Andrew Colbert, David Doman, Gilbert Guyah, and Christina LoConte

Abstract

This project is to choose a data set, analyze it using methods gained from class, and draw conclusions based on the analysis of the data.

Final Project

IST 687

Table of Contents

[Introduction 2](#_Toc35888560)

[Analysis Methods 2](#_Toc35888561)

[Basic Analysis 2](#_Toc35888562)

[Summary: 2](#_Toc35888563)

[Histograms: 3](#_Toc35888564)

[Box Plots: 5](#_Toc35888565)

[Pairs: 7](#_Toc35888566)

[Linear Modeling 7](#_Toc35888567)

[Correlations between variables 8](#_Toc35888568)

[Linear Regression 11](#_Toc35888569)

[Machine Learning 14](#_Toc35888570)

[Random Forest 14](#_Toc35888571)

[SVM 21](#_Toc35888572)

[KSVM 23](#_Toc35888573)

[Limitations 23](#_Toc35888574)

[Conclusion 24](#_Toc35888575)

[References 24](#_Toc35888576)

[Data Set: 24](#_Toc35888577)

[Research on wine quality: 24](#_Toc35888578)

[Code Research 24](#_Toc35888579)

# Introduction

The first step of the project is to acquire data. We chose the wine quality data set from the UCI Machine Learning Repository. This data set had data for both red and white wine that were from Portugal. Based on the data attribute tittles, we decided that this data was provided by a vineyard and they requested analysis of their data from our team. Eleven of the attributes were measurable while the last attribute was not. The last attribute is titled ‘quality’ and is the median of at least 3 wine expert evaluations. The last thing to note is that there were far more white wine values than red wine values.

# Analysis Methods

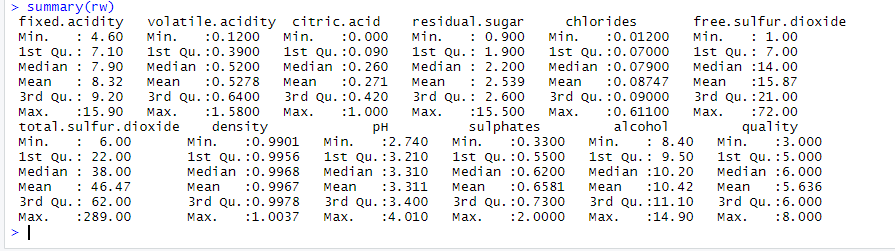
Please note that the data was split into red wine and white wine with the white wine set having more than double the number of samples. The analysis was performed on each data set separately.

## Basic Analysis

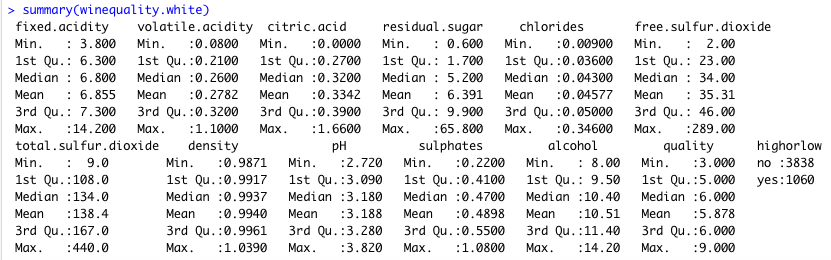
We started by running the summary function on the data and plotting graphs (histograms and box plots.

### Summary:

Red Wine:

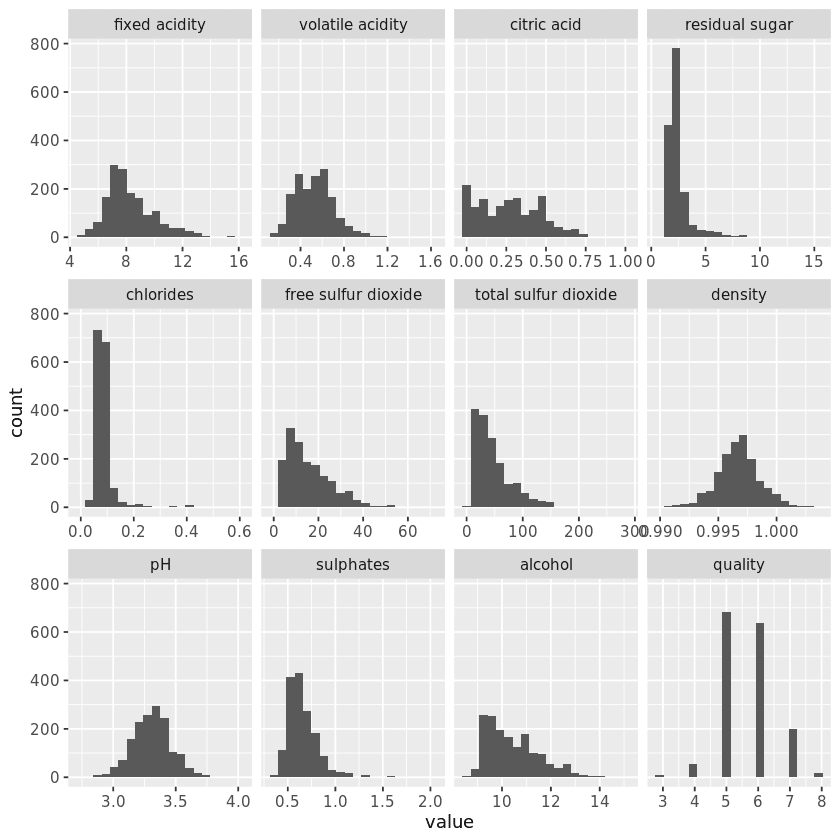


White Wine:

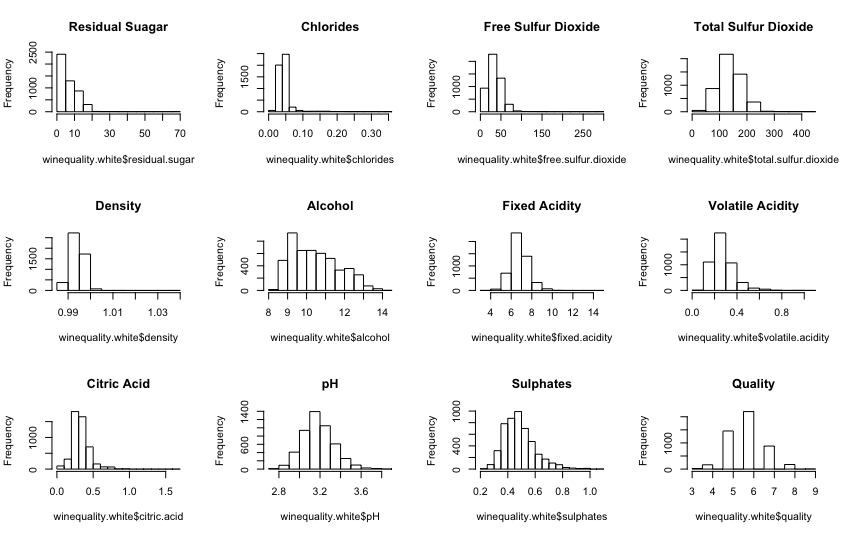


### Histograms:

Red Wine:

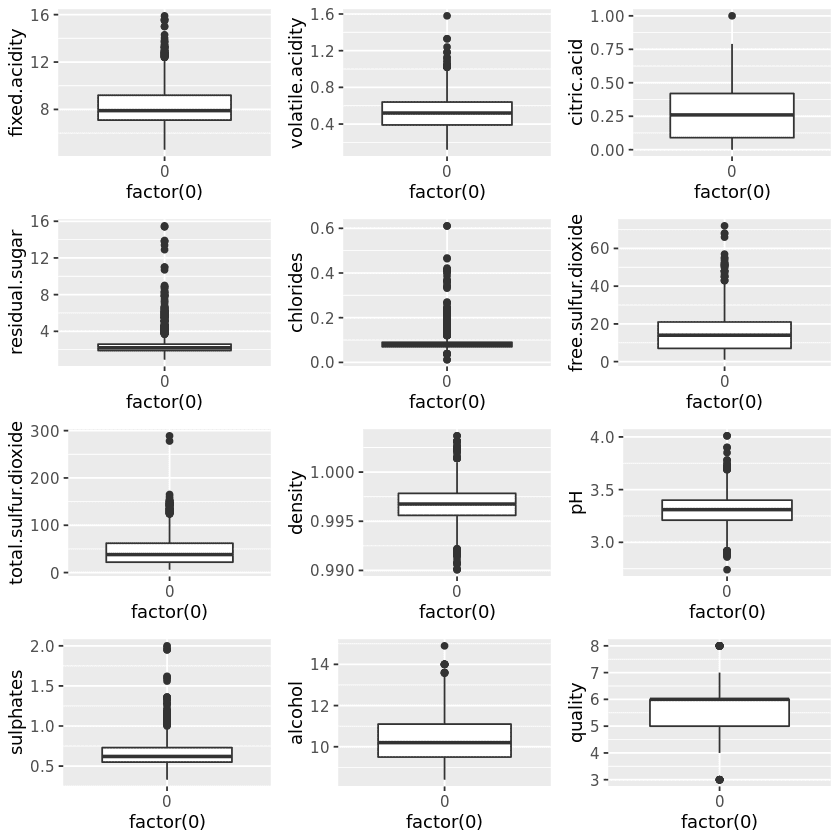


White Wine:

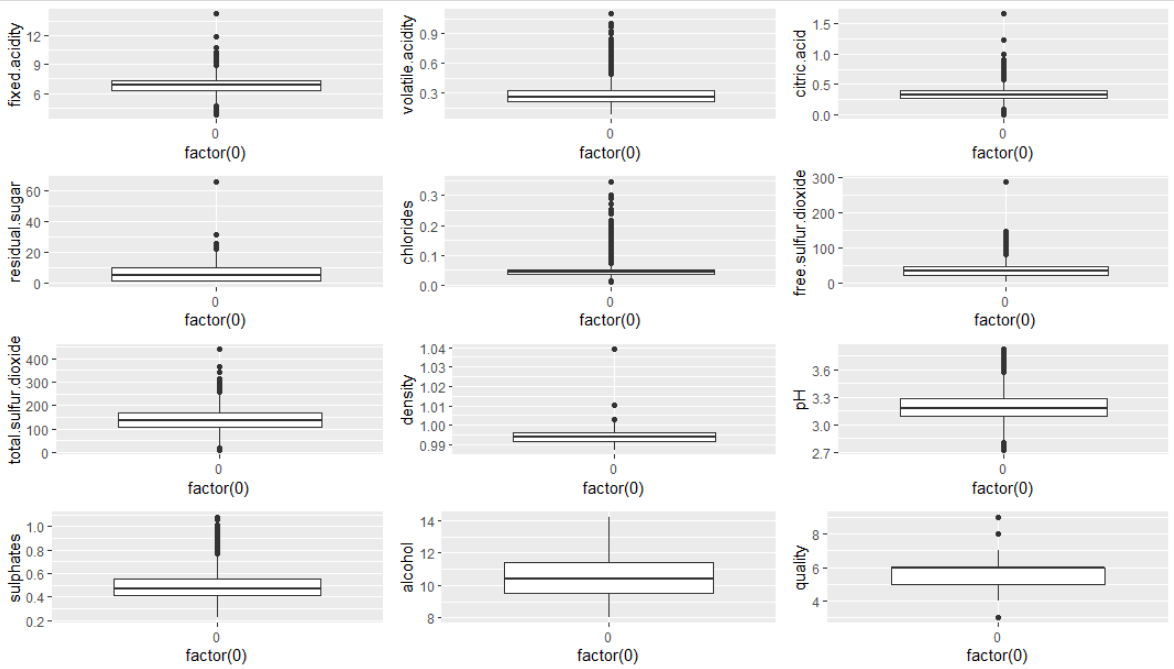


### Box Plots:

Red Wine:



White Wine:

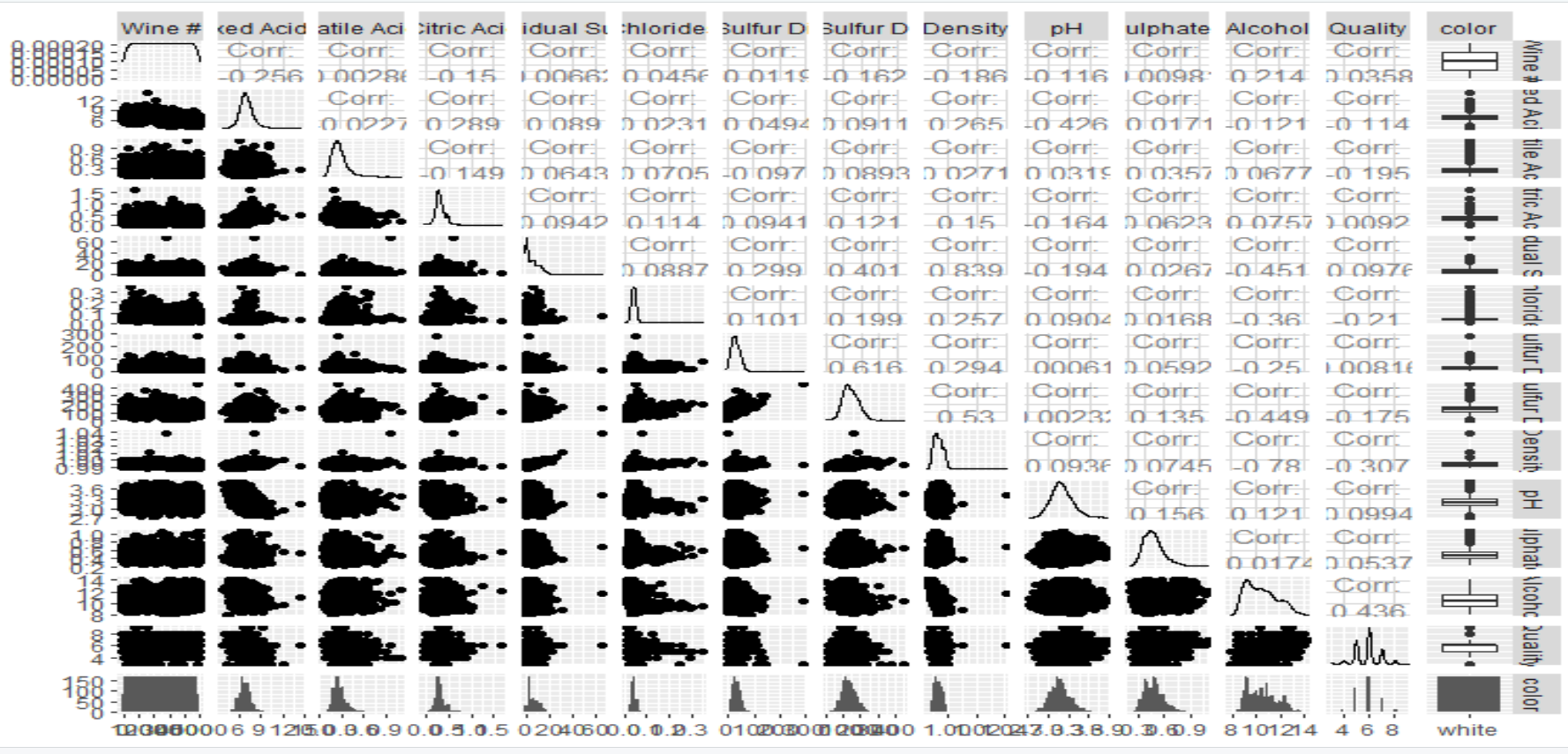


### Pairs:

White Wine:

Code:

ggpairs(white)



## Linear Modeling

We started with correlation graphs to see if there were any attributes linked to each other either in a positive or a negative way. This was followed by running linear regression and modeling.

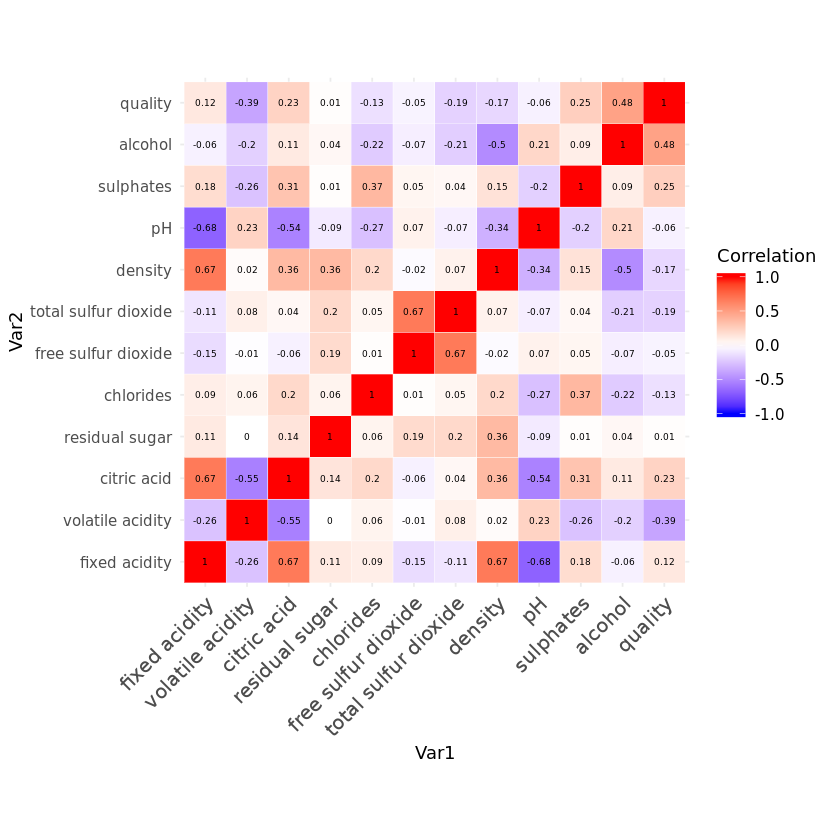
### Correlations between variables

Red Wine:

cor(winequality\_red)

ggplot(data= melt\_cormat,aes(x=Var1, y=Var2, fill= value))+ geom\_tile(color="white")+ scale\_fill\_gradient2(low="blue", mid="white", high= "red", midpoint=0, limit=c(-1, 1), space= "lab", name="Correlation")+ theme\_minimal()+ theme(axis.text.x=element\_text(angle=45, vjust=1, size=12, hjust=1))+coord\_fixed()

winemap + geom\_text(aes(Var2, Var1, label=value), color="black",size=2)



From this we noticed, pH had a negative correlation with fixed acidity. Citric acid had a positive correlation with fix acidity and total sulfur dioxide one with free sulfur dioxide, neither were surprising. Density having a correlation with fix acidity was surprising. Alcohol having a positive correlation with quality was surprising until we started doing some research. Through our research, we found that higher alcohol was a factor in wines being considered bolder. (Puckette 2015)

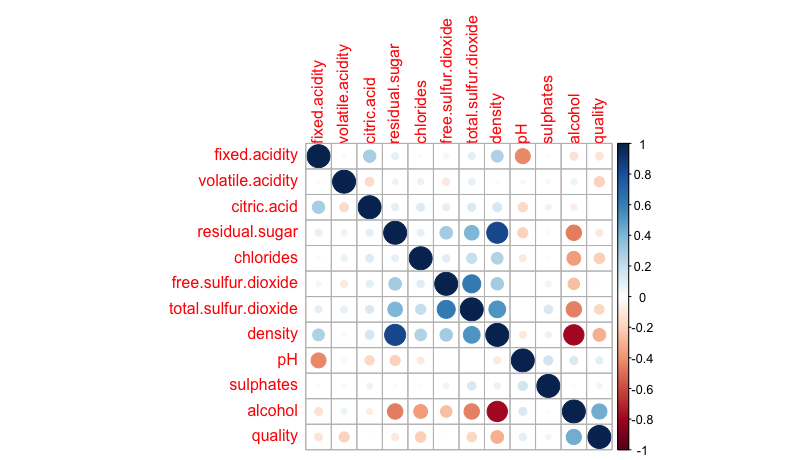
White Wine:

install.packages("corrplot")

library("corrplot")

correlations <- cor(winequality.white)

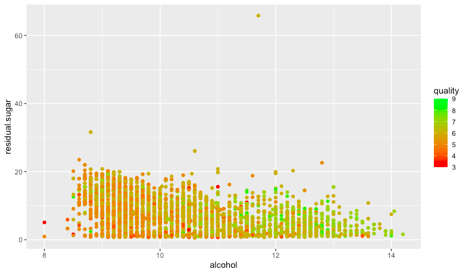
corrplot(correlations)



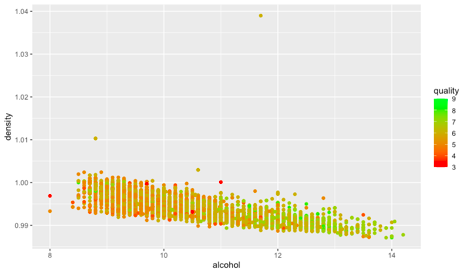
From the white wine correlation matrix, we noticed there was a positive correlation between density and residual sugar. There was a negative correlation between alcohol and density. Both of these were a little surprising as we were expecting the correlations to be a lot more similar to the red wine set.

alcoholandsugar <- ggplot(mywinedata, aes(x= alcohol, y= residual.sugar, color= quality)) + geom\_point() + scale\_color\_gradient(low= "red", high= "green")

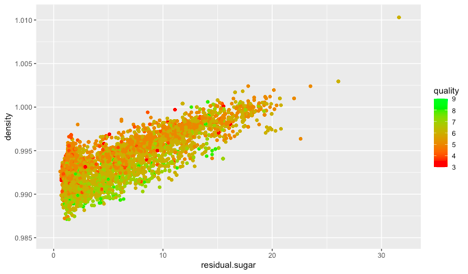
alcoholandsugar



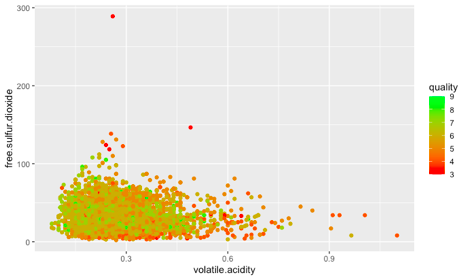
alcoholanddensity <- ggplot(winequality.white, aes(x= alcohol, y=density, color= quality)) + geom\_point() + scale\_color\_gradient(low= "red", high= "green")



sugaranddensity <- ggplot(winequality.white, aes(x= residual.sugar, y= density, color= quality)) + geom\_point() + scale\_color\_gradient(low= "red", high= "green") + xlim(0,32) + ylim(0.985, 1.0105)



freesulfurandvolatile <- ggplot(winequality.white, aes(y= free.sulfur.dioxide, x= volatile.acidity, color= quality)) + geom\_point() + scale\_color\_gradient(low= "red", high= "green")



### Linear Regression

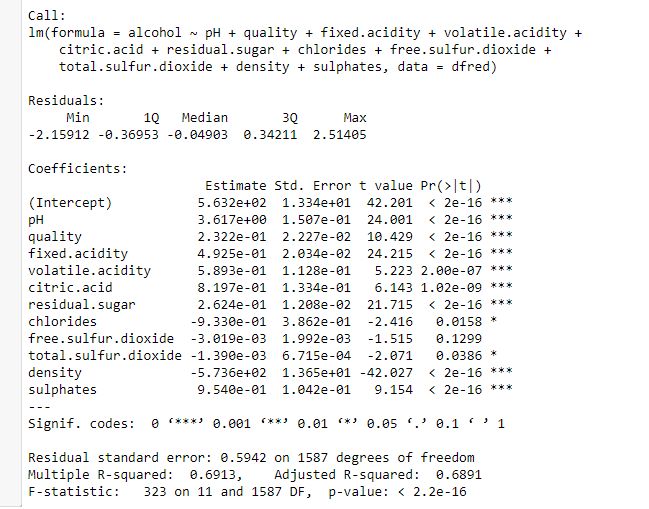
Red Wine:

dfred <- data.frame(winequality\_red)

str(dfred)

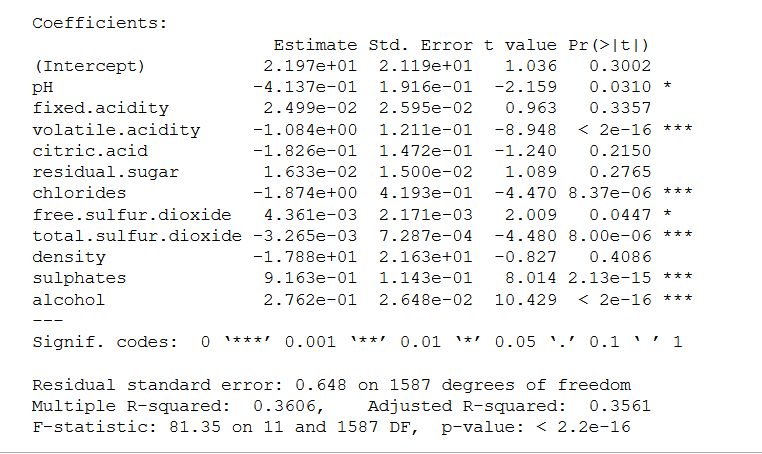
mod1 <- lm(alcohol ~ pH + quality+ fixed.acidity + volatile.acidity + citric.acid + residual.sugar + chlorides+ free.sulfur.dioxide + total.sulfur.dioxide+ density + sulphates, data= dfred)

summary(mod1)



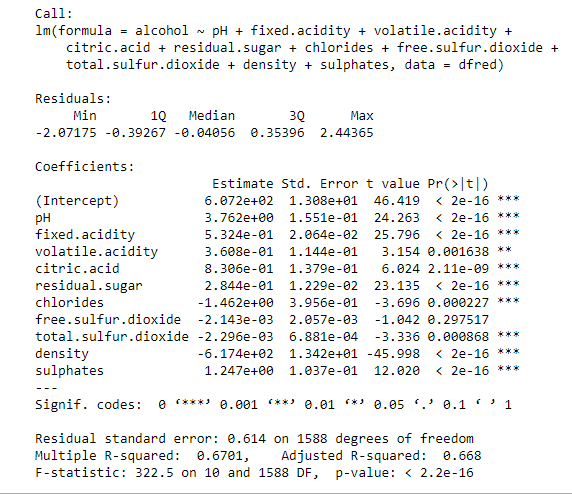
mod3 <- lm(quality ~ pH + fixed.acidity + volatile.acidity + citric.acid + residual.sugar + chlorides+ free.sulfur.dioxide + total.sulfur.dioxide+ density + sulphates+ alcohol, data= dfred)

summary(mod3)



mod31 <- lm(alcohol ~ pH + fixed.acidity + volatile.acidity + citric.acid + residual.sugar + chlorides+ free.sulfur.dioxide + total.sulfur.dioxide+ density + sulphates, data= dfred)

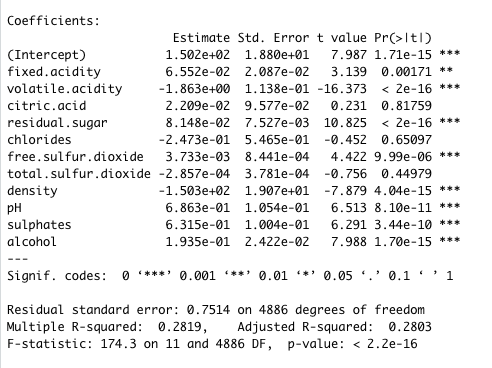
summary(mod31)



White Wine:

linearmodel <- lm(formula= quality ~ fixed.acidity + volatile.acidity + citric.acid + residual.sugar + chlorides + free.sulfur.dioxide + total.sulfur.dioxide + density + pH + sulphates + alcohol, data= winequality.white)

summary(linearmodel)



## Machine Learning

### Random Forest

Red Wine:

> str(rw)

Classes ‘tbl\_df’, ‘tbl’ and 'data.frame': 1599 obs. of 12 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 : num 5 5 5 6 5 5 5 7 7 5 ...

> rw$class <-rw$quality

> rw$class <-gsub("3","low", rw$class)

> rw$class <-gsub("4","low", rw$class)

> rw$class <-gsub("5","medium", rw$class)

> rw$class <-gsub("6","medium", rw$class)

> rw$class <-gsub("7","high", rw$class)

> rw$class <-gsub("8","high", rw$class)

> rw$class <-gsub("9","high", rw$class)

> rw$class <-as.factor(rw$class)

> str(rw)

Classes ‘tbl\_df’, ‘tbl’ and 'data.frame': 1599 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 : num 5 5 5 6 5 5 5 7 7 5 ...

$ class : Factor w/ 3 levels "high","low","medium": 3 3 3 3 3 3 3 1 1 3 ...

> table(rw$class)

high low medium

217 63 1319

Change rw to a dataframe

> dfrw2 <- data.frame(rw)

> str(dfrw2)

'data.frame': 1599 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 : num 5 5 5 6 5 5 5 7 7 5 ...

$ class : Factor w/ 3 levels "high","low","medium": 3 3 3 3 3 3 3 1 1 3 ...

Split data to train and test

> set.seed (123)

> ind <- sample(2, nrow(dfrw2), replace = TRUE, prob = c(0.7, 0.3))

> traindata <- dfrw2[ind==1,]

> testdata <- dfrw2[ind==2,]

> library(randomForest)

> rftrain <- randomForest(class ~.,data=traindata)

> print(rftrain)

Call:

randomForest(formula = class ~ ., data = traindata)

Type of random forest: classification

Number of trees: 500

No. of variables tried at each split: 3

OOB estimate of error rate: 0.09%

Confusion matrix:

high low medium class.error

high 159 0 0 0.00000000

low 0 43 1 0.02272727

medium 0 0 924 0.00000000

> attributes(rftrain)

$names

[1] "call" "type" "predicted" "err.rate" "confusion" "votes" "oob.times"

[8] "classes" "importance" "importanceSD" "localImportance" "proximity" "ntree" "mtry"

[15] "forest" "y" "test" "inbag" "terms"

$class

[1] "randomForest.formula" "randomForest"

> rftrain$confusion

high low medium class.error

high 159 0 0 0.00000000

low 0 43 1 0.02272727

medium 0 0 924 0.00000000

> p1 <- predict(rftrain, traindata)

> head(p1)

1 3 6 7 9 10

medium medium medium medium high medium

Levels: high low medium

> head(traindata$class)

[1] medium medium medium medium high medium

Levels: high low medium

> library("caret")

Loading required package: lattice

Loading required package: ggplot2

Attaching package: ‘ggplot2’

The following object is masked from ‘package:randomForest’:

margin

Warning messages:

1: package ‘caret’ was built under R version 3.6.3

2: package ‘ggplot2’ was built under R version 3.6.2

> confusionMatrix(p1, traindata$class)

Confusion Matrix and Statistics

Reference

Prediction high low medium

high 159 0 0

low 0 44 0

medium 0 0 924

Overall Statistics

Accuracy : 1

95% CI : (0.9967, 1)

No Information Rate : 0.8199

P-Value [Acc > NIR] : < 2.2e-16

Kappa : 1

Mcnemar's Test P-Value : NA

Statistics by Class:

Class: high Class: low Class: medium

Sensitivity 1.0000 1.00000 1.0000

Specificity 1.0000 1.00000 1.0000

Pos Pred Value 1.0000 1.00000 1.0000

Neg Pred Value 1.0000 1.00000 1.0000

Prevalence 0.1411 0.03904 0.8199

Detection Rate 0.1411 0.03904 0.8199

Detection Prevalence 0.1411 0.03904 0.8199

Balanced Accuracy 1.0000 1.00000 1.0000

> p2 <- predict(rftrain, testdata)

> confusionMatrix(p2, testdata$class)

Confusion Matrix and Statistics

Reference

Prediction high low medium

high 58 0 0

low 0 19 0

medium 0 0 395

Overall Statistics

Accuracy : 1

95% CI : (0.9922, 1)

No Information Rate : 0.8369

P-Value [Acc > NIR] : < 2.2e-16

Kappa : 1

Mcnemar's Test P-Value : NA

Statistics by Class:

Class: high Class: low Class: medium

Sensitivity 1.0000 1.00000 1.0000

Specificity 1.0000 1.00000 1.0000

Pos Pred Value 1.0000 1.00000 1.0000

Neg Pred Value 1.0000 1.00000 1.0000

Prevalence 0.1229 0.04025 0.8369

Detection Rate 0.1229 0.04025 0.8369

Detection Prevalence 0.1229 0.04025 0.8369

Balanced Accuracy 1.0000 1.00000 1.0000

> plot(rftrain)

> rf2 <- randomForest(class ~., data= traindata, ntree=300, mtry=8, importance= TRUE, proximity = TRUE)

> print(rf2)

Call:

randomForest(formula = class ~ ., data = traindata, ntree = 300, mtry = 8, importance = TRUE, proximity = TRUE)

Type of random forest: classification

Number of trees: 300

No. of variables tried at each split: 8

OOB estimate of error rate: 0%

Confusion matrix:

high low medium class.error

high 159 0 0 0

low 0 44 0 0

medium 0 0 924 0

> rf3 <- randomForest(class ~., data= testdata, ntree=300, mtry=8, importance= TRUE, proximity = TRUE)

> print(rf3)

Call:

randomForest(formula = class ~ ., data = testdata, ntree = 300, mtry = 8, importance = TRUE, proximity = TRUE)

Type of random forest: classification

Number of trees: 300

No. of variables tried at each split: 8

OOB estimate of error rate: 0%

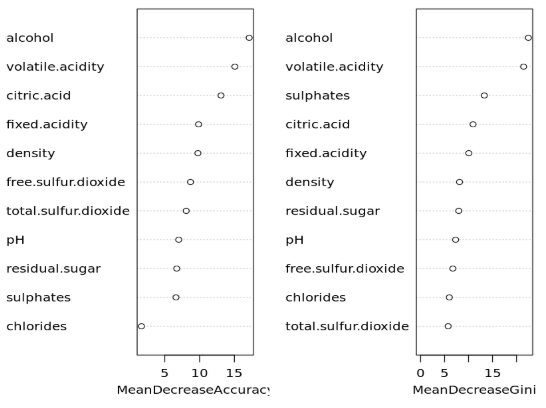
Confusion matrix:

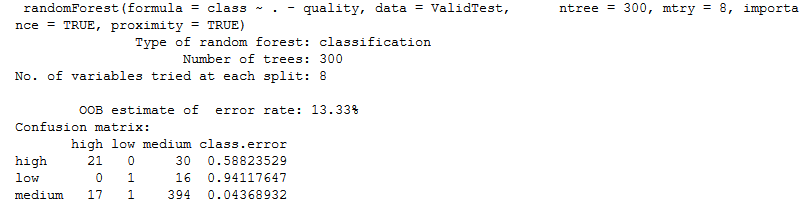
high low medium class.error

high 58 0 0 0

low 0 19 0 0

medium 0 0 395 0

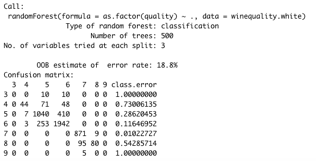


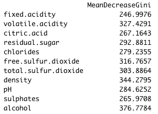


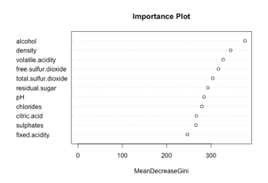
White Wine:

library("randomForest")

quality.rf <- randomForest(as.factor(quality) ~ ., data= winequality.white)







**#group together the high quality wines (7, 8, 9)**

highquality <- winequality.white[which(winequality.white$quality >= 7),]

**#group together the low quality wines (3,4,5,6)**

lowquality <- winequality.white[which(winequality.white$quality <= 6),]

**Run a t-test to determine which variables are significant**

#create a new data frame with only the columns with a significant difference

hqsigdif <- highquality[,c(-1,-2,-3,-4,-6,-9,-10)]

lqsigdif <- lowquality[,(-1,-2,-3,-4,-6,-9,-10)]

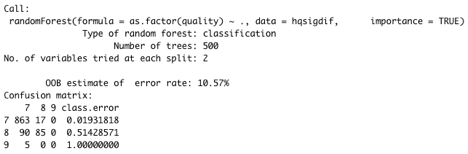
**#run a randomForest of the significant columns for low and high quality data frames**

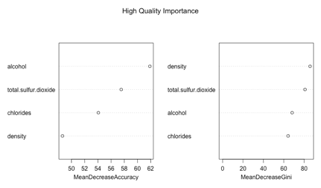
library(randomForest)

hq1 <- randomForest(as.factor(quality) ~., data= hqsigdif, importance = TRUE)

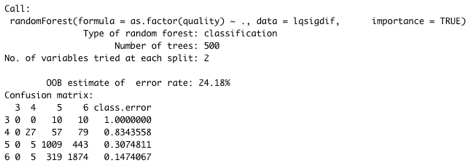
lq1 <- randomForest(as.factor(quality) ~., data= lqsigdif, importance= TRUE)

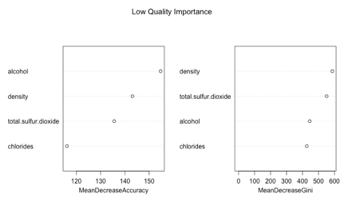
hq1





Lq1





### SVM

Red Wine:

library(e1071)

> mysvm <- svm(class~., data= dfrw2)

> summary(mysvm)

Call:

svm(formula = class ~ ., data = dfrw2)

Parameters:

SVM-Type: C-classification

SVM-Kernel: radial

cost: 1

Number of Support Vectors: 340

( 233 71 36 )

Number of Classes: 3

Levels:

high low medium

> mysvm2 <- svm(class~., data= testdata)

> summary(mysvm2)

Call:

svm(formula = class ~ ., data = testdata)

Parameters:

SVM-Type: C-classification

SVM-Kernel: radial

cost: 1

Number of Support Vectors: 192

( 135 38 19 )

Number of Classes: 3

Levels:

high low medium

predwine2 <- predict(mysvm2, dfrw2)

> confusionMatrix(dfrw2$class, predwine2)

Confusion Matrix and Statistics

Reference

Prediction high low medium

high 204 0 13

low 0 51 12

medium 0 0 1319

Overall Statistics

Accuracy : 0.9844

95% CI : (0.977, 0.9899)

No Information Rate : 0.8405

P-Value [Acc > NIR] : < 2.2e-16

Kappa : 0.9457

Mcnemar's Test P-Value : NA

Statistics by Class:

Class: high Class: low Class: medium

Sensitivity 1.0000 1.00000 0.9814

Specificity 0.9907 0.99225 1.0000

Pos Pred Value 0.9401 0.80952 1.0000

Neg Pred Value 1.0000 1.00000 0.9107

Prevalence 0.1276 0.03189 0.8405

Detection Rate 0.1276 0.03189 0.8249

Detection Prevalence 0.1357 0.03940 0.8249

Balanced Accuracy 0.9953 0.99612 0.9907

----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

|  |
| --- |
| predwine <- predict(mysvm3, dfrw2)  > confusionMatrix(dfrw2$class, predwine)  Confusion Matrix and Statistics  Reference  Prediction high low medium  high 215 0 2  low 0 61 2  medium 0 0 1319  Overall Statistics    Accuracy : 0.9975  95% CI : (0.9936, 0.9993)  No Information Rate : 0.8274  P-Value [Acc > NIR] : < 2.2e-16    Kappa : 0.9916    Mcnemar's Test P-Value : NA  Statistics by Class:  Class: high Class: low Class: medium  Sensitivity 1.0000 1.00000 0.9970  Specificity 0.9986 0.99870 1.0000  Pos Pred Value 0.9908 0.96825 1.0000  Neg Pred Value 1.0000 1.00000 0.9857  Prevalence 0.1345 0.03815 0.8274  Detection Rate 0.1345 0.03815 0.8249  Detection Prevalence 0.1357 0.03940 0.8249  Balanced Accuracy 0.9993 0.99935 0.9985 |

### KSVM

White Wine:

install.packages("kernlab")

library("kernlab")

winequality.white$highorlow <- ifelse(winequality.white$quality >= 7, "yes", "no")

winequality.white$highorlow <- as.factor(winequality.white$highorlow)

nrows <- nrow(winequality.white)

cutpoint <- floor(nrows/3\*2)

rand <- sample(1:nrows)

wine.train <- winequality.white[rand[1:cutpoint],]

wine.test <- winequality.white[rand[(cutpoint+1):nrows],]

model <- ksvm(highorlow ~., data= wine.train)

pred <- predict(model, wine.test)

table(pred, wine.test$highorlow)



# Limitations

We found there to be several limitations. Some of the limitations we found when doing our research about wine. We did not have access to a wine expert to better understand the science behind the process of wine testing. We found that a many of the factors for wine quality are unmeasurable. These include: the amount of time they were fermented, the grapes they were made with, the barrels they sat in, and where they were made. These factors are huge in determining the quality of wine and we do not have this data. Another limitation is that quality is a very subjective variable. People have a variety of palettes that can determine the way they rate the wine. What our analysis does is give a general idea of what the person/people who rated this wine think. It is hard to come up with a definitive way of determining the quality of the wine because of the subjectivity.

# Conclusion

We can create models that can accurately predict whether a wine will be high quality or low quality based on the variables provided. The ksvm model had 99.9% accuracy in classifying whether a wine was high or low quality. When we ran the randomForest, we got the significance of the variables that were used in the randomForest. The top 4 variables were alcohol, density, volatile acidity, and free sulfur dioxide. Based on this information, we made 4 graphs showing how these variables effect the quality. The color represents the quality where red is low quality, orange is medium quality, and green in the higher quality. You can see the patterns in some of the graphs more clearly than others. The thing that we looked at the most is where the clusters of colors were. You can clearly see that lower density and higher alcohol content usually resulted in the wine being rated as a higher quality. There also seems to be a pattern where lower volatile acidity results in a higher to medium level qualities, but this trend isn’t as strong as the others. One reason the wines with high volatile acidity may be rated lower is because these wines tend to smell more strongly of vinegar which may result in a lower rating.

# References

## Data Set:

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.

## Research on wine quality:

Gardner, D. M. (2015, September 30). *Volatile Acidity in Wine*. Retrieved from <https://extension.psu.edu/volatile-acidity-in-wine>

Maggrett, E. (2017, September 28). *Qualities of a Good Wine*. Retrieved from <https://oureverydaylife.com/qualities-of-a-good-wine-12259526.html>

McCarthy, E., & Ewing-Mulligan, M. (n.d.). *Wine Quality: How to Judge Good or Bad Wines*. Retrieved from <https://www.dummies.com/food-drink/drinks/wine/wine-quality-how-to-judge-good-or-bad-wines/>

Puckette, M. (2017, January 25). *What is Brix? Winemaking Secrets*. Retrieved from <https://winefolly.com/deep-dive/brix-secrets-wine/>

Puckette, M., & Hammack, J. (2015). *Wine folly: The essential guide to wine*. New York, NY: Avery, un imprint of Penguin Random House.

## Code Research

Facer, C. (2018, October 23). *How to Create a Correlation Matrix in R*. Retrieved from <https://www.displayr.com/how-to-create-a-correlation-matrix-in-r/>

Spector, P. (n.d.). *Using t-tests in R*. Retrieved from <https://statistics.berkeley.edu/computing/r-t-tests>

Wei, T. (n.d.). *corrplot*. Retrieved from <https://www.rdocumentation.org/packages/corrplot/versions/0.84/topics/corrplot>