Final Project

Nitya Kari

2023-12-15

Load Data and Packages

```
data <- read.csv("/Users/nitwit/Desktop/winequality-red.csv")
```

This dataset contains information about physicochemical factors of different types of red Portuguese "Vinho Verde" wine. There are several predictor variables in this dataset (fixed acidity, volatile acidity, citric acid, residual sugar, chlorides, free sulfur dioxide, total sulfur dioxide, density, pH, sulphates, alcohol) with one outcome variable (quality). All of the predictor variables are continuous numeric variables as they can take any number of values while the outcome variable (quality) is an ordinal numeric variable as it rates each wine on a scale of 0 to 10. There are 1599 seperate values in this dataset.

This dataset was obtained from Kaggle.com (https://www.kaggle.com/datasets/uciml/red-wine-quality-cortez-et-al-2009/) but it can also be found on the UCI Machine Learning repository (https://archive.ics.uci.edu/ml/datasets/wine+quality).

Here is the citation for this dataset: 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.

```
require(corrplot)
```

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

```
## corrplot 0.92 loaded
```

```
require(dplyr)
```

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

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

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

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

```
require(tidyverse)
```

```
## Loading required package: tidyverse
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.2 --
## v ggplot2 3.4.0     v purrr   1.0.1
## v tibble  3.2.1     v stringr 1.5.0
## v tidyr   1.3.0     v forcats 0.5.2
## v readr   2.1.3
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
require(ggplot2)
require(e1071)
```

```
## Loading required package: e1071
```

```
require(stringr)
require(tidytext)
```

```
## Loading required package: tidytext
```

```
require(ggplot2)
require(psych)
```

```
## Loading required package: psych
##
## Attaching package: 'psych'
##
## The following objects are masked from 'package:ggplot2':
##
##     %+%, alpha
```

```
require(caret)
```

```
## Loading required package: caret
## Loading required package: lattice
##
## Attaching package: 'caret'
##
## The following object is masked from 'package:purrr':
##
##     lift
```

```
require(GGally)
```

```
## Loading required package: GGally
## Registered S3 method overwritten by 'GGally':
##   method from
##   +.gg   ggplot2
```

```
require(glmnet)
```

```
## Loading required package: glmnet
## Loading required package: Matrix
```

```
## Warning: package 'Matrix' was built under R version 4.2.3
```

```
##
## Attaching package: 'Matrix'
##
## The following objects are masked from 'package:tidyr':
##
##     expand, pack, unpack
##
## Loaded glmnet 4.1-8
```

```
library(factoextra)
```

```
## Welcome! Want to learn more? See two factoextra-related books at https://goo.gl/ve3WBa
```

```
library(rms)
```

```
## Loading required package: Hmisc
##
## Attaching package: 'Hmisc'
##
## The following object is masked from 'package:psych':
##
##     describe
##
## The following object is masked from 'package:e1071':
##
##     impute
##
## The following objects are masked from 'package:dplyr':
##
##     src, summarize
##
## The following objects are masked from 'package:base':
##
##     format.pval, units
```

Research Question

Can the predictor variables attained from physicochemical tests in this dataset be aggregated to be used to accurately predict the quality of red wine? This question is important because it will allow people to get a better understanding of which physicochemical factors of wine would make a wine good. This can help people understand which wines they should gravitate towards if given this information. It can also help wine companies figure out which physicochemical properties they should focus on when they develop their alcohol.

Variables of Interest

Predictor Variable (all continuous numeric variables): 1 - fixed acidity: most acids involved with wine or fixed or nonvolatile (do not evaporate readily) 2 - volatile acidity: the amount of acetic acid in wine, which at too high of levels can lead to an unpleasant, vinegar taste 3 - citric acid: found in small quantities, citric acid can add 'freshness' and flavor to wines 4 - residual sugar: the amount of sugar remaining after fermentation stops 5 - chlorides: the amount of salt in the wine 6 - free sulfur dioxide: the free form of SO2 exists in equilibrium between molecular SO2 (as a dissolved gas) and bisulfite ion 7 - total sulfur dioxide: amount of free and bound forms of S02 8 - density: the density of water is close to that of water depending on the percent alcohol and sugar content 9 - pH: describes how acidic or basic a wine is on a scale from 0 (very acidic) to 14 (very basic) 10 - sulphates: a wine additive which can contribute to sulfur dioxide gas (S02) levels 11 - alcohol: the percent alcohol content of the wine

Outcome Variable (ordinal numeric variable): quality -> output variable (based on sensory data, score between 0 and 10)

Data Wrangling

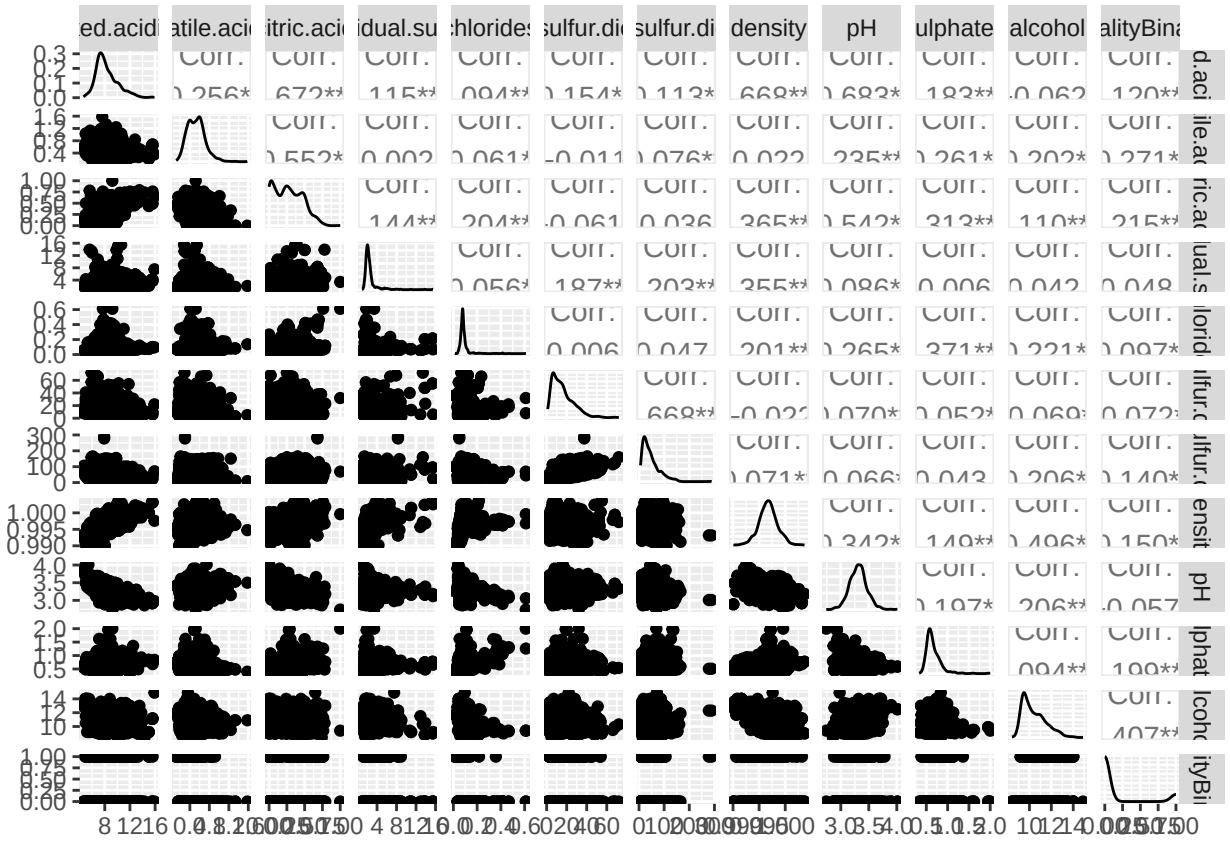
```
data <- na.omit(data) #drop the NAs in dataset

# creating a binary variable based off of the quality variable
data <- data %>%
  mutate(qualityBinary = case_when(
    quality < 7 ~ 0, # if the quality of the wine is rated less than 7, then it is not good quality and
    quality >= 7 ~ 1 # if the quality of the wine is rated 7 or higher, then it is good quality and is
  ))

# Selecting relevant variables
dataSelection <- data[, c("fixed.acidity", "volatile.acidity", "citric.acid", "residual.sugar", "chlori
```

After filtering out NA values, I created a binary variable of quality where I assigned a value of "0" to any wine that recieved a quality score of less than 7 on the 0-10 scale and I assigned a value of "1" to any wine that recieved a score of 7 or higher. Essentially, 0 means that the quality was not good and 1 means that the quality was good.

Visualization

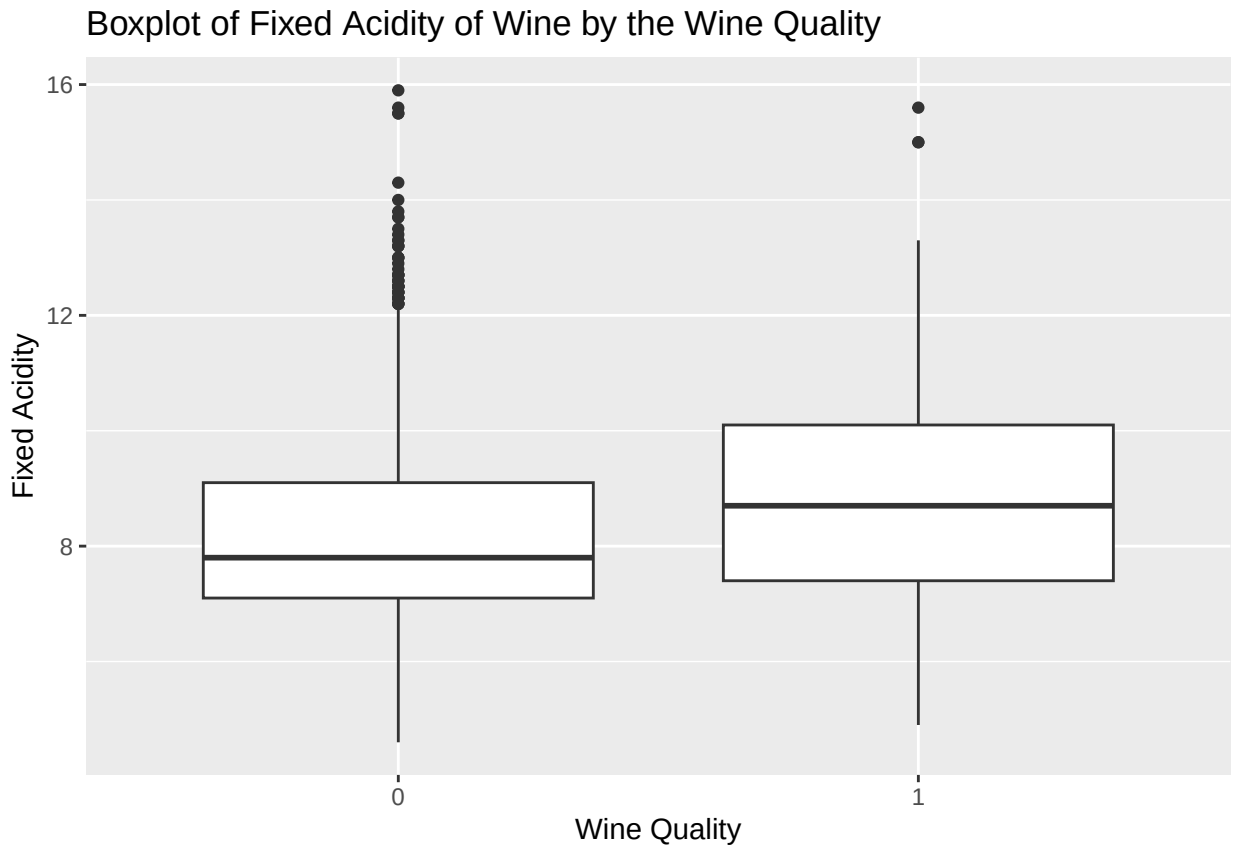
Since the outcome variable (qualityBinary) is now a binary variable, we will be using box plots to visualize the predictor variables.

```
ggpairs(dataSelection)
```

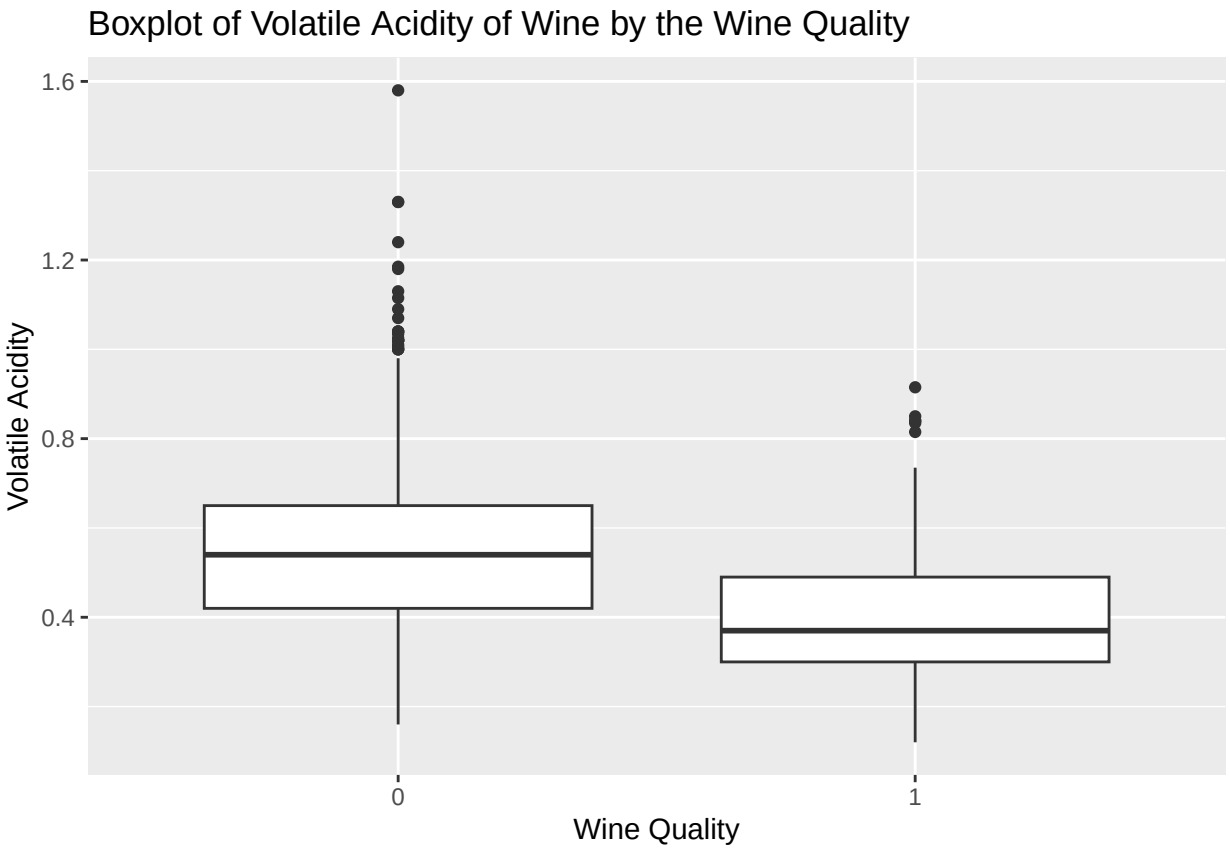



```
# Visualize the predictors by outcome variable
plot <- ggplot(dataSelection, aes(x = factor(qualityBinary), y = fixed.acidity)) +
  geom_boxplot() +
  labs(x = "Wine Quality", y = "Fixed Acidity") +
  ggtitle("Boxplot of Fixed Acidity of Wine by the Wine Quality")

plot2 <- ggplot(dataSelection, aes(x = factor(qualityBinary), y = volatile.acidity)) +
  geom_boxplot() +
  labs(x = "Wine Quality", y = "Volatile Acidity") +
  ggtitle("Boxplot of Volatile Acidity of Wine by the Wine Quality")

plot3 <- ggplot(dataSelection, aes(x = factor(qualityBinary), y = citric.acid)) +
  geom_boxplot() +
  labs(x = "Wine Quality", y = "Citric Acid") +
  ggtitle("Boxplot of Citric Acid in Wine by the Wine Quality")

plot4 <- ggplot(dataSelection, aes(x = factor(qualityBinary), y = residual.sugar)) +
  geom_boxplot() +
  labs(x = "Wine Quality", y = "Residual Sugar") +
  ggtitle("Boxplot of Residual Sugar in Wine by the Wine Quality")

plot5 <- ggplot(dataSelection, aes(x = factor(qualityBinary), y = chlorides)) +
  geom_boxplot() +
```

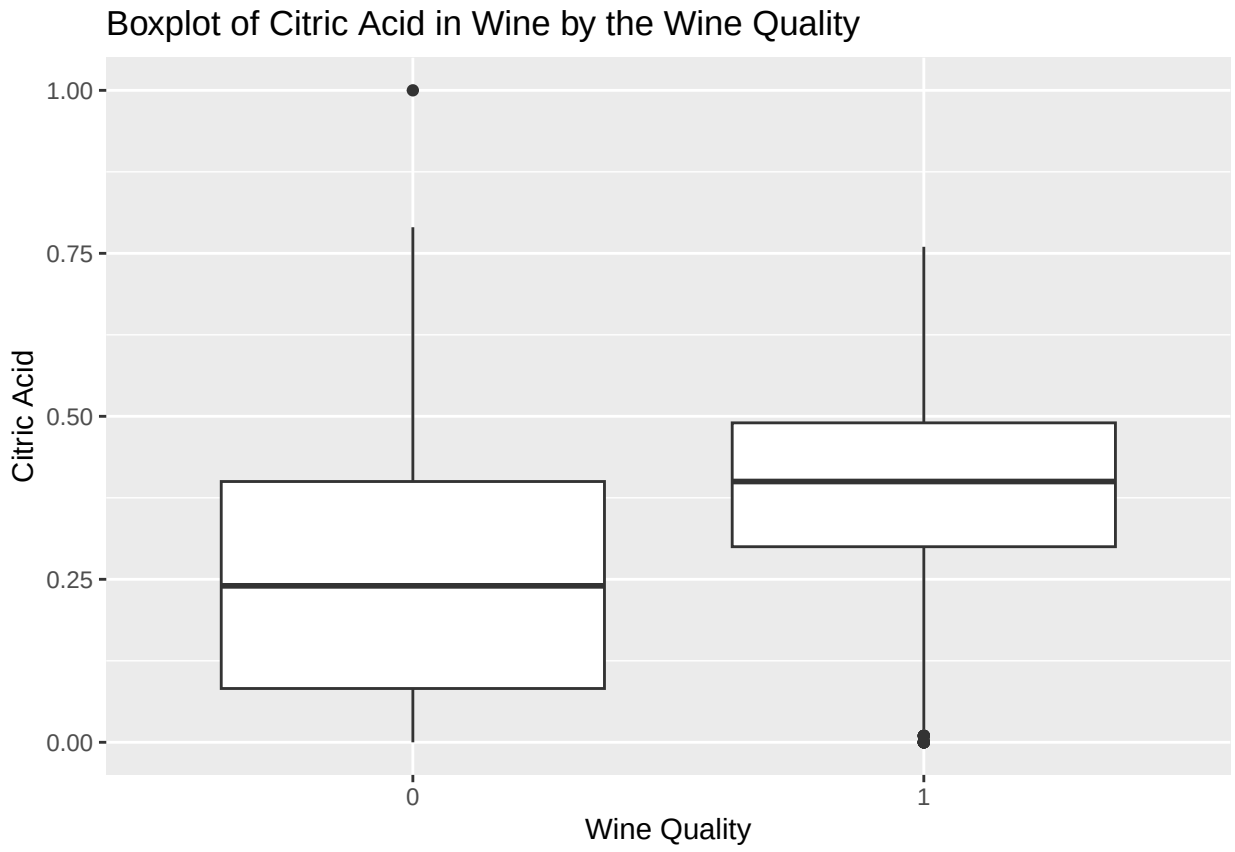
```
  labs(x = "Wine Quality", y = "Chlorides") +
  ggtitle("Boxplot of Chlorides in Wine by the Wine Quality")

plot6 <- ggplot(dataSelection, aes(x = factor(qualityBinary), y = free.sulfur.dioxide)) +
  geom_boxplot() +
  labs(x = "Wine Quality", y = "Free Sulfur Dioxide") +
  ggtitle("Boxplot of Free Sulfur Dioxide in Wine by the Wine Quality")

plot7 <- ggplot(dataSelection, aes(x = factor(qualityBinary), y = total.sulfur.dioxide)) +
  geom_boxplot() +
  labs(x = "Wine Quality", y = "Total Sulfur Dioxide") +
  ggtitle("Boxplot of Total Sulfur Dioxide in Wine by the Wine Quality")

plot8 <- ggplot(dataSelection, aes(x = factor(qualityBinary), y = density)) +
  geom_boxplot() +
  labs(x = "Wine Quality", y = "Density") +
  ggtitle("Boxplot of Density of Wine by the Wine Quality")

plot9 <- ggplot(dataSelection, aes(x = factor(qualityBinary), y = pH)) +
  geom_boxplot() +
  labs(x = "Wine Quality", y = "pH") +
  ggtitle("Boxplot of pH of Wine by the Wine Quality")

plot10 <- ggplot(dataSelection, aes(x = factor(qualityBinary), y = sulphates)) +
  geom_boxplot() +
  labs(x = "Wine Quality", y = "Sulphates") +
  ggtitle("Boxplot of Sulphates of Wine by the Wine Quality")

plot11 <- ggplot(dataSelection, aes(x = factor(qualityBinary), y = alcohol)) +
  geom_boxplot() +
  labs(x = "Wine Quality", y = "Alcohol") +
  ggtitle("Boxplot of Alcohol Content of Wine by the Wine Quality")

plot
```


Boxplot of Fixed Acidity of Wine by the Wine Quality
16
12
8
Fixed Acidity
0
1
Wine Quality

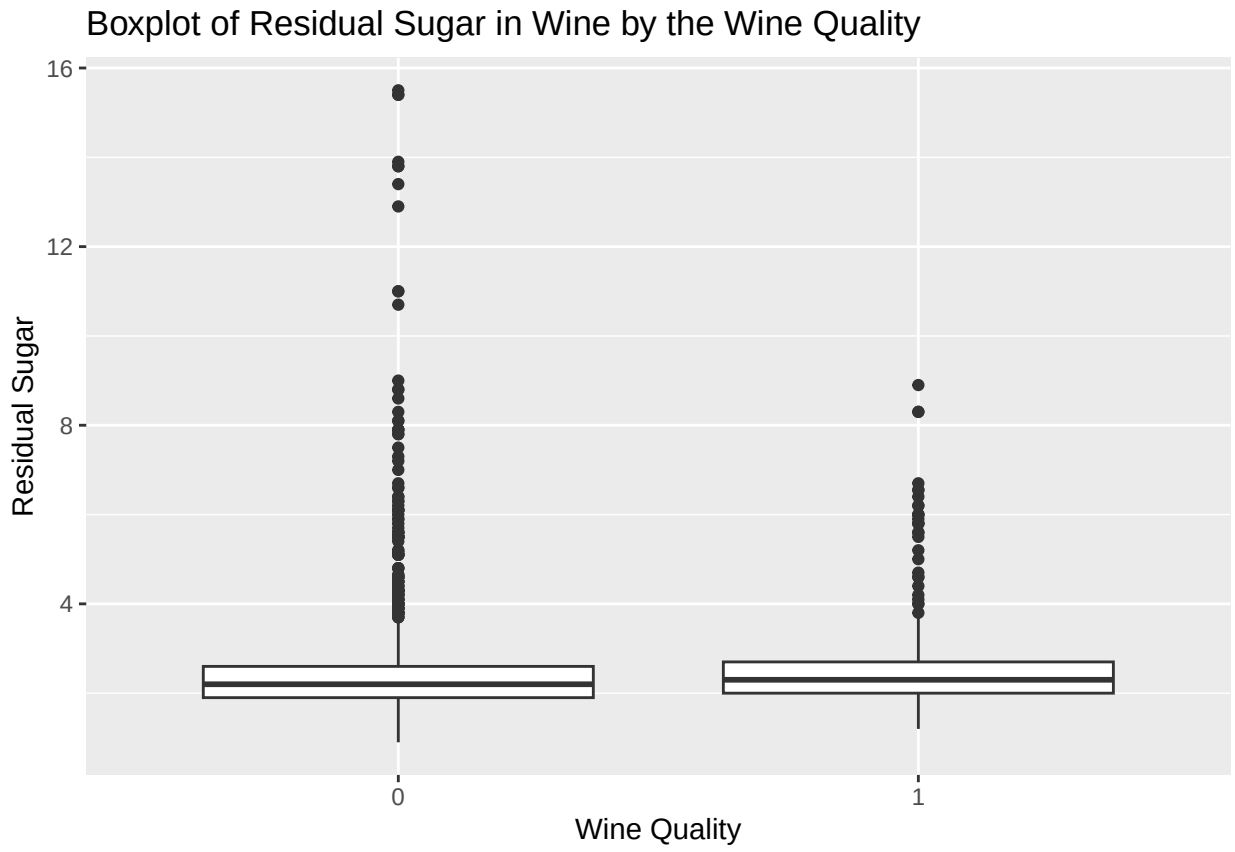
```
plot2
```


Boxplot of Volatile Acidity of Wine by the Wine Quality
Volatile Acidity
1.6
1.2
0.8
0.4
0
1
Wine Quality

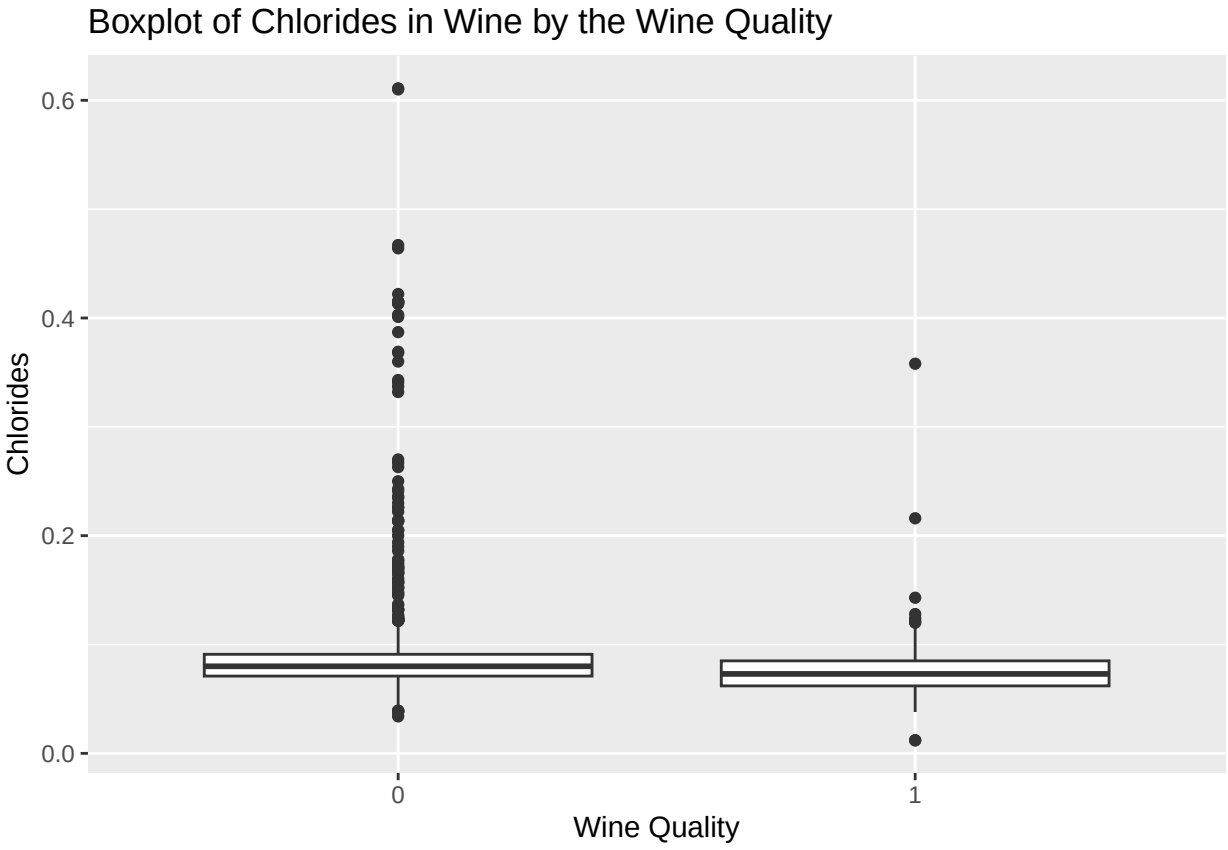
```
plot3
```


Boxplot of Citric Acid in Wine by the Wine Quality
1.00
0.75
0.50
0.25
0.00
Citric Acid
0
1
Wine Quality

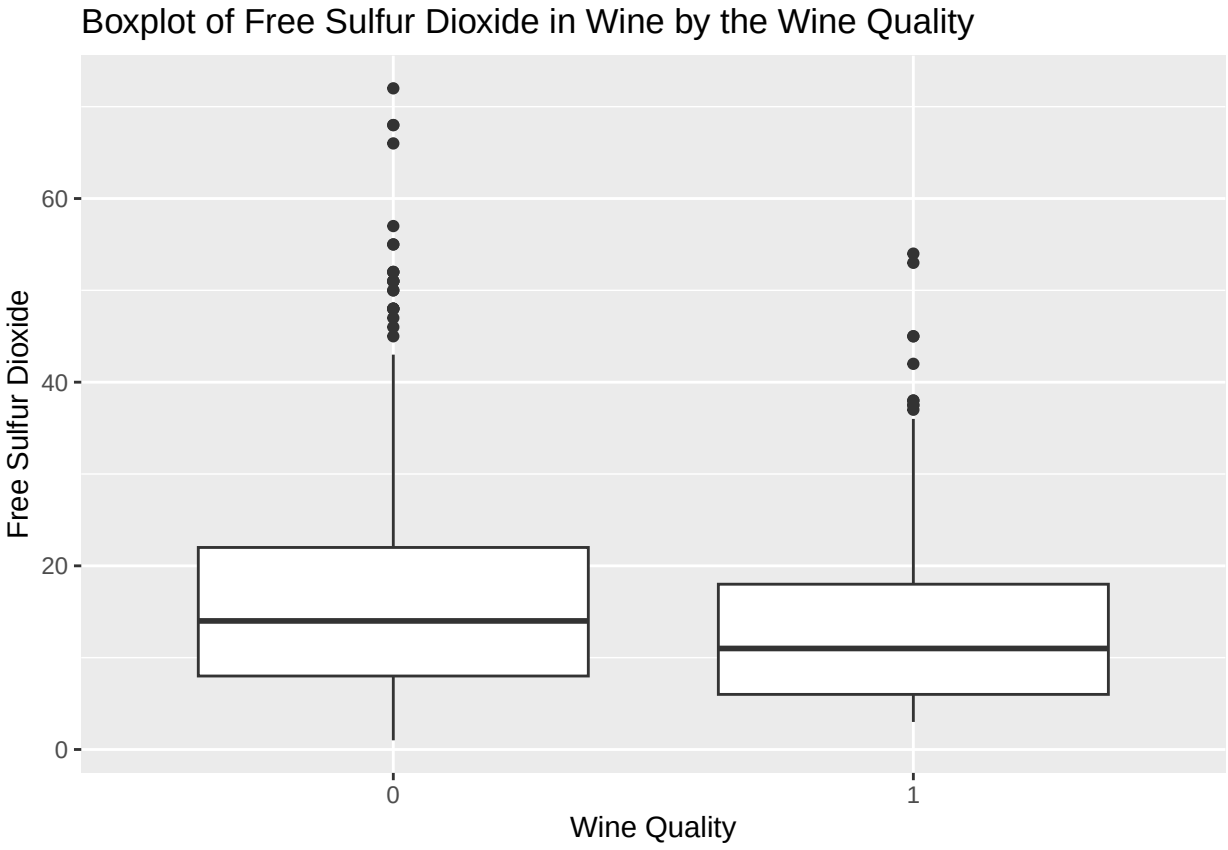
```
plot4
```


Boxplot of Residual Sugar in Wine by the Wine Quality
16
12
8
4
Residual Sugar
0
1
Wine Quality

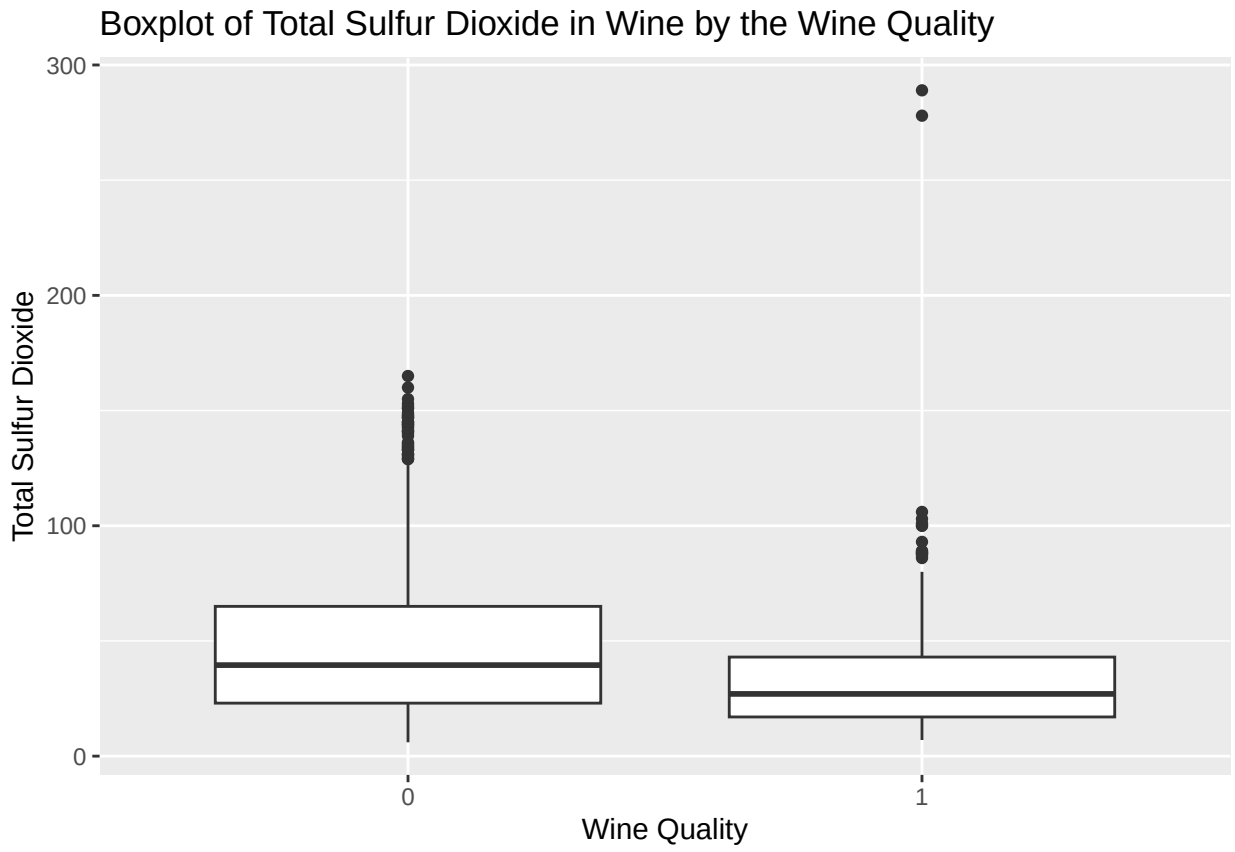
```
plot5
```


Boxplot of Chlorides in Wine by the Wine Quality
Chlorides
0.6
0.4
0.2
0.0
0
1
Wine Quality

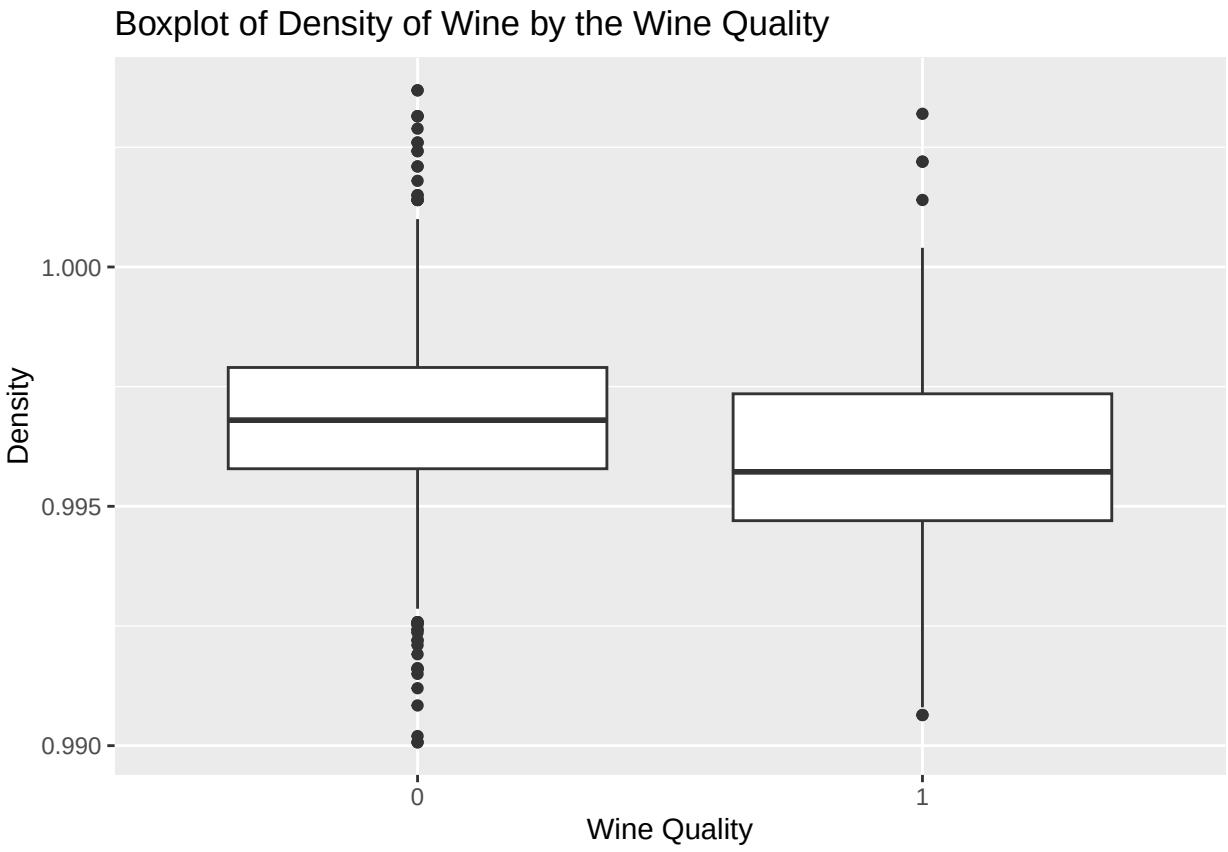
```
plot6
```


Boxplot of Free Sulfur Dioxide in Wine by the Wine Quality
Free Sulfur Dioxide
60
40
20
0
0
1
Wine Quality

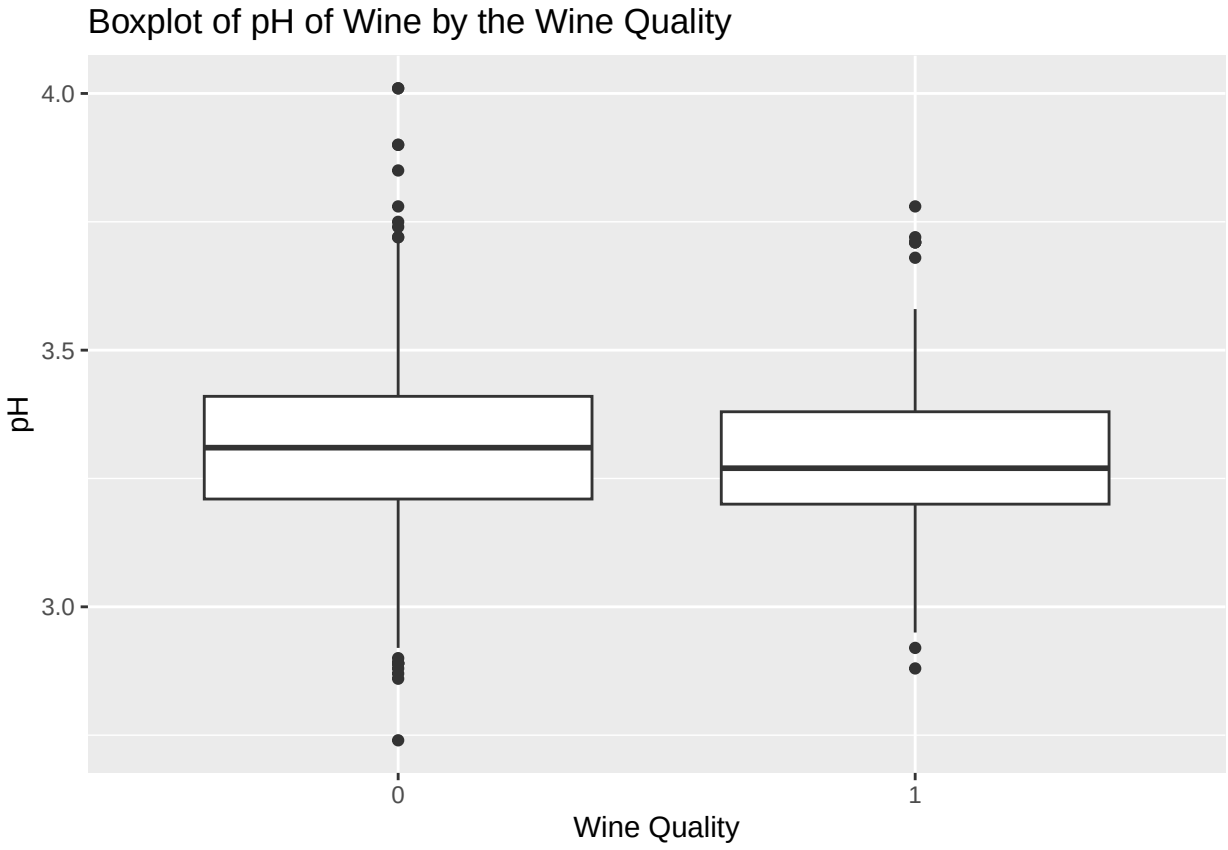
```
plot7
```


Boxplot of Total Sulfur Dioxide in Wine by the Wine Quality
Total Sulfur Dioxide
300
200
100
0
0
1
Wine Quality

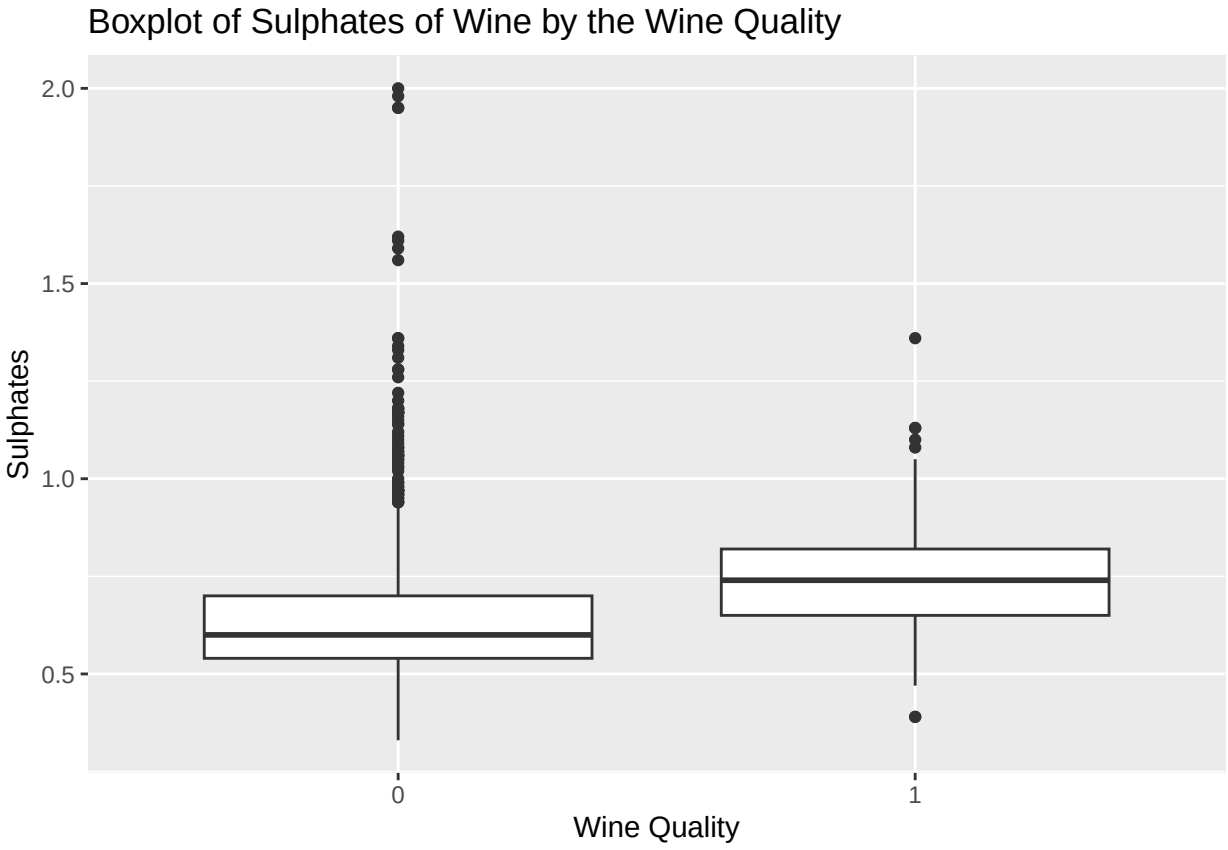
```
plot8
```


Boxplot of Density of Wine by the Wine Quality
Density
1.000
0.995
0.990
0
1
Wine Quality

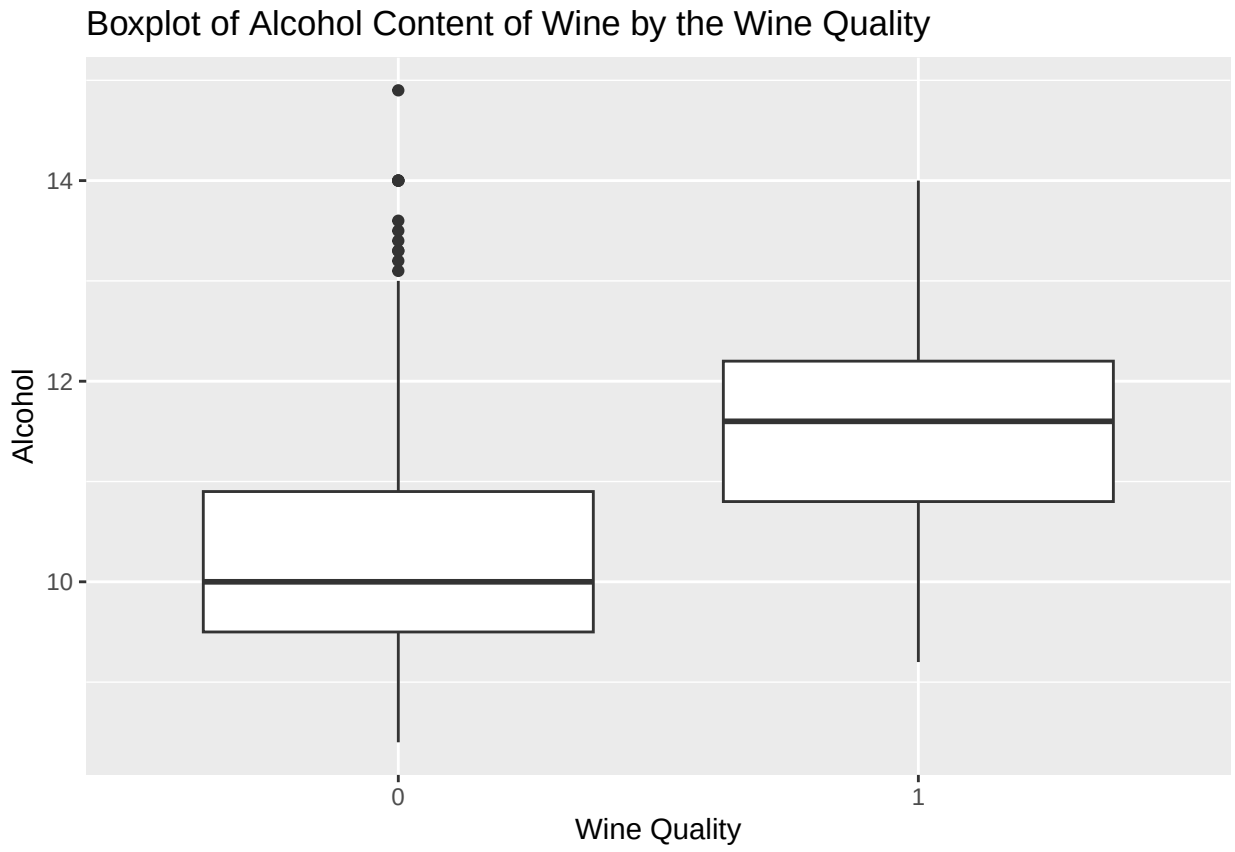
```
plot9
```


Boxplot of pH of Wine by the Wine Quality
4.0
3.5
3.0
pH
0
1
Wine Quality

```
plot10
```


Boxplot of Sulphates of Wine by the Wine Quality
Sulphates
2.0
1.5
1.0
0.5
0
1
Wine Quality

```
plot11
```



Using the boxplots, we can easily see which physicochemical factors are more or less present in better quality red wines. Fixed acidity, citric acid, residual sugar, sulphates, and the alcohol content all appear to be higher in better quality wines while volatile acidity, chlorides, free sulfur dioxide, total sulfur dioxide, density, and pH appear to be lower.

Dimension Reduction through PCA

First, I will be checking for multicollinearity

```
corr_data <- cor(dataSelection) #find correlations of all variables
corr_data
```

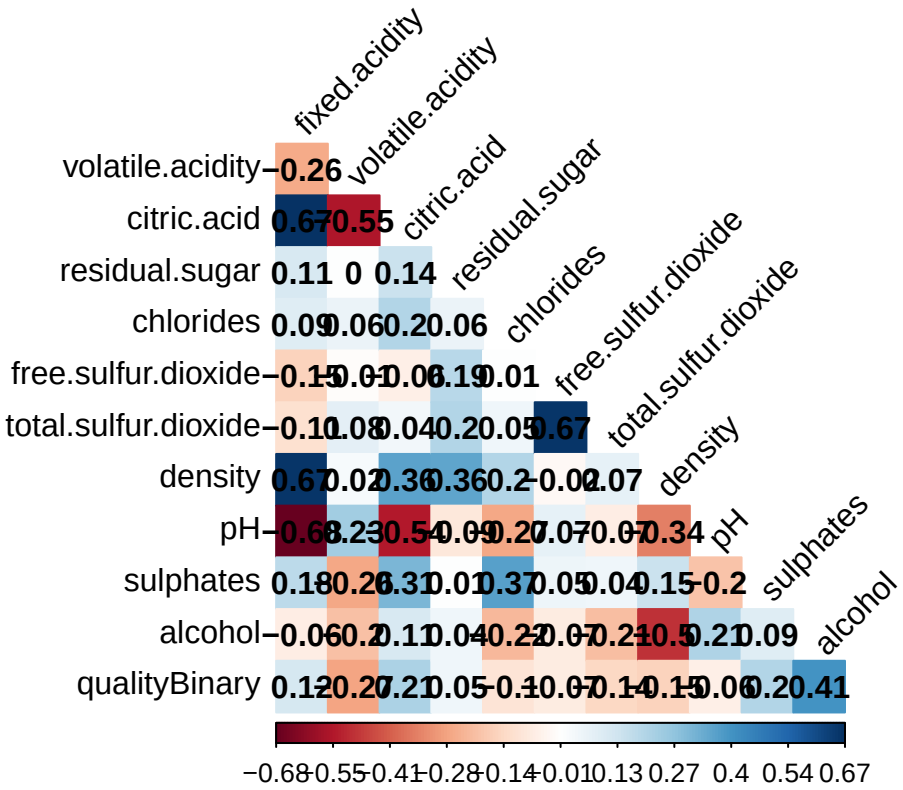
```
##                       fixed.acidity volatile.acidity citric.acid residual.sugar
## fixed.acidity            1.00000000     -0.256130895  0.67170343    0.114776724
## volatile.acidity        -0.25613089      1.000000000 -0.55249568    0.001917882
## citric.acid              0.67170343     -0.552495685  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
```

```
## qualityBinary           0.12006104     -0.270711532  0.21471559    0.047778946
##                        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
## qualityBinary        -0.097307638        -0.071747296          -0.13951655
##                           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
## qualityBinary        -0.15045968 -0.05728334  0.199485209  0.40731485
##                      qualityBinary
## fixed.acidity           0.12006104
## volatile.acidity       -0.27071153
## citric.acid             0.21471559
## residual.sugar          0.04777895
## chlorides              -0.09730764
## free.sulfur.dioxide    -0.07174730
## total.sulfur.dioxide   -0.13951655
## density                -0.15045968
## pH                     -0.05728334
## sulphates               0.19948521
## alcohol                 0.40731485
## qualityBinary           1.00000000
```

```
par(oma = c(0, 0, 0, 0))

corrplot(
  corr_data,
  type = "lower",
  tl.pos = "ld",
  diag = FALSE,
  tl.cex = 1,
  method = "color",
  addCoef.col = "black",
  tl.col = "black",
  tl.srt = 45,
  is.corr = FALSE,
```

```
  addCoef.cex = 0.5,  # Adjust the size of the coefficient text
  number.digits = 2   # Set the number of digits after the decimal point
)
```

```
## Warning in text.default(pos.xlabel[, 1], pos.xlabel[, 2], newcolnames, srt =
## tl.srt, : "addCoef.cex" is not a graphical parameter

## Warning in text.default(pos.ylabel[, 1], pos.ylabel[, 2], newrownames, col =
## tl.col, : "addCoef.cex" is not a graphical parameter

## Warning in title(title, ...): "addCoef.cex" is not a graphical parameter
```



> Although there is correlation in the data, none of the correlation values are higher than threshold of 0.899. As such, there is no need to take into consideration multicollinearity for this dataset.

Before scaling the predictor variables, I will remove the outcome variable.

```
data_pca <- dataSelection[, c("fixed.acidity", "volatile.acidity", "citric.acid", "residual.sugar", "chl

#Scaling the variables
scaled_data <- data_pca %>%
  mutate_at(c(1:11), ~(scale(.) %>% as.vector)) #scale the variables

#Visualize PCA Data
viz_pca <- prcomp(scaled_data, center = TRUE,scale. = TRUE)
summary(viz_pca)
```

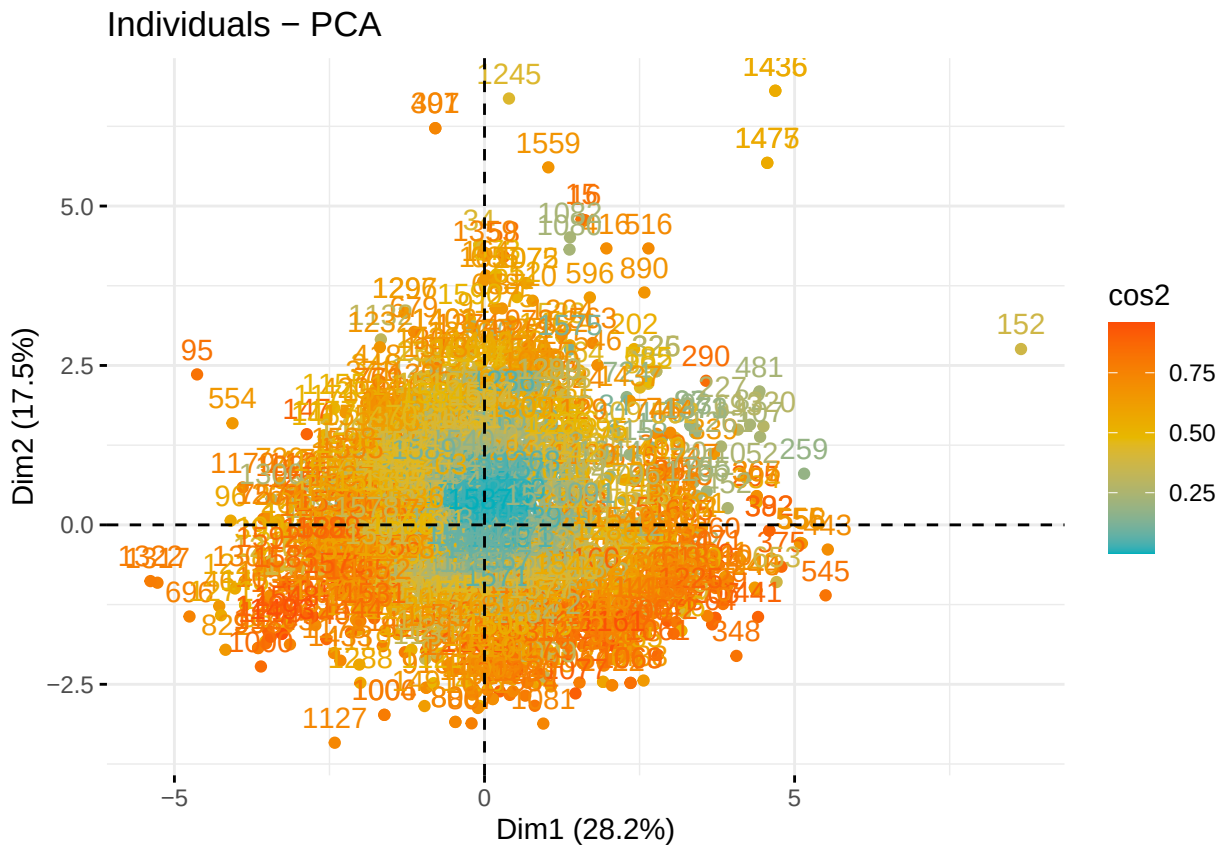
```
## Importance of components:
##                           PC1    PC2    PC3    PC4     PC5     PC6     PC7
## Standard deviation     1.7604 1.3878 1.2452 1.1015 0.97943 0.81216 0.76406
## Proportion of Variance 0.2817 0.1751 0.1410 0.1103 0.08721 0.05996 0.05307
## Cumulative Proportion  0.2817 0.4568 0.5978 0.7081 0.79528 0.85525 0.90832
##                           PC8     PC9    PC10    PC11
## Standard deviation     0.65035 0.58706 0.42583 0.24405
## Proportion of Variance 0.03845 0.03133 0.01648 0.00541
## Cumulative Proportion  0.94677 0.97810 0.99459 1.00000
```

```
viz_pca$rotation
```

```
##                              PC1          PC2         PC3          PC4
## fixed.acidity         0.48931422 -0.110502738  0.12330157 -0.229617370
## volatile.acidity     -0.23858436  0.274930480  0.44996253  0.078959783
## citric.acid           0.46363166 -0.151791356 -0.23824707 -0.079418256
## residual.sugar        0.14610715  0.272080238 -0.10128338 -0.372792562
## chlorides             0.21224658  0.148051555  0.09261383  0.666194756
## free.sulfur.dioxide  -0.03615752  0.513566812 -0.42879287 -0.043537818
## total.sulfur.dioxide  0.02357485  0.569486959 -0.32241450 -0.034577115
## density               0.39535301  0.233575490  0.33887135 -0.174499758
## pH                   -0.43851962  0.006710793 -0.05769735 -0.003787746
## sulphates             0.24292133 -0.037553916 -0.27978615  0.550872362
## alcohol              -0.11323206 -0.386180959 -0.47167322 -0.122181088
##                              PC5         PC6         PC7         PC8
## fixed.acidity         0.08261366 -0.10147858  0.35022736 -0.17759545
## volatile.acidity     -0.21873452 -0.41144893  0.53373510 -0.07877531
## citric.acid           0.05857268 -0.06959338 -0.10549701 -0.37751558
## residual.sugar       -0.73214429 -0.04915555 -0.29066341  0.29984469
## chlorides            -0.24650090 -0.30433857 -0.37041337 -0.35700936
## free.sulfur.dioxide   0.15915198  0.01400021  0.11659611 -0.20478050
## total.sulfur.dioxide  0.22246456 -0.13630755  0.09366237  0.01903597
## density              -0.15707671  0.39115230  0.17048116 -0.23922267
## pH                   -0.26752977  0.52211645  0.02513762 -0.56139075
## sulphates            -0.22596222  0.38126343  0.44746911  0.37460432
## alcohol              -0.35068141 -0.36164504  0.32765090 -0.21762556
##                             PC9         PC10         PC11
## fixed.acidity         0.194020908  0.24952314  0.639691452
## volatile.acidity     -0.129110301 -0.36592473  0.002388597
## citric.acid          -0.381449669 -0.62167708 -0.070910304
## residual.sugar        0.007522949 -0.09287208  0.184029964
## chlorides             0.111338666  0.21767112  0.053065322
## free.sulfur.dioxide   0.635405218 -0.24848326 -0.051420865
## total.sulfur.dioxide -0.592115893  0.37075027  0.068701598
## density               0.020718675  0.23999012 -0.567331898
## pH                   -0.167745886  0.01096960  0.340710903
## sulphates            -0.058367062 -0.11232046  0.069555381
## alcohol               0.037603106  0.30301450 -0.314525906
```

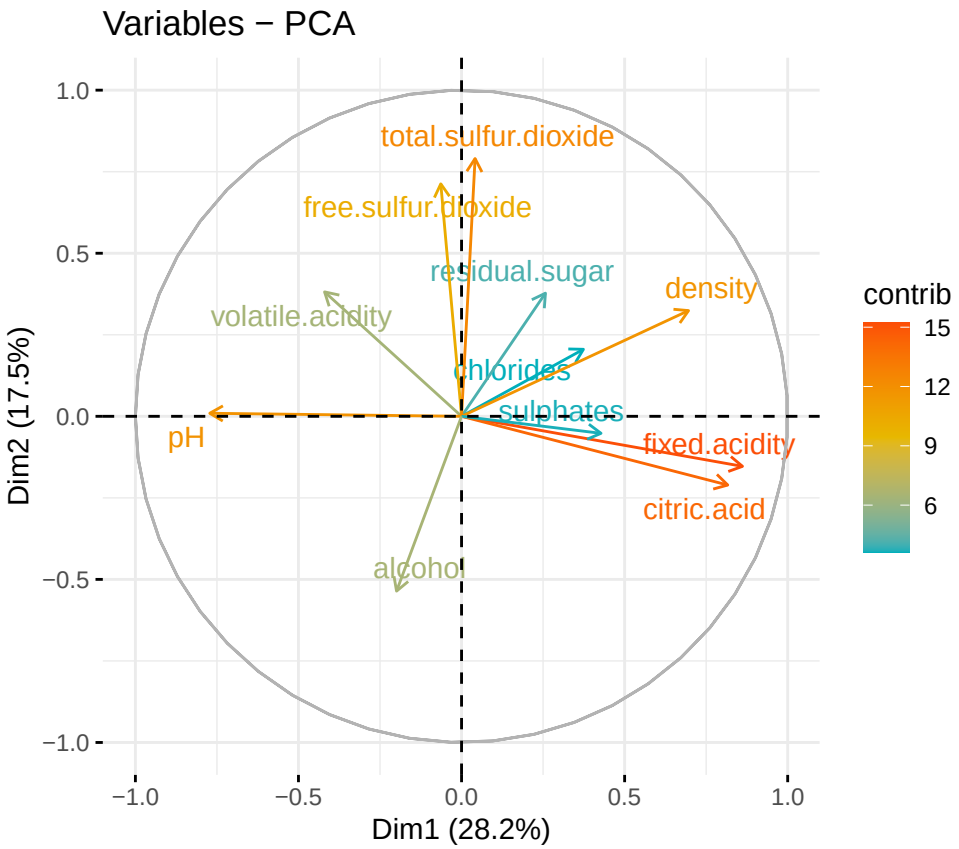
```
fviz_pca_ind(viz_pca,
             c = "point", # setting c to point
             col.ind = "cos2", # coloring based on the quality of representation,
             gradient.cols = c("#00AFBB", "#E7B800", "#FC4E07"), # setting the color gradient
```

```
             repel = FALSE
)
```



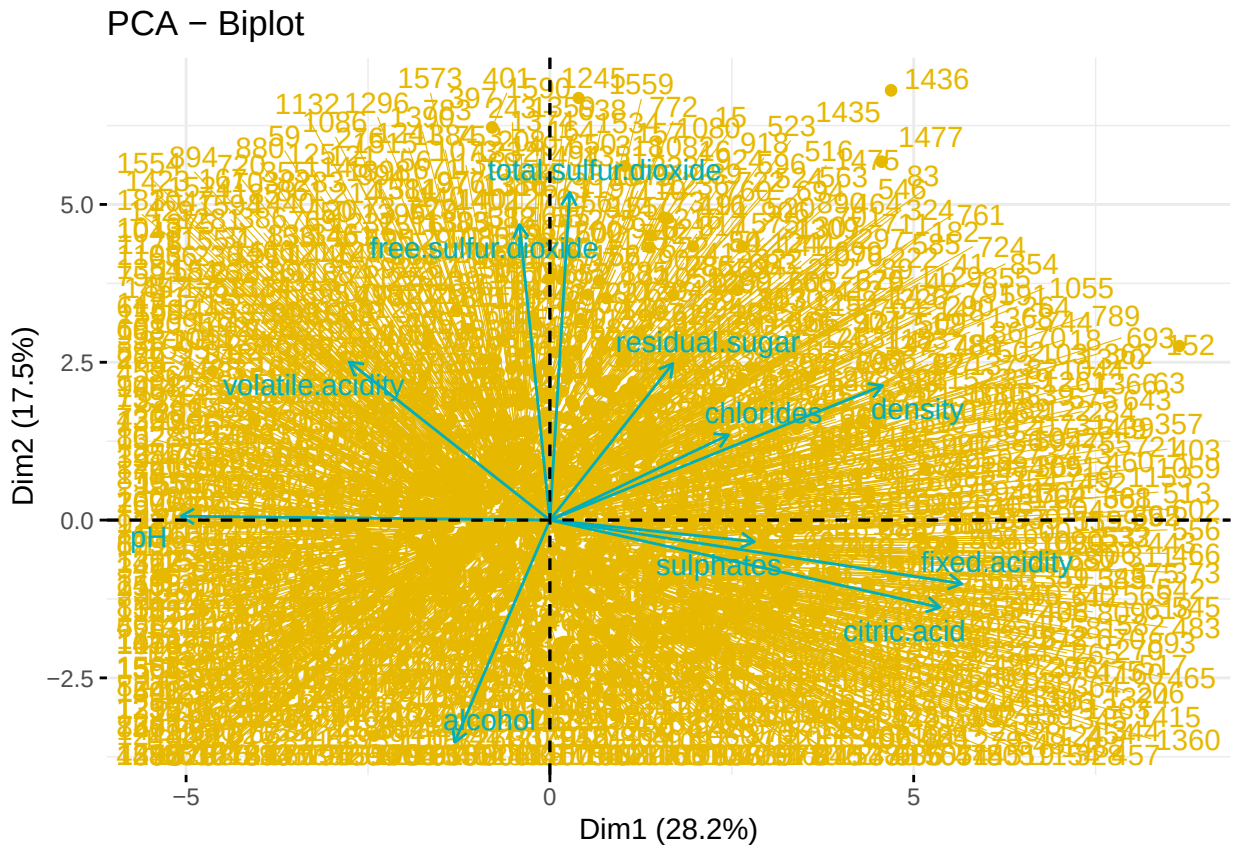
> It is hard to glean any specific information from this plot.

```
fviz_pca_var(viz_pca,
             col.var = "contrib", # coloring based on contribution to the PC
             gradient.cols = c("#00AFBB", "#E7B800", "#FC4E07"),
             repel = TRUE # mitigating text overlap
)
```



> Looking at this, we can see thta there may be some groupings forming. Fixed acidity, citric acid, and sulphates appear to be grouped together while density, chlorides, residual sugar, total and free sulfur dioxide could be grouped together.

```
fviz_pca_biplot(viz_pca, repel = TRUE,
 col.var = "#00AFBB",
 col.ind = "#E7B800" #this one combines the first two graphs
 )
```




```
# Barlett's test
cortest.bartlett(scaled_data, 1599)
```

```
## R was not square, finding R from data

## $chisq
## [1] 8017.566
##
## $p.value
## [1] 0
##
## $df
## [1] 55
```

```
# KMO
KMO(scaled_data)
```

```
## Kaiser-Meyer-Olkin factor adequacy
## Call: KMO(r = scaled_data)
## Overall MSA =  0.43
## MSA for each item =
##        fixed.acidity     volatile.acidity          citric.acid
##                 0.45                 0.52                 0.70
##       residual.sugar            chlorides  free.sulfur.dioxide
```

```
##                 0.21                 0.46                 0.48
## total.sulfur.dioxide              density                   pH
##                 0.45                 0.37                 0.45
##            sulphates              alcohol
##                 0.51                 0.23
```

Since the p-value indicates that this is significant, we can use PCA.

The Overall MSA value and the MSA values for each of the predictor variables are pretty subpar for the most part. The only outlier is citric acid with a MSA value of 0.7. For the sake of this project, I will be dropping variables with MSA values beneath 0.4–residual sugar and density.

```
scaled_data_pca <- scaled_data %>%
  dplyr::select(!density) %>%
  dplyr::select(!residual.sugar)

KMO(scaled_data_pca)
```

```
## Kaiser-Meyer-Olkin factor adequacy
## Call: KMO(r = scaled_data_pca)
## Overall MSA =  0.54
## MSA for each item =
##        fixed.acidity     volatile.acidity          citric.acid
##                 0.57                 0.49                 0.59
##            chlorides  free.sulfur.dioxide total.sulfur.dioxide
##                 0.39                 0.47                 0.40
##                   pH            sulphates              alcohol
##                 0.70                 0.64                 0.47
```

This is slightly better, but there are still low MSA values

```
scaled_data_pca <- scaled_data_pca %>%
  dplyr::select(!chlorides) %>%
  dplyr::select(!total.sulfur.dioxide )

KMO(scaled_data_pca)
```

```
## Kaiser-Meyer-Olkin factor adequacy
## Call: KMO(r = scaled_data_pca)
## Overall MSA =  0.68
## MSA for each item =
##       fixed.acidity    volatile.acidity         citric.acid free.sulfur.dioxide
##                0.67                0.66                0.69                0.52
##                  pH           sulphates             alcohol
##                0.73                0.84                0.48
```

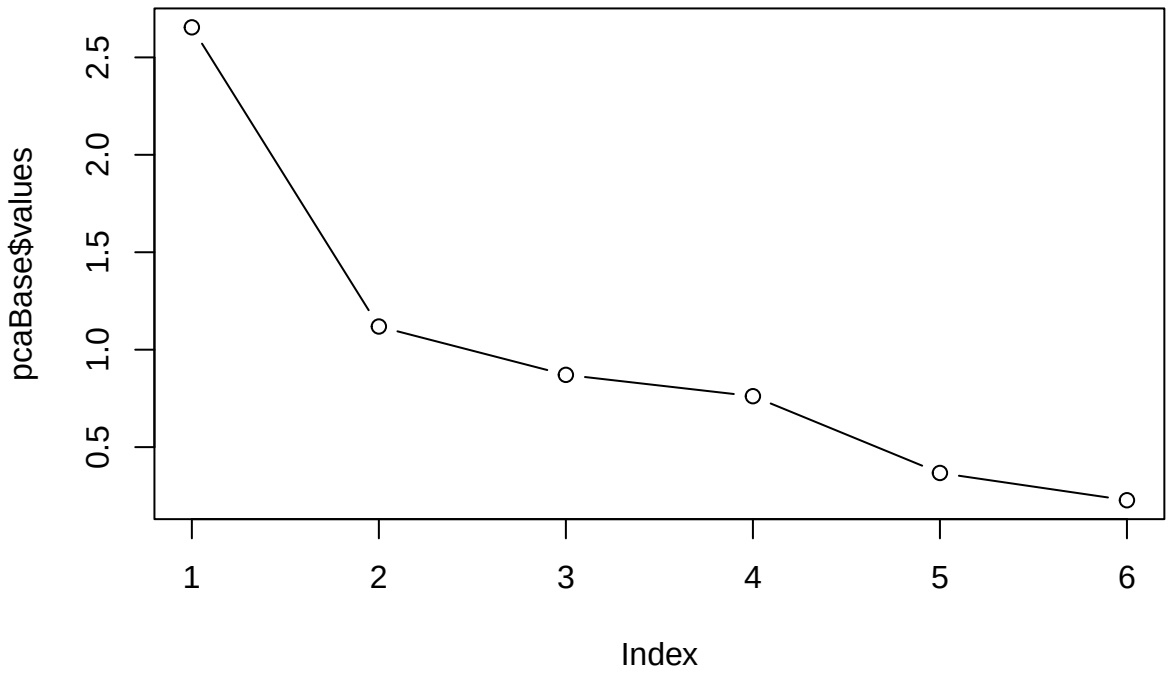
This is much better, but there are still low MSA values

```
scaled_data_pca <- scaled_data_pca %>%
  dplyr::select(!alcohol)

KMO(scaled_data_pca)
```

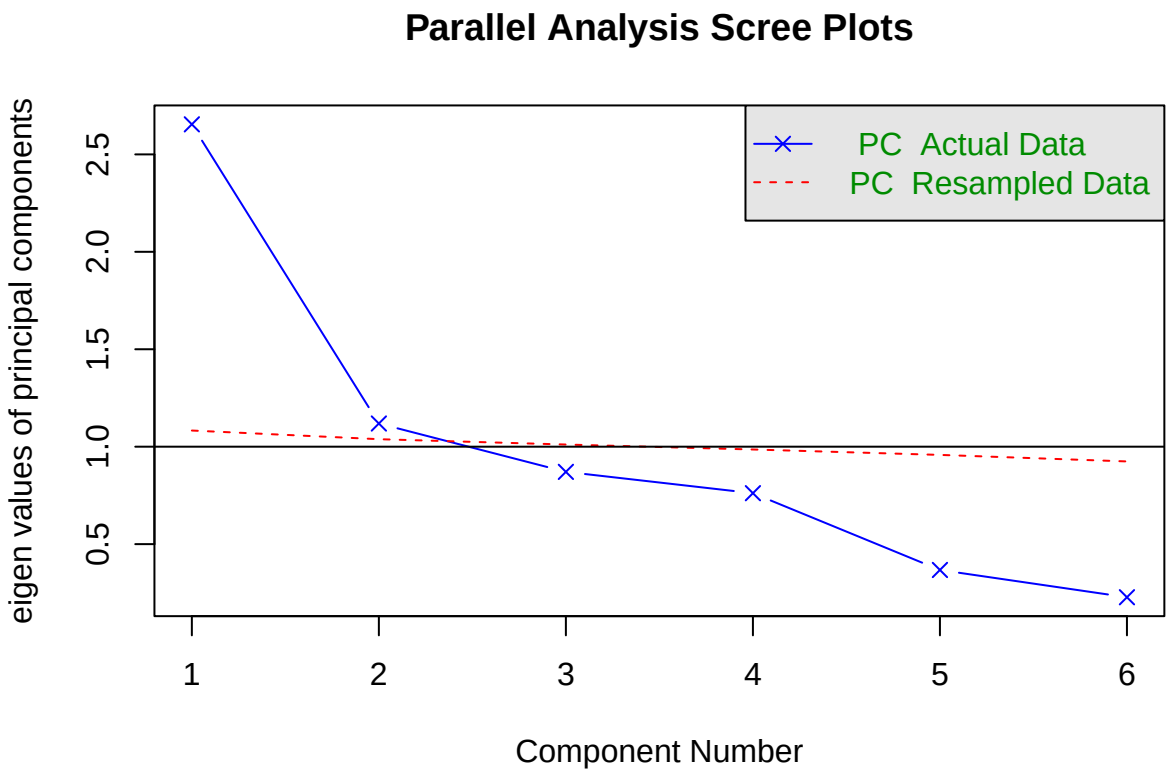
```
## Kaiser-Meyer-Olkin factor adequacy
## Call: KMO(r = scaled_data_pca)
## Overall MSA =  0.69
## MSA for each item =
##       fixed.acidity    volatile.acidity         citric.acid free.sulfur.dioxide
##                0.66                0.64                0.68                0.57
##                  pH           sulphates
##                0.76                0.84
```

We can now proceed!

```r
# base PCA
pcaBase <- principal(scaled_data_pca, nfactors = 6, rotate = "none")
plot(pcaBase$values, type = "b")
```




```r
parallel_pca <- fa.parallel(
  x = scaled_data_pca, fa = "pc",
  sim = FALSE # ensures resampling
)
```




```
## Parallel analysis suggests that the number of factors =  NA  and the number of components =  2
```

This indicates that 2 components would be best.

```
pca_resid <- principal(scaled_data_pca, nfactors = 2, rotate = "oblimin", residuals = TRUE)
```

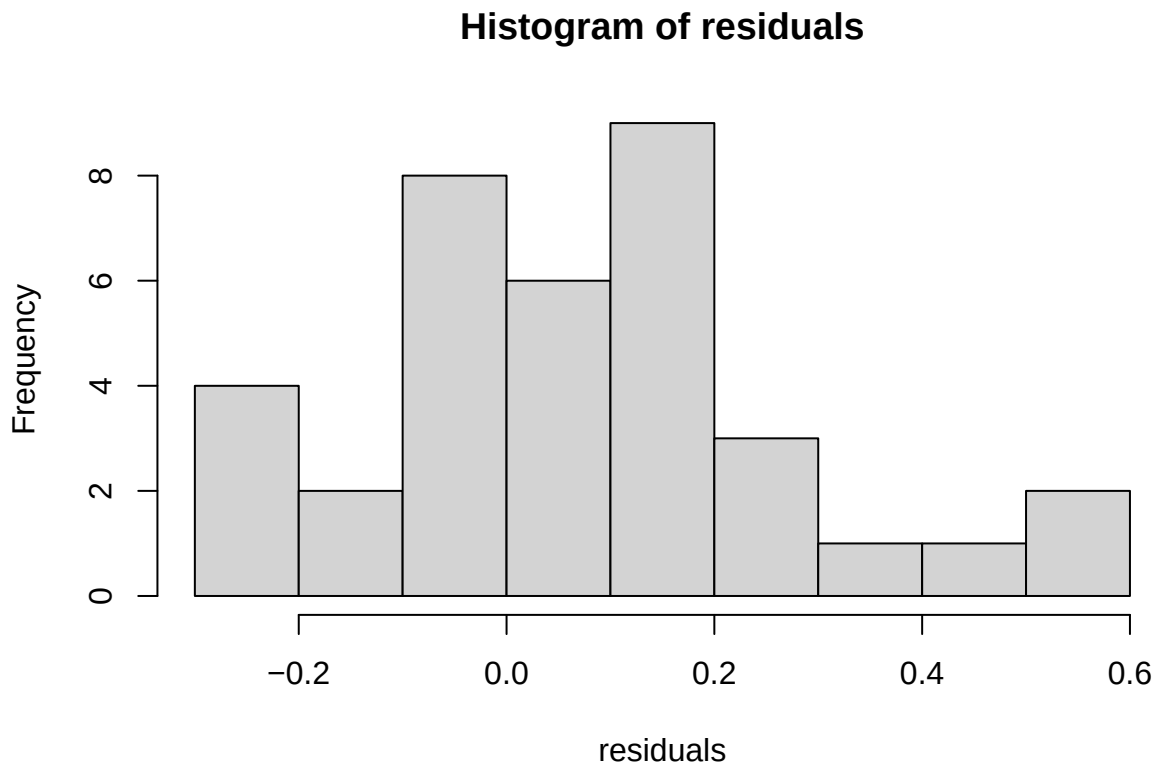
```
## Loading required namespace: GPArotation
```

```
pca_resid #results
```

```
## Principal Components Analysis
## Call: principal(r = scaled_data_pca, nfactors = 2, residuals = TRUE,
##     rotate = "oblimin")
## Standardized loadings (pattern matrix) based upon correlation matrix
##                        TC1   TC2   h2   u2 com
## fixed.acidity         0.88 -0.17 0.78 0.22 1.1
## volatile.acidity     -0.50 -0.48 0.52 0.48 2.0
## citric.acid           0.85  0.20 0.78 0.22 1.1
## free.sulfur.dioxide -0.28  0.72 0.57 0.43 1.3
## pH                   -0.81  0.11 0.66 0.34 1.0
## sulphates             0.32  0.58 0.46 0.54 1.6
##
##                        TC1  TC2
## SS loadings           2.60 1.17
```

```
## Proportion Var        0.43 0.19
## Cumulative Var        0.43 0.63
## Proportion Explained  0.69 0.31
## Cumulative Proportion 0.69 1.00
##
##  With component correlations of
##      TC1  TC2
## TC1 1.00 0.07
## TC2 0.07 1.00
##
## Mean item complexity =  1.3
## Test of the hypothesis that 2 components are sufficient.
##
## The root mean square of the residuals (RMSR) is  0.14
##  with the empirical chi square  968.38  with prob <  2.5e-208
##
## Fit based upon off diagonal values = 0.84
```

```
#Residuals
corMatrix<-cor(scaled_data_pca)
residuals<-factor.residuals(corMatrix, pca_resid$loadings)
hist(residuals)
```



> The residuals appear to be mostly normally distributed with a slight right skew.

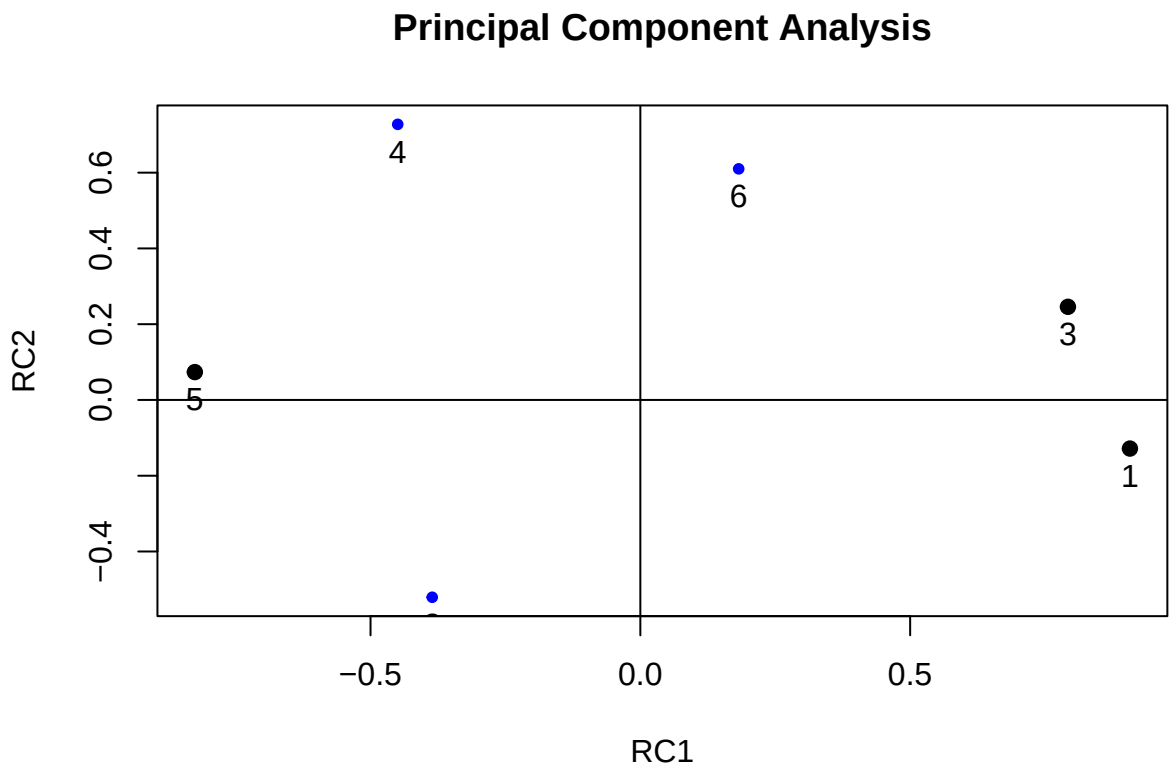
```r
# Check loadings
loadings <- round(pca_resid$loadings[,1:2], 6)
# For interpretation, set less than 0.30 to ""
loadings[loadings < 0.30] <- ""
# Print loadings
View(as.data.frame(loadings))
```

Looking at this, it appears that two groups have formed one with fixed acidity, sulphates, and citric acid. I will name this group "Acidity". The other group is the free sulphur dioxide, and and suplphates. I will name this group "Sulphur".

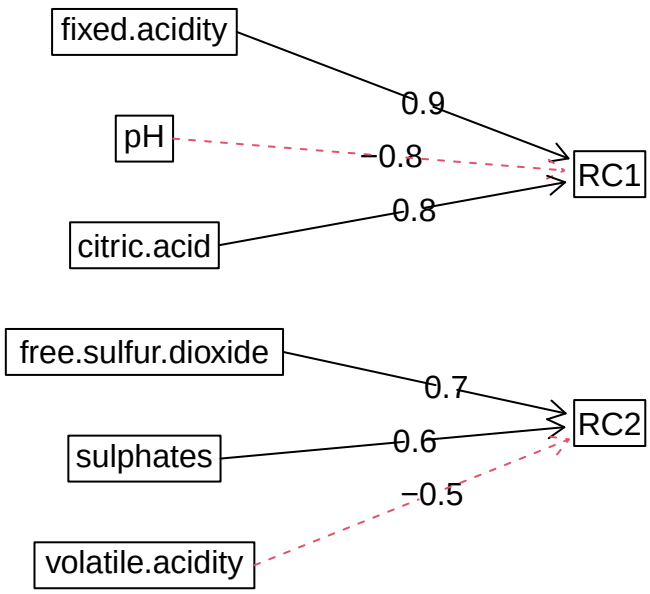
```r
# Informed PCA
pca_final <- principal(scaled_data_pca, nfactors = 2, rotate = "promax")
pca_final
```

```
## Principal Components Analysis
## Call: principal(r = scaled_data_pca, nfactors = 2, rotate = "promax")
## Standardized loadings (pattern matrix) based upon correlation matrix
##                        RC1   RC2   h2   u2 com
## fixed.acidity         0.91 -0.13 0.78 0.22 1.0
## volatile.acidity     -0.39 -0.52 0.52 0.48 1.8
## citric.acid           0.79  0.25 0.78 0.22 1.2
## free.sulfur.dioxide -0.45  0.73 0.57 0.43 1.7
## pH                  -0.83  0.07 0.66 0.34 1.0
## sulphates             0.18  0.61 0.46 0.54 1.2
##
##                          RC1  RC2
## SS loadings             2.52 1.26
## Proportion Var          0.42 0.21
## Cumulative Var          0.42 0.63
## Proportion Explained    0.67 0.33
## Cumulative Proportion 0.67 1.00
##
##  With component correlations of
##      RC1  RC2
## RC1 1.00 0.24
## RC2 0.24 1.00
##
## Mean item complexity =  1.3
## Test of the hypothesis that 2 components are sufficient.
##
## The root mean square of the residuals (RMSR) is  0.14
##  with the empirical chi square  968.38  with prob <  2.5e-208
##
## Fit based upon off diagonal values = 0.84
```

```r
plot(pca_final)
```




```
fa.diagram(pca_final) #seeing which sentiments make up each dimensions
```



> The two groups remain "Acidity" and "Sulphur".

Modeling the Data

```
#adding the outcome variable back into the dataset
regressionData <- bind_cols(pca_final$scores, dataSelection$qualityBinary) %>%
  rename(qualityBinary = ...3)
```

```
## New names:
## * `` -> `...3`
```

```
cor(regressionData)
```

```
##                       RC1       RC2 qualityBinary
## RC1           1.0000000 0.2445371     0.1982207
## RC2           0.2445371 1.0000000     0.1937249
## qualityBinary 0.1982207 0.1937249     1.0000000
```

None of the variables are heavily correlated so we cannot use them in the regression

```
logm <- glm(
  formula = qualityBinary ~ .,
  data = regressionData,
```

```
  family = "binomial" # logistic
)
# Print summary
summary(logm)
```

```
##
## Call:
## glm(formula = qualityBinary ~ ., family = "binomial", data = regressionData)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.9504  -0.5788  -0.4311  -0.3297   2.6106
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept) -2.04273    0.08482 -24.083  < 2e-16 ***
## RC1          0.48192    0.07559   6.375 1.83e-10 ***
## RC2          0.45963    0.07533   6.101 1.05e-09 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1269.9  on 1598  degrees of freedom
## Residual deviance: 1171.4  on 1596  degrees of freedom
## AIC: 1177.4
##
## Number of Fisher Scoring iterations: 5
```

```
car::vif(logm)
```

```
##      RC1      RC2
## 1.015446 1.015446
```

All predictors are significant so we do not remove anything. Since the VIF values low, there is not much multicollinearity.

```
coef(logm)
```

```
## (Intercept)         RC1         RC2
##  -2.0427298   0.4819213   0.4596335
```

```
oddsRatio <- exp(coef(logm))
oddsRatio
```

```
## (Intercept)         RC1         RC2
##   0.1296742   1.6191824   1.5834936
```

For each one-unit increase in RC1 (Acidity), the odds of the outcome (qualityBinary) increase by approximately 1.62 (or 1.58) times, assuming all other variables are constant. For each one-unit increase in RC2 (Sulphur), the odds of the outcome increase by approximately 1.58 times, holding other variables constant.

```r
# Obtain probabilities
probs <- predict(
  logm,
  type = "response"
# needed for probabilities
)

# Obtain classes
classes <- factor(
ifelse(
  probs > 0.50,
  0, # if TRUE -> high quality wine
  1 # if FALSE -> lower quality wine
  )
)
levels(classes)
```

```
## [1] "0" "1"
```

```r
regressionData$qualityBinary <- factor(regressionData$qualityBinary)
levels(regressionData$qualityBinary)
```

```
## [1] "0" "1"
```

```r
trainSet <- regressionData%>% #creating a training set with 70% of the data
 sample_frac(0.7)

testSet <- anti_join(regressionData, trainSet)
```

```
## Joining with `by = join_by(RC1, RC2, qualityBinary)`
```

```r
set.seed(135) #setting the seed for replicability
trainControl <- trainControl(method = "cv", number = 10) #10 time cross training for the training set
lmtenStep <- train(qualityBinary~., data=trainSet, method="glm", family=binomial(), trControl=trainCont
summary(lmtenStep)
```

```
##
## Call:
## NULL
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.9456  -0.5898  -0.4394  -0.3349   2.6004
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept) -1.99069    0.09959 -19.989  < 2e-16 ***
## RC1          0.51400    0.08858   5.803 6.53e-09 ***
## RC2          0.42988    0.08837   4.865 1.15e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 918.35  on 1118  degrees of freedom
## Residual deviance: 843.82  on 1116  degrees of freedom
## AIC: 849.82
##
## Number of Fisher Scoring iterations: 5
```

```
finalModel <- glm(qualityBinary ~ RC1 + RC2, data = trainSet, family = binomial) #making a new logistic
summary(finalModel)
```

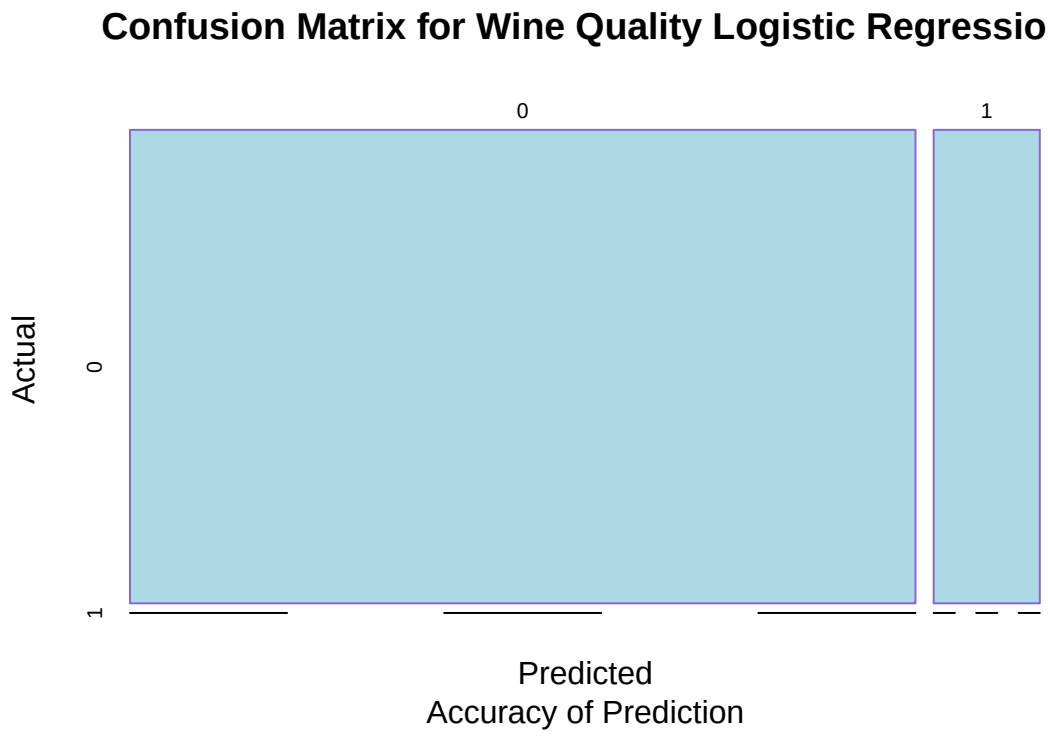
```
##
## Call:
## glm(formula = qualityBinary ~ RC1 + RC2, family = binomial, data = trainSet)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.9456  -0.5898  -0.4394  -0.3349   2.6004
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept) -1.99069    0.09959 -19.989  < 2e-16 ***
## RC1          0.51400    0.08858   5.803 6.53e-09 ***
## RC2          0.42988    0.08837   4.865 1.15e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 918.35  on 1118  degrees of freedom
## Residual deviance: 843.82  on 1116  degrees of freedom
## AIC: 849.82
##
## Number of Fisher Scoring iterations: 5
```

```
finalModelStat <- lrm(qualityBinary ~ RC1 + RC2, data = trainSet) #done for additional statistics
finalModelStat
```

```
## Logistic Regression Model
##
## lrm(formula = qualityBinary ~ RC1 + RC2, data = trainSet)
##
##                       Model Likelihood      Discrimination    Rank Discrim.
##                             Ratio Test             Indexes          Indexes
## Obs          1119    LR chi2      74.53      R2       0.115    C       0.721
##  0            959    d.f.             2     R2(2,1119)0.063    Dxy     0.442
##  1            160    Pr(> chi2) <0.0001    R2(2,411.4)0.162    gamma   0.443
## max |deriv| 2e-07                           Brier    0.115    tau-a   0.109
##
##           Coef    S.E.   Wald Z Pr(>|Z|)
## Intercept -1.9907 0.0996 -19.99 <0.0001
## RC1        0.5140 0.0886   5.80 <0.0001
## RC2        0.4299 0.0884   4.86 <0.0001
```

```r
predicted <- predict(finalModel, testSet)

# Create ActualPredicted dataframe including qualityBinary, predicted, and aboveMedian
ActualPredicted <- testSet %>%
  dplyr::select(qualityBinary) %>%
  mutate(predicted = as.factor(ifelse(predicted >= 0.5, 1, 0)),  # Using 0.5 as threshold for binary cl
         aboveMedian = as.factor(ifelse(qualityBinary == "1", "1", "0")))  # Assuming "1" is your posit

# Ensure predicted column has the same levels as qualityBinary column
levels(ActualPredicted$predicted) <- levels(ActualPredicted$qualityBinary)

# Create confusion matrix
conf_matrix <- confusionMatrix(ActualPredicted$predicted, ActualPredicted$qualityBinary,
                               mode = "everything", positive = "1")
```

```r
mosaic <- table(ActualPredicted$aboveMedian, ActualPredicted$predicted) #making a beautiful mosaic plot
mosaicplot(mosaic,
 main = "Confusion Matrix for Wine Quality Logistic Regression",
 sub = "Accuracy of Prediction",
 xlab = "Predicted",
 ylab = "Actual",
 color = "lightblue",
 border = "mediumpurple3")
```



Discussion

For this project, I started out by creating visual representations of the data through box plots and then conducted a PCA to find relevant variables. I also used PCA to conduct dimension reduction. I then conducted a 10-fold cross validation to train a model and then created a confusion matrix.

Overall, the model appears to perform well with a high accuracy rate of 88.68%. Additionally, with a p-value of 1.061e-09, we can reject the null hypothesis and accept the alternative hypothesis that we can use predictor variables attained from physicochemical tests in this dataset be aggregated to be used to accurately predict the quality of red wine. Additionally, specificity of this model is also quite high at 88.92% which means that it has a decent ability at correctly identifying the quality of wine (good or low quality). However, it is important to note that the precision score and F1 score is quite low. Additionally, the Kappa score is negative (-0.0052), indicating poor agreement between predicted and observed classes. Although this model appears to be accurate, it does not perform well in terms of prediction when you compare the values to the actual values in the dataset. The mosaic graph also showcases this.

As mentioned earlier, an ideal version of this model would be incredibly helpful in allowing people to get a better understanding of which physicochemical factors of wine would make a wine good. It can also help wine companies priorities certain physicochemical elements of tehir wine to create a better tasting wine. Certain limitations of this dataset would be that there is no data about grape types, wine brands, wine selling price, etc. This would give us extra information as to why certain wines would be in higher quality compared to others. This could help address some of the variances that we see in the dataset, regardless of physiocochemical properties.

If I were to redo this project in teh future, I would perform ordinal logistic regression rather than binary logistic regression. I think this would be helpful because it would account for the entire quality scale (0-10) of the original outcome variable. This may also help account for more of the variances in the physicochemical factor levels. It is quite hard to firmly declare all wines with a score of 7 or higher as being good and vice versa as a wine with a quality of 10 would have much different properties to a wine with a quality of 7. The same can be said about a wine with a quality score of 6 to a score of 0.