005 0.03553302 0.203027882  
## density 0.66804729 0.022026232 0.36494718 0.355283371  
## pH -0.68297819 0.234937294 -0.54190414 -0.085652422  
## sulphates 0.18300566 -0.260986685 0.31277004 0.005527121  
## alcohol -0.06166827 -0.202288027 0.10990325 0.042075437  
## quality 0.12405165 -0.390557780 0.22637251 0.013731637  
## chlorides free.sulfur.dioxide total.sulfur.dioxide  
## fixed.acidity 0.093705186 -0.153794193 -0.11318144  
## volatile.acidity 0.061297772 -0.010503827 0.07647000  
## citric.acid 0.203822914 -0.060978129 0.03553302  
## residual.sugar 0.055609535 0.187048995 0.20302788  
## chlorides 1.000000000 0.005562147 0.04740047  
## free.sulfur.dioxide 0.005562147 1.000000000 0.66766645  
## total.sulfur.dioxide 0.047400468 0.667666450 1.00000000  
## density 0.200632327 -0.021945831 0.07126948  
## pH -0.265026131 0.070377499 -0.06649456  
## sulphates 0.371260481 0.051657572 0.04294684  
## alcohol -0.221140545 -0.069408354 -0.20565394  
## quality -0.128906560 -0.050656057 -0.18510029  
## density pH sulphates alcohol  
## fixed.acidity 0.66804729 -0.68297819 0.183005664 -0.06166827  
## volatile.acidity 0.02202623 0.23493729 -0.260986685 -0.20228803  
## citric.acid 0.36494718 -0.54190414 0.312770044 0.10990325  
## residual.sugar 0.35528337 -0.08565242 0.005527121 0.04207544  
## chlorides 0.20063233 -0.26502613 0.371260481 -0.22114054  
## free.sulfur.dioxide -0.02194583 0.07037750 0.051657572 -0.06940835  
## total.sulfur.dioxide 0.07126948 -0.06649456 0.042946836 -0.20565394  
## density 1.00000000 -0.34169933 0.148506412 -0.49617977  
## pH -0.34169933 1.00000000 -0.196647602 0.20563251  
## sulphates 0.14850641 -0.19664760 1.000000000 0.09359475  
## alcohol -0.49617977 0.20563251 0.093594750 1.00000000  
## quality -0.17491923 -0.05773139 0.251397079 0.47616632  
## quality  
## fixed.acidity 0.12405165  
## volatile.acidity -0.39055778  
## citric.acid 0.22637251  
## residual.sugar 0.01373164  
## chlorides -0.12890656  
## free.sulfur.dioxide -0.05065606  
## total.sulfur.dioxide -0.18510029  
## density -0.17491923  
## pH -0.05773139  
## sulphates 0.25139708  
## alcohol 0.47616632  
## quality 1.00000000

train2 <- (winedata$density < 0.99675)  
class(train2)

## [1] "logical"

winedata\_test <- winedata[!train2, ]  
dim(winedata\_test)

## [1] 801 12

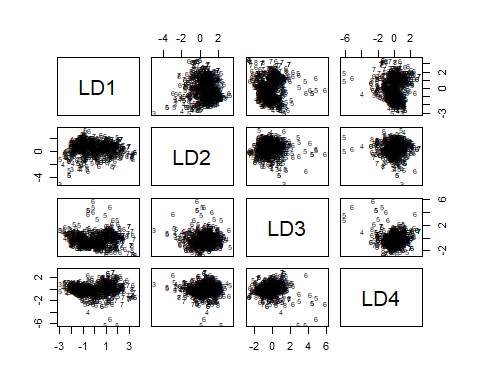
quality\_test <- winedata$quality[!train2]

A correlation test was run on the data to give a starting point for classification using the 4 variables with the strongest correlation. This is on a 0-1 scale and a higher score means higher correlation between the terms. The data was then divided into training and test sections.

## LDA - Linear Discriminant Analysis 4 variables  
lda.fit4 <- lda(quality ~ volatile.acidity + citric.acid + sulphates + alcohol,  
 data = winedata, subset = train2)  
lda.fit4

## Call:  
## lda(quality ~ volatile.acidity + citric.acid + sulphates + alcohol,   
## data = winedata, subset = train2)  
##   
## Prior probabilities of groups:  
## 3 4 5 6 7 8   
## 0.005012531 0.038847118 0.359649123 0.427318296 0.154135338 0.015037594   
##   
## Group means:  
## volatile.acidity citric.acid sulphates alcohol  
## 3 1.0850000 0.0050000 0.5200000 10.33750  
## 4 0.7261290 0.1051613 0.5490323 10.57903  
## 5 0.5992683 0.1672125 0.5835540 10.10087  
## 6 0.5030938 0.2173021 0.6472141 11.10254  
## 7 0.3996748 0.3133333 0.7216260 11.85298  
## 8 0.4433333 0.3066667 0.7775000 12.25833  
##   
## Coefficients of linear discriminants:  
## LD1 LD2 LD3 LD4  
## volatile.acidity -2.3385319 -5.5789303 3.2529493 0.3995607  
## citric.acid 0.1481706 -1.0864323 5.9211903 4.4341103  
## sulphates 2.5752889 -0.2452499 2.7986522 -6.3033344  
## alcohol 0.8503196 -0.5940594 -0.3118945 0.2010721  
##   
## Proportion of trace:  
## LD1 LD2 LD3 LD4   
## 0.9083 0.0784 0.0113 0.0020

plot(lda.fit4)



lda.pred4 = predict(lda.fit4, winedata\_test)  
names(lda.pred4)

## [1] "class" "posterior" "x"

lda.class4 = lda.pred4$class  
table(lda.class4, quality\_test)

## quality\_test  
## lda.class4 3 4 5 6 7 8  
## 3 0 0 2 1 0 0  
## 4 0 0 0 0 0 0  
## 5 5 18 308 174 14 0  
## 6 1 3 80 106 46 3  
## 7 0 0 3 14 16 3  
## 8 0 1 1 2 0 0

mean(lda.class4 == quality\_test)

## [1] 0.536829

sum(lda.pred4$posterior[ , 1] >= 0.5)

## [1] 3

sum(lda.pred4$posterior[ , 1] < 0.5)

## [1] 798

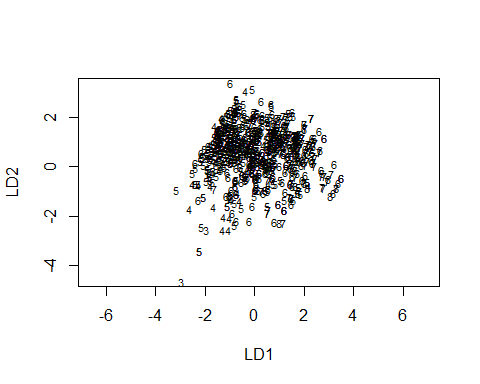
sum(lda.pred4$posterior[ , 1] > .9)

## [1] 1

## LDA - Linear Discriminant Analysis 2 variables  
lda.fit2 <- lda(quality ~ volatile.acidity + alcohol,  
 data = winedata, subset = train2)  
lda.fit2

## Call:  
## lda(quality ~ volatile.acidity + alcohol, data = winedata, subset = train2)  
##   
## Prior probabilities of groups:  
## 3 4 5 6 7 8   
## 0.005012531 0.038847118 0.359649123 0.427318296 0.154135338 0.015037594   
##   
## Group means:  
## volatile.acidity alcohol  
## 3 1.0850000 10.33750  
## 4 0.7261290 10.57903  
## 5 0.5992683 10.10087  
## 6 0.5030938 11.10254  
## 7 0.3996748 11.85298  
## 8 0.4433333 12.25833  
##   
## Coefficients of linear discriminants:  
## LD1 LD2  
## volatile.acidity -2.973397 -4.9743996  
## alcohol 0.902222 -0.6313157  
##   
## Proportion of trace:  
## LD1 LD2   
## 0.9116 0.0884

plot(lda.fit2)



lda.pred2 = predict(lda.fit2, winedata\_test)  
names(lda.pred2)

## [1] "class" "posterior" "x"

lda.class2 = lda.pred2$class  
table(lda.class2, quality\_test)

## quality\_test  
## lda.class2 3 4 5 6 7 8  
## 3 0 0 1 0 0 0  
## 4 0 0 0 0 0 0  
## 5 6 21 332 188 16 1  
## 6 0 1 59 104 59 2  
## 7 0 0 2 5 1 3  
## 8 0 0 0 0 0 0

mean(lda.class2 == quality\_test)

## [1] 0.545568

sum(lda.pred2$posterior[,1] >= 0.5)

## [1] 1

sum(lda.pred2$posterior[,1] < 0.5)

## [1] 800

sum(lda.pred2$posterior[ , 1] > .9)

## [1] 0

The quality variable has 6 possible categories. Linear Discriminant Analysis (LDA) was used as a method capable of doing classification with two or more classes. It was performed with 4 variables with the highest correlation, and again with 2 variables. The mean success rate of 4 and 2 variables was 53.7% and 54.6% respectively. The 4-variable model only had 3 predictions above 50% confidence, and 1 above 90% confidence. The 2-variable model only had 1 above 50% confidence, and none above 90% confidence. Neither model showed strong prediction power.

## Try with 2 highest correlated variables  
## Split into training and test data   
train.X <- cbind(winedata$volatile.acidity, winedata$alcohol)[train2, ]  
test.X <- cbind(winedata$volatile.acidity, winedata$alcohol)[!train2, ]  
train.Quality <- winedata$quality[train2]  
  
## Perform KNN with k=1  
set.seed(6)  
knn.pred <- knn(train.X, test.X, train.Quality, k = 1)  
table(knn.pred, quality\_test)

## quality\_test  
## knn.pred 3 4 5 6 7 8  
## 3 0 0 2 0 0 0  
## 4 3 5 20 24 0 0  
## 5 0 10 241 138 20 1  
## 6 2 7 124 105 44 3  
## 7 1 0 6 27 10 2  
## 8 0 0 1 3 2 0

## Calculate results  
dim(winedata\_test)

## [1] 801 12

(5+241+105+10) / 801

## [1] 0.4506866

## Repeat KNN with k=3  
set.seed(7)  
knn.pred2 <- knn(train.X, test.X, train.Quality, k = 3)  
table(knn.pred2, quality\_test)

## quality\_test  
## knn.pred2 3 4 5 6 7 8  
## 3 0 0 1 0 0 0  
## 4 2 1 14 20 2 0  
## 5 3 14 280 150 20 0  
## 6 1 7 91 112 46 3  
## 7 0 0 8 13 8 3  
## 8 0 0 0 2 0 0

mean(knn.pred2 == quality\_test)

## [1] 0.5006242

## Repeat KNN with k=10  
set.seed(9)  
knn.pred3 <- knn(train.X, test.X, train.Quality, k = 10)  
table(knn.pred3, quality\_test)

## quality\_test  
## knn.pred3 3 4 5 6 7 8  
## 3 0 0 0 0 0 0  
## 4 1 1 0 0 0 0  
## 5 4 15 313 173 14 0  
## 6 1 6 76 118 54 3  
## 7 0 0 5 6 8 3  
## 8 0 0 0 0 0 0

mean(knn.pred3 == quality\_test)

## [1] 0.5493134

## See if standardizing the data helps with KNN  
standardized.X <- scale(winedata[,-12])  
  
##Resplit the standardized data  
test <- 1:1120  
train.X2 <- standardized.X[-test, ]  
test.X2 <- standardized.X[test, ]  
train.Y2 <- winedata$quality[-test]  
test.Y2 <- winedata$quality[test]  
  
## Perform KNN Standardized with k=1   
set.seed(4)  
knn.pred4 <- knn(train.X2, test.X2, train.Y2, k=1)  
table(knn.pred4, test.Y2)

## test.Y2  
## knn.pred4 3 4 5 6 7 8  
## 3 1 2 8 5 0 0  
## 4 0 4 25 8 2 0  
## 5 1 10 266 151 19 1  
## 6 3 16 168 225 84 6  
## 7 0 0 26 29 51 5  
## 8 0 0 1 2 1 0

mean(test.Y2 != knn.pred4)

## [1] 0.5116071

## Perform KNN Standardized with k=3  
set.seed(5)  
knn.pred5 <- knn(train.X2, test.X2, train.Y2, k=3)  
table(knn.pred5, test.Y2)

## test.Y2  
## knn.pred5 3 4 5 6 7 8  
## 3 1 0 4 2 0 0  
## 4 1 4 25 6 2 0  
## 5 2 18 289 175 33 2  
## 6 1 10 167 213 89 5  
## 7 0 0 9 20 32 5  
## 8 0 0 0 4 1 0

mean(test.Y2 != knn.pred5)

## [1] 0.51875

## Perform KNN Standarized with k=6  
set.seed(6)  
knn.pred6 <- knn(train.X2, test.X2, train.Y2, k=6)  
table(knn.pred6, test.Y2)

## test.Y2  
## knn.pred6 3 4 5 6 7 8  
## 3 0 0 0 0 0 0  
## 4 2 3 20 4 0 0  
## 5 2 14 306 173 32 0  
## 6 1 15 164 224 105 10  
## 7 0 0 4 19 20 2  
## 8 0 0 0 0 0 0

mean(test.Y2 != knn.pred6)

## [1] 0.50625

KNN was performed as another classification method. The data was analyzed raw, and also standardized to make the scales similar between different variables. Three different values of K were tested for each, to see if increasing the number of nearest neighbours helped the model. Success rates for raw data ranged from 45.1% to 55.0%, at increasing rates of K. Success rates for standardized data ranged from 48.8% to 49.4% at increasing rates of K.

## Perform SVM  
set.seed(4)  
train\_SVM <- sample(1:nrow(winedata),1200)  
x <- winedata[ ,1:11]  
y <- winedata[ ,"quality"]  
xtrainSVM <- x[train\_SVM, ]  
ytrainSVM <- y[train\_SVM]  
xtestSVM <- x[-train\_SVM, ]  
ytestSVM <- y[-train\_SVM]  
table(ytrainSVM)

## ytrainSVM  
## 3 4 5 6 7 8   
## 9 40 499 489 150 13

dat <- data.frame(  
 x=xtrainSVM,  
 y=as.factor(ytrainSVM)  
)  
out <- svm(y ~ ., data = dat, kernel = "linear", cost = 10)  
summary(out)

##   
## Call:  
## svm(formula = y ~ ., data = dat, kernel = "linear", cost = 10)  
##   
##   
## Parameters:  
## SVM-Type: C-classification   
## SVM-Kernel: linear   
## cost: 10   
##   
## Number of Support Vectors: 1029  
##   
## ( 454 150 363 40 13 9 )  
##   
##   
## Number of Classes: 6   
##   
## Levels:   
## 3 4 5 6 7 8

table(out$fitted, dat$y)

##   
## 3 4 5 6 7 8  
## 3 0 0 0 0 0 0  
## 4 0 0 0 0 0 0  
## 5 8 25 353 153 11 0  
## 6 1 15 146 336 139 13  
## 7 0 0 0 0 0 0  
## 8 0 0 0 0 0 0

## Calculate training success rate  
(353+336)/1200

## [1] 0.5741667

dat.te <- data.frame(x = xtestSVM,  
 y = as.factor(ytestSVM))  
pred.te <- predict(out, newdata = dat.te)  
table(pred.te, dat.te$y)

##   
## pred.te 3 4 5 6 7 8  
## 3 0 0 0 0 0 0  
## 4 0 0 0 0 0 0  
## 5 1 11 134 48 2 0  
## 6 0 2 48 101 47 5  
## 7 0 0 0 0 0 0  
## 8 0 0 0 0 0 0

## Calculate test data success rate  
(134+101)/(1599-1200)

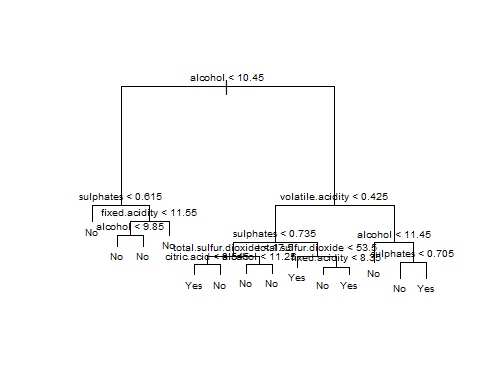
## [1] 0.5889724

A Support Vector Machine was tried as a classification method as they also have the capability of differentiating between more than two levels. This method was again performed on training and test data. The training classification success rate was 57.4% and the test success rate was 58.9%, which shows an improvement from the other models.

## Quality of 7 or 8 = High   
attach(winedata)  
High <- factor(ifelse (quality < 7, "No", "Yes"))  
winedataDT <- data.frame(winedata, High)  
tree.winedata <- tree(High ~ . -quality, winedataDT)  
summary(tree.winedata)

##   
## Classification tree:  
## tree(formula = High ~ . - quality, data = winedataDT)  
## Variables actually used in tree construction:  
## [1] "alcohol" "sulphates" "fixed.acidity"   
## [4] "volatile.acidity" "total.sulfur.dioxide" "citric.acid"   
## Number of terminal nodes: 14   
## Residual mean deviance: 0.4803 = 761.2 / 1585   
## Misclassification error rate: 0.09944 = 159 / 1599

## Plotting decision trees  
plot(tree.winedata)  
text(tree.winedata, pretty = 0, cex = 0.6)



tree.winedata

## node), split, n, deviance, yval, (yprob)  
## \* denotes terminal node  
##   
## 1) root 1599 1270.00 No ( 0.864290 0.135710 )   
## 2) alcohol < 10.45 916 257.30 No ( 0.968341 0.031659 )   
## 4) sulphates < 0.615 517 36.88 No ( 0.994197 0.005803 ) \*  
## 5) sulphates > 0.615 399 192.30 No ( 0.934837 0.065163 )   
## 10) fixed.acidity < 11.55 363 125.00 No ( 0.958678 0.041322 )   
## 20) alcohol < 9.85 243 23.18 No ( 0.991770 0.008230 ) \*  
## 21) alcohol > 9.85 120 82.32 No ( 0.891667 0.108333 ) \*  
## 11) fixed.acidity > 11.55 36 44.32 No ( 0.694444 0.305556 ) \*  
## 3) alcohol > 10.45 683 803.80 No ( 0.724744 0.275256 )   
## 6) volatile.acidity < 0.425 300 410.00 No ( 0.570000 0.430000 )   
## 12) sulphates < 0.735 169 208.60 No ( 0.692308 0.307692 )   
## 24) total.sulfur.dioxide < 17.5 55 76.23 Yes ( 0.490909 0.509091 )   
## 48) citric.acid < 0.545 45 59.67 Yes ( 0.377778 0.622222 ) \*  
## 49) citric.acid > 0.545 10 0.00 No ( 1.000000 0.000000 ) \*  
## 25) total.sulfur.dioxide > 17.5 114 117.30 No ( 0.789474 0.210526 )   
## 50) alcohol < 11.25 51 22.82 No ( 0.941176 0.058824 ) \*  
## 51) alcohol > 11.25 63 80.20 No ( 0.666667 0.333333 ) \*  
## 13) sulphates > 0.735 131 177.50 Yes ( 0.412214 0.587786 )   
## 26) total.sulfur.dioxide < 53.5 104 130.00 Yes ( 0.317308 0.682692 ) \*  
## 27) total.sulfur.dioxide > 53.5 27 28.60 No ( 0.777778 0.222222 )   
## 54) fixed.acidity < 8.35 16 0.00 No ( 1.000000 0.000000 ) \*  
## 55) fixed.acidity > 8.35 11 15.16 Yes ( 0.454545 0.545455 ) \*  
## 7) volatile.acidity > 0.425 383 329.10 No ( 0.845953 0.154047 )   
## 14) alcohol < 11.45 244 123.40 No ( 0.930328 0.069672 ) \*  
## 15) alcohol > 11.45 139 170.30 No ( 0.697842 0.302158 )   
## 30) sulphates < 0.705 91 78.14 No ( 0.846154 0.153846 ) \*  
## 31) sulphates > 0.705 48 65.20 Yes ( 0.416667 0.583333 ) \*

## Perform on training and test data  
set.seed(2)  
train\_tree <- sample(1:nrow(winedataDT), 1100)   
winedata.test.tree <- winedataDT[-train\_tree, ]  
High.test <- High[-train\_tree]  
tree.winedata2 <- tree(High ~ . -quality,  
 winedataDT, subset = train\_tree)  
tree.pred <- predict(tree.winedata2, winedata.test.tree, type = "class")  
table(tree.pred, High.test)

## High.test  
## tree.pred No Yes  
## No 392 35  
## Yes 38 34

## Calculate % correct  
(392+34) / (392+35+38+34)

## [1] 0.8537074

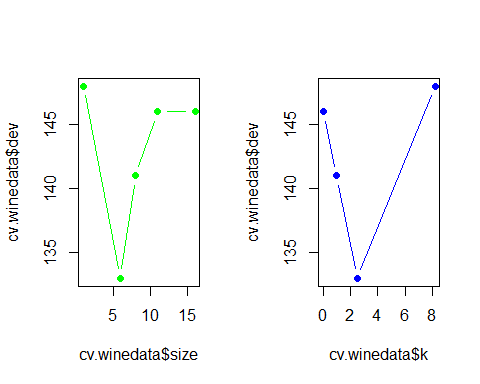
## Prune the tree  
set.seed(7)  
cv.winedata <- cv.tree(tree.winedata2, FUN = prune.misclass)  
names(cv.winedata)

## [1] "size" "dev" "k" "method"

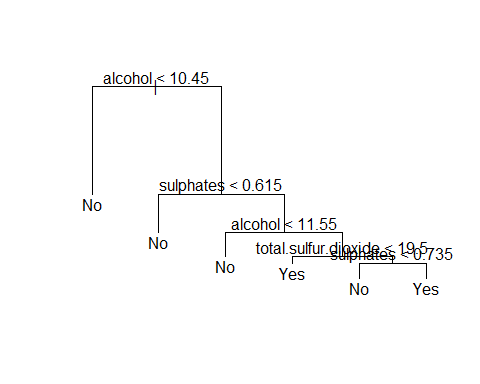
cv.winedata #size vs. dev shows optimal # nodes

## $size  
## [1] 16 11 8 6 1  
##   
## $dev  
## [1] 146 146 141 133 148  
##   
## $k  
## [1] -Inf 0.0 1.0 2.5 8.2  
##   
## $method  
## [1] "misclass"  
##   
## attr(,"class")  
## [1] "prune" "tree.sequence"

## Plot data to see optimal # nodes  
par(mfrow = c(1,2))  
plot(cv.winedata$size, cv.winedata$dev, type = "b",  
 pch=16, col = "green")  
plot(cv.winedata$k, cv.winedata$dev, type = "b",  
 pch=16, col = "blue")



prune.winedata <- prune.misclass(tree.winedata2, best = 6)  
par(mfrow=c(1,1))  
plot(prune.winedata)  
text(prune.winedata, pretty = 0)



## Test pruned tree on test data  
tree.pred2 <- predict(prune.winedata, winedata.test.tree, type = "class")  
table(tree.pred2, High.test)

## High.test  
## tree.pred2 No Yes  
## No 416 48  
## Yes 14 21

(416+21)/(416+48+14+21)

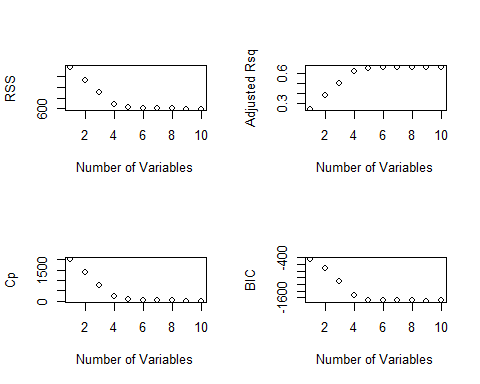
## [1] 0.8757515

Decision trees proved to be an interesting model for the classification problem. Here a quality of 7 or 8 was denoted as “High”, and the other 4 categories were not “High”. Originally the misclassification error was 9.9% and when the tree was applied to the test data the success error was 85.4%. The tree was pruned in an attempt to seek improvement. The tree was shrunk to 6 nodes and had a success error rate of 87.6%.These were the highest success rates of all the classification models.

## Best subset  
regfit.full <- regsubsets(alcohol ~ . - quality,   
 data = winedata,  
 nvmax = 10)  
reg.summary <- summary(regfit.full)  
reg.summary

## Subset selection object  
## Call: regsubsets.formula(alcohol ~ . - quality, data = winedata, nvmax = 10)  
## 10 Variables (and intercept)  
## Forced in Forced out  
## fixed.acidity FALSE FALSE  
## volatile.acidity FALSE FALSE  
## citric.acid FALSE FALSE  
## residual.sugar FALSE FALSE  
## chlorides FALSE FALSE  
## free.sulfur.dioxide FALSE FALSE  
## total.sulfur.dioxide FALSE FALSE  
## density FALSE FALSE  
## pH FALSE FALSE  
## sulphates FALSE FALSE  
## 1 subsets of each size up to 10  
## Selection Algorithm: exhaustive  
## fixed.acidity volatile.acidity citric.acid residual.sugar chlorides  
## 1 ( 1 ) " " " " " " " " " "   
## 2 ( 1 ) "\*" " " " " " " " "   
## 3 ( 1 ) "\*" " " " " " " " "   
## 4 ( 1 ) "\*" " " " " "\*" " "   
## 5 ( 1 ) "\*" " " " " "\*" " "   
## 6 ( 1 ) "\*" " " "\*" "\*" " "   
## 7 ( 1 ) "\*" " " "\*" "\*" " "   
## 8 ( 1 ) "\*" " " "\*" "\*" "\*"   
## 9 ( 1 ) "\*" "\*" "\*" "\*" "\*"   
## 10 ( 1 ) "\*" "\*" "\*" "\*" "\*"   
## free.sulfur.dioxide total.sulfur.dioxide density pH sulphates  
## 1 ( 1 ) " " " " "\*" " " " "   
## 2 ( 1 ) " " " " "\*" " " " "   
## 3 ( 1 ) " " " " "\*" "\*" " "   
## 4 ( 1 ) " " " " "\*" "\*" " "   
## 5 ( 1 ) " " " " "\*" "\*" "\*"   
## 6 ( 1 ) " " " " "\*" "\*" "\*"   
## 7 ( 1 ) " " "\*" "\*" "\*" "\*"   
## 8 ( 1 ) " " "\*" "\*" "\*" "\*"   
## 9 ( 1 ) " " "\*" "\*" "\*" "\*"   
## 10 ( 1 ) "\*" "\*" "\*" "\*" "\*"

## Plotting specs to decide which model is best  
par(mfrow = c(2,2))  
plot(reg.summary$rss,  
 xlab = "Number of Variables",  
 ylab = "RSS")  
plot(reg.summary$adjr2,  
 xlab = "Number of Variables",  
 ylab = "Adjusted Rsq")  
plot(reg.summary$cp,  
 xlab = "Number of Variables",  
 ylab = "Cp")  
plot(reg.summary$bic,   
 xlab = "Number of Variables",  
 ylab = "BIC")



which.max(reg.summary$adjr2)

## [1] 10

which.min(reg.summary$cp)

## [1] 9

which.min(reg.summary$bic)

## [1] 9

## Stepwise regression - Forward  
regfit.fwd <- regsubsets(alcohol ~ . -quality,  
 data = winedata,  
 nvmax = 10,  
 method = "forward")  
summary(regfit.fwd)

## Subset selection object  
## Call: regsubsets.formula(alcohol ~ . - quality, data = winedata, nvmax = 10,   
## method = "forward")  
## 10 Variables (and intercept)  
## Forced in Forced out  
## fixed.acidity FALSE FALSE  
## volatile.acidity FALSE FALSE  
## citric.acid FALSE FALSE  
## residual.sugar FALSE FALSE  
## chlorides FALSE FALSE  
## free.sulfur.dioxide FALSE FALSE  
## total.sulfur.dioxide FALSE FALSE  
## density FALSE FALSE  
## pH FALSE FALSE  
## sulphates FALSE FALSE  
## 1 subsets of each size up to 10  
## Selection Algorithm: forward  
## fixed.acidity volatile.acidity citric.acid residual.sugar chlorides  
## 1 ( 1 ) " " " " " " " " " "   
## 2 ( 1 ) "\*" " " " " " " " "   
## 3 ( 1 ) "\*" " " " " " " " "   
## 4 ( 1 ) "\*" " " " " "\*" " "   
## 5 ( 1 ) "\*" " " " " "\*" " "   
## 6 ( 1 ) "\*" " " "\*" "\*" " "   
## 7 ( 1 ) "\*" " " "\*" "\*" " "   
## 8 ( 1 ) "\*" " " "\*" "\*" "\*"   
## 9 ( 1 ) "\*" "\*" "\*" "\*" "\*"   
## 10 ( 1 ) "\*" "\*" "\*" "\*" "\*"   
## free.sulfur.dioxide total.sulfur.dioxide density pH sulphates  
## 1 ( 1 ) " " " " "\*" " " " "   
## 2 ( 1 ) " " " " "\*" " " " "   
## 3 ( 1 ) " " " " "\*" "\*" " "   
## 4 ( 1 ) " " " " "\*" "\*" " "   
## 5 ( 1 ) " " " " "\*" "\*" "\*"   
## 6 ( 1 ) " " " " "\*" "\*" "\*"   
## 7 ( 1 ) " " "\*" "\*" "\*" "\*"   
## 8 ( 1 ) " " "\*" "\*" "\*" "\*"   
## 9 ( 1 ) " " "\*" "\*" "\*" "\*"   
## 10 ( 1 ) "\*" "\*" "\*" "\*" "\*"

## Stepwise regression - Backward  
regfit.bwd <- regsubsets(alcohol ~ . -quality,  
 data = winedata,  
 nvmax = 10,   
 method = "backward")  
summary(regfit.bwd)

## Subset selection object  
## Call: regsubsets.formula(alcohol ~ . - quality, data = winedata, nvmax = 10,   
## method = "backward")  
## 10 Variables (and intercept)  
## Forced in Forced out  
## fixed.acidity FALSE FALSE  
## volatile.acidity FALSE FALSE  
## citric.acid FALSE FALSE  
## residual.sugar FALSE FALSE  
## chlorides FALSE FALSE  
## free.sulfur.dioxide FALSE FALSE  
## total.sulfur.dioxide FALSE FALSE  
## density FALSE FALSE  
## pH FALSE FALSE  
## sulphates FALSE FALSE  
## 1 subsets of each size up to 10  
## Selection Algorithm: backward  
## fixed.acidity volatile.acidity citric.acid residual.sugar chlorides  
## 1 ( 1 ) " " " " " " " " " "   
## 2 ( 1 ) "\*" " " " " " " " "   
## 3 ( 1 ) "\*" " " " " " " " "   
## 4 ( 1 ) "\*" " " " " "\*" " "   
## 5 ( 1 ) "\*" " " " " "\*" " "   
## 6 ( 1 ) "\*" " " "\*" "\*" " "   
## 7 ( 1 ) "\*" " " "\*" "\*" " "   
## 8 ( 1 ) "\*" " " "\*" "\*" "\*"   
## 9 ( 1 ) "\*" "\*" "\*" "\*" "\*"   
## 10 ( 1 ) "\*" "\*" "\*" "\*" "\*"   
## free.sulfur.dioxide total.sulfur.dioxide density pH sulphates  
## 1 ( 1 ) " " " " "\*" " " " "   
## 2 ( 1 ) " " " " "\*" " " " "   
## 3 ( 1 ) " " " " "\*" "\*" " "   
## 4 ( 1 ) " " " " "\*" "\*" " "   
## 5 ( 1 ) " " " " "\*" "\*" "\*"   
## 6 ( 1 ) " " " " "\*" "\*" "\*"   
## 7 ( 1 ) " " "\*" "\*" "\*" "\*"   
## 8 ( 1 ) " " "\*" "\*" "\*" "\*"   
## 9 ( 1 ) " " "\*" "\*" "\*" "\*"   
## 10 ( 1 ) "\*" "\*" "\*" "\*" "\*"

## Divide data into training and test set to   
## choose best model  
  
set.seed(1)  
train <- sample(c(TRUE, FALSE), nrow(winedata),  
 replace = TRUE)  
test <- (!train)  
regfit.best <- regsubsets(alcohol ~ . - quality,  
 data = winedata[train, ],  
 nvmax = 10)  
test.mat <- model.matrix(alcohol ~ . - quality,  
 data = winedata[test, ])  
val.errors <- rep(NA, 10)  
for(i in 1:10) {  
 coefi <- coef(regfit.best, id = i)  
 pred <- test.mat[ ,names(coefi)] %\*% coefi  
 val.errors[i] <- mean((winedata$alcohol[test] - pred)^2)  
}  
val.errors

## [1] 0.8730455 0.7007587 0.5400362 0.4046926 0.3801010 0.3788554 0.3711247  
## [8] 0.3692187 0.3657098 0.3657800

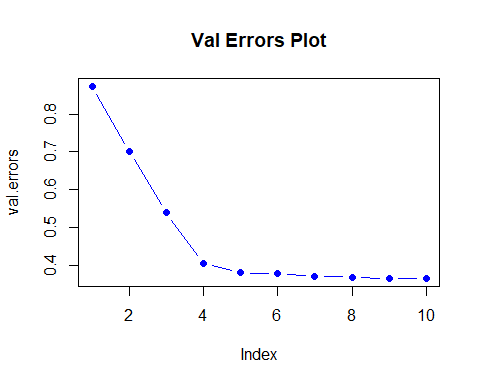
which.min(val.errors)

## [1] 9

coef(regfit.best, 9)

## (Intercept) fixed.acidity volatile.acidity   
## 5.904164e+02 5.221004e-01 3.074080e-01   
## citric.acid residual.sugar chlorides   
## 6.965482e-01 2.988153e-01 -1.365311e+00   
## total.sulfur.dioxide density pH   
## -3.164675e-03 -6.004151e+02 3.749767e+00   
## sulphates   
## 1.351780e+00

par(mfrow = c(1,1))  
plot(val.errors, type = "b",  
 pch = 16, col = "blue",  
 main = "Val Errors Plot")



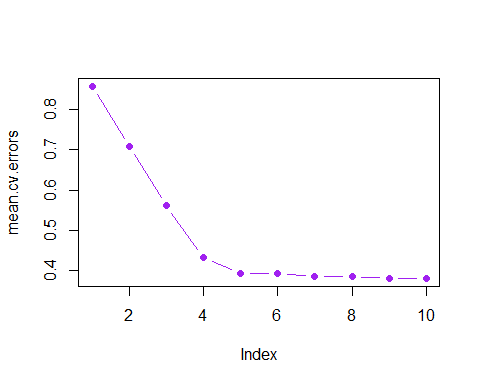
## Automating those steps  
predict.regsubsets <- function(object,  
 newdata, id, ...){  
 form <- as.formula(object$call[[ 2 ]])  
 mat <- model.matrix(form, newdata )  
 coefi <- coef (object, id = id)  
 xvars <- names (coefi)  
 mat [, xvars] %\*% coefi  
}  
  
#Find best fit with whole set,  
#not just training data  
#see if best variables are different  
regfit.best <- regsubsets(alcohol ~ . - quality,  
 data = winedata, nvmax = 10)  
coef(regfit.best, 9)

## (Intercept) fixed.acidity volatile.acidity   
## 6.059030e+02 5.299743e-01 3.808628e-01   
## citric.acid residual.sugar chlorides   
## 8.548181e-01 2.827306e-01 -1.487198e+00   
## total.sulfur.dioxide density pH   
## -2.774562e-03 -6.159814e+02 3.738814e+00   
## sulphates   
## 1.241551e+00

## Choose a model using cross validation  
k <- 10  
n <- nrow(winedata)  
set.seed(1)  
folds <- sample(rep(1:k, length = n))  
cv.errors <- matrix(NA, k, 10,  
 dimnames =   
 list(NULL, paste (1:10)))  
#nested for() loops  
for(j in 1:k){  
 best.fit <- regsubsets(alcohol ~ . -quality,  
 data = winedata[folds != j, ], nvmax = 10)  
 for(i in 1:10){  
 pred <- predict(best.fit, winedata[folds == j, ], id = i)  
 cv.errors[j, i] <- mean((winedata$alcohol[folds == j] - pred)^2)  
 }  
}  
mean.cv.errors <- apply(cv.errors, 2, mean)  
mean.cv.errors

## 1 2 3 4 5 6 7 8   
## 0.8579169 0.7093646 0.5625482 0.4325110 0.3938363 0.3940480 0.3851523 0.3847925   
## 9 10   
## 0.3808930 0.3811345

par(mfrow = c(1,1))  
plot(mean.cv.errors, type = "b",  
 pch = 16, col = "purple")



which.min(mean.cv.errors)

## 9   
## 9

## Perform best subset selection on full data to get best model  
reg.best <- regsubsets(alcohol ~ . -quality, data = winedata,  
 nvmax = 10)  
coef(reg.best, 9)

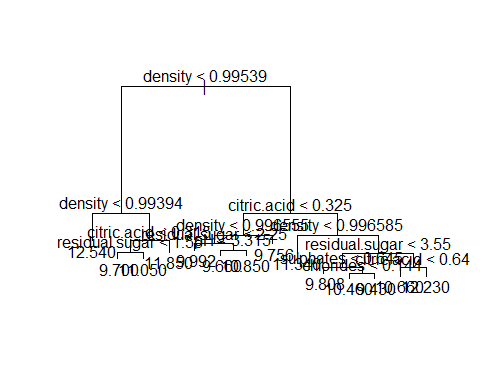
## (Intercept) fixed.acidity volatile.acidity   
## 6.059030e+02 5.299743e-01 3.808628e-01   
## citric.acid residual.sugar chlorides   
## 8.548181e-01 2.827306e-01 -1.487198e+00   
## total.sulfur.dioxide density pH   
## -2.774562e-03 -6.159814e+02 3.738814e+00   
## sulphates   
## 1.241551e+00

The regression question was started with model selection analysis. Although an important question is whether the data fits a linear model or not, with 10 variables as putative additions to the linear model, including polynomial terms and potential interactions between variables, the model selection was a method to gain a direction on the regression question. Model selection was completed on the full data, in stepwise forward and backward directions, and applied to training and test data. The results regarding the important variables were very similar with forward and backward stepwise selection highlighting the same top 5 variables: density, fixed acid, pH, residual sugar and sulphates.

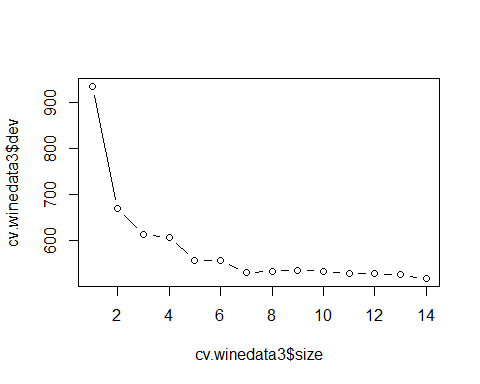
set.seed(1)  
train3 <- sample(1:nrow(winedata), nrow(winedata)/2)  
tree.winedata3 <- tree(alcohol ~ .-quality, winedata, subset= train3)  
summary(tree.winedata3)

##   
## Regression tree:  
## tree(formula = alcohol ~ . - quality, data = winedata, subset = train3)  
## Variables actually used in tree construction:  
## [1] "density" "citric.acid" "residual.sugar" "pH"   
## [5] "sulphates" "chlorides"   
## Number of terminal nodes: 14   
## Residual mean deviance: 0.463 = 363.5 / 785   
## Distribution of residuals:  
## Min. 1st Qu. Median Mean 3rd Qu. Max.   
## -2.22700 -0.44750 -0.09242 0.00000 0.38400 2.67300

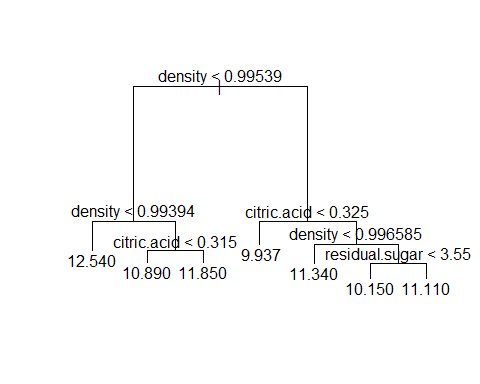
plot(tree.winedata3)  
text(tree.winedata3, pretty = 0)



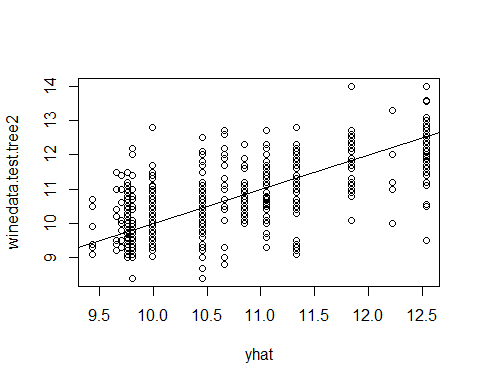
## Prune tree  
cv.winedata3 <- cv.tree(tree.winedata3)  
plot(cv.winedata3$size, cv.winedata3$dev, type = "b")



prune.winedata3 <- prune.tree(tree.winedata3, best = 7)  
plot(prune.winedata3)  
text(prune.winedata3, pretty = 0)



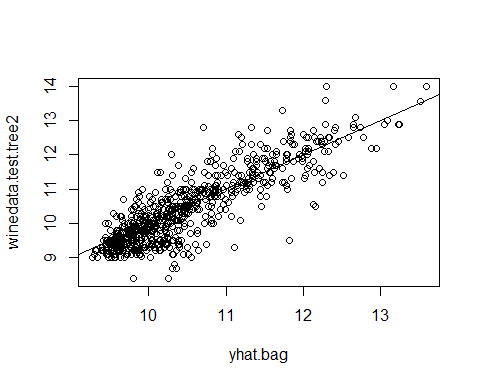
## Cross validation  
yhat <- predict(tree.winedata3, newdata = winedata[-train3, ])  
winedata.test.tree2 <- winedata[-train3, "alcohol"]  
plot(yhat, winedata.test.tree2)  
abline(0,1)



mean((yhat - winedata.test.tree2)^2) #need sqrt(MSE)

## [1] 0.5570267

## Bagging  
set.seed(1)  
bag.winedata <- randomForest(alcohol ~ . -quality,  
 data = winedata,  
 subset = train3, mtry = 10,  
 importance = TRUE)  
yhat.bag <- predict(bag.winedata, newdata = winedata[-train3, ])  
plot(yhat.bag, winedata.test.tree2)  
abline(0,1)



mean((yhat.bag - winedata.test.tree2)^2)

## [1] 0.2670029

## Change number of trees  
bag.winedata2 <- randomForest(alcohol ~ . -quality,  
 data = winedata,  
 subset=train3,  
 mtry=10,   
 ntree=25)  
yhat.bag2 <- predict(bag.winedata2, newdata = winedata[-train3, ])  
mean((yhat.bag2 - winedata.test.tree2)^2)

## [1] 0.2914859

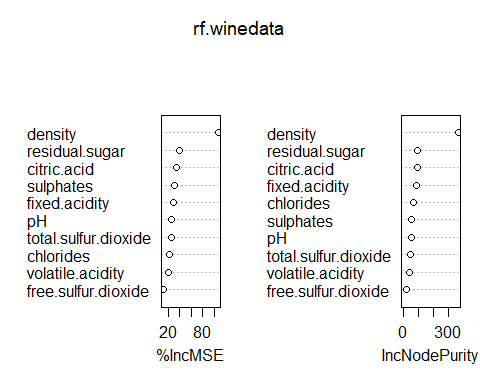
## Testing growing random forest  
set.seed(1)  
rf.winedata <- randomForest(alcohol ~ . -quality,  
 data = winedata,  
 subset = train3,  
 mtry = 6,  
 importance = TRUE)  
yhat.rf <- predict(rf.winedata, newdata = winedata[-train3, ])  
mean((yhat.rf - winedata.test.tree2)^2)

## [1] 0.278371

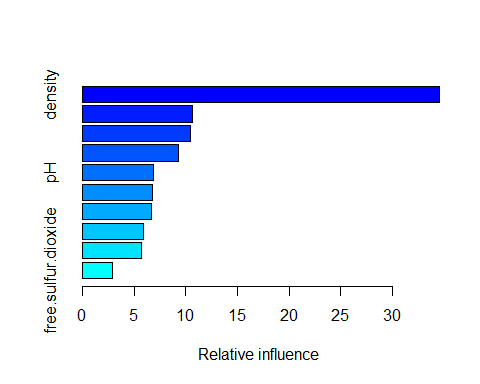
importance(rf.winedata)

## %IncMSE IncNodePurity  
## fixed.acidity 28.05899 85.36610  
## volatile.acidity 20.94591 41.17629  
## citric.acid 34.62452 89.96925  
## residual.sugar 38.75918 91.24657  
## chlorides 21.35687 69.45357  
## free.sulfur.dioxide 11.36211 20.92275  
## total.sulfur.dioxide 25.68723 44.89521  
## density 107.20093 363.87155  
## pH 25.88238 50.23057  
## sulphates 30.72715 55.61053

varImpPlot(rf.winedata)



## Boosting  
set.seed(1)  
boost.winedata <- gbm(alcohol ~ . -quality,  
 data = winedata[train3, ],  
 distribution = "gaussian",  
 n.trees = 5000,  
 interaction.depth = 4)  
summary(boost.winedata)



## var rel.inf  
## density density 34.557812  
## residual.sugar residual.sugar 10.645934  
## citric.acid citric.acid 10.481514  
## fixed.acidity fixed.acidity 9.340004  
## pH pH 6.919771  
## sulphates sulphates 6.827214  
## total.sulfur.dioxide total.sulfur.dioxide 6.665275  
## volatile.acidity volatile.acidity 5.896106  
## chlorides chlorides 5.700039  
## free.sulfur.dioxide free.sulfur.dioxide 2.966331

## Use boosted model to predict alcohol on test set  
yhat.boost <- predict(boost.winedata,  
 newdata = winedata[-train3, ],  
 n.trees = 5000)  
mean((yhat.boost - winedata.test.tree2)^2)

## [1] 0.2513281

boost.winedata2 <- gbm(alcohol ~ . -quality,  
 data = winedata[train3, ],  
 distribution = "gaussian",  
 n.trees = 5000,  
 interaction.depth = 4,   
 shrinkage = 0.2,  
 verbose = F)  
yhat.boost2 <- predict(boost.winedata2,  
 newdata = winedata[-train3, ],  
 n.trees = 5000)  
mean((yhat.boost2 - winedata.test.tree2)^2)

## [1] 0.2565991

## Bayesian additive regression trees  
x <- winedata[ ,1:10]  
y <- winedata[ ,"alcohol"]  
xtrain <- x[train3, ]  
ytrain <- y[train3]  
xtest <- x[-train3, ]  
ytest <- y[-train3]  
set.seed(1)  
bartfit <- gbart(xtrain, ytrain, x.test = xtest)

## \*\*\*\*\*Calling gbart: type=1  
## \*\*\*\*\*Data:  
## data:n,p,np: 799, 10, 800  
## y1,yn: 2.140488, -0.659512  
## x1,x[n\*p]: 8.900000, 0.640000  
## xp1,xp[np\*p]: 7.400000, 0.660000  
## \*\*\*\*\*Number of Trees: 200  
## \*\*\*\*\*Number of Cut Points: 89 ... 79  
## \*\*\*\*\*burn,nd,thin: 100,1000,1  
## \*\*\*\*\*Prior:beta,alpha,tau,nu,lambda,offset: 2,0.95,0.113137,3,0.0785483,10.4595  
## \*\*\*\*\*sigma: 0.635014  
## \*\*\*\*\*w (weights): 1.000000 ... 1.000000  
## \*\*\*\*\*Dirichlet:sparse,theta,omega,a,b,rho,augment: 0,0,1,0.5,1,10,0  
## \*\*\*\*\*printevery: 100  
##   
## MCMC  
## done 0 (out of 1100)  
## done 100 (out of 1100)  
## done 200 (out of 1100)  
## done 300 (out of 1100)  
## done 400 (out of 1100)  
## done 500 (out of 1100)  
## done 600 (out of 1100)  
## done 700 (out of 1100)  
## done 800 (out of 1100)  
## done 900 (out of 1100)  
## done 1000 (out of 1100)  
## time: 10s  
## trcnt,tecnt: 1000,1000

## Find test error  
yhat.bart <- bartfit$yhat.test.mean  
mean((ytest - yhat.bart)^2)

## [1] 0.2515497

## Check for variable appearances  
ord <- order(bartfit$varcount.mean, decreasing = T)  
bartfit$varcount.mean[ord]

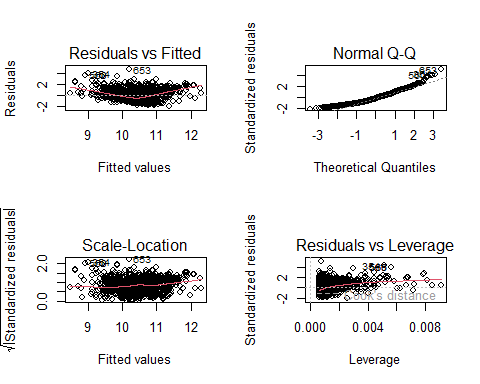
## density residual.sugar fixed.acidity   
## 37.253 28.792 28.307   
## pH citric.acid sulphates   
## 27.074 25.250 24.975   
## chlorides free.sulfur.dioxide volatile.acidity   
## 20.276 19.980 18.763   
## total.sulfur.dioxide   
## 13.766

Various methods using decision trees were also implemented to focus on important variables. Bagging ordered density, residual sugar, citric acid, and fixed acid as the top 4 variables, slightly different from the stepwise methods. Boosting still highlighted density and residual sugar as the top 2 variables. The Bayesian additive regression trees yielded the lowest test error and designated density, residual sugar, fixed acid and citric acid as the top 4 variables. In all cases, density was identified as the most important variable in alcohol levels.

## Simple linear model using density  
lm\_density <- lm(alcohol ~ density, data = winedata)  
lm\_density

##   
## Call:  
## lm(formula = alcohol ~ density, data = winedata)  
##   
## Coefficients:  
## (Intercept) density   
## 289.7 -280.2

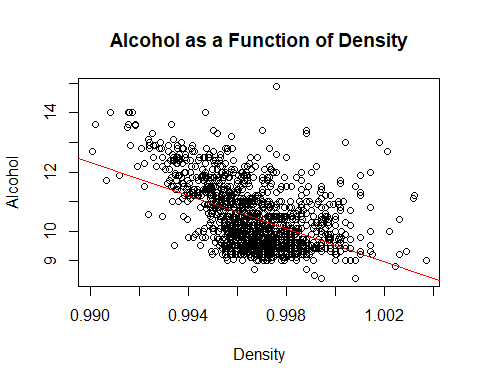
par(mfrow = c(2,2))  
plot(lm\_density)



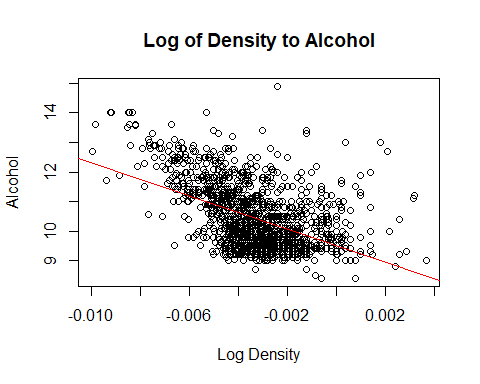
par(mfrow = c(1,1))  
summary(lm\_density)

##   
## Call:  
## lm(formula = alcohol ~ density, data = winedata)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -1.8578 -0.7240 -0.1048 0.5588 4.7161   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 289.68 12.23 23.69 <2e-16 \*\*\*  
## density -280.16 12.27 -22.84 <2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 0.9255 on 1597 degrees of freedom  
## Multiple R-squared: 0.2462, Adjusted R-squared: 0.2457   
## F-statistic: 521.6 on 1 and 1597 DF, p-value: < 2.2e-16

plot(x= winedata$density, y = winedata$alcohol,  
 xlab = "Density",  
 ylab = "Alcohol",   
 main = "Alcohol as a Function of Density")  
abline(lm\_density, col = "red")



## Take log of density  
lm\_density\_log <- lm(alcohol ~ log(density), data = winedata)  
plot(x=log(winedata$density), y = winedata$alcohol,   
 xlab= "Log Density", ylab = "Alcohol", main = "Log of Density to Alcohol")  
abline(lm\_density\_log, col = "red")



summary(lm\_density\_log)

##   
## Call:  
## lm(formula = alcohol ~ log(density), data = winedata)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -1.8600 -0.7234 -0.1056 0.5585 4.7168   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 9.51143 0.04608 206.40 <2e-16 \*\*\*  
## log(density) -279.58221 12.22370 -22.87 <2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 0.9252 on 1597 degrees of freedom  
## Multiple R-squared: 0.2467, Adjusted R-squared: 0.2463   
## F-statistic: 523.1 on 1 and 1597 DF, p-value: < 2.2e-16

## Multiple regression - multiple variables in linear model  
## Try using density and residual sugar  
lm\_sugar\_den <- lm(alcohol ~ residual.sugar + density, data = winedata)  
summary(lm\_sugar\_den)

##   
## Call:  
## lm(formula = alcohol ~ residual.sugar + density, data = winedata)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -2.0612 -0.6936 -0.0969 0.5452 3.8218   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 339.16547 12.58724 26.95 <2e-16 \*\*\*  
## residual.sugar 0.18889 0.01692 11.16 <2e-16 \*\*\*  
## density -330.29659 12.64355 -26.12 <2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 0.8917 on 1596 degrees of freedom  
## Multiple R-squared: 0.3008, Adjusted R-squared: 0.2999   
## F-statistic: 343.2 on 2 and 1596 DF, p-value: < 2.2e-16

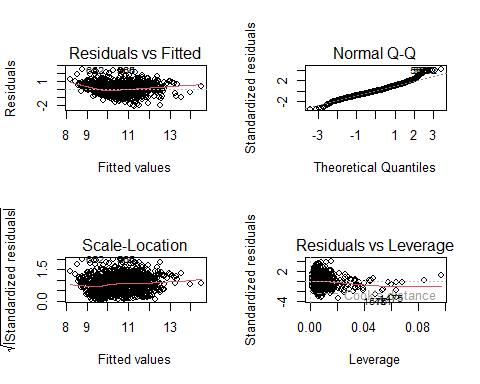
## Try using density, residual sugar and citric acid  
lm\_dsc <- lm(alcohol ~ density + residual.sugar + citric.acid, data = winedata)  
summary(lm\_dsc)

##   
## Call:  
## lm(formula = alcohol ~ density + residual.sugar + citric.acid,   
## data = winedata)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -2.9381 -0.5945 -0.0513 0.5495 3.6824   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 405.76966 12.43057 32.64 <2e-16 \*\*\*  
## density -397.60141 12.49477 -31.82 <2e-16 \*\*\*  
## residual.sugar 0.18489 0.01573 11.75 <2e-16 \*\*\*  
## citric.acid 1.81493 0.11434 15.87 <2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 0.8289 on 1595 degrees of freedom  
## Multiple R-squared: 0.3961, Adjusted R-squared: 0.395   
## F-statistic: 348.8 on 3 and 1595 DF, p-value: < 2.2e-16

## Try using all variables  
lm\_all <- lm(alcohol ~ . - quality, data = winedata)  
summary(lm\_all)

##   
## Call:  
## lm(formula = alcohol ~ . - quality, data = winedata)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -2.07175 -0.39267 -0.04056 0.35396 2.44365   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 6.072e+02 1.308e+01 46.419 < 2e-16 \*\*\*  
## fixed.acidity 5.324e-01 2.064e-02 25.796 < 2e-16 \*\*\*  
## volatile.acidity 3.608e-01 1.144e-01 3.154 0.001638 \*\*   
## citric.acid 8.306e-01 1.379e-01 6.024 2.11e-09 \*\*\*  
## residual.sugar 2.844e-01 1.229e-02 23.135 < 2e-16 \*\*\*  
## chlorides -1.462e+00 3.956e-01 -3.696 0.000227 \*\*\*  
## free.sulfur.dioxide -2.143e-03 2.057e-03 -1.042 0.297517   
## total.sulfur.dioxide -2.296e-03 6.881e-04 -3.336 0.000868 \*\*\*  
## density -6.174e+02 1.342e+01 -45.998 < 2e-16 \*\*\*  
## pH 3.762e+00 1.551e-01 24.263 < 2e-16 \*\*\*  
## sulphates 1.247e+00 1.037e-01 12.020 < 2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 0.614 on 1588 degrees of freedom  
## Multiple R-squared: 0.6701, Adjusted R-squared: 0.668   
## F-statistic: 322.5 on 10 and 1588 DF, p-value: < 2.2e-16

## Diagnostic plots  
par(mfrow = c(2,2))  
plot(lm\_all)



par(mfrow = c(1,1))  
## Try with all except free sulfur dioxide  
lm\_9v <- lm(alcohol ~ . - quality -free.sulfur.dioxide, data = winedata)  
summary(lm\_9v)

##   
## Call:  
## lm(formula = alcohol ~ . - quality - free.sulfur.dioxide, data = winedata)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -2.06145 -0.39706 -0.03917 0.34928 2.44848   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 6.059e+02 1.302e+01 46.535 < 2e-16 \*\*\*  
## fixed.acidity 5.300e-01 2.051e-02 25.846 < 2e-16 \*\*\*  
## volatile.acidity 3.809e-01 1.128e-01 3.377 0.000749 \*\*\*  
## citric.acid 8.548e-01 1.359e-01 6.289 4.12e-10 \*\*\*  
## residual.sugar 2.827e-01 1.219e-02 23.198 < 2e-16 \*\*\*  
## chlorides -1.487e+00 3.949e-01 -3.766 0.000172 \*\*\*  
## total.sulfur.dioxide -2.775e-03 5.123e-04 -5.416 7.02e-08 \*\*\*  
## density -6.160e+02 1.335e+01 -46.125 < 2e-16 \*\*\*  
## pH 3.739e+00 1.534e-01 24.369 < 2e-16 \*\*\*  
## sulphates 1.242e+00 1.036e-01 11.984 < 2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 0.614 on 1589 degrees of freedom  
## Multiple R-squared: 0.6699, Adjusted R-squared: 0.668   
## F-statistic: 358.2 on 9 and 1589 DF, p-value: < 2.2e-16

## Test lm on 5 variables  
lm.5v <- lm(alcohol ~ density + fixed.acidity + residual.sugar + pH + sulphates,   
 data = winedata)  
summary(lm.5v)

##   
## Call:  
## lm(formula = alcohol ~ density + fixed.acidity + residual.sugar +   
## pH + sulphates, data = winedata)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -2.41642 -0.39972 -0.05842 0.36731 2.59205   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 633.15668 12.14236 52.14 <2e-16 \*\*\*  
## density -644.84558 12.38685 -52.06 <2e-16 \*\*\*  
## fixed.acidity 0.62480 0.01620 38.56 <2e-16 \*\*\*  
## residual.sugar 0.28683 0.01212 23.66 <2e-16 \*\*\*  
## pH 4.01876 0.14347 28.01 <2e-16 \*\*\*  
## sulphates 1.18680 0.09455 12.55 <2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 0.625 on 1593 degrees of freedom  
## Multiple R-squared: 0.6571, Adjusted R-squared: 0.656   
## F-statistic: 610.4 on 5 and 1593 DF, p-value: < 2.2e-16

## Test interactions with density  
lm\_denint2 <- lm(alcohol ~ density\*residual.sugar,   
 data = winedata)  
summary(lm\_denint2)

##   
## Call:  
## lm(formula = alcohol ~ density \* residual.sugar, data = winedata)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -2.3196 -0.6979 -0.1115 0.5383 3.8749   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 311.624 18.942 16.451 <2e-16 \*\*\*  
## density -302.711 18.996 -15.935 <2e-16 \*\*\*  
## residual.sugar 10.700 5.406 1.979 0.048 \*   
## density:residual.sugar -10.523 5.412 -1.944 0.052 .   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 0.8909 on 1595 degrees of freedom  
## Multiple R-squared: 0.3024, Adjusted R-squared: 0.3011   
## F-statistic: 230.5 on 3 and 1595 DF, p-value: < 2.2e-16

lm\_denint3 <- lm(alcohol ~ density\*citric.acid,   
 data = winedata)  
summary(lm\_denint2)

##   
## Call:  
## lm(formula = alcohol ~ density \* residual.sugar, data = winedata)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -2.3196 -0.6979 -0.1115 0.5383 3.8749   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 311.624 18.942 16.451 <2e-16 \*\*\*  
## density -302.711 18.996 -15.935 <2e-16 \*\*\*  
## residual.sugar 10.700 5.406 1.979 0.048 \*   
## density:residual.sugar -10.523 5.412 -1.944 0.052 .   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 0.8909 on 1595 degrees of freedom  
## Multiple R-squared: 0.3024, Adjusted R-squared: 0.3011   
## F-statistic: 230.5 on 3 and 1595 DF, p-value: < 2.2e-16

## Test polynomial terms with density  
lm\_den2 <- lm(alcohol ~ density + I(density^2), data = winedata)  
summary(lm\_den2)

##   
## Call:  
## lm(formula = alcohol ~ density + I(density^2), data = winedata)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -2.5140 -0.6073 -0.1249 0.5323 4.8935   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 60108 3471 17.32 <2e-16 \*\*\*  
## density -120300 6965 -17.27 <2e-16 \*\*\*  
## I(density^2) 60202 3493 17.23 <2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 0.8501 on 1596 degrees of freedom  
## Multiple R-squared: 0.3645, Adjusted R-squared: 0.3637   
## F-statistic: 457.6 on 2 and 1596 DF, p-value: < 2.2e-16

## Anova to compare the models  
anova(lm\_density, lm\_den2)

## Analysis of Variance Table  
##   
## Model 1: alcohol ~ density  
## Model 2: alcohol ~ density + I(density^2)  
## Res.Df RSS Df Sum of Sq F Pr(>F)   
## 1 1597 1368.0   
## 2 1596 1153.4 1 214.61 296.97 < 2.2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1

## Higher order polynomials  
lm\_den6 <- lm(alcohol ~ poly(density, 6), data = winedata)  
summary(lm\_den6)

##   
## Call:  
## lm(formula = alcohol ~ poly(density, 6), data = winedata)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -2.6554 -0.5783 -0.1505 0.5297 4.9723   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 10.42298 0.02082 500.568 <2e-16 \*\*\*  
## poly(density, 6)1 -21.13728 0.83263 -25.386 <2e-16 \*\*\*  
## poly(density, 6)2 14.64954 0.83263 17.594 <2e-16 \*\*\*  
## poly(density, 6)3 0.96224 0.83263 1.156 0.248   
## poly(density, 6)4 -6.91162 0.83263 -8.301 <2e-16 \*\*\*  
## poly(density, 6)5 0.59639 0.83263 0.716 0.474   
## poly(density, 6)6 0.78788 0.83263 0.946 0.344   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 0.8326 on 1592 degrees of freedom  
## Multiple R-squared: 0.3918, Adjusted R-squared: 0.3895   
## F-statistic: 170.9 on 6 and 1592 DF, p-value: < 2.2e-16

Various forms of the linear model were examined to fit with the wine data. This included the simple model with only density, taking the log, adding residual sugar and citric acid, using interactions and adding polynomial terms. The best value of adjusted R2 out of all these variations was 0.40. The best adjusted R2 came from the model using all the continuous variables at 0.67. Removing the free sulfur dioxide term that did not show as significant in the model did not change the adjusted R2. Reducing the model to 5 variables (from model selection) yielded an adjusted R2 of 0.66, with the advantage of losing 4 variables from the model.

## Create training and test data  
set.seed(5)  
trainCV <- sample(1599, 800)  
lm.cv <- lm(alcohol ~ density, data = winedata, subset = trainCV)  
  
## Use predict on test data  
attach(winedata)

## The following objects are masked from winedata (pos = 3):  
##   
## alcohol, chlorides, citric.acid, density, fixed.acidity,  
## free.sulfur.dioxide, pH, quality, residual.sugar, sulphates,  
## total.sulfur.dioxide, volatile.acidity

mean((alcohol - predict(lm.cv, winedata))[-trainCV]^2)

## [1] 0.8728497

## Test CV on poly data  
## Poly2  
lm.cv2 <- lm(alcohol ~ poly(density, 2),  
 data = winedata, subset = trainCV)  
mean((alcohol - predict(lm.cv2, winedata))[-trainCV]^2)

## [1] 0.7577617

## Poly3  
lm.cv3 <- lm(alcohol ~ poly(density, 3),  
 data = winedata, subset = trainCV)  
mean((alcohol - predict(lm.cv3, winedata))[-trainCV]^2)

## [1] 0.757074

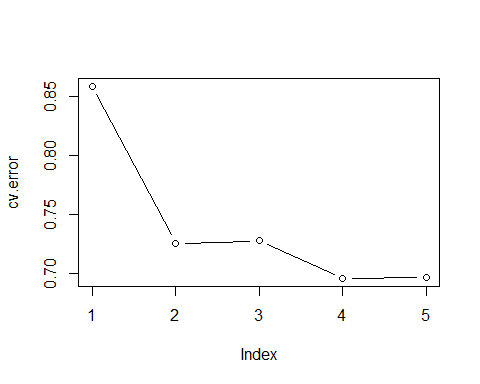
## Leave-one-out CV  
glm.fit <- glm(alcohol ~ density, data = winedata)  
cv.err <- cv.glm(winedata, glm.fit)  
cv.err$delta

## [1] 0.8583933 0.8583924

## Test on higher order polynomials  
cv.error <- rep(0, 5)  
for (i in 1:5){  
 glm.fit <- glm(alcohol ~ poly(density, i), data = winedata)  
 cv.error[i] <- cv.glm(winedata, glm.fit)$delta[1]  
}  
cv.error

## [1] 0.8583933 0.7254887 0.7276844 0.6958566 0.6967864

plot(cv.error, type='b')



## Test CV on full model  
lm.cv4 <- lm(alcohol ~ . -quality, data = winedata, subset = trainCV)  
attach(winedata)

## The following objects are masked from winedata (pos = 3):  
##   
## alcohol, chlorides, citric.acid, density, fixed.acidity,  
## free.sulfur.dioxide, pH, quality, residual.sugar, sulphates,  
## total.sulfur.dioxide, volatile.acidity

## The following objects are masked from winedata (pos = 4):  
##   
## alcohol, chlorides, citric.acid, density, fixed.acidity,  
## free.sulfur.dioxide, pH, quality, residual.sugar, sulphates,  
## total.sulfur.dioxide, volatile.acidity

mean((alcohol - predict(lm.cv4, winedata))[-trainCV]^2)

## [1] 0.4106752

## Test CV on 5 variable model  
lm.cv5 <- lm(alcohol ~ density + fixed.acidity + residual.sugar + pH + sulphates , data = winedata, subset = trainCV)  
attach(winedata)

## The following objects are masked from winedata (pos = 3):  
##   
## alcohol, chlorides, citric.acid, density, fixed.acidity,  
## free.sulfur.dioxide, pH, quality, residual.sugar, sulphates,  
## total.sulfur.dioxide, volatile.acidity  
##   
## The following objects are masked from winedata (pos = 4):  
##   
## alcohol, chlorides, citric.acid, density, fixed.acidity,  
## free.sulfur.dioxide, pH, quality, residual.sugar, sulphates,  
## total.sulfur.dioxide, volatile.acidity

## The following objects are masked from winedata (pos = 5):  
##   
## alcohol, chlorides, citric.acid, density, fixed.acidity,  
## free.sulfur.dioxide, pH, quality, residual.sugar, sulphates,  
## total.sulfur.dioxide, volatile.acidity

mean((alcohol - predict(lm.cv5, winedata))[-trainCV]^2)

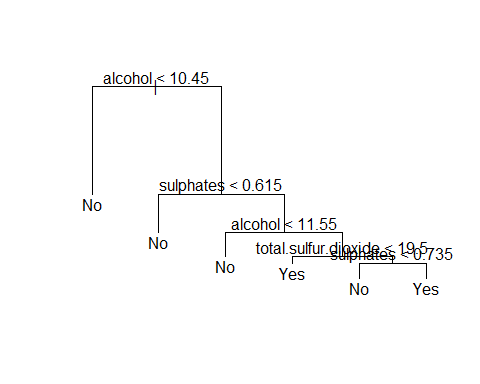
## [1] 0.4279337

Cross validation was performed on some of the linear models. The results fit the previous conclusions on the different models, where a polynomial fit of higher orders was not advantageous, and the squared term added little benefit. The full model with all terms provided the lowest MSE at 0.41, which is less than half of the MSE of 0.87 of the simple model with only density as a variable. Testing with a 5 variable model only increased the MSE to 0.43.

**Results**

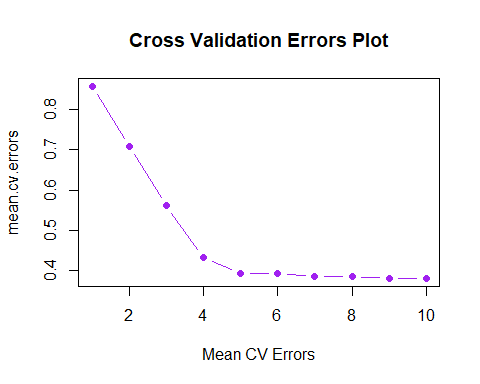
There are many applications of data science which seek to avoid human interpretation, including in wine classification. The method that performed the best for designating the 6 categories based on lab tests was support vector machines, although the average success rate was 58.2%. The decision tree method showed a high success rate of 86.5%, but it was not performing the same task as support vector machines. The decision trees were grouping the good wines with a quality value of 7 or 8 into a “High” category, and everything below into one group. This was not a balanced data set, with most of the wines falling into the 5 and 6 range of quality. In the right context, a producer may only want to differentiate which wines would qualify for a higher tier of product[4], and based on easily measurable lab tests, the decision tree could be a very useful tool.

plot(prune.winedata)  
text(prune.winedata, pretty = 0)



The second objective was to find a model where alcohol was estimated using lab tests of the wine. Model selection was used to find the most relevant predictor variables for alcohol content and this was taken forward to the linear models. Through model comparing and cross-validation, a linear model containing all variables provided the best fit to the data with an adjusted R-squared value of 0.67. This is further validated by the fact that although model selection identifies the most important predictor variables, the best model had 9 variables, out of the optional 10. The many model selection tools produced plots comparing number of variables to model error. The full model did have the lowest error, but the graphs all showed a similar pattern of a plateau beginning at 5 variables. It was then worth evaluating if adding the complexity of the extra variables greatly improved the model, or if remaining with 5 variables only sacrificed a small amount of fit. The adjusted R2 only decreased by 0.1 and the CV error only increased by 0.02. These differences show the retention of an improved model, while successfully decreasing the complexity by taking 9 variables down to 5.

plot(mean.cv.errors, type = "b", pch = 16, col = "purple",   
 xlab = "Mean CV Errors", main = "Cross Validation Errors Plot")



Relationships to the physicochemical tests are apparent in the data, although there is not one strong indicator and the interactions are complex. To further improve on the quality classification a more balanced data set with more quality ratings at the higher and lower ends of the scale, could help further train the model. Alcohol content is mainly dependent on 5 different variables. If alcohol content was going to be altered after wine production, it would be beneficial to see which parameters are heavily affected by changes in alcohol content.

**Conclusion**

Many factors go into making a good wine. Some factors can be altered during the growing season in the vineyard, and some can be changed during management of the wine making[2]. The producer needs to be able to monitor the changes occurring at all stages. Finding important relationships for how easily quantifiable measurements translate to different parameters will assist them in maintaining product standards, protecting human health, and advance their business based upon consumer needs and potential future markets.

**Literature Cited**

[1] 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.

[2] P. Demiglio, and G. J. Pickering. The influence of ethanol and pH on the taste and mouthfeel sensations elicited by red wine. Journal of Food, Agriculture & Environment. 6(3&4):143-150, 2008.

[3] R. Gawel, A. Oberholster, and I. Leigh Francis. A ‘Mouth-feel Wheel’: terminology for communicating the mouth-feel characteristics of wine. Australian Journal of Grape and Wine Research. 6(3):203-207, 2008. doi.org/10.1111/j.1755-0238.2000.tb00180

[4] A. Manuel Jordao, A. Vilela, and F. Cosme. From Sugar of Grape to Alcohol of Wine: Sensorial Impact of Alcohol in Wine. Beverages. 1:292-310, 2015. <doi:10.3390/beverages1040292>