Wine

Nathania

June 2, 2019

Introduction

The white wine dataset contains a physicochemical properties and quality ratings. Each wine sample comes with a quality rating from three to ten, and results from several physical chemical tests, such as: alcohol content, acidity level and residual sugar, etc. There are 11 columns describing their chemical properties, and a column for quality ratings.The objectives to build a predict the quality of the wine.

Description of features:

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, it's rare to find wines with less than 1 gram/liter and wines with greater than 45 grams/liter are considered sweet

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; it prevents microbial growth and the oxidation of wine

7 - total sulfur dioxide: amount of free and bound forms of S02; in low concentrations, SO2 is mostly undetectable in wine, but at free SO2 concentrations over 50 ppm, SO2 becomes evident in the nose and taste of wine

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); most wines are between 3-4 on the pH scale

10 - sulphates: a wine additive which can contribute to sulfur dioxide gas (S02) levels, wich acts as an antimicrobial and antioxidant

11 - alcohol: the percent alcohol content of the wine

Output va (based on sensory data):

12 - quality (score between 3 and 9)

The objectives is to discover which of these chemical propeties influence the quality of wine and to understand how these characteristics influence the quality.At the end, we will create a model to predict the quality of wine.

```
library(ggplot2)
```

```
## Warning: package 'ggplot2' was built under R version 3.5.1
```

```
library(gridExtra)
```

```
## Warning: package 'gridExtra' was built under R version 3.5.1
```

```
library(grid)
library(dplyr)
```

```
## Warning: package 'dplyr' was built under R version 3.5.3
##
## Attaching package: 'dplyr'
## The following object is masked from 'package:gridExtra':
##
##     combine
## The following objects are masked from 'package:stats':
##
##     filter, lag
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
library(reshape2)
library(GGally)
## Warning: package 'GGally' was built under R version 3.5.1
##
## Attaching package: 'GGally'
## The following object is masked from 'package:dplyr':
##
##     nasa
library(scales)
## Warning: package 'scales' was built under R version 3.5.1
library(ggpubr)
## Warning: package 'ggpubr' was built under R version 3.5.1
## Loading required package: magrittr
library(memisc)
## Warning: package 'memisc' was built under R version 3.5.3
## Loading required package: lattice
## Loading required package: MASS
##
## Attaching package: 'MASS'
## The following object is masked from 'package:dplyr':
##
##     select
##
## Attaching package: 'memisc'
## The following object is masked from 'package:scales':
##
##     percent
## The following objects are masked from 'package:dplyr':
##
##     collect, recode, rename, syms
```

```
## The following object is masked from 'package:ggplot2':
##
##     syms
## The following objects are masked from 'package:stats':
##
##     contr.sum, contr.treatment, contrasts
## The following object is masked from 'package:base':
##
##     as.array
```

```
library(caret)
```

```
## Warning: package 'caret' was built under R version 3.5.1
```

```
library(tidyverse)
```

```
## Warning: package 'tidyverse' was built under R version 3.5.1
## -- Attaching packages ---------------------------------------------------------- tidyverse 1.2.1 --
## v tibble  2.1.1     v purrr   0.2.5
## v tidyr   0.8.1     v stringr 1.3.1
## v readr   1.1.1     v forcats 0.3.0
## Warning: package 'tibble' was built under R version 3.5.3
## Warning: package 'tidyr' was built under R version 3.5.1
## Warning: package 'stringr' was built under R version 3.5.1
## -- Conflicts ------------------------------------------------------------ tidyverse_conflicts() --
## x purrr::%@%()        masks memisc::%@%()
## x readr::col_factor() masks scales::col_factor()
## x memisc::collect()   masks dplyr::collect()
## x dplyr::combine()    masks gridExtra::combine()
## x purrr::discard()    masks scales::discard()
## x tidyr::extract()    masks magrittr::extract()
## x dplyr::filter()     masks stats::filter()
## x dplyr::lag()        masks stats::lag()
## x purrr::lift()       masks caret::lift()
## x memisc::recode()    masks dplyr::recode()
## x memisc::rename()    masks dplyr::rename()
## x MASS::select()      masks dplyr::select()
## x purrr::set_names()  masks magrittr::set_names()
## x memisc::syms()      masks dplyr::syms(), ggplot2::syms()
```

```
fileName <- 'winequality-white.csv';
if (!file.exists(fileName)) {
  download.file(paste0('https://archive.ics.uci.edu/ml/machine-learning-databases/wine-quality/', fileNa
}
```

Take a look of the data

```
df <- read.csv('winequality-white.csv', sep=';')
glimpse(df)
```

```
## Observations: 4,898
## Variables: 12
## $ fixed.acidity        <dbl> 7.0, 6.3, 8.1, 7.2, 7.2, 8.1, 6.2, 7.0, 6...
```

```
## $ volatile.acidity     <dbl> 0.27, 0.30, 0.28, 0.23, 0.23, 0.28, 0.32,...
## $ citric.acid          <dbl> 0.36, 0.34, 0.40, 0.32, 0.32, 0.40, 0.16,...
## $ residual.sugar       <dbl> 20.70, 1.60, 6.90, 8.50, 8.50, 6.90, 7.00...
## $ chlorides            <dbl> 0.045, 0.049, 0.050, 0.058, 0.058, 0.050,...
## $ free.sulfur.dioxide  <dbl> 45, 14, 30, 47, 47, 30, 30, 45, 14, 28, 1...
## $ total.sulfur.dioxide <dbl> 170, 132, 97, 186, 186, 97, 136, 170, 132...
## $ density              <dbl> 1.0010, 0.9940, 0.9951, 0.9956, 0.9956, 0...
## $ pH                   <dbl> 3.00, 3.30, 3.26, 3.19, 3.19, 3.26, 3.18,...
## $ sulphates            <dbl> 0.45, 0.49, 0.44, 0.40, 0.40, 0.44, 0.47,...
## $ alcohol              <dbl> 8.8, 9.5, 10.1, 9.9, 9.9, 10.1, 9.6, 8.8,...
## $ quality              <int> 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 5, 5, 5, 7,...
```

Data Cleaning

We'll check missing values,as well as the duplicated values.

```
#Check missing values
sum(is.na.data.frame(df))
```

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

```
#Check duplicated values
sum(duplicated(df))
```

```
## [1] 937
```

Now we have no duplicated values

```
#filter any duplicated rows
df <- df %>% distinct()
#confirm there are no duplicated rows
sum(duplicated(df))
```

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

Let's summarize the data, notice here the quality of wine lies between 3 and 9

```
summary(df)
```

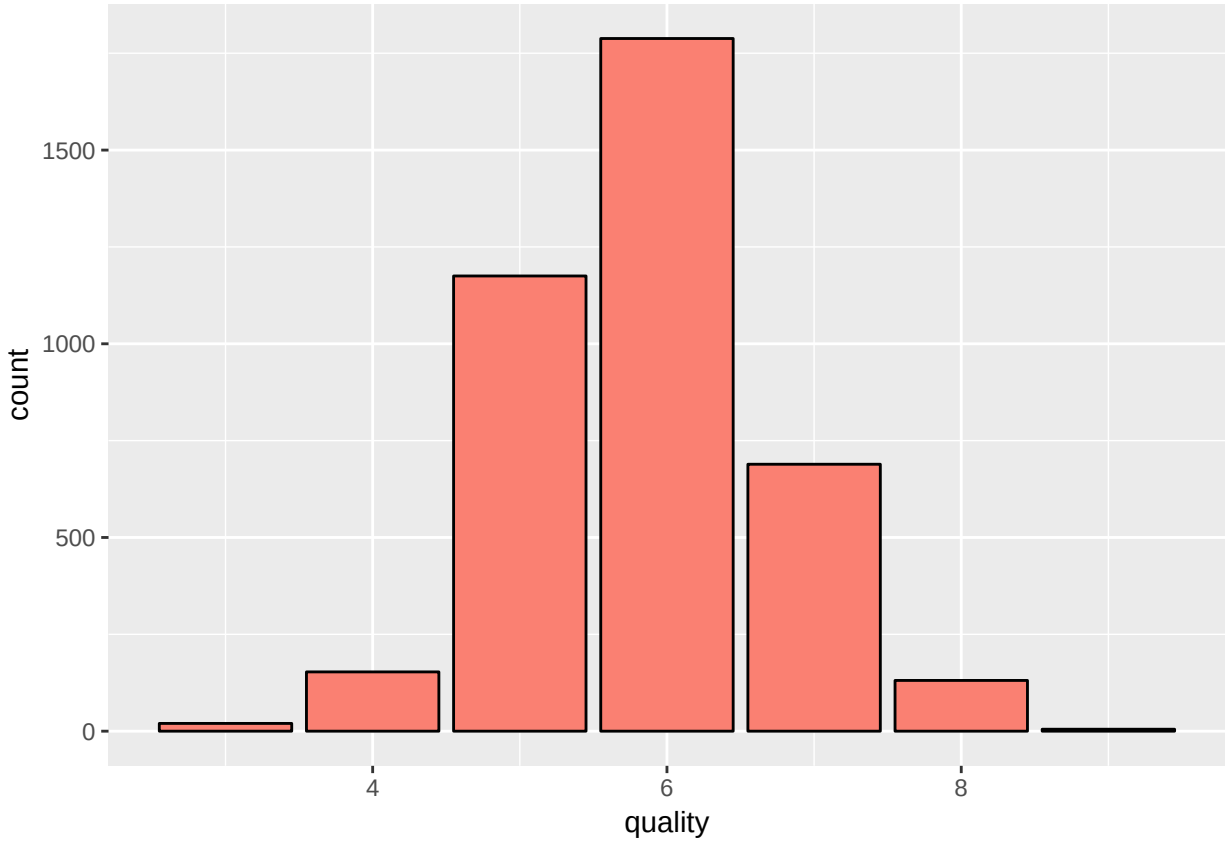
```
##  fixed.acidity    volatile.acidity  citric.acid     residual.sugar
##  Min.   : 3.800   Min.   :0.0800   Min.   :0.0000   Min.   : 0.600
##  1st Qu.: 6.300   1st Qu.:0.2100   1st Qu.:0.2700   1st Qu.: 1.600
##  Median : 6.800   Median :0.2600   Median :0.3200   Median : 4.700
##  Mean   : 6.839   Mean   :0.2805   Mean   :0.3343   Mean   : 5.915
##  3rd Qu.: 7.300   3rd Qu.:0.3300   3rd Qu.:0.3900   3rd Qu.: 8.900
##  Max.   :14.200   Max.   :1.1000   Max.   :1.6600   Max.   :65.800
##    chlorides       free.sulfur.dioxide total.sulfur.dioxide
##  Min.   :0.00900   Min.   :  2.00      Min.   :  9.0
##  1st Qu.:0.03500   1st Qu.: 23.00      1st Qu.:106.0
##  Median :0.04200   Median : 33.00      Median :133.0
##  Mean   :0.04591   Mean   : 34.89      Mean   :137.2
##  3rd Qu.:0.05000   3rd Qu.: 45.00      3rd Qu.:166.0
##  Max.   :0.34600   Max.   :289.00      Max.   :440.0
##     density             pH          sulphates         alcohol
##  Min.   :0.9871   Min.   :2.720   Min.   :0.2200   Min.   : 8.00
##  1st Qu.:0.9916   1st Qu.:3.090   1st Qu.:0.4100   1st Qu.: 9.50
##  Median :0.9935   Median :3.180   Median :0.4800   Median :10.40
##  Mean   :0.9938   Mean   :3.195   Mean   :0.4904   Mean   :10.59
##  3rd Qu.:0.9957   3rd Qu.:3.290   3rd Qu.:0.5500   3rd Qu.:11.40
```

```
##  Max.   :1.0390   Max.   :3.820   Max.   :1.0800   Max.   :14.20
##     quality
##  Min.   :3.000
##  1st Qu.:5.000
##  Median :6.000
##  Mean   :5.855
##  3rd Qu.:6.000
##  Max.   :9.000
```

Data Exploratory

The target va is Quality therefore it's important to know it's distribution, let's look at the distribution of the wine's quality.

```
ggplot(aes(quality), data = df) +
  geom_bar(color = "black", fill = "salmon")
```




```
table(df$quality)
```

```
##
##    3    4    5    6    7    8    9
##   20  153 1175 1788  689  131    5
```

Most wine samples are of 5 and 6 (almost 80% of the dataset). Moreover, it seems to be that wines which received the highest score (9) have a few observations, and this situation repeats in the lowest level(3, 4). Wines with a score of 7 are 855 observations. Then, I will compare the median and mean of physicochemical properties for the 3, 5, 6, 9 quality levels to understand the main differences among them.

```r
#exclude (droping) index of the dataset
col <- names(df) %in% "X"
winequality <- df[!col]

# Create a function to compare the mean and median between highest, average and lowest
# scores of all vas
mean_median <- function(func.) {
# this va creates a new data frame by selecting only rows with quality equal 3
q3 <- data.frame(summarize_all(subset(winequality, quality == 3),
                                    .funs= func.))

# this va will creates a new data frame by selecting only rows with quality equal
# 5 or 6
q56 <- data.frame(summarize_all(subset(winequality, quality == 5 |
                                    quality == 6), .funs= func.))

# this va will creates a new data frame by selecting only rows with quality equal 9
q9 <- data.frame(summarize_all(subset(winequality, quality == 9),
                                    .funs= func.))

# this va joins our three new data frames (q3, q56, q9) vertically.
comb <- rbind(q3, q56, q9)

# convert quality colunm in integer number
comb$quality = as.integer(comb$quality)

# reorder the columns in our new dataframe
comb [, c(12, 1:11) ]
}

head(df)
```

```
##   fixed.acidity volatile.acidity citric.acid residual.sugar chlorides
## 1           7.0             0.27        0.36           20.7     0.045
## 2           6.3             0.30        0.34            1.6     0.049
## 3           8.1             0.28        0.40            6.9     0.050
## 4           7.2             0.23        0.32            8.5     0.058
## 5           6.2             0.32        0.16            7.0     0.045
## 6           8.1             0.22        0.43            1.5     0.044
##   free.sulfur.dioxide total.sulfur.dioxide density   pH sulphates alcohol
## 1                  45                  170  1.0010 3.00      0.45     8.8
## 2                  14                  132  0.9940 3.30      0.49     9.5
## 3                  30                   97  0.9951 3.26      0.44    10.1
## 4                  47                  186  0.9956 3.19      0.40     9.9
## 5                  30                  136  0.9949 3.18      0.47     9.6
## 6                  28                  129  0.9938 3.22      0.45    11.0
##   quality
## 1       6
## 2       6
## 3       6
## 4       6
## 5       6
## 6       6
```

```
#Check the mean of all vas based on highest, average and lowest score.
mean_median(mean)
```

```
##   quality fixed.acidity volatile.acidity citric.acid residual.sugar
## 1       3      7.600000        0.3332500    0.336000       6.392500
## 2       5      6.860007        0.2781573    0.337327       6.349106
## 3       9      7.420000        0.2980000    0.386000       4.120000
##    chlorides free.sulfur.dioxide total.sulfur.dioxide   density       pH
## 1 0.05430000             53.3250             170.6000 0.9948840 3.187500
## 2 0.04801114             35.5837             141.8694 0.9942577 3.186412
## 3 0.02740000             33.4000             116.0000 0.9914600 3.308000
##   sulphates  alcohol
## 1 0.4745000 10.34500
## 2 0.4884948 10.33604
## 3 0.4660000 12.18000
```

There are significant variations of mean and median with sulfur dioxide (free and total sulfur) and acidity (fixed, volatile and citric) vas and residual sugar. In order to avoid problems with outliers, we will consider only the median.

Notice that wine with highest score, i.e. 9 have the lowest level of density, sulfur dioxide and sugar (the lowest score has the same median). Furthermore, they have highest level of alcohol, citric acid, and pH.

Summarizing above, we can see that: - What attributes increase values with a better rating? Alcohol, citric acid, pH - What attributes decrease values with a better rating? density, sulfur dioxide and sugar

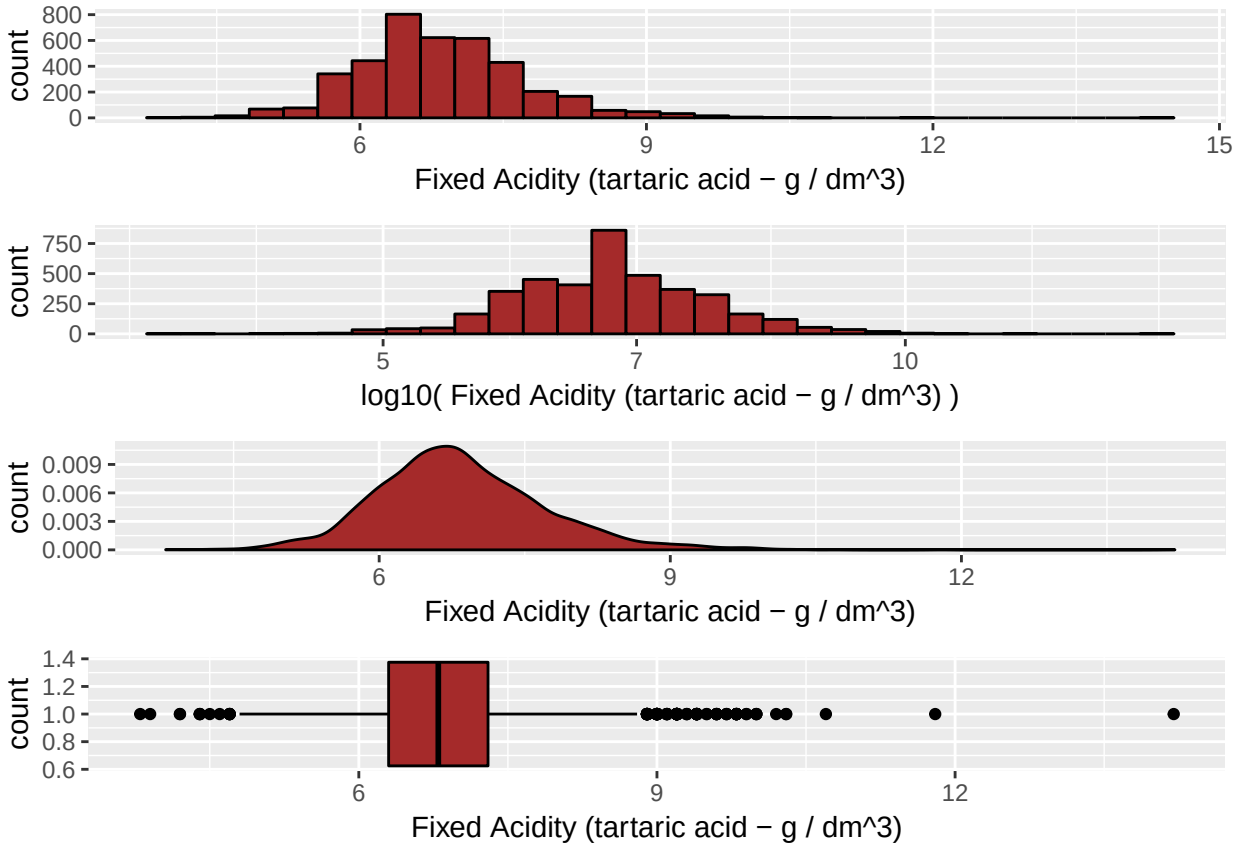
```
# The objective of this function is to create charts to analyze the distribution and outliers of featur

# wine distribution
wdist <- function(va, varName = '', bins = 30) {

#Print charts with outliers
## Building a Histogram:
his <- ggplot(data = df) +
    geom_histogram(aes(x = va), bins = bins,
                   fill = 'brown', colour='black') +
    labs(x = varName)

## histogram with scale log10
hislog10 <- his + scale_x_log10() +
    labs(x = paste('log10(', varName,')'))

## Building a boxplot:
bxplot <- ggplot(df, aes(x = 1, y = va)) +
    geom_boxplot(color = 'black', fill = 'brown') +
    labs(x ='count', y = varName) +
  coord_flip()

## Building density plot
dsplot <- ggplot(aes(x = va,
                         y = ..count../sum(..count..)), data = df ) +
geom_density(fill = 'brown', binwidth = 10) +
scale_x_continuous() +
 labs(x = varName, y = 'count')

## Arranging all the plots:
```

```
ggarrange(his, hislog10, dsplot, bxplot,  nrow = 4)

}
```

Fixed Acidity

```
wdist(df$fixed.acidity,
        varName = 'Fixed Acidity (tartaric acid - g / dm^3)')
```

```
## Warning: Ignoring unknown parameters: binwidth
```

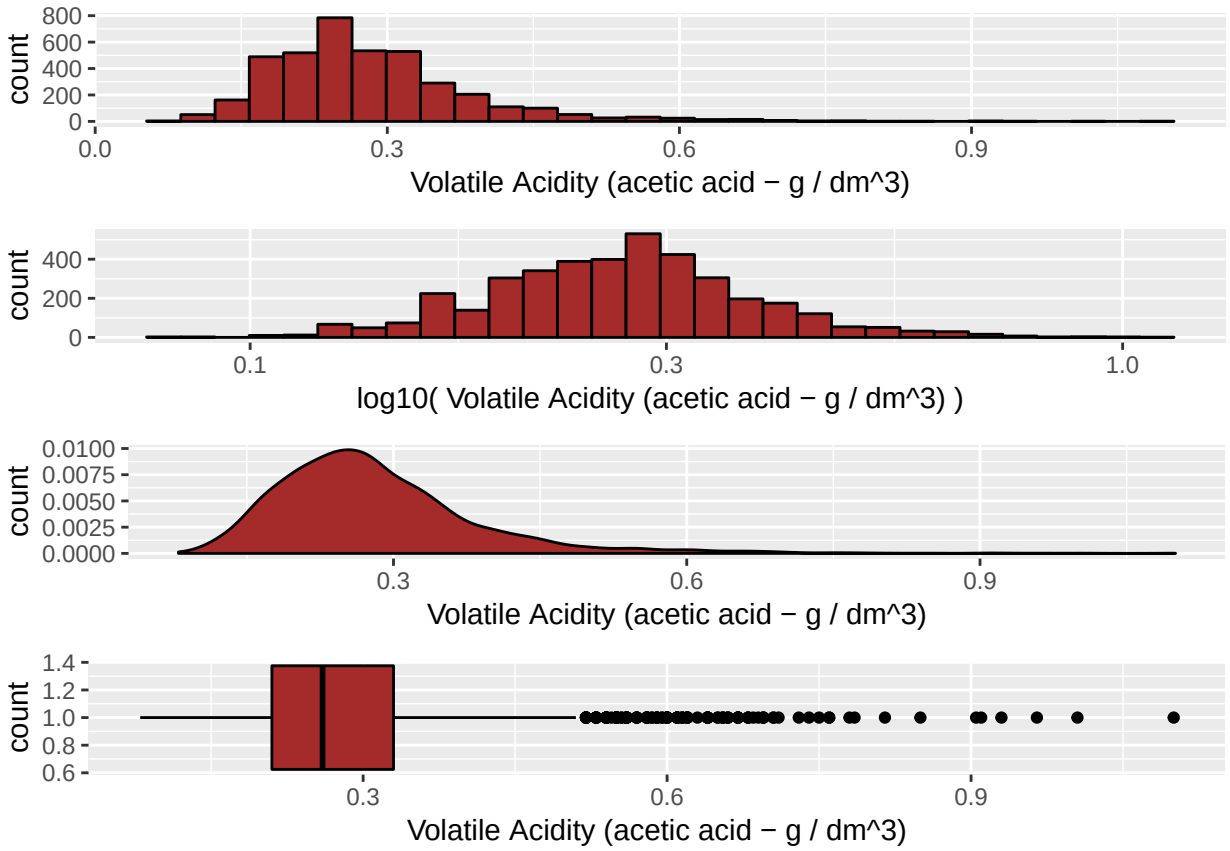


There are two things to notice: Distribution: Right-skewed, there is a long tail. Outliers: The boxplot shows a few outliers from 8 to 16.

Volatile Acidity

```
wdist(df$volatile.acidity,
        varName = 'Volatile Acidity (acetic acid - g / dm^3)')
```

```
## Warning: Ignoring unknown parameters: binwidth
```



Distribution: The first histogram appears bimodal with peaks around 0.27, but when we zoom into the histogram with log10, it seems there's right-skewed distribution. Further, there is a long tail. Outliers: There are a few outliers between the higher range, around 0.5 to 12.0

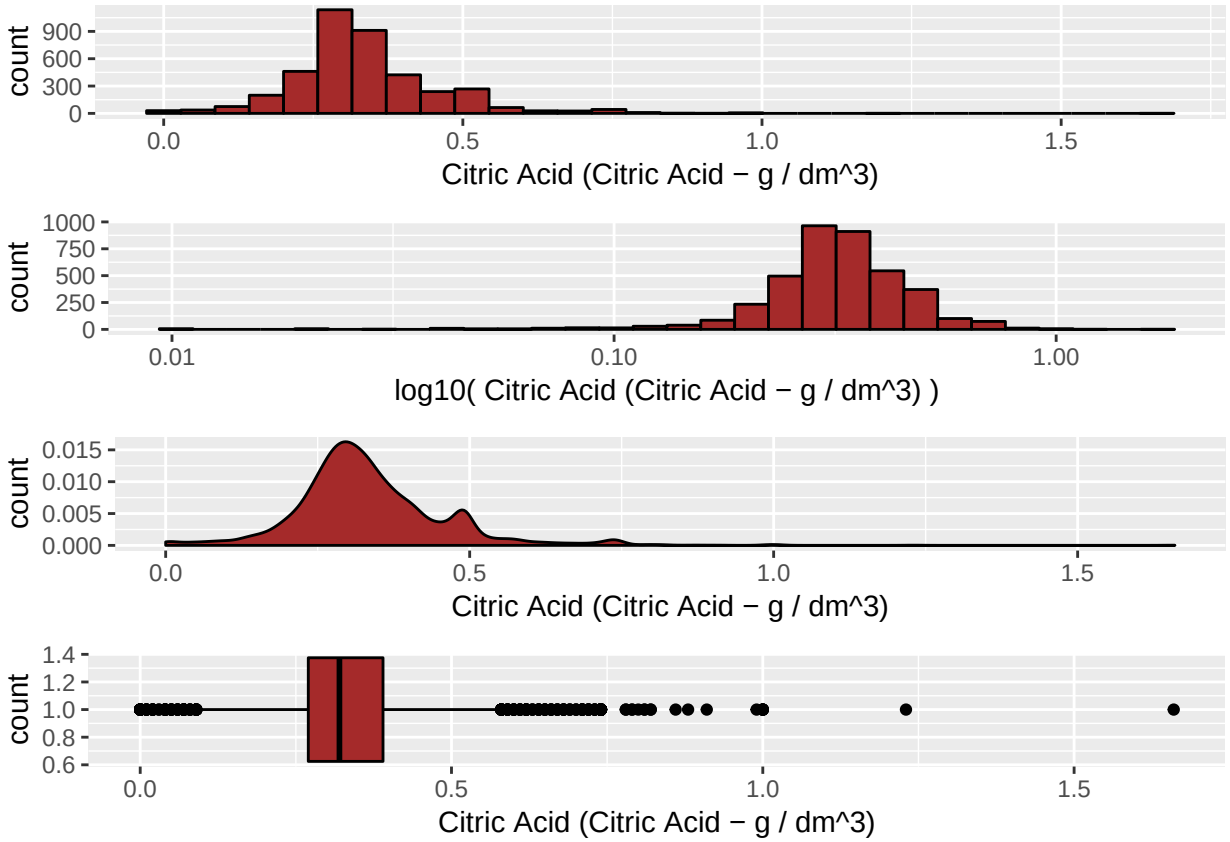
Citric Acid

```
wdist(df$citric.acid ,
        varName = 'Citric Acid (Citric Acid - g / dm^3)')
```

```
## Warning: Ignoring unknown parameters: binwidth
```

```
## Warning: Transformation introduced infinite values in continuous x-axis
```

```
## Warning: Removed 18 rows containing non-finite values (stat_bin).
```



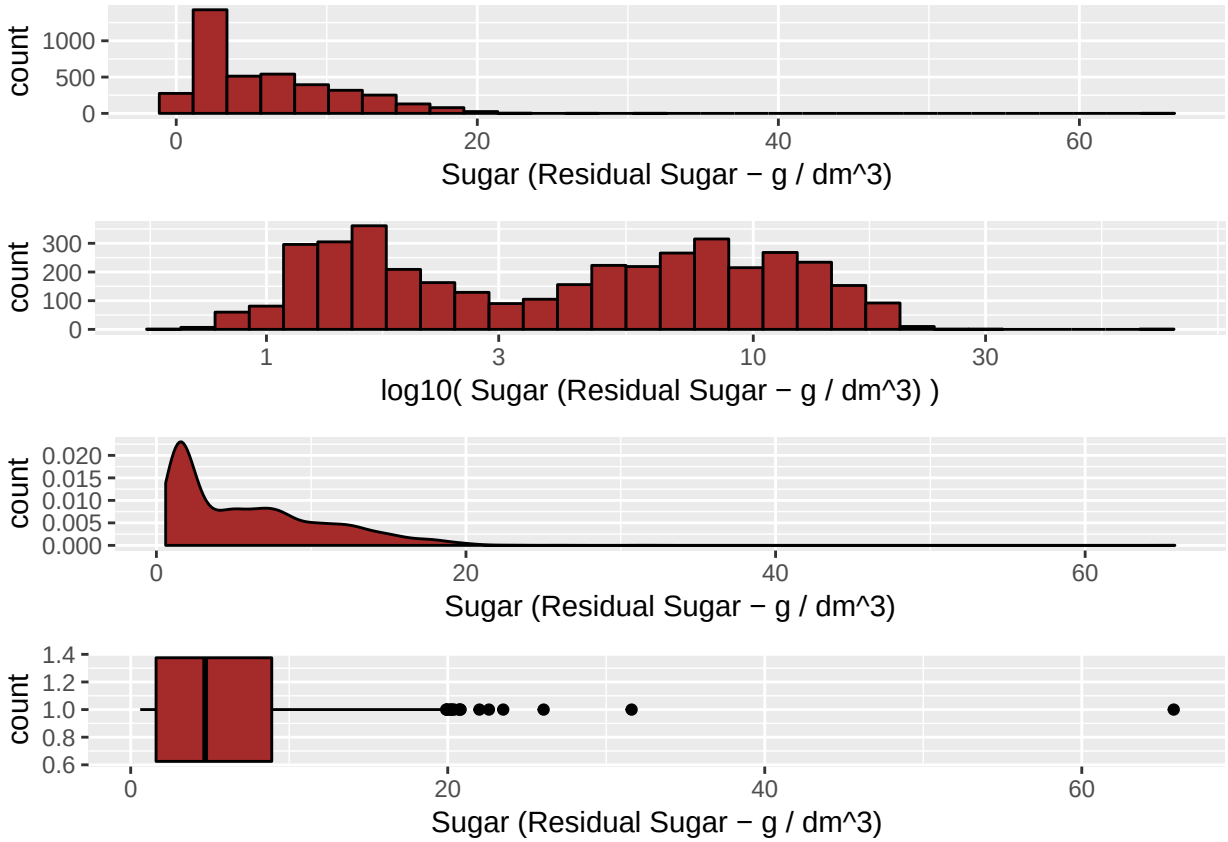
Distribution: The first histogram appears right-skewed with high peak around 0.30 and with a short tail. When we use the histogram with log10, it seems to be the distribution changes the direction to the left and creates a long tail.

Outliers: There is There are a few outliers between the higher range, around 0.6 to 1.7

Sugar

```
wdist(df$residual.sugar ,
        varName = 'Sugar (Residual Sugar - g / dm^3)')
```

```
## Warning: Ignoring unknown parameters: binwidth
```



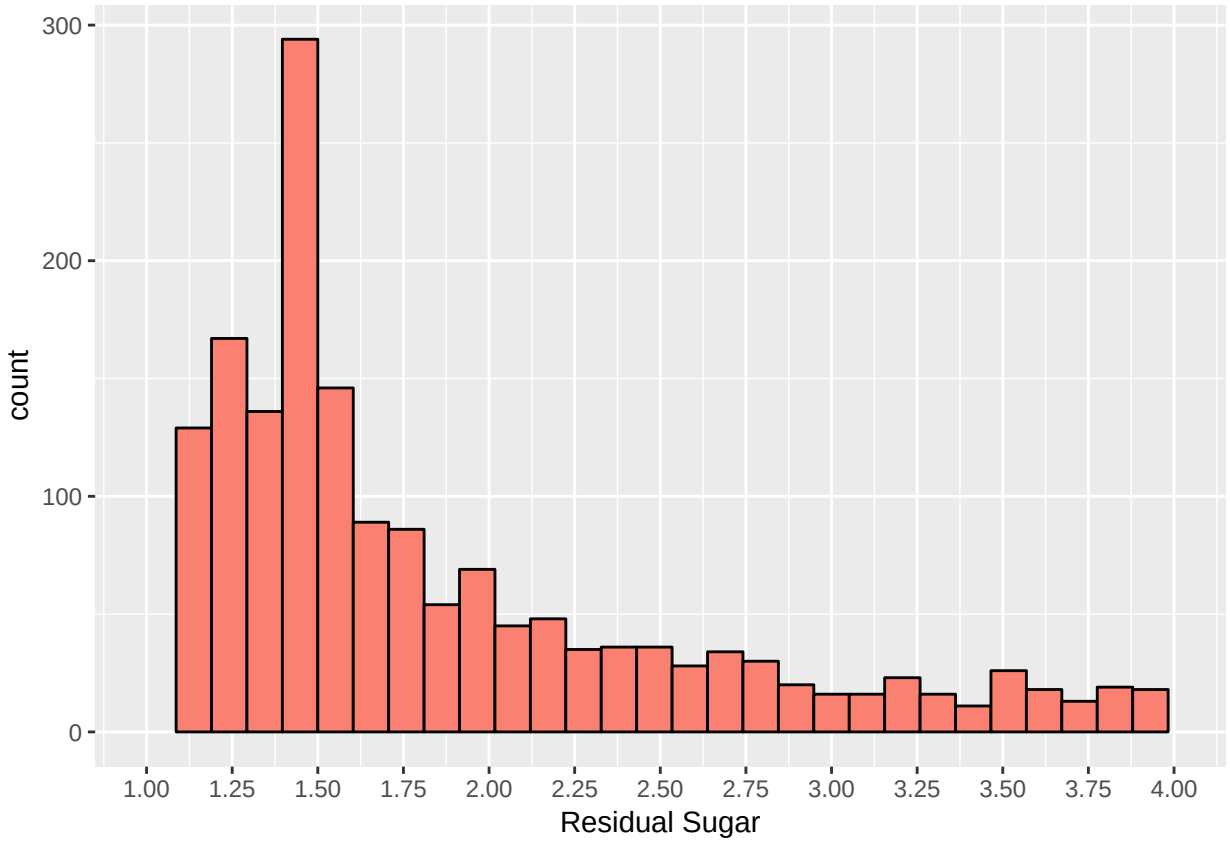
It's hard to clearly see the distribution of sugar because of the long tail, we will create new a histogram with breaks and limits.

```
ggplot(data = df) +
    geom_histogram(aes(x = residual.sugar),
                   fill = 'salmon', colour='black') +
  scale_x_continuous(breaks = seq(1, 4, 0.25),
                     limits = c(1, 4)) +
    labs(x = "Residual Sugar")
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

```
## Warning: Removed 2207 rows containing non-finite values (stat_bin).
```

```
## Warning: Removed 2 rows containing missing values (geom_bar).
```



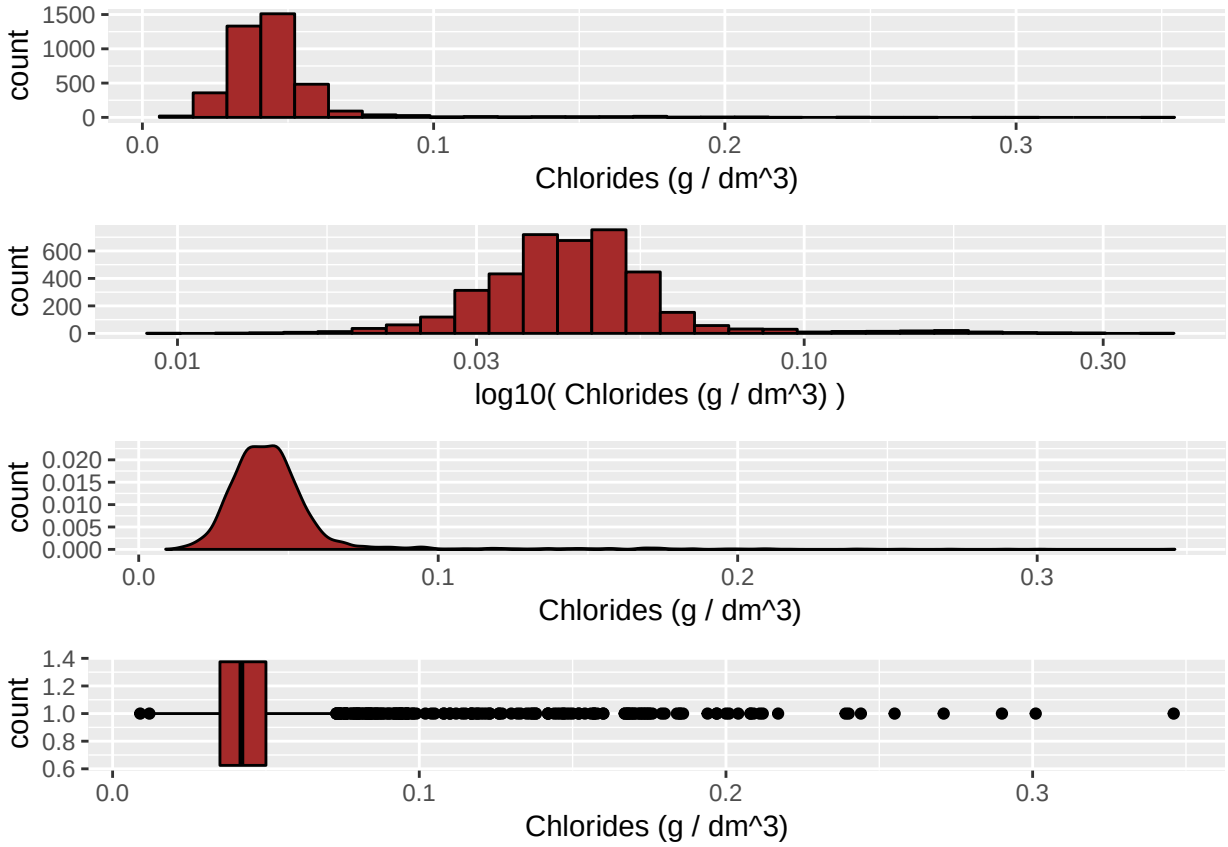
Distribution: It has a right-skewed distribution with a long tail (from 20 to 65). The distribution with log10 appears right-skewed with a long tail to the right.

Outliers: There are many outliers with a large range (from 20 to 65).

Chlorides

```
wdist(df$chlorides,
        varName = 'Chlorides (g / dm^3)')
```

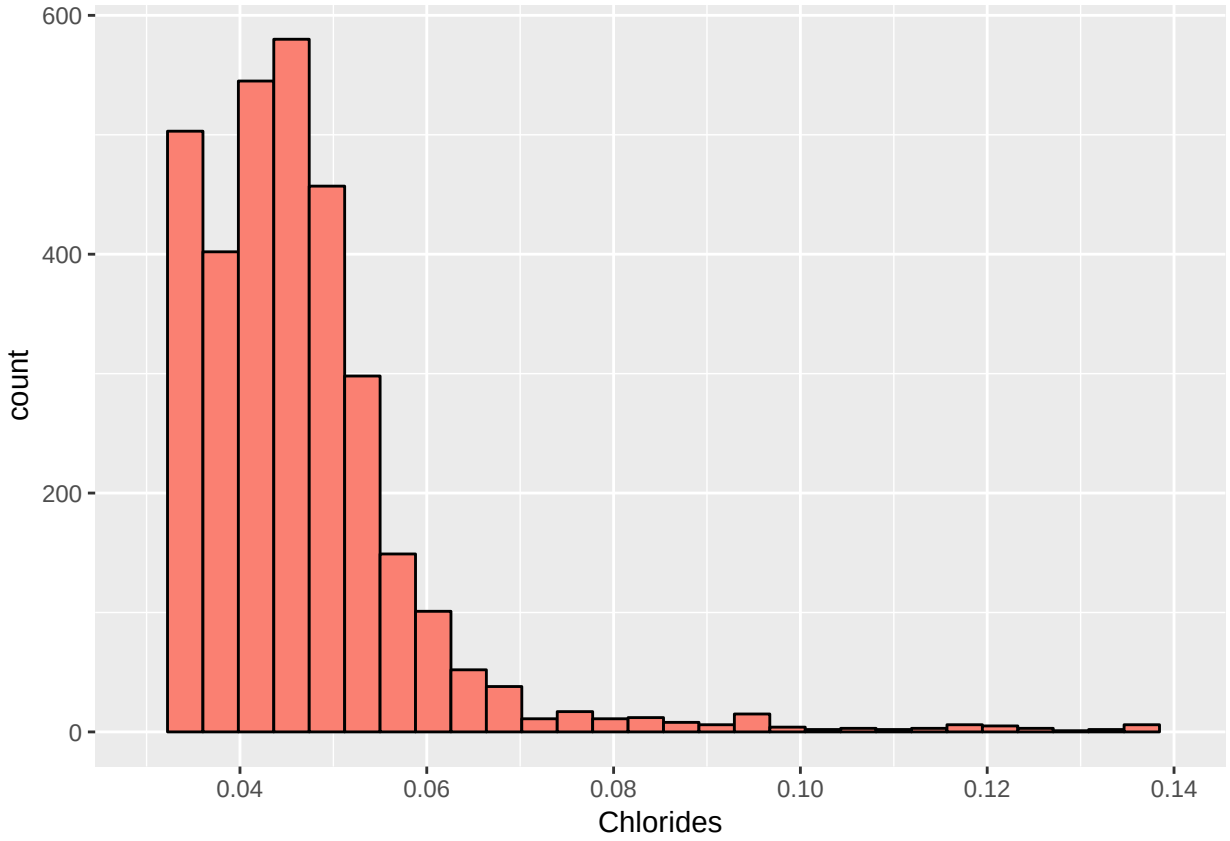
```
## Warning: Ignoring unknown parameters: binwidth
```



Again, we will create a new histogram with breaks and limits.

```
ggplot(data = df) +
    geom_histogram(aes(x = chlorides),
                   fill = 'salmon', colour='black') +
  scale_x_continuous(breaks = seq(0, 0.15, 0.020),
                     limits = c(0.03, 0.14)) +
    labs(x = "Chlorides")
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## Warning: Removed 444 rows containing non-finite values (stat_bin).
## Warning: Removed 2 rows containing missing values (geom_bar).
```



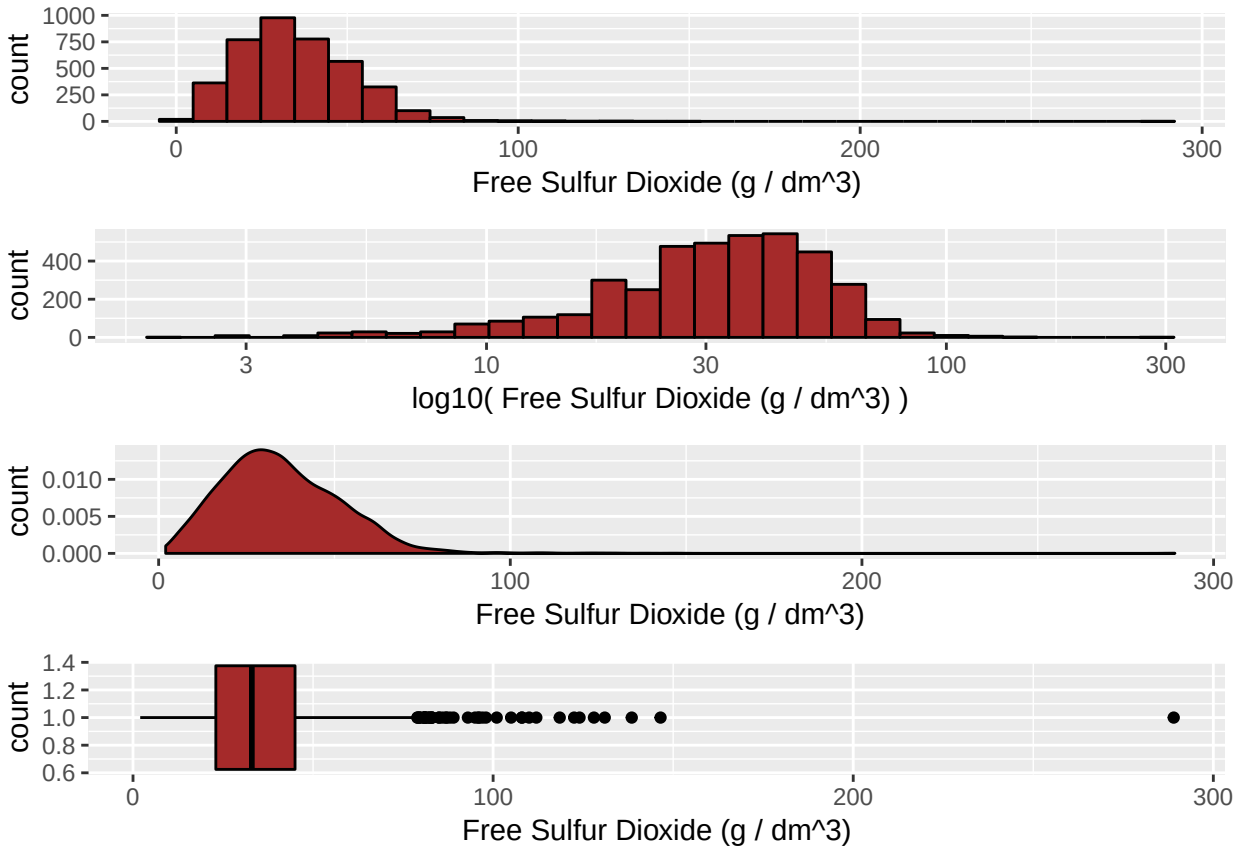
Distribution: It has right-skewerd distribution with a long tail (from 0.06 to 0.35). The distribution with log10 appears symmetrical with a long tail to the right. Aftedfards, adjusting the distribution continues to be symmetrical.

Outliers: There are many upper outliers with an extensitve range (around 0.06 and 0.35) and a few lower outliers (around 0.1 and 0.2).

Free Sulfur Dioxide

```
wdist(df$free.sulfur.dioxide,
       varName = 'Free Sulfur Dioxide (g / dm^3)')
```

```
## Warning: Ignoring unknown parameters: binwidth
```



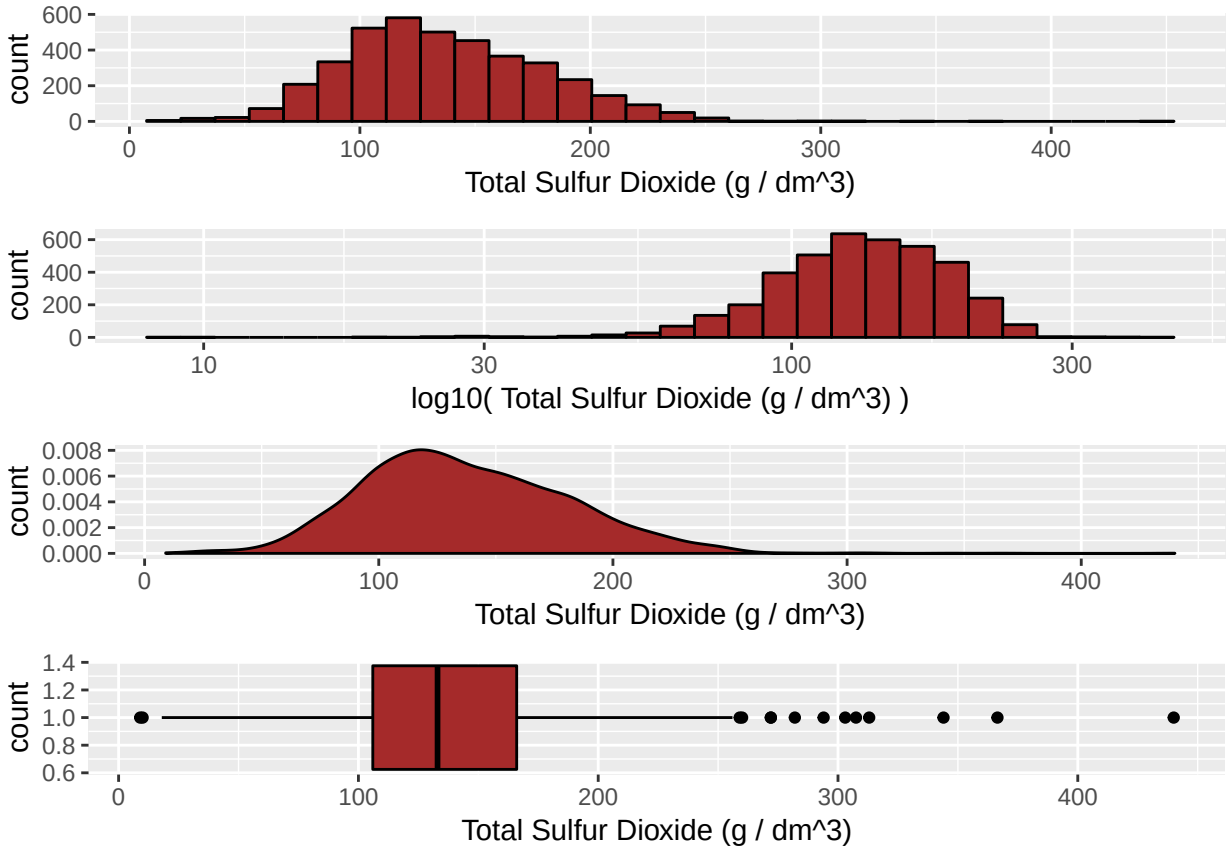
Distribution: The first histogram appears right-skewed, but when we zoom into the histogram with log10, it seems to be bimodal with peaks around 25 and 30. Further, there is also a short tail.

Outliers: There are a few outliers with a range around 80 to 140.

Total Sulfur Dioxide

```
wdist(df$total.sulfur.dioxide,
        varName = 'Total Sulfur Dioxide (g / dm^3)')
```

```
## Warning: Ignoring unknown parameters: binwidth
```



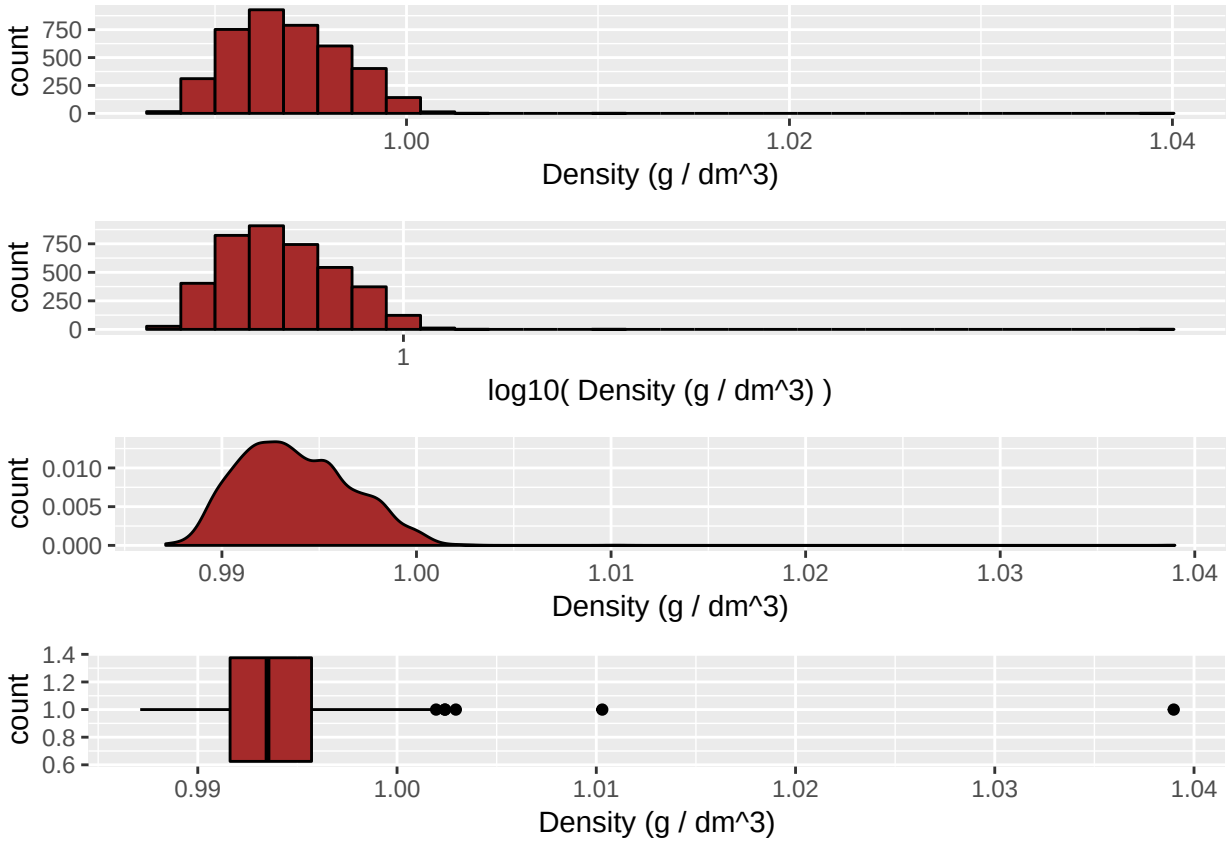
Distribution: The first histogram appears right-skewed, but when we zoom into the histogram with log10, change direction to left-skewed. Furthermore, there is a long tail.

Outliers: There are many outliers with a higher range, around 260 to 430.

Density

```
wdist(df$density,
        varName = 'Density (g / dm^3)')
```

```
## Warning: Ignoring unknown parameters: binwidth
```



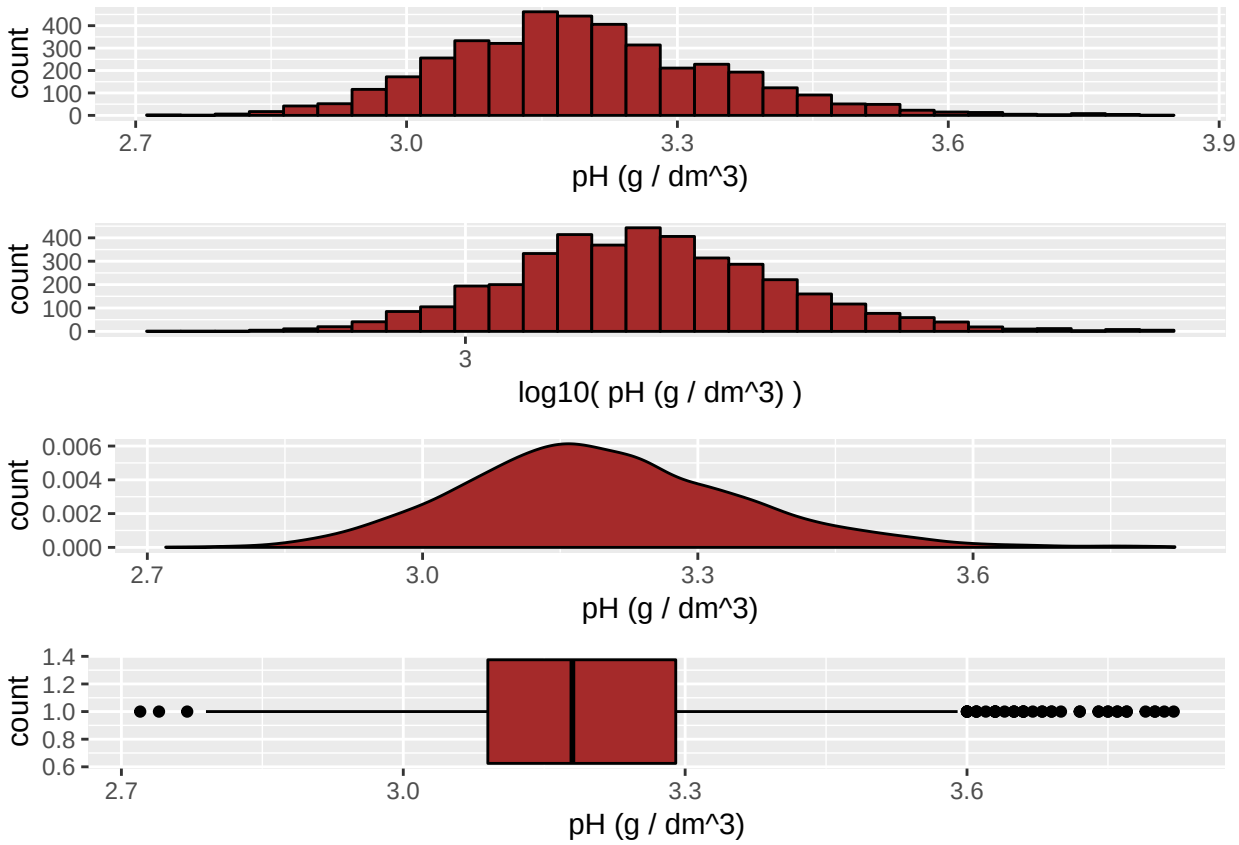
Distribution: The distribution of density is left-skewed for all plots.

Outliers: There is outliers with higher range, around 1.025 and 1.028.

pH

```
wdist(df$pH,
        varName = 'pH (g / dm^3)')
```

```
## Warning: Ignoring unknown parameters: binwidth
```



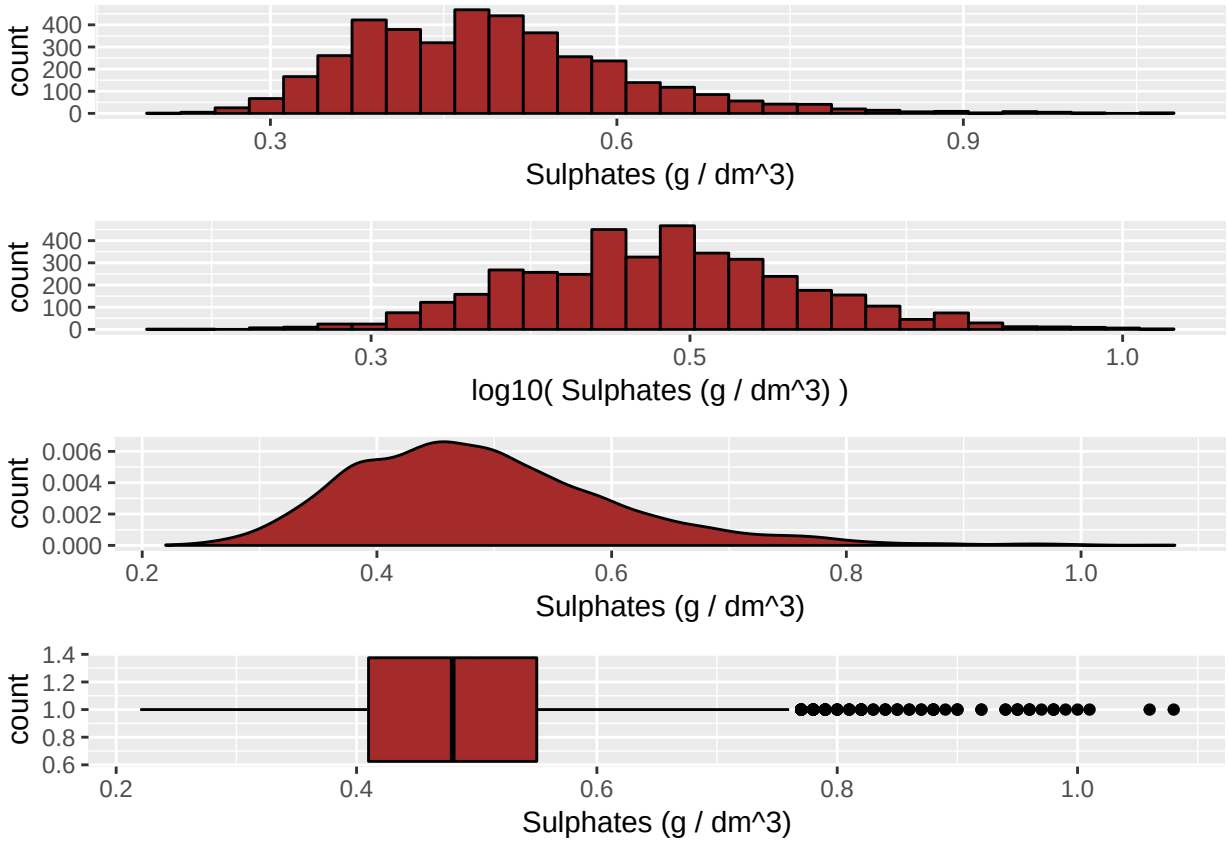
Distribution: The distribution of density is symmetrical for all plots.

Outliers: There is outliers which lies in 3.6 to 4.3

Sulphates

```
wdist(df$sulphates,
        varName = 'Sulphates (g / dm^3)')
```

```
## Warning: Ignoring unknown parameters: binwidth
```



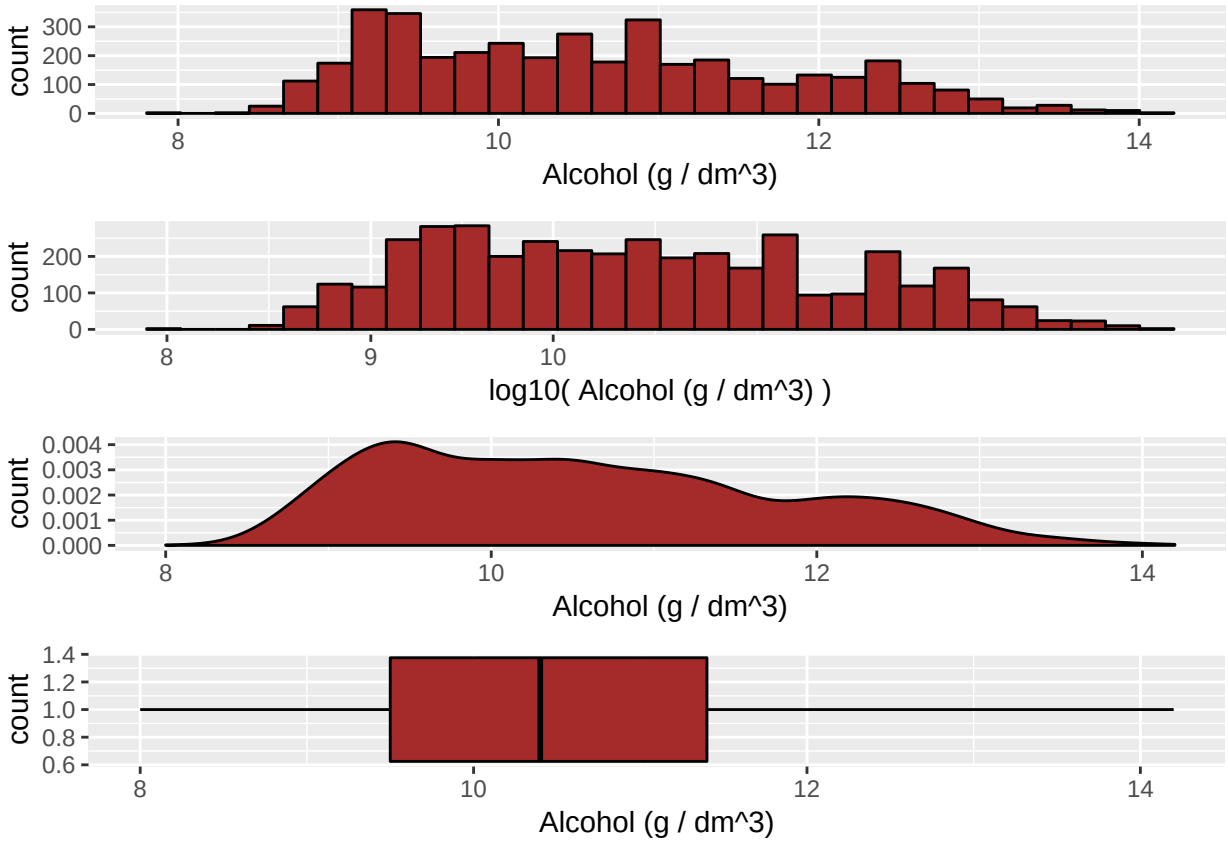
Distribution: It left-skewed, as we zoomed into log10, it change to right-skewed. It also has a long tail.

Outliers: There are many outliers with a higher range, around 0.77 to 1.4.

Alcohol

```
wdist(df$alcohol,
        varName = 'Alcohol (g / dm^3)')
```

```
## Warning: Ignoring unknown parameters: binwidth
```



Distribution: It seems almost a symmetrical distribution but instead it has a short tail.

Outliers: There are a no outlier.

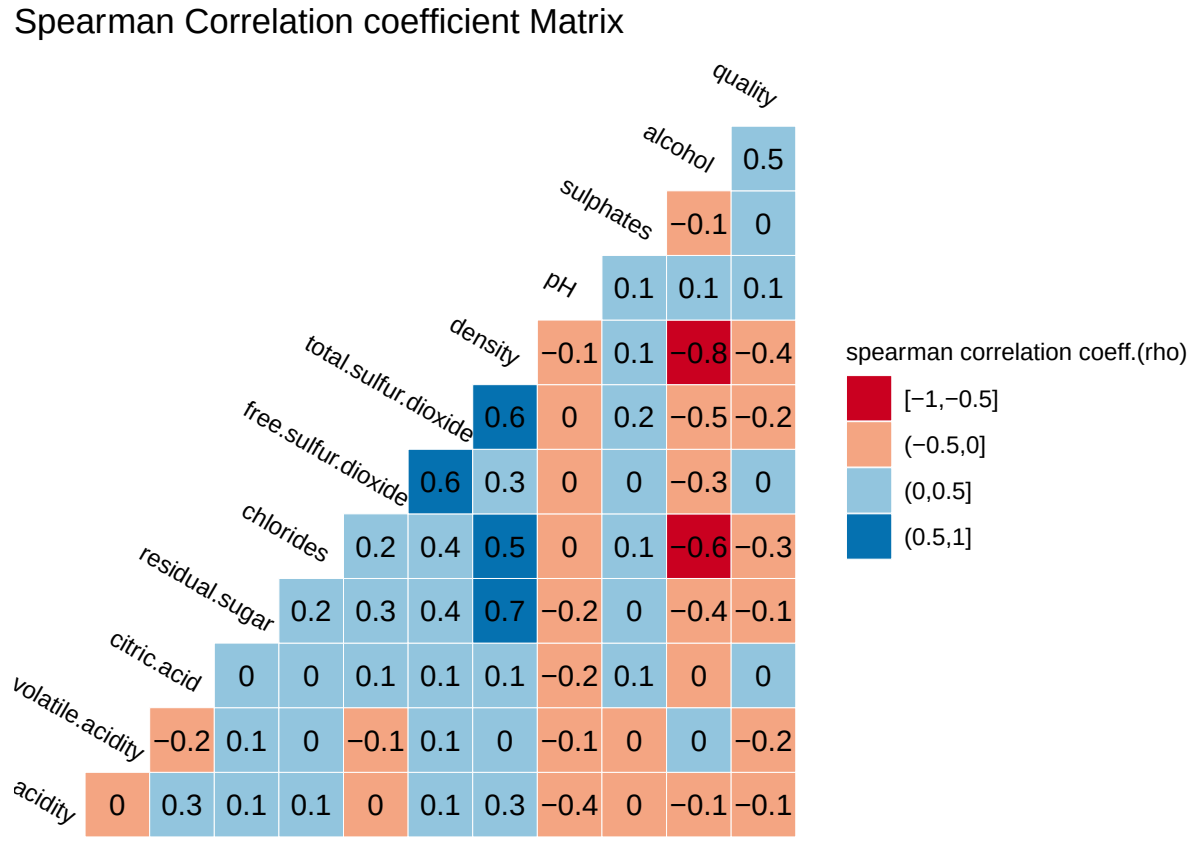
Correlation Analysis

```
#Investigate the relationship between all vas with correlation matrix

# Building Correlation Matrix with all vas
library(corrplot)
```

```
## Warning: package 'corrplot' was built under R version 3.5.1
```

```
## corrplot 0.84 loaded
```

```
#wq is data frame without index that was created in line 109
# ggcorr will use spearman method in all observations to plot a correlation matrix
ggcorr(winequality,
       method = c("all.obs","spearman"),
       nbreaks = 4, palette = 'RdBu', label = TRUE,
       name = "spearman correlation coeff.(rho)",
       hjust = 0.8, angle = -30, size = 3) +
  ggtitle("Spearman Correlation coefficient Matrix")
```



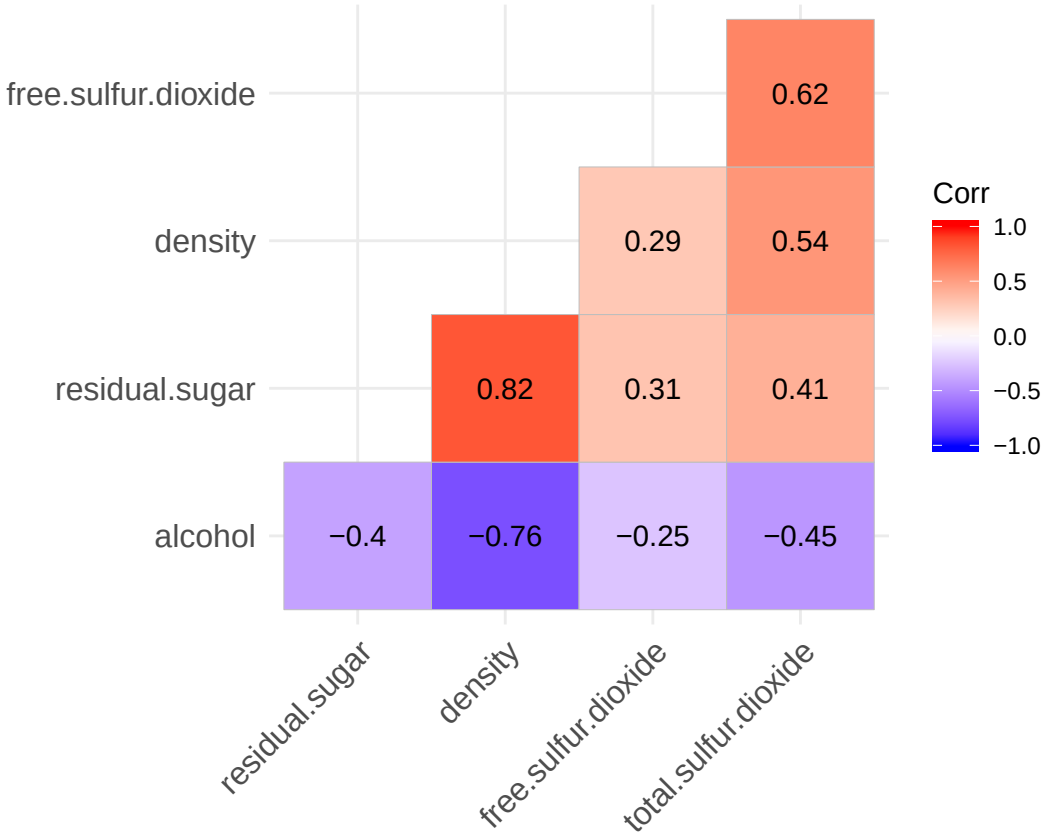
Positive correlation between: - Quality - Alcohol : 0.5 - Density - Sugar: 0.7 - Density- Chlorides: 0.5 - Density - Total Sulfur Dioxide: 0.6 - Total Sulfur Dioxide - Free Sulfur Dioxide: 0.6 Negative correlation between: - Quality - Volatile Acid: -0.4 - Density - Alcohol : -0.8 - pH - Fixed Acid: -0.4 - pH - Citric Acid: -0.2 - pH -Sugar: -0.2 - Citric Acid - Volatile Acid: -0.2

Correlation Network

```
library(ggcorrplot)
```

```
## Warning: package 'ggcorrplot' was built under R version 3.5.3
```

```
#Investigate the relationship between all vas with network analysis
correlations <- cor(as.matrix(df))
threshold <- 0.6
cc0 <- correlations
diag(cc0) <- 0
ok <- apply(abs(cc0) >= threshold, 1, any)
correlations <- correlations[ok, ok]
correlations <- as.data.frame(correlations)

ggcorrplot(correlations, hc.order = TRUE, type = "lower",
           lab = TRUE)
```



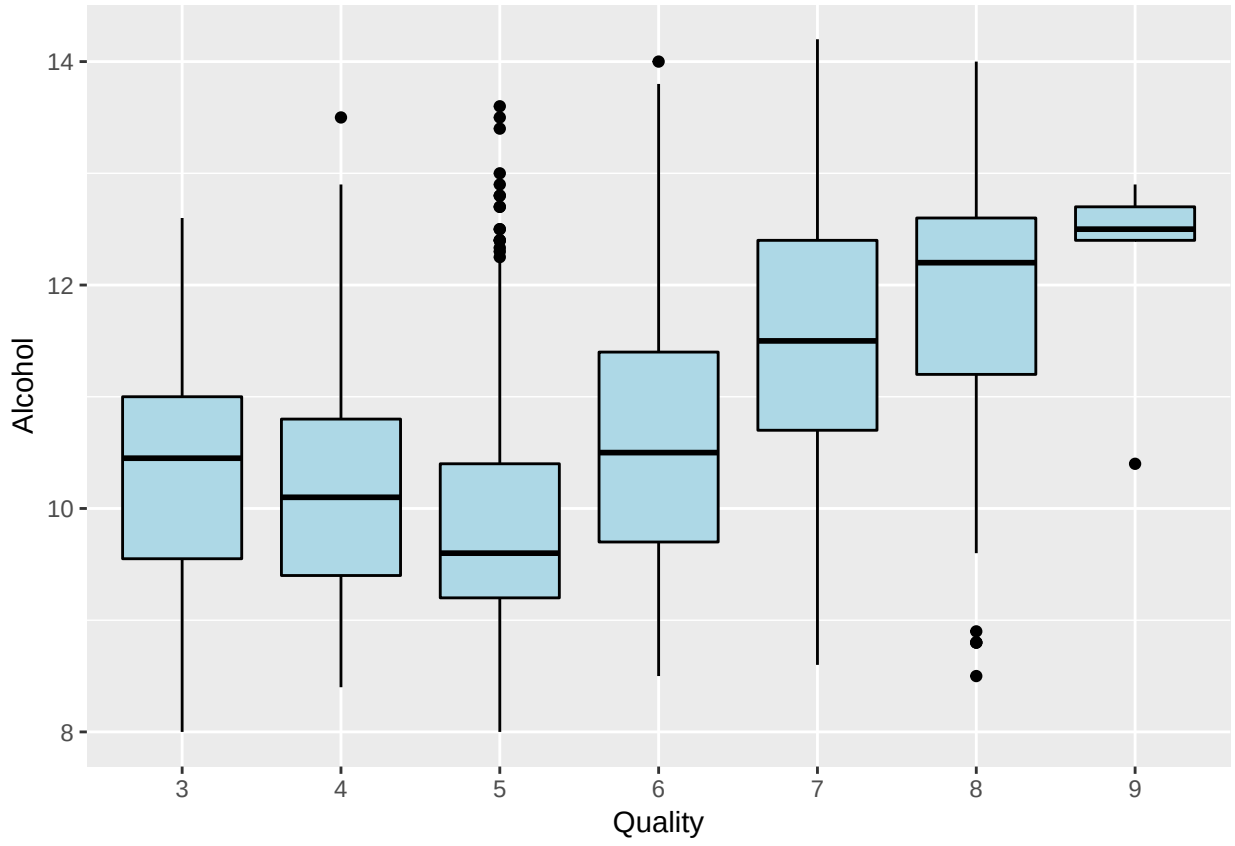
Bivariate Analysis

```
#Bulding function to analyze categorical va and continuos va
w_bxplot <- function (va1, varName = '',
                      va2, varName1 = '') {

  boxplot <- ggplot(aes(x = factor(va1), y = va2),
                    data = df) +
  geom_boxplot(color = 'black', fill = "lightblue") +
    labs(x = varName, y = varName1)
  plot(boxplot)
}


library(ggpubr)
#Bulding function to analyze continuos vas
w_scplot <- function(varName = '', varName1 = '',
                       varName2 = '', varName3 = '' ) {


  #Building scatter plot
  scatter <- ggscatter(df, x = varName, y = varName1,
              conf.int = TRUE, color = "orange",
              fill = "orange", cor.coef = TRUE,
              cor.method = "pearson") +
    theme_dark() +
```

```
    labs(title = paste(varName2, "vs", varName3, "scatter plot"))

  # Building scatter plot and add a linear regression line
  scatter2 <- ggscatter(df, x = varName, y = varName1,
              add = "reg.line", conf.int = TRUE, color = "orange",
              add.params = list(color = "grey20"), cor.coef = TRUE,
              cor.method = "pearson", alpha = 0.3) +
    theme_dark() +
    labs(title = paste(varName2, "vs", varName3,
                       "scatter plot with linear regression"))

  # Building scatter plot and add a fiting regression line
  scatter3 <- ggscatter(df, x = varName, y = varName1,
              add = "loess", conf.int = TRUE, color = "orange",
              add.params = list(color = "grey20"), cor.coef = TRUE,
              cor.method = "pearson", alpha = 0.3) +
    theme_dark() +
    labs(title = paste(varName2, "vs", varName3,
                       "scatter plot with fiting regression"))

  ggarrange(scatter, scatter2, scatter3,  nrow = 3)
}
```

Quality vs Alcohol

```
w_bxplot(df$quality, varName = 'Quality', df$alcohol,
           varName1 = 'Alcohol')
```




```
by(df$alcohol, df$quality, summary)
```

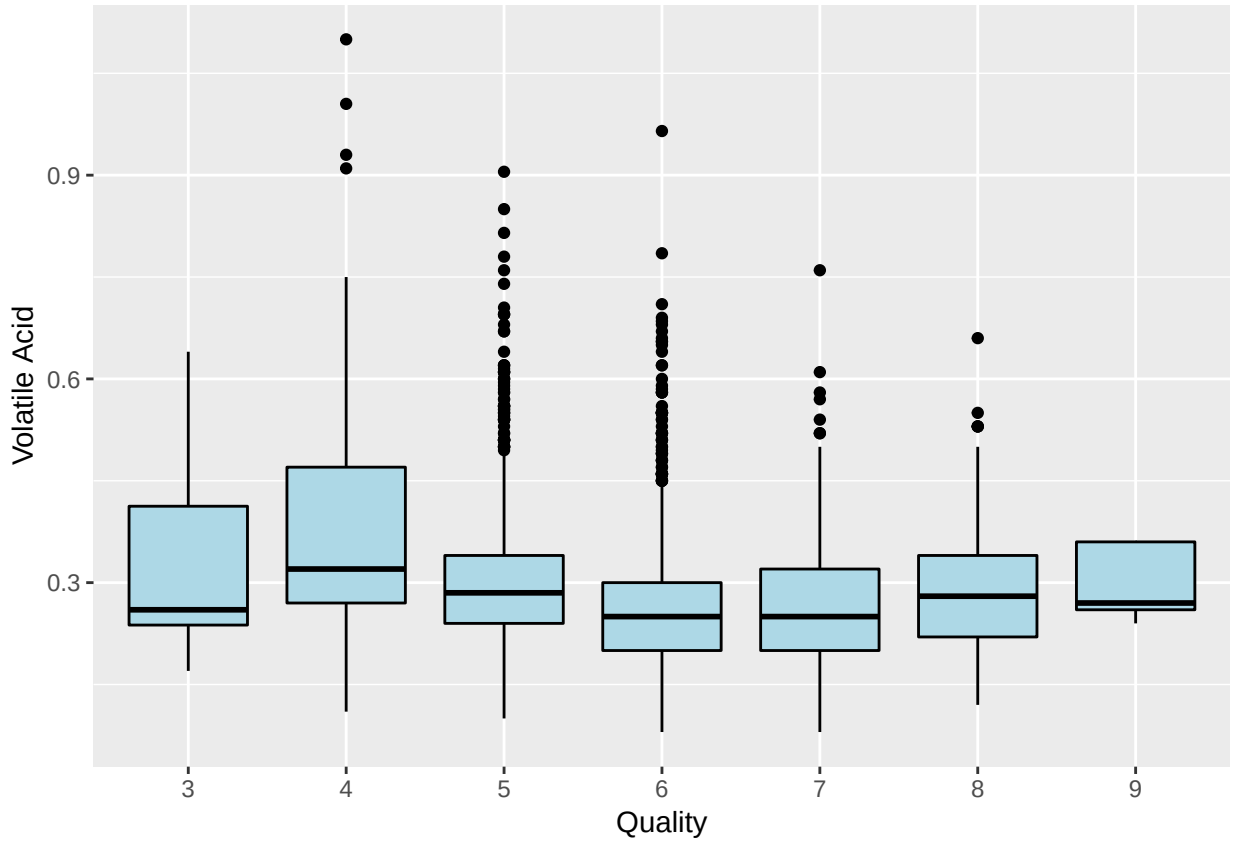
```
## df$quality: 3
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    8.00    9.55   10.45   10.35   11.00   12.60
## ------------------------------------------------------------
## df$quality: 4
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##     8.4     9.4    10.1    10.2    10.8    13.5
## ------------------------------------------------------------
## df$quality: 5
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   8.000   9.200   9.600   9.864  10.400  13.600
## ------------------------------------------------------------
## df$quality: 6
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    8.50    9.70   10.50   10.65   11.40   14.00
## ------------------------------------------------------------
## df$quality: 7
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    8.60   10.70   11.50   11.52   12.40   14.20
## ------------------------------------------------------------
## df$quality: 8
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    8.50   11.20   12.20   11.88   12.60   14.00
## ------------------------------------------------------------
```

```
## df$quality: 9
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   10.40   12.40   12.50   12.18   12.70   12.90
```

The trend between alcohol and quality is clearer,the amount of alcohol increases with better quality ranking. Additionally, most outliers have a score of 5, and that explains why the median is lower than score of 4.

Quality vs Volatile Acidity

```
w_bxplot(df$quality, varName = 'Quality', df$volatile.acidity,
           varName1 = 'Volatile Acid')
```




```
by(df$quality, df$volatile.acidity, summary)
```

```
## df$volatile.acidity: 0.08
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    6.00    6.25    6.50    6.50    6.75    7.00
## ------------------------------------------------------------
## df$volatile.acidity: 0.085
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##       6       6       6       6       6       6
## ------------------------------------------------------------
## df$volatile.acidity: 0.09
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##       6       6       6       6       6       6
## ------------------------------------------------------------
## df$volatile.acidity: 0.1
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
```

```
##   5.000   6.000   6.500   6.333   7.000   7.000
## ------------------------------------------------------------
## df$volatile.acidity: 0.105
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##     6.0     6.0     6.5     6.5     7.0     7.0
## ------------------------------------------------------------
## df$volatile.acidity: 0.11
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   4.000   6.000   6.000   6.222   7.000   7.000
## ------------------------------------------------------------
## df$volatile.acidity: 0.115
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   6.000   6.000   6.000   6.333   6.500   7.000
## ------------------------------------------------------------
## df$volatile.acidity: 0.12
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   5.000   6.000   6.000   6.286   7.000   8.000
## ------------------------------------------------------------
## df$volatile.acidity: 0.125
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   6.000   6.500   7.000   6.667   7.000   7.000
## ------------------------------------------------------------
## df$volatile.acidity: 0.13
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   5.000   6.000   6.000   6.306   7.000   8.000
## ------------------------------------------------------------
## df$volatile.acidity: 0.135
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##       6       6       6       6       6       6
## ------------------------------------------------------------
## df$volatile.acidity: 0.14
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   5.000   6.000   6.000   5.958   6.000   7.000
## ------------------------------------------------------------
## df$volatile.acidity: 0.145
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    6.00    6.00    6.00    6.25    6.25    7.00
## ------------------------------------------------------------
## df$volatile.acidity: 0.15
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   5.000   6.000   6.000   6.164   7.000   8.000
## ------------------------------------------------------------
## df$volatile.acidity: 0.155
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##     6.0     6.5     7.0     7.0     7.5     8.0
## ------------------------------------------------------------
## df$volatile.acidity: 0.16
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   4.000   6.000   6.000   6.055   6.000   8.000
## ------------------------------------------------------------
## df$volatile.acidity: 0.165
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    5.00    5.25    5.50    5.50    5.75    6.00
## ------------------------------------------------------------
```

```
## df$volatile.acidity: 0.17
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   3.000   6.000   6.000   6.097   7.000   8.000
## ------------------------------------------------------------
## df$volatile.acidity: 0.175
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##       6       6       6       6       6       6
## ------------------------------------------------------------
## df$volatile.acidity: 0.18
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   4.000   6.000   6.000   6.067   7.000   8.000
## ------------------------------------------------------------
## df$volatile.acidity: 0.185
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##       6       6       6       6       6       6
## ------------------------------------------------------------
## df$volatile.acidity: 0.19
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   4.000   6.000   6.000   6.095   7.000   8.000
## ------------------------------------------------------------
## df$volatile.acidity: 0.2
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   3.000   6.000   6.000   5.937   6.000   8.000
## ------------------------------------------------------------
## df$volatile.acidity: 0.205
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##       6       6       6       6       6       6
## ------------------------------------------------------------
## df$volatile.acidity: 0.21
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   4.000   5.000   6.000   5.994   6.000   8.000
## ------------------------------------------------------------
## df$volatile.acidity: 0.215
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##       3       3       3       3       3       3
## ------------------------------------------------------------
## df$volatile.acidity: 0.22
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   4.000   6.000   6.000   6.028   6.000   8.000
## ------------------------------------------------------------
## df$volatile.acidity: 0.225
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##     6.0     6.0     6.0     6.5     6.5     8.0
## ------------------------------------------------------------
## df$volatile.acidity: 0.23
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   3.000   5.000   6.000   5.861   6.000   8.000
## ------------------------------------------------------------
## df$volatile.acidity: 0.235
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    5.00    5.75    6.00    6.00    6.25    7.00
## ------------------------------------------------------------
## df$volatile.acidity: 0.24
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
```

```
##   3.000   5.000   6.000   5.894   6.000   9.000
## ------------------------------------------------------------
## df$volatile.acidity: 0.245
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##     5.0     5.0     5.5     5.5     6.0     6.0
## ------------------------------------------------------------
## df$volatile.acidity: 0.25
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   3.000   5.000   6.000   5.855   6.000   8.000
## ------------------------------------------------------------
## df$volatile.acidity: 0.255
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   4.000   6.000   6.000   5.889   6.000   7.000
## ------------------------------------------------------------
## df$volatile.acidity: 0.26
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   3.000   5.000   6.000   5.792   6.000   9.000
## ------------------------------------------------------------
## df$volatile.acidity: 0.265
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    5.00    5.75    6.50    6.25    7.00    7.00
## ------------------------------------------------------------
## df$volatile.acidity: 0.27
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   4.000   5.000   6.000   5.663   6.000   9.000
## ------------------------------------------------------------
## df$volatile.acidity: 0.275
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##     5.0     5.5     6.0     6.0     6.5     7.0
## ------------------------------------------------------------
## df$volatile.acidity: 0.28
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   4.000   5.000   6.000   5.817   6.000   8.000
## ------------------------------------------------------------
## df$volatile.acidity: 0.285
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   5.000   5.500   6.000   6.333   7.000   8.000
## ------------------------------------------------------------
## df$volatile.acidity: 0.29
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   4.000   5.000   6.000   5.752   6.000   8.000
## ------------------------------------------------------------
## df$volatile.acidity: 0.295
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##     5.0     5.5     6.0     6.0     6.5     7.0
## ------------------------------------------------------------
## df$volatile.acidity: 0.3
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   4.000   5.000   6.000   5.851   6.000   8.000
## ------------------------------------------------------------
## df$volatile.acidity: 0.305
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##     5.0     5.0     5.0     5.5     5.5     7.0
## ------------------------------------------------------------
```

```
## df$volatile.acidity: 0.31
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   4.000   5.000   5.000   5.627   6.000   8.000
## ------------------------------------------------------------
## df$volatile.acidity: 0.315
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    5.00    5.75    6.00    6.25    6.50    8.00
## ------------------------------------------------------------
## df$volatile.acidity: 0.32
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   3.000   5.000   6.000   5.778   6.000   8.000
## ------------------------------------------------------------
## df$volatile.acidity: 0.325
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##       5       5       5       5       5       5
## ------------------------------------------------------------
## df$volatile.acidity: 0.33
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   3.000   5.000   6.000   5.864   6.000   8.000
## ------------------------------------------------------------
## df$volatile.acidity: 0.335
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   5.000   5.000   5.000   5.333   5.750   6.000
## ------------------------------------------------------------
## df$volatile.acidity: 0.34
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   4.000   5.000   6.000   5.767   6.000   8.000
## ------------------------------------------------------------
## df$volatile.acidity: 0.345
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   5.000   5.000   5.000   5.167   5.000   6.000
## ------------------------------------------------------------
## df$volatile.acidity: 0.35
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   4.000   5.000   6.000   5.736   6.000   8.000
## ------------------------------------------------------------
## df$volatile.acidity: 0.355
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##       6       6       6       6       6       6
## ------------------------------------------------------------
## df$volatile.acidity: 0.36
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   4.000   5.000   6.000   5.966   7.000   9.000
## ------------------------------------------------------------
## df$volatile.acidity: 0.365
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    5.00    5.25    5.50    5.50    5.75    6.00
## ------------------------------------------------------------
## df$volatile.acidity: 0.37
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   4.000   5.000   6.000   5.907   6.000   8.000
## ------------------------------------------------------------
## df$volatile.acidity: 0.375
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
```

```
##    5.00    5.25    5.50    5.50    5.75    6.00
## ------------------------------------------------------------
## df$volatile.acidity: 0.38
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   4.000   5.000   6.000   5.612   6.000   7.000
## ------------------------------------------------------------
## df$volatile.acidity: 0.385
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##       6       6       6       6       6       6
## ------------------------------------------------------------
## df$volatile.acidity: 0.39
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   3.000   5.000   6.000   5.571   6.000   8.000
## ------------------------------------------------------------
## df$volatile.acidity: 0.395
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##     4.0     4.5     5.0     5.0     5.5     6.0
## ------------------------------------------------------------
## df$volatile.acidity: 0.4
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   4.000   5.000   6.000   5.894   6.500   8.000
## ------------------------------------------------------------
## df$volatile.acidity: 0.405
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##       4       4       4       4       4       4
## ------------------------------------------------------------
## df$volatile.acidity: 0.41
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##     4.0     5.0     6.0     5.8     6.0     8.0
## ------------------------------------------------------------
## df$volatile.acidity: 0.415
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##     5.0     5.0     5.0     5.5     5.5     7.0
## ------------------------------------------------------------
## df$volatile.acidity: 0.42
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   5.000   5.000   6.000   5.938   6.250   8.000
## ------------------------------------------------------------
## df$volatile.acidity: 0.425
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    5.00    5.25    5.50    5.50    5.75    6.00
## ------------------------------------------------------------
## df$volatile.acidity: 0.43
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   4.000   5.000   6.000   5.586   6.000   8.000
## ------------------------------------------------------------
## df$volatile.acidity: 0.435
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##       6       6       6       6       6       6
## ------------------------------------------------------------
## df$volatile.acidity: 0.44
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   4.000   5.000   6.000   6.118   7.000   8.000
## ------------------------------------------------------------
```

```
## df$volatile.acidity: 0.445
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##     5.0     5.0     5.5     5.5     6.0     6.0
## ------------------------------------------------------------
## df$volatile.acidity: 0.45
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    4.00    5.00    5.00    5.81    7.00    8.00
## ------------------------------------------------------------
## df$volatile.acidity: 0.455
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##       7       7       7       7       7       7
## ------------------------------------------------------------
## df$volatile.acidity: 0.46
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   5.000   5.000   6.000   5.889   7.000   8.000
## ------------------------------------------------------------
## df$volatile.acidity: 0.47
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##     4.0     5.0     5.0     5.5     6.0     8.0
## ------------------------------------------------------------
## df$volatile.acidity: 0.475
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##     5.0     5.5     6.0     6.0     6.5     7.0
## ------------------------------------------------------------
## df$volatile.acidity: 0.48
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   3.000   5.000   6.000   5.538   7.000   7.000
## ------------------------------------------------------------
## df$volatile.acidity: 0.485
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    4.00    4.25    4.50    4.50    4.75    5.00
## ------------------------------------------------------------
## df$volatile.acidity: 0.49
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   3.000   5.000   5.000   5.364   6.000   7.000
## ------------------------------------------------------------
## df$volatile.acidity: 0.495
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    5.00    5.25    5.50    5.50    5.75    6.00
## ------------------------------------------------------------
## df$volatile.acidity: 0.5
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   4.000   5.000   5.500   5.833   7.000   8.000
## ------------------------------------------------------------
## df$volatile.acidity: 0.51
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    5.00    5.00    5.00    5.30    5.75    6.00
## ------------------------------------------------------------
## df$volatile.acidity: 0.52
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   4.000   5.000   6.000   5.778   6.000   7.000
## ------------------------------------------------------------
## df$volatile.acidity: 0.53
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
```

```
##   4.000   4.500   6.000   6.143   8.000   8.000
## ------------------------------------------------------------
## df$volatile.acidity: 0.54
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   4.000   5.000   5.000   5.333   6.000   7.000
## ------------------------------------------------------------
## df$volatile.acidity: 0.545
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##       5       5       5       5       5       5
## ------------------------------------------------------------
## df$volatile.acidity: 0.55
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   3.000   5.000   6.000   5.385   6.000   8.000
## ------------------------------------------------------------
## df$volatile.acidity: 0.555
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##       5       5       5       5       5       5
## ------------------------------------------------------------
## df$volatile.acidity: 0.56
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##       4       5       5       5       5       6
## ------------------------------------------------------------
## df$volatile.acidity: 0.57
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    4.00    4.75    5.00    5.25    5.50    7.00
## ------------------------------------------------------------
## df$volatile.acidity: 0.58
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   5.000   5.250   6.000   5.833   6.000   7.000
## ------------------------------------------------------------
## df$volatile.acidity: 0.585
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    5.00    5.25    5.50    5.50    5.75    6.00
## ------------------------------------------------------------
## df$volatile.acidity: 0.59
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    3.00    3.75    4.50    4.50    5.25    6.00
## ------------------------------------------------------------
## df$volatile.acidity: 0.595
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    4.00    4.25    4.50    4.50    4.75    5.00
## ------------------------------------------------------------
## df$volatile.acidity: 0.6
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   4.000   4.000   5.000   4.714   5.000   6.000
## ------------------------------------------------------------
## df$volatile.acidity: 0.61
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   4.000   5.000   5.000   5.167   5.000   7.000
## ------------------------------------------------------------
## df$volatile.acidity: 0.615
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   4.000   4.000   4.000   4.333   4.500   5.000
## ------------------------------------------------------------
```

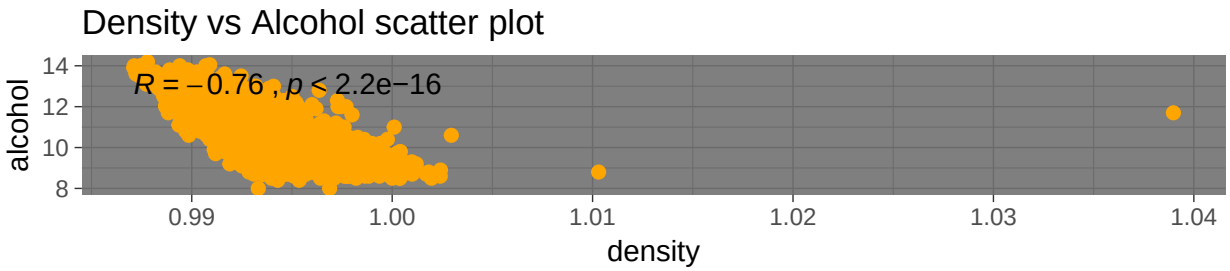
```
## df$volatile.acidity: 0.62
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##     5.0     5.0     5.0     5.4     6.0     6.0
## ------------------------------------------------------------
## df$volatile.acidity: 0.63
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##       4       4       4       4       4       4
## ------------------------------------------------------------
## df$volatile.acidity: 0.64
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   3.000   4.000   4.000   4.333   4.750   6.000
## ------------------------------------------------------------
## df$volatile.acidity: 0.65
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##     4.0     4.5     5.0     5.0     5.5     6.0
## ------------------------------------------------------------
## df$volatile.acidity: 0.655
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   4.000   5.000   6.000   5.333   6.000   6.000
## ------------------------------------------------------------
## df$volatile.acidity: 0.66
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##       4       5       6       6       7       8
## ------------------------------------------------------------
## df$volatile.acidity: 0.67
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##     4.0     4.0     5.0     4.8     5.0     6.0
## ------------------------------------------------------------
## df$volatile.acidity: 0.68
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##     4.0     4.5     5.0     5.0     5.5     6.0
## ------------------------------------------------------------
## df$volatile.acidity: 0.685
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##       6       6       6       6       6       6
## ------------------------------------------------------------
## df$volatile.acidity: 0.69
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##       6       6       6       6       6       6
## ------------------------------------------------------------
## df$volatile.acidity: 0.695
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##       5       5       5       5       5       5
## ------------------------------------------------------------
## df$volatile.acidity: 0.705
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    4.00    4.25    4.50    4.50    4.75    5.00
## ------------------------------------------------------------
## df$volatile.acidity: 0.71
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##       6       6       6       6       6       6
## ------------------------------------------------------------
## df$volatile.acidity: 0.73
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
```

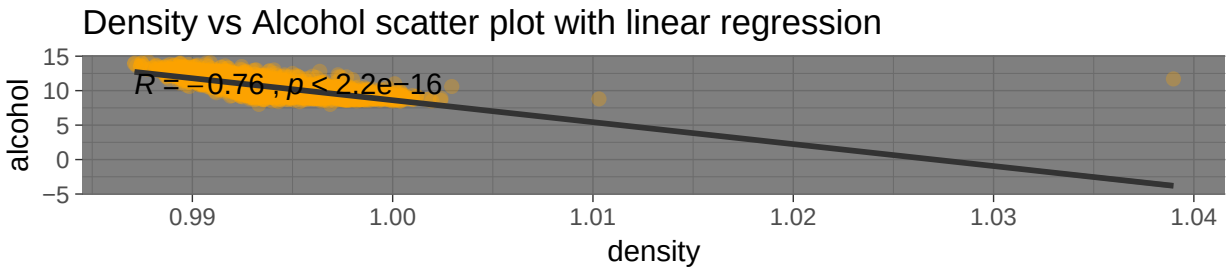
```
##       4       4       4       4       4       4
## ------------------------------------------------------------
## df$volatile.acidity: 0.74
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##       5       5       5       5       5       5
## ------------------------------------------------------------
## df$volatile.acidity: 0.75
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##       4       4       4       4       4       4
## ------------------------------------------------------------
## df$volatile.acidity: 0.76
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##     5.0     5.5     6.0     6.0     6.5     7.0
## ------------------------------------------------------------
## df$volatile.acidity: 0.78
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##       5       5       5       5       5       5
## ------------------------------------------------------------
## df$volatile.acidity: 0.785
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##       6       6       6       6       6       6
## ------------------------------------------------------------
## df$volatile.acidity: 0.815
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##       5       5       5       5       5       5
## ------------------------------------------------------------
## df$volatile.acidity: 0.85
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##       5       5       5       5       5       5
## ------------------------------------------------------------
## df$volatile.acidity: 0.905
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##       5       5       5       5       5       5
## ------------------------------------------------------------
## df$volatile.acidity: 0.91
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##       4       4       4       4       4       4
## ------------------------------------------------------------
## df$volatile.acidity: 0.93
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##       4       4       4       4       4       4
## ------------------------------------------------------------
## df$volatile.acidity: 0.965
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##       6       6       6       6       6       6
## ------------------------------------------------------------
## df$volatile.acidity: 1.005
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##       4       4       4       4       4       4
## ------------------------------------------------------------
## df$volatile.acidity: 1.1
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##       4       4       4       4       4       4
```

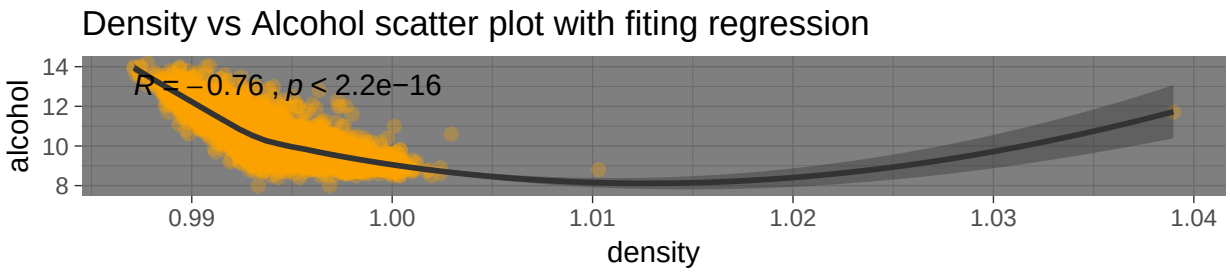
Volatile acidity shows an opposite trend, the worst score quality having the largest median. By way of explanation, the amount of volatile acidity decreases with a better quality ranking.

Density vs Alcohol scatter plot

```
w_scplot(varName = "density", varName1 = "alcohol",
           varName2 = "Density", varName3 = "Alcohol")
```



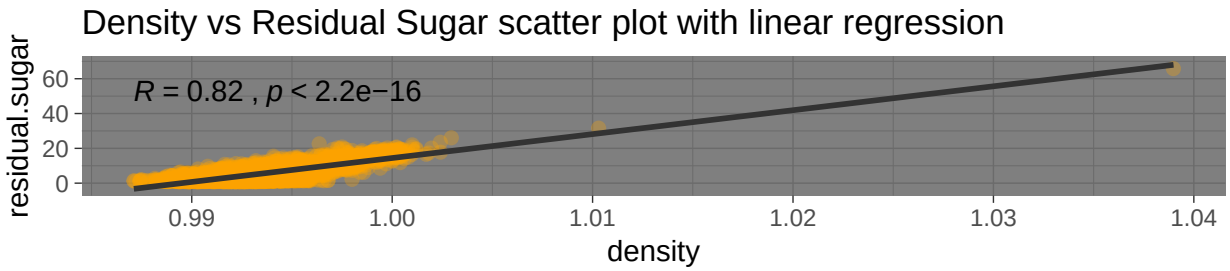


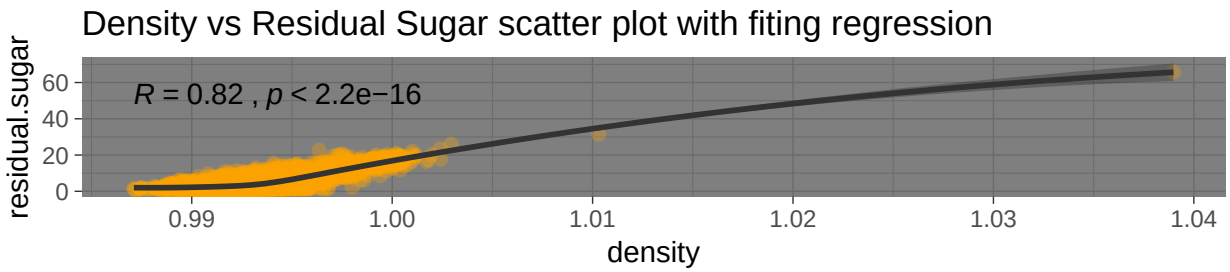


There is a moderate negative correlation between density vs alcohol. That means when the level of alcohol decreases, the density increases. We can see this pattern with the line models.

Density vs Sugar scatter plot

```
w_scplot(varName = "density", varName1 = "residual.sugar",
           varName2 = "Density", varName3 = "Residual Sugar")
```

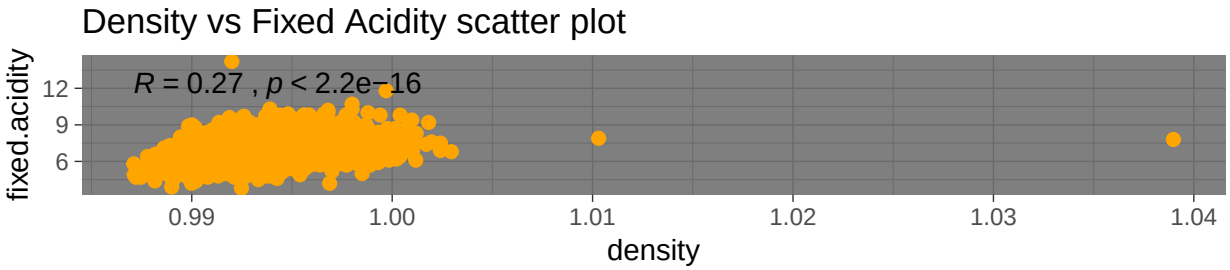


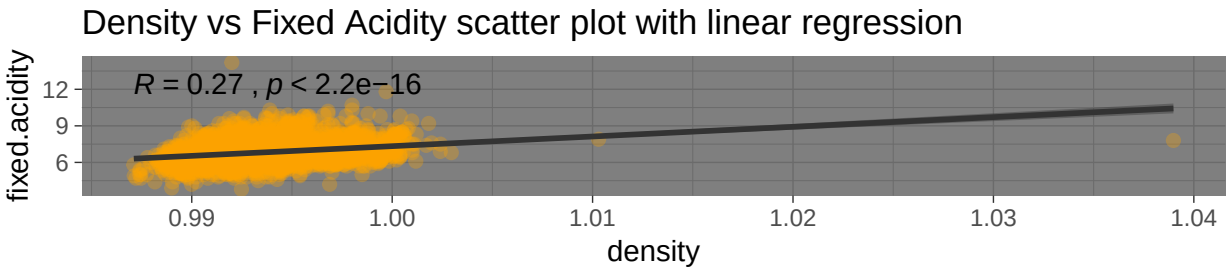


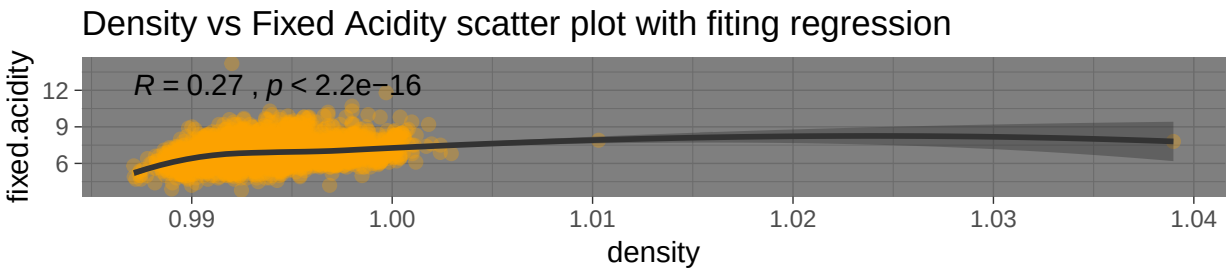
There is a weak positive correlation between density and sugar . That means the level of fixed acidity increases as the density increases. We can see this trend with the line models.

Density vs Fixed.acidity scatter plot

```
w_scplot(varName = "density", varName1 = "fixed.acidity",
           varName2 = "Density", varName3 = "Fixed Acidity")
```



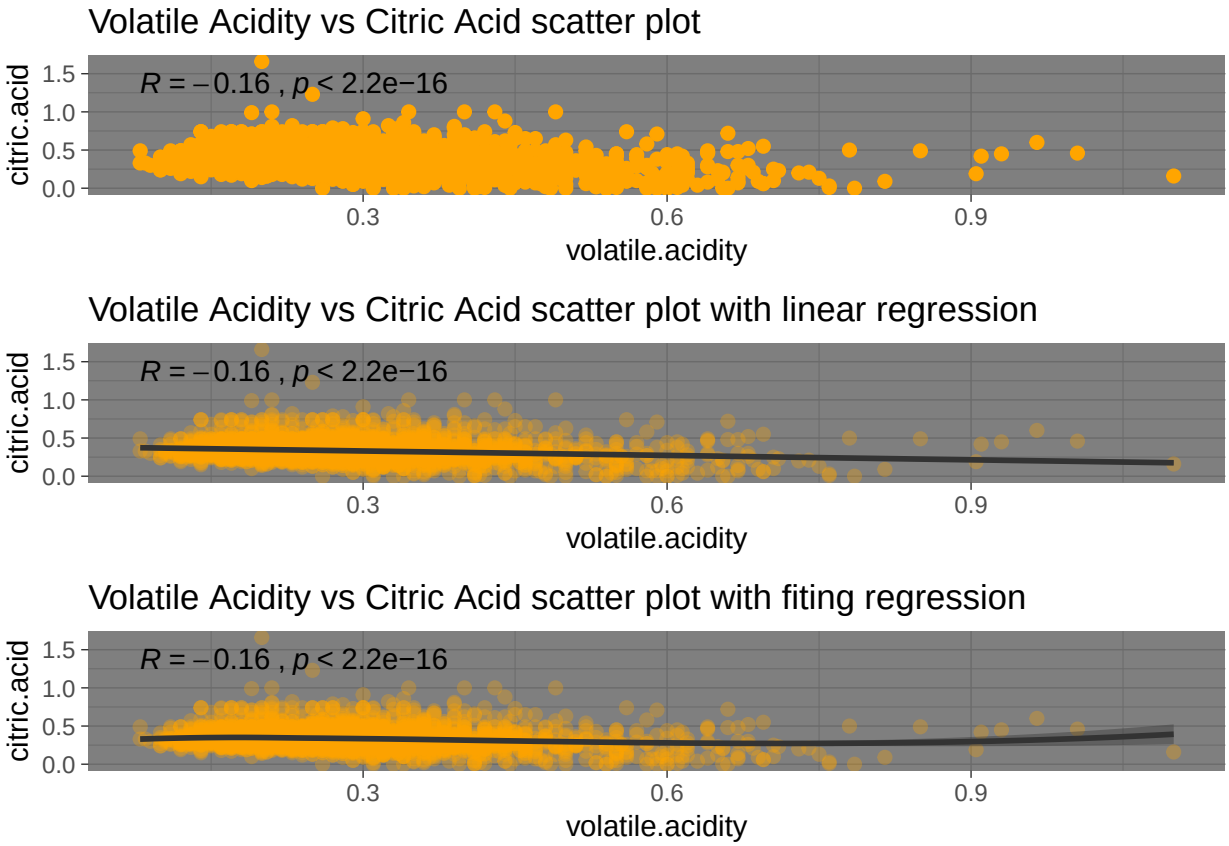




There is a weak positive correlation between density and fixed acidity. We can see with the line model a slight increased trend. So, as the fixed acidity rises, the density grows slightly.

Volatile vs Citric Acid scatter plot

```
w_scplot(varName = "volatile.acidity", varName1 = "citric.acid",
          varName2 = "Volatile Acidity", varName3 = "Citric Acid")
```



Volatite acidity and citric acid have a moderate negative correlation, almost zero correlation. It seems they don't have relationship to each other.

Bivariate Analysis Summary

Since quality is a categorical ordinal va, we analyzed it vs continuous va with a box plot. Then, we examined continuous vas between each other with a scatter plot. We further applied a regression line that allows us to understand the pattern between these correlations.

The box plots show how alcohol and volatile acidity influence the quality rating. In others words, the quality va has only a moderate positive correlation with alcohol (0.5), and a moderate negative correlation with volatile acidity (-0.4). Quality does not have other moderate or strong correlations.

In addition, we saw how alcohol, sugar, and chlorides affect the va density. Density has a strong negative correlation with alcohol (-0.5), and a strong positive correlation with residual sugar, The sulfur dioxide. The pH va has a strong negative correlation with fixed acid and moderate negative correlation with citric acidity and Sugar.

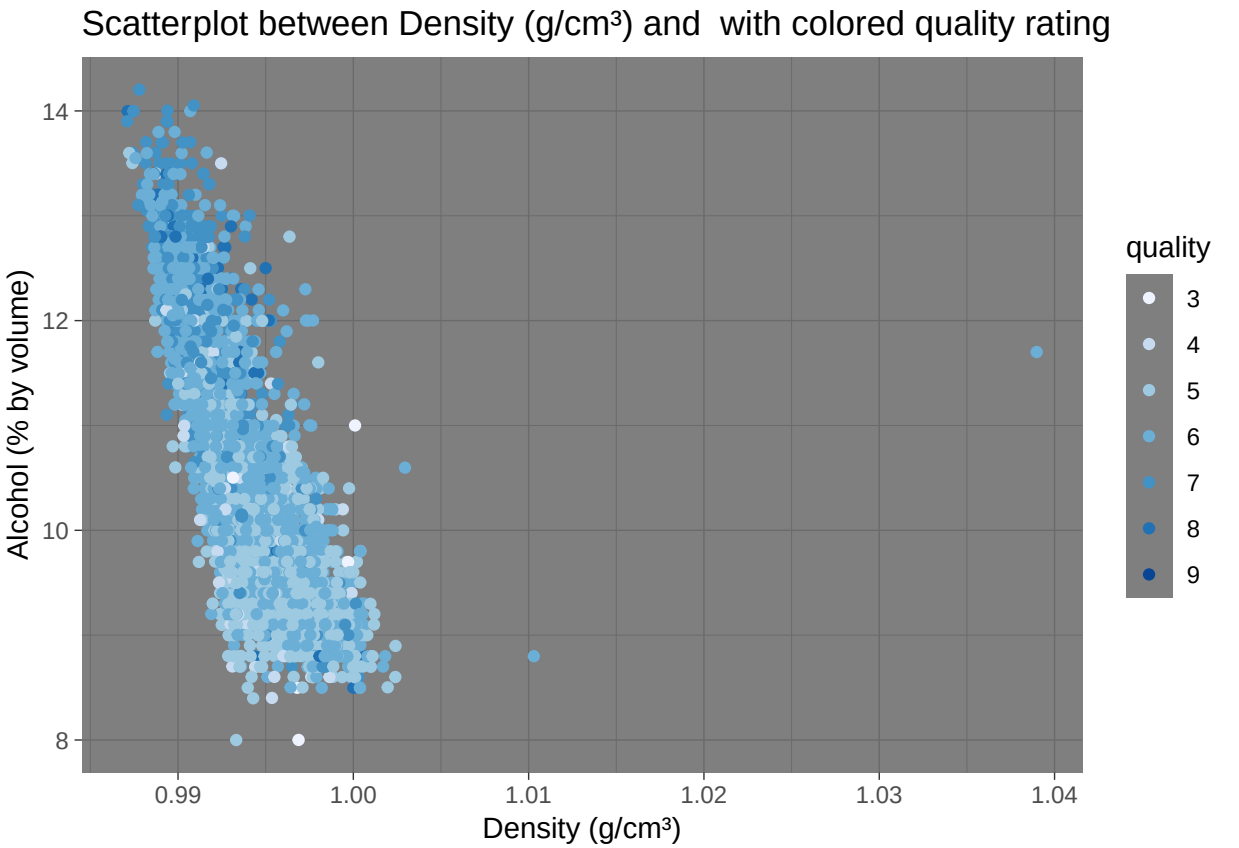
Multivariate Analysis

```
# use function "factor" to enconde "quality" as factor with default order
df$quality = factor(df$quality)


# building function with scatterplot colored by quality
multi_ <- function(va, varName = '', va1, varName1 ='') {

    ggplot(df, aes(x = va, y = va1, color = quality)) +
```

```
    geom_jitter() +
    # color enconding
    scale_color_brewer(type = 'div', palette = "Blues") +
    # darken the background in order to see light colored points
    theme_dark() +
    labs(x = varName,
         y = varName1,
         title = paste("Scatterplot between", varName, "and",
           varName1 = '', "with colored quality rating"))

}

#Building function to facet the scatterplot by quality score with regression
multi_2 <- function(varName = '', varName1 = '', varName2 = '',
                        varName3 = '' ) {

  s2 <- ggscatter(df, x = varName, y = varName1, facet.by = "quality",
            add = "reg.line", add.params = list(color = "grey20"),
          conf.int = TRUE, color = "lightblue", fill = "grey",
            cor.coef = TRUE, cor.method = "pearson", alpha = 0.5) +
    theme_dark() +
    labs(title = paste(varName2, "vs", varName3,
                        "scatter plot with linear regression"))

  s3 <- ggscatter(df, x = varName, y = varName1, facet.by = "quality",
              add = "loess", add.params = list(color = "grey20"),
            conf.int = TRUE, color = "lightblue", fill = "grey",
              cor.coef = TRUE, cor.method = "pearson", alpha = 0.5) +
    theme_dark() +
    labs(title = paste(varName2, "vs", varName3,
                        "scatter plot with fiting regression"))

ggarrange(s2, s3, nrow = 2)

}
```

Density VS Alcohol

```
multi_(df$density, varName = 'Density (g/cm³)', df$alcohol,
       varName1 = 'Alcohol (% by volume)')
```



Most of the samples with a 7 and 8 quality score appear above 10% of alcohol and below 0.995 of density. Wines with a low score are distributed around 9% to 10% of alcohol and a density concentration of about 0.995 to 1.000.

```
multi_2(varName = "density", varName1 = "alcohol",
        varName2 = "Density", varName3 = "Alcohol")
```

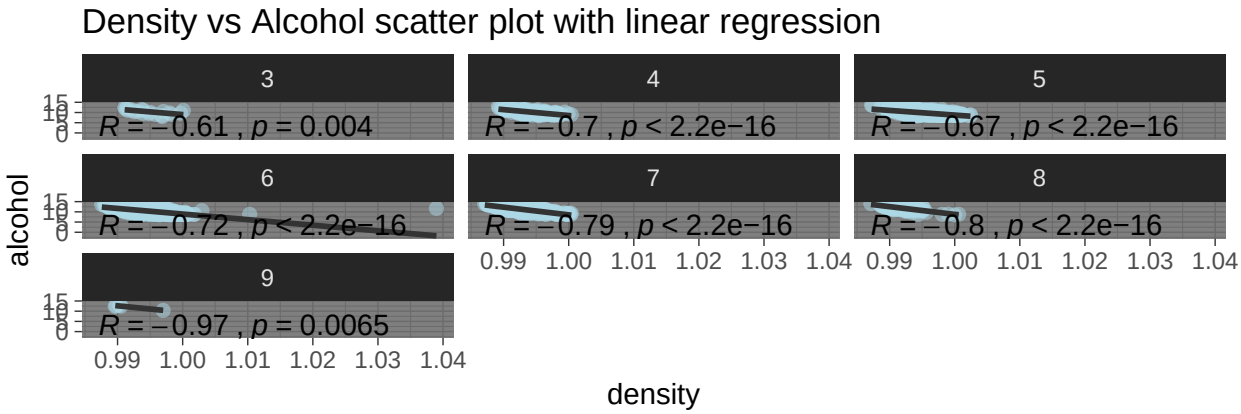
```
## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : span too small. fewer data values than degrees of freedom.

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : pseudoinverse used at 0.98961

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : neighborhood radius 0.00068675

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : reciprocal condition number 0

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : There are other near singularities as well. 4.5384e-005

## Warning in predLoess(object$y, object$x, newx = if
## (is.null(newdata)) object$x else if (is.data.frame(newdata))
## as.matrix(model.frame(delete.response(terms(object)), : span too small.
## fewer data values than degrees of freedom.

## Warning in predLoess(object$y, object$x, newx = if
## (is.null(newdata)) object$x else if (is.data.frame(newdata))
## as.matrix(model.frame(delete.response(terms(object)), : pseudoinverse used
```

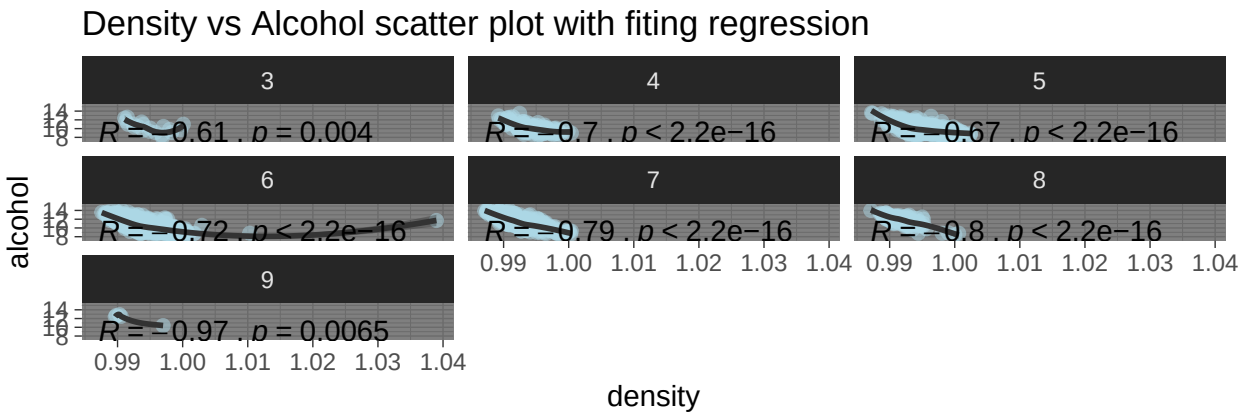
```
## at 0.98961
```

```
## Warning in predLoess(object$y, object$x, newx = if
## (is.null(newdata)) object$x else if (is.data.frame(newdata))
## as.matrix(model.frame(delete.response(terms(object)), : neighborhood radius
## 0.00068675
```

```
## Warning in predLoess(object$y, object$x, newx = if
## (is.null(newdata)) object$x else if (is.data.frame(newdata))
## as.matrix(model.frame(delete.response(terms(object)), : reciprocal
## condition number 0
```

```
## Warning in predLoess(object$y, object$x, newx = if
## (is.null(newdata)) object$x else if (is.data.frame(newdata))
## as.matrix(model.frame(delete.response(terms(object)), : There are other
## near singularities as well. 4.5384e-005
```

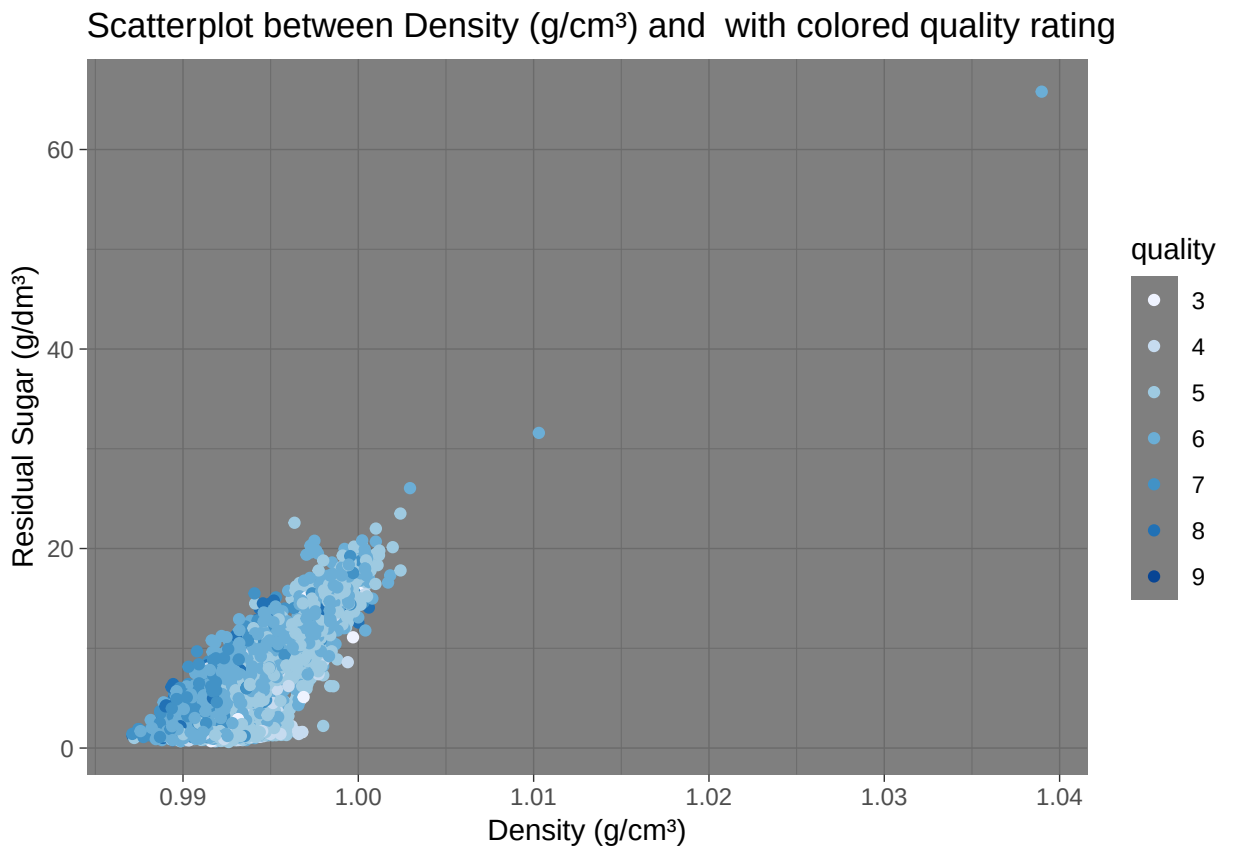




Comparing density VS alcohol by quality, we can see a strong negative correlation between density and alcohol by quality scores, except a score of 9. The samples with a score of 9 show the most strong negative correlation, i.e. 0.97.

Density vs Sugar

```
multi_(df$density, varName = 'Density (g/cm³)', df$residual.sugar,
       varName1 = 'Residual Sugar (g/dm³)')
```



Density and Sugar analyzed by quality show a strong positive linear correlation. We can see sugar increases as density increases.

```
multi_2(varName = "density", varName1 = "residual.sugar",
        varName2 = "Density", varName3 = "Sugar")
```

```
## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : span too small. fewer data values than degrees of freedom.
```

```
## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : pseudoinverse used at 0.98961
```

```
## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : neighborhood radius 0.00068675
```

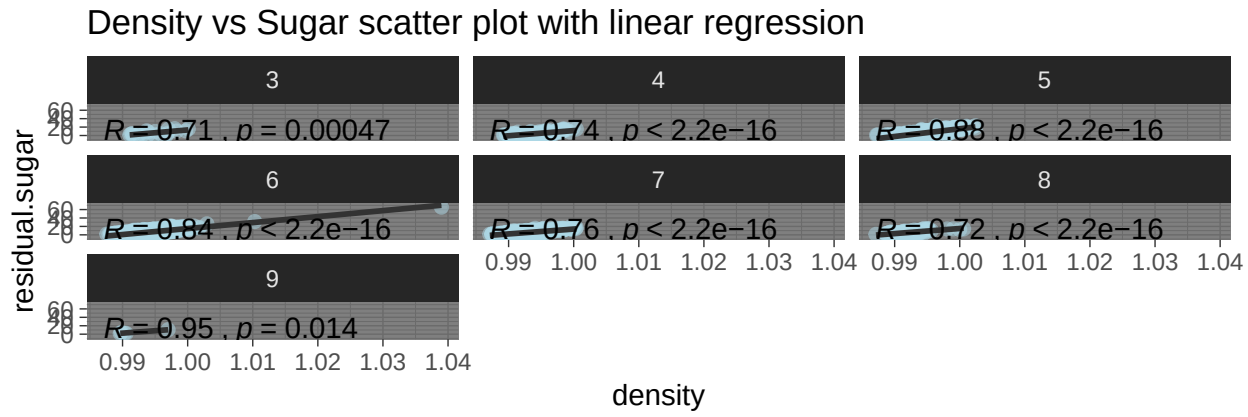
```
## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : reciprocal condition number 0
```

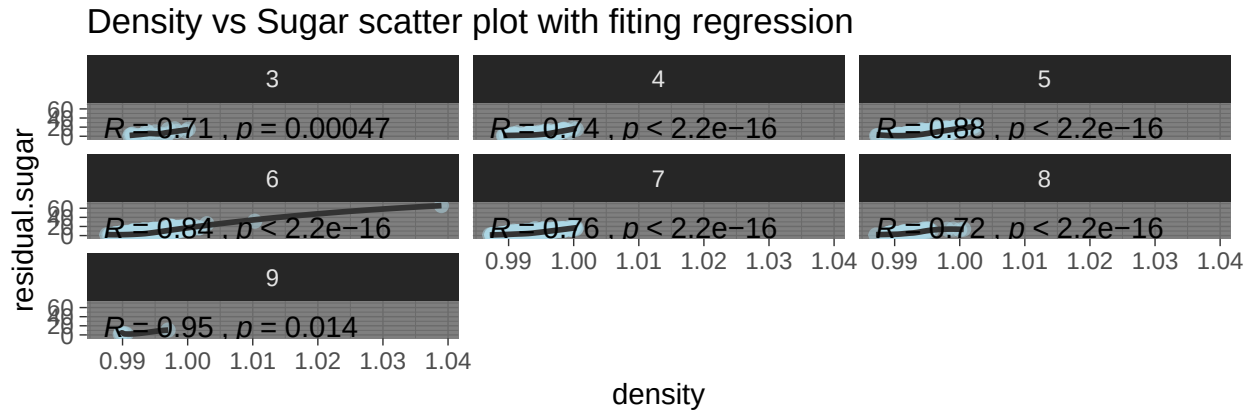
```
## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : There are other near singularities as well. 4.5384e-005
```

```
## Warning in predLoess(object$y, object$x, newx = if
## (is.null(newdata)) object$x else if (is.data.frame(newdata))
## as.matrix(model.frame(delete.response(terms(object)), : span too small.
## fewer data values than degrees of freedom.
```

```
## Warning in predLoess(object$y, object$x, newx = if
## (is.null(newdata)) object$x else if (is.data.frame(newdata))
## as.matrix(model.frame(delete.response(terms(object)), : pseudoinverse used
## at 0.98961
```

```
## Warning in predLoess(object$y, object$x, newx = if
## (is.null(newdata)) object$x else if (is.data.frame(newdata))
## as.matrix(model.frame(delete.response(terms(object)), : neighborhood radius
## 0.00068675

## Warning in predLoess(object$y, object$x, newx = if
## (is.null(newdata)) object$x else if (is.data.frame(newdata))
## as.matrix(model.frame(delete.response(terms(object)), : reciprocal
## condition number 0

## Warning in predLoess(object$y, object$x, newx = if
## (is.null(newdata)) object$x else if (is.data.frame(newdata))
## as.matrix(model.frame(delete.response(terms(object)), : There are other
## near singularities as well. 4.5384e-005
```

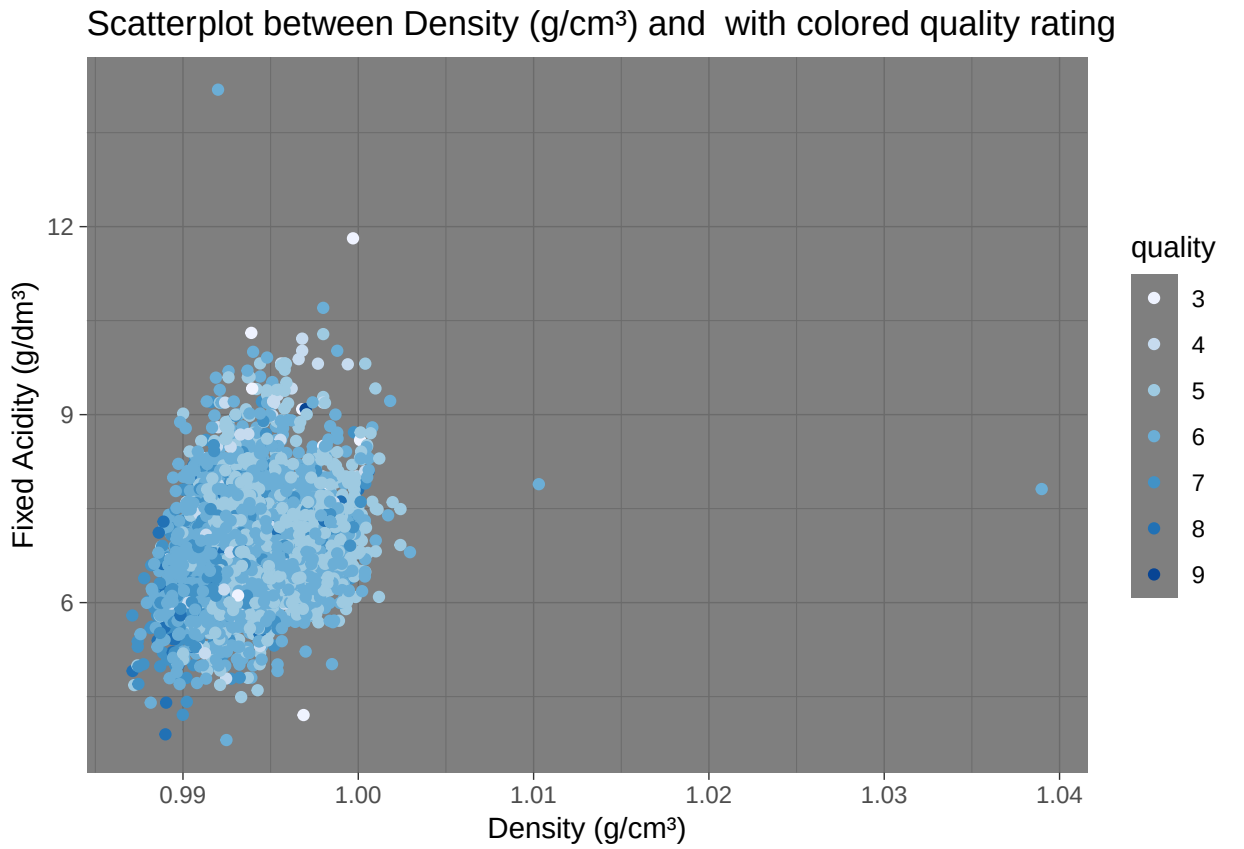




The density and sugar have a strong positive correlation when analyzed by quality, except for 9 it is the strongest.

Density vs Fixed acidity

```
multi_(df$density, varName = 'Density (g/cm³)', df$fixed.acidity,
       varName1 = 'Fixed Acidity (g/dm³)')
```



Density vs Fixed acidity analyzed by quality show a strong positive linear correlation. We can see fixed acidity increases as density increases.

```
multi_2(varName = "density", varName1 = "fixed.acidity",
        varName2 = "Density", varName3 = "Fixed Acidity")
```

```
## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : span too small. fewer data values than degrees of freedom.
```

```
## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : pseudoinverse used at 0.98961
```

```
## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : neighborhood radius 0.00068675
```

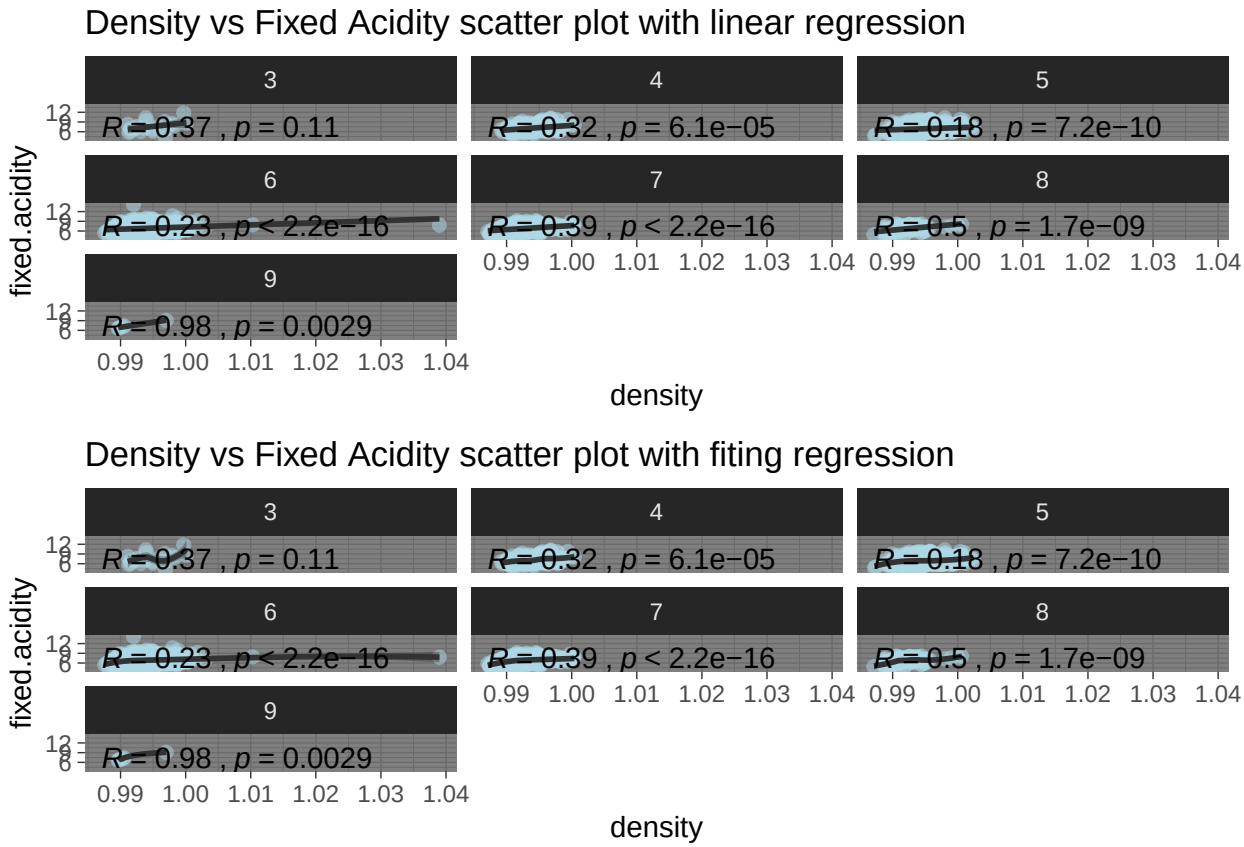
```
## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : reciprocal condition number 0
```

```
## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : There are other near singularities as well. 4.5384e-005
```

```
## Warning in predLoess(object$y, object$x, newx = if
## (is.null(newdata)) object$x else if (is.data.frame(newdata))
## as.matrix(model.frame(delete.response(terms(object)), : span too small.
## fewer data values than degrees of freedom.
```

```
## Warning in predLoess(object$y, object$x, newx = if
## (is.null(newdata)) object$x else if (is.data.frame(newdata))
## as.matrix(model.frame(delete.response(terms(object)), : pseudoinverse used
## at 0.98961
```

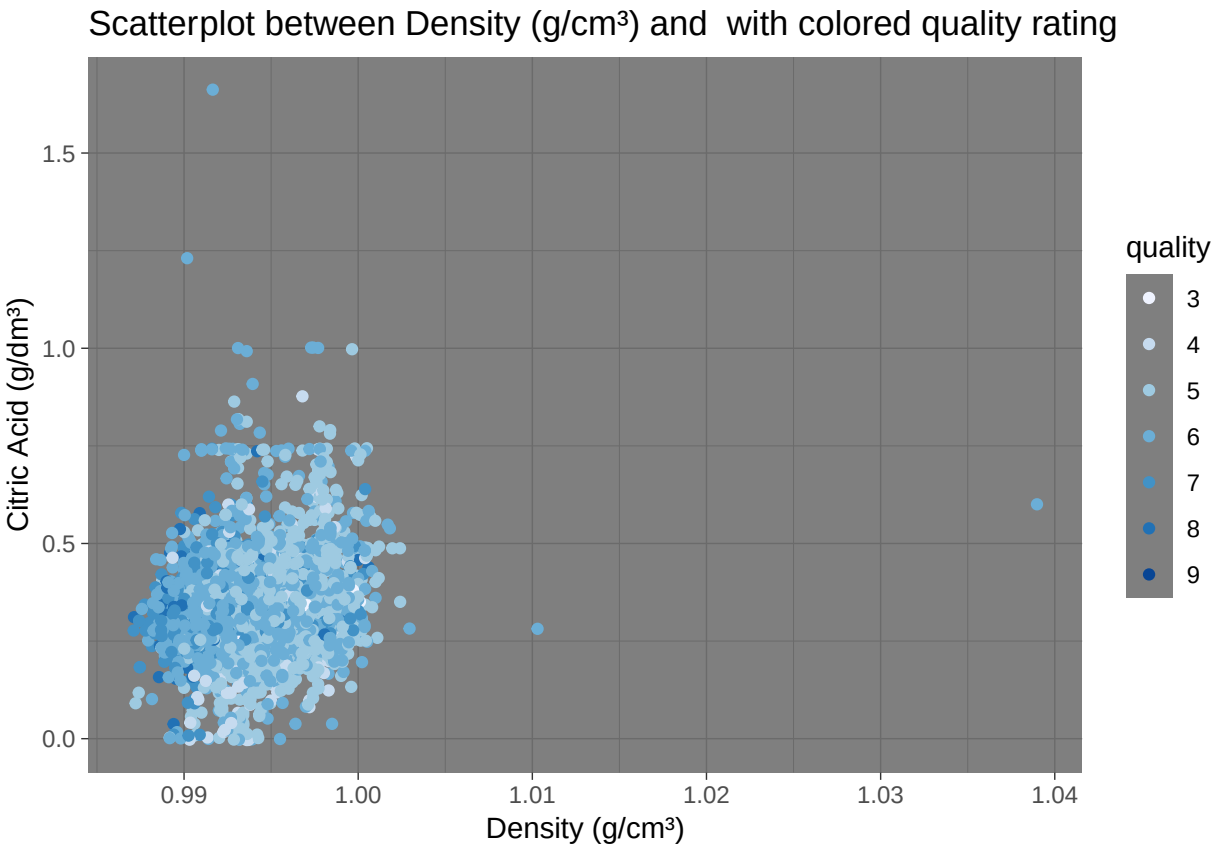
```
## Warning in predLoess(object$y, object$x, newx = if
## (is.null(newdata)) object$x else if (is.data.frame(newdata))
## as.matrix(model.frame(delete.response(terms(object)), : neighborhood radius
## 0.00068675

## Warning in predLoess(object$y, object$x, newx = if
## (is.null(newdata)) object$x else if (is.data.frame(newdata))
## as.matrix(model.frame(delete.response(terms(object)), : reciprocal
## condition number 0

## Warning in predLoess(object$y, object$x, newx = if
## (is.null(newdata)) object$x else if (is.data.frame(newdata))
## as.matrix(model.frame(delete.response(terms(object)), : There are other
## near singularities as well. 4.5384e-005
```



When we compare density vs fixed acidity with regression, the plots show a linear trend, and all scores have a weak positive correlation with the highlight being a score of 9 that has a 0.98 correlation.

Density vs Citric Acid

```
multi_(df$density, varName = 'Density (g/cm³)', df$citric.acid,
       varName1 = 'Citric Acid (g/dm³)')
```



Density vs Fixed acidity analyzed by quality shows a weak positive correlation and most of the samples appear less than 0.5 of citric acid. Moreover, both quality scores seem to be distributed around 0.99 to 1.0 of citric acid.

```
multi_2(varName = "density", varName1 = "citric.acid",
        varName2 = "Density", varName3 = "Citric Acid")
```

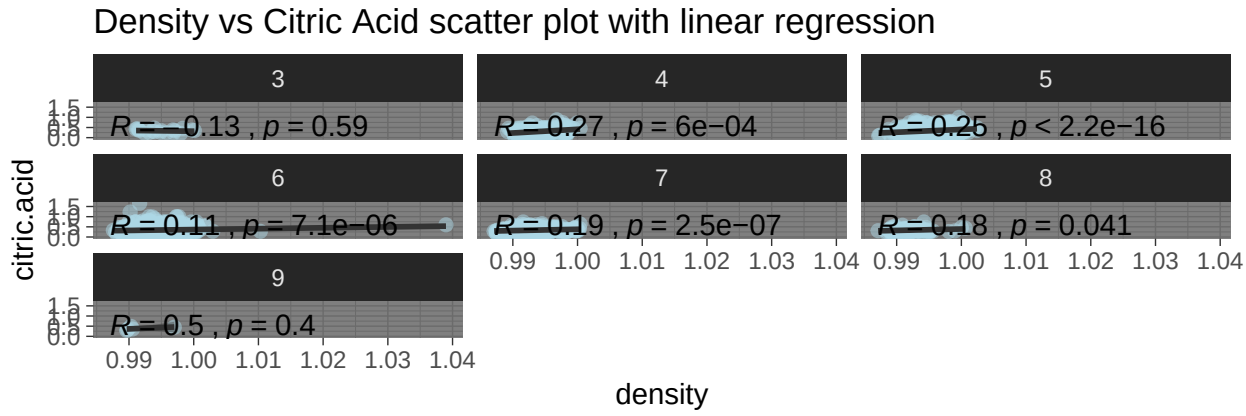
```
## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : span too small. fewer data values than degrees of freedom.

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : pseudoinverse used at 0.98961

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : neighborhood radius 0.00068675

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : reciprocal condition number 0

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : There are other near singularities as well. 4.5384e-005

## Warning in predLoess(object$y, object$x, newx = if
## (is.null(newdata)) object$x else if (is.data.frame(newdata))
## as.matrix(model.frame(delete.response(terms(object)), : span too small.
## fewer data values than degrees of freedom.

## Warning in predLoess(object$y, object$x, newx = if
## (is.null(newdata)) object$x else if (is.data.frame(newdata))
## as.matrix(model.frame(delete.response(terms(object)), : pseudoinverse used
```

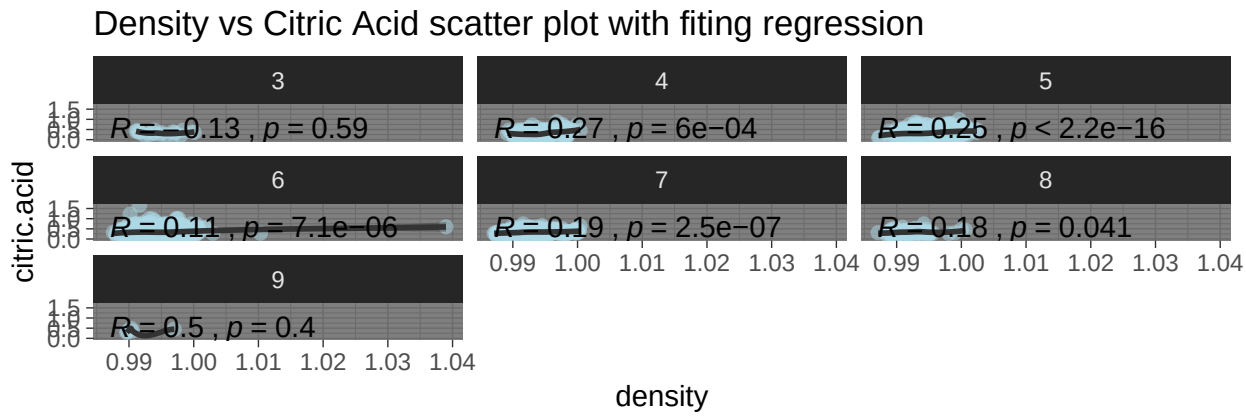
```
## at 0.98961
```

```
## Warning in predLoess(object$y, object$x, newx = if
## (is.null(newdata)) object$x else if (is.data.frame(newdata))
## as.matrix(model.frame(delete.response(terms(object)), : neighborhood radius
## 0.00068675
```

```
## Warning in predLoess(object$y, object$x, newx = if
## (is.null(newdata)) object$x else if (is.data.frame(newdata))
## as.matrix(model.frame(delete.response(terms(object)), : reciprocal
## condition number 0
```

```
## Warning in predLoess(object$y, object$x, newx = if
## (is.null(newdata)) object$x else if (is.data.frame(newdata))
## as.matrix(model.frame(delete.response(terms(object)), : There are other
## near singularities as well. 4.5384e-005
```

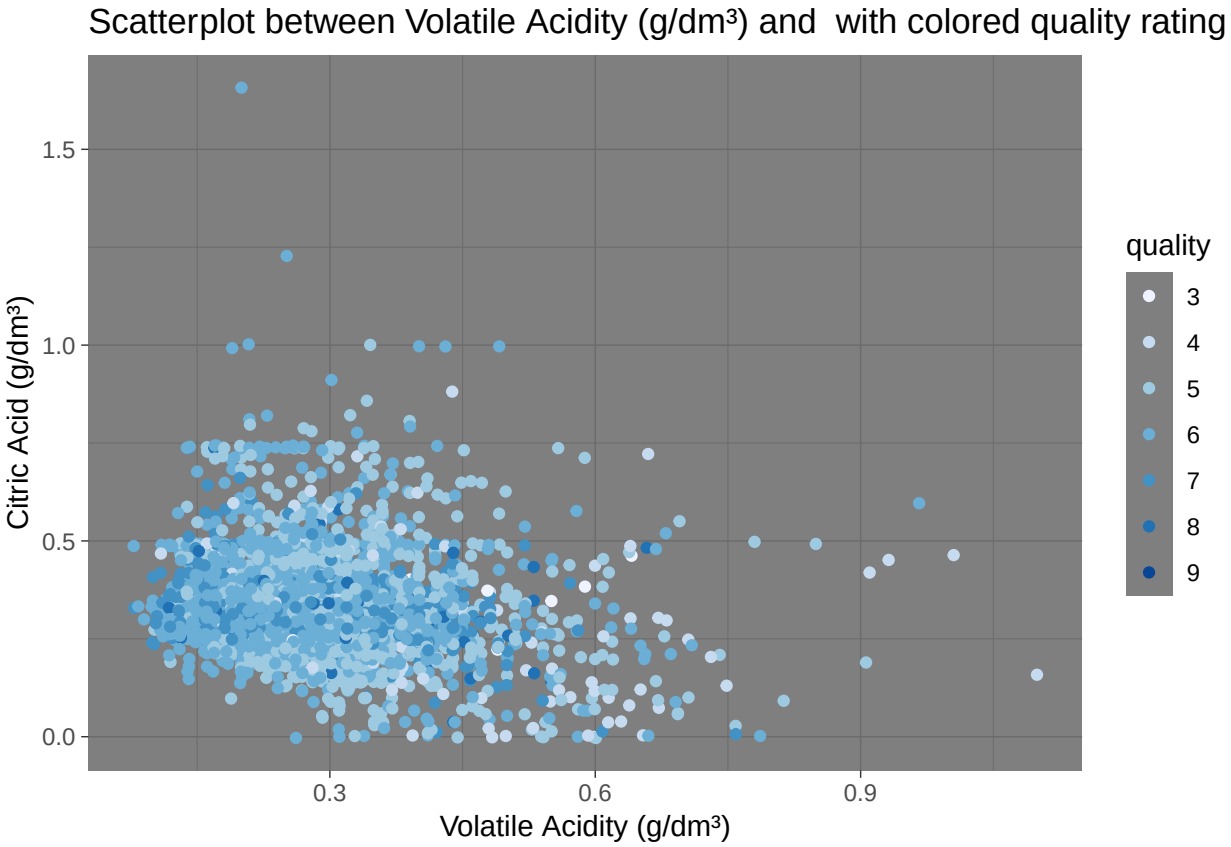




Comparing density VS citric acid analyzed for each quality scores separately, we can see a weak positive correlation for all of them, except for 9 it has a moderate positive correlation.

Volatile vs Citric Acid

```
multi_(df$volatile.acidity, varName = "Volatile Acidity (g/dm³)",
       df$citric.acid, varName1 = "Citric Acid (g/dm³)")
```



It seems to be that most of the observations are distributed in volite acidity between 0.15 and 0.45.

```
multi_2(varName = "volatile.acidity", varName1 = "citric.acid",
        varName2 = "Volatile Acidity", varName3 = "Citric Acid")
```

```
## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : span too small. fewer data values than degrees of freedom.
```

```
## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : pseudoinverse used at 0.2394
```

```
## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : neighborhood radius 0.0306
```

```
## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : reciprocal condition number 0
```

```
## Warning in simpleLoess(y, x, w, span, degree = degree, parametric =
## parametric, : There are other near singularities as well. 0.0082084
```

```
## Warning in predLoess(object$y, object$x, newx = if
## (is.null(newdata)) object$x else if (is.data.frame(newdata))
## as.matrix(model.frame(delete.response(terms(object)), : span too small.
## fewer data values than degrees of freedom.
```

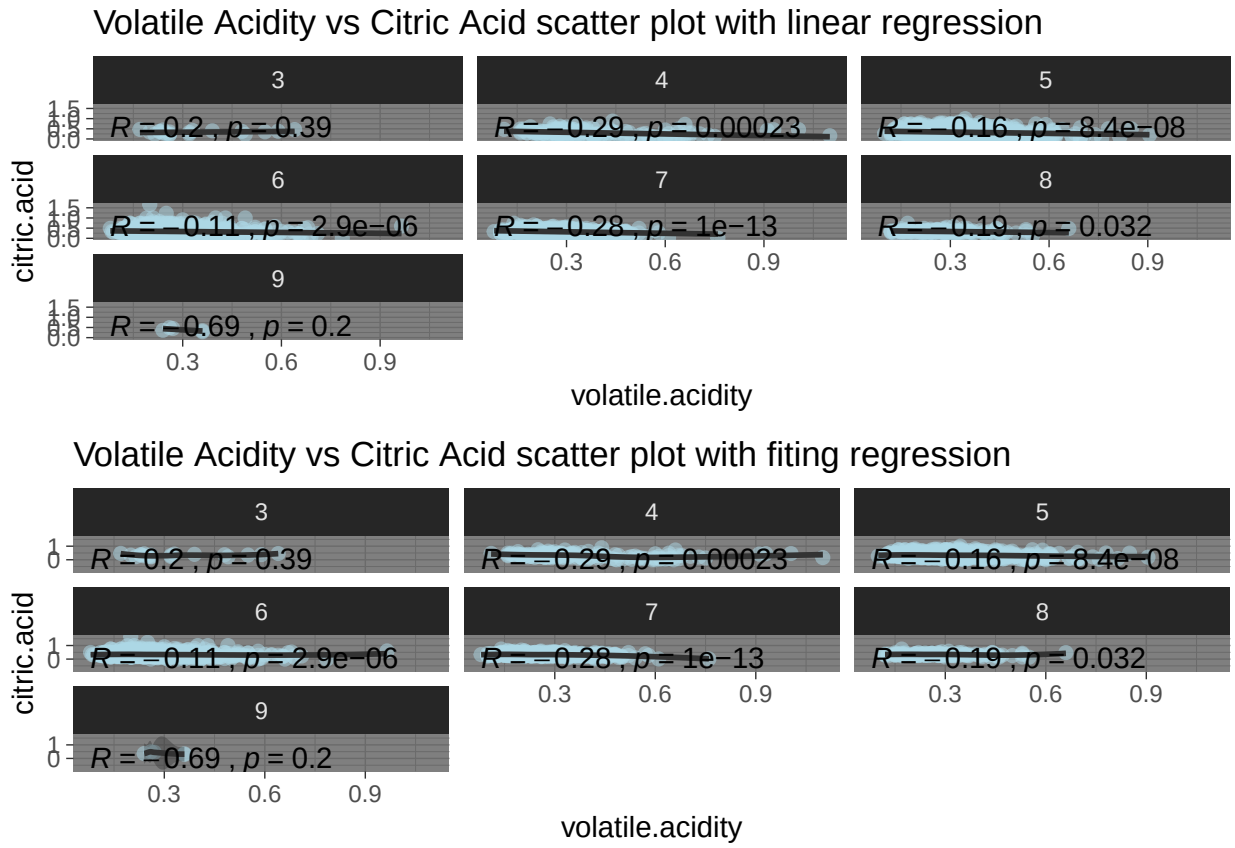
```
## Warning in predLoess(object$y, object$x, newx = if
## (is.null(newdata)) object$x else if (is.data.frame(newdata))
## as.matrix(model.frame(delete.response(terms(object)), : pseudoinverse used
## at 0.2394
```

```
## Warning in predLoess(object$y, object$x, newx = if
```

```
## (is.null(newdata)) object$x else if (is.data.frame(newdata))
## as.matrix(model.frame(delete.response(terms(object)), : neighborhood radius
## 0.0306
```

```
## Warning in predLoess(object$y, object$x, newx = if
## (is.null(newdata)) object$x else if (is.data.frame(newdata))
## as.matrix(model.frame(delete.response(terms(object)), : reciprocal
## condition number 0
```

```
## Warning in predLoess(object$y, object$x, newx = if
## (is.null(newdata)) object$x else if (is.data.frame(newdata))
## as.matrix(model.frame(delete.response(terms(object)), : There are other
## near singularities as well. 0.0082084
```



Volatile vs citric acid show a weak negative correlation, except for the highest quality score, which has a strong negative correlation, i.e. 0.69.

Multivariate Analysis Summary

There is a strong positive correlation between density vs fixed acidity, alcohol, and sugar, i.e. 0.98, 0.97 and 0.95 respectively.

For highest wine quality: - There is a strong negative correlation between Volatile vs citric acid, i.e. 0.69 - There is a moderate positive correlation between density vs citric acid, i.e. 0.5 - There is a strong positive correlation between density and sugar, i.e. 0.95

Model

There are 4 algorithm which will be used and evaluated, such as Classification and Regression Trees (CART). k-Nearest Neighbors (kNN). Support Vector Machines (SVM) with a linear kernel. Random Forest (RF)

The range of wine quality is 3-9, which will divide into three levels, [high, medium, low]. The validation set is about 20% from the dataset. We'll perform 10 cross validation on validation set.

```
#Classification model

#Use a copy of the original data set
wine <-read.csv('winequality-white.csv', sep=';')



wine$quality_levels <- cut(wine$quality, breaks = c(3, 4, 6, 9),
                           labels = c('low','medium', 'high'))


# create a list of 80% of the rows in the original dataset we can use for training
validation_index <- createDataPartition(wine$quality_levels,
                                        p = 0.80, list = FALSE)

# select 20% of the data for validation
validation <- wine[-validation_index,]

# use the remaining 80% of data to training and testing the models
wine1 <- wine[validation_index,]
str(wine)
```

```
## 'data.frame':    4898 obs. of  13 variables:
##  $ fixed.acidity       : num  7 6.3 8.1 7.2 7.2 8.1 6.2 7 6.3 8.1 ...
##  $ volatile.acidity    : num  0.27 0.3 0.28 0.23 0.23 0.28 0.32 0.27 0.3 0.22 ...
##  $ citric.acid         : num  0.36 0.34 0.4 0.32 0.32 0.4 0.16 0.36 0.34 0.43 ...
##  $ residual.sugar      : num  20.7 1.6 6.9 8.5 8.5 6.9 7 20.7 1.6 1.5 ...
##  $ chlorides           : num  0.045 0.049 0.05 0.058 0.058 0.05 0.045 0.045 0.049 0.044 ...
##  $ free.sulfur.dioxide : num  45 14 30 47 47 30 30 45 14 28 ...
##  $ total.sulfur.dioxide: num  170 132 97 186 186 97 136 170 132 129 ...
##  $ density             : num  1.001 0.994 0.995 0.996 0.996 ...
##  $ pH                  : num  3 3.3 3.26 3.19 3.19 3.26 3.18 3 3.3 3.22 ...
##  $ sulphates           : num  0.45 0.49 0.44 0.4 0.4 0.44 0.47 0.45 0.49 0.45 ...
##  $ alcohol             : num  8.8 9.5 10.1 9.9 9.9 10.1 9.6 8.8 9.5 11 ...
##  $ quality             : int  6 6 6 6 6 6 6 6 6 6 ...
##  $ quality_levels      : Factor w/ 3 levels "low","medium",..: 2 2 2 2 2 2 2 2 2 2 ...
```

```
#Active packages
library(memisc)
library(RColorBrewer)
library(caret)
library(ggpubr)
library(magrittr)
```

```
# Run algorithms using 10-fold cross validation
control <- trainControl(method ="cv", number = 10)
metric <- "Accuracy"
```

```
# a) nonlinear algorithms
```

```
# CART
fit.cart <- train(quality_levels ~., data = wine, metric = metric,
                  method="rpart", trControl = control,
                  na.action=na.exclude)

# kNN
fit.knn <- train(quality_levels~., data = wine, metric = metric,
                 method="knn", trControl = control,
                 na.action = na.exclude)


# b) advanced algorithms

# SVM
fit.svm <- train(quality_levels~., data = wine, method ="svmRadial",
                 trControl = control, na.action = na.exclude)
# Random Forest
fit.rf <- train(quality_levels~., data = wine, method ="rf",
                trControl=control, na.action = na.exclude)
```

Evaluate the Best Model

```
# summarize accuracy of models
results <- resamples(list(cart = fit.cart, knn = fit.knn,
                          svm = fit.svm, rf = fit.rf))
summary(results)
```

```
##
## Call:
## summary.resamples(object = results)
##
## Models: cart, knn, svm, rf
## Number of resamples: 10
##
## Accuracy
##           Min.   1st Qu.    Median      Mean   3rd Qu.      Max. NA's
## cart 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000    0
## knn  0.7663934 0.7790311 0.7905544 0.7898785 0.8004143 0.8135246    0
## svm  0.9958932 0.9958974 0.9969283 0.9973348 0.9979508 1.0000000    0
## rf   1.0000000 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000    0
##
## Kappa
##           Min.   1st Qu.    Median      Mean   3rd Qu.     Max. NA's
## cart 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000 1.000000    0
## knn  0.3243296 0.3702208 0.3879581 0.3881933 0.4042026 0.469096    0
## svm  0.9893847 0.9894156 0.9921341 0.9931396 0.9947466 1.000000    0
## rf   1.0000000 1.0000000 1.0000000 1.0000000 1.0000000 1.000000    0
```

CART and RF models achieve 100% accuracy. Now, I will summarize both models to figure out which one is the best.

Summarize the Best Model

```
# summarize Best Model
print(fit.cart)
```

```
## CART
##
## 4898 samples
##   12 predictor
##    3 classes: 'low', 'medium', 'high'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 4390, 4391, 4390, 4390, 4391, 4389, ...
## Resampling results across tuning parameters:
##
##   cp         Accuracy   Kappa
##   0.0000000  1.0000000  1.0000000
##   0.1332788  0.9895634  0.9715602
##   0.8667212  0.8143581  0.2715602
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was cp = 0.
```

```
print(fit.rf)
```

```
## Random Forest
##
## 4898 samples
##   12 predictor
##    3 classes: 'low', 'medium', 'high'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 4390, 4390, 4391, 4391, 4391, 4389, ...
## Resampling results across tuning parameters:
##
##   mtry  Accuracy   Kappa
##    2    0.9987688  0.99682
##    7    1.0000000  1.00000
##   12    1.0000000  1.00000
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was mtry = 7.
```

It seems that the random forest model was more accurate than the classification and regression trees model.

Generate Predictions

The RF was the most accurate model.Therefore, we'll apply it to the validation set, i.e. 20% from the wine dataset and used confusion matrix as metric evaluator.

```
# estimate skill of RF on the validation dataset
predictions <- predict(fit.rf, validation)
confusionMatrix(predictions, validation$quality_levels)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction low medium high
##     low     32      0    0
```

```
##     medium   0    731    0
##     high     0      0  212
##
## Overall Statistics
##
##                Accuracy : 1
##                  95% CI : (0.9962, 1)
##     No Information Rate : 0.7497
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 1
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: low Class: medium Class: high
## Sensitivity             1.00000        1.0000      1.0000
## Specificity             1.00000        1.0000      1.0000
## Pos Pred Value          1.00000        1.0000      1.0000
## Neg Pred Value          1.00000        1.0000      1.0000
## Prevalence              0.03282        0.7497      0.2174
## Detection Rate          0.03282        0.7497      0.2174
## Detection Prevalence    0.03282        0.7497      0.2174
## Balanced Accuracy       1.00000        1.0000      1.0000
```

```
dim(validation)
```

```
## [1] 979  13
```

Our model gets 100% accuracy in the test with the validation set. It is important to remember that the validation only contains a small part (20%) of our original dataset. This explains why our accuracy was so high. We need to test this model in other wine data sets to evaluate if it is a reliably accurate mode

```
predtb <- cbind.data.frame(predictions, validation$quality_levels)
colnames(predtb) <- c('predictions', 'actual')
predtb[1:30,]
```

```
##    predictions actual
## 1       medium medium
## 2       medium medium
## 3         high   high
## 4         high   high
## 5       medium medium
## 6         high   high
## 7       medium medium
## 8       medium medium
## 9       medium medium
## 10      medium medium
## 11      medium medium
## 12      medium medium
## 13        high   high
## 14        high   high
## 15      medium medium
## 16        high   high
## 17      medium medium
## 18      medium medium
```

```
## 19      medium medium
## 20      medium medium
## 21      medium medium
## 22        high   high
## 23      medium medium
## 24      medium medium
## 25      medium medium
## 26      medium medium
## 27        high   high
## 28      medium medium
## 29      medium medium
## 30        high   high
```

Conclusions

The white wine data set contains information on almost 4698 white wine samples across 12 chemical properties. Descriptive statistics using a histogram, histogram log10, density and box plots is used to find the distribution for each separated variable. Statistics computation was calculated such as mean, median, max, etc. The mean was not totally reliable in a few attributes as sugar and chlorides. These attributes had a significant differenctrom the median, and this situation is common for outliers. Therefore, the median is used to gather the main characteristics for the quality levels.

For highest wine quality: - High level of alcohol, citric acid, and pH. - Low level of density, sulfur dioxide and sugar. - Medium level of fixed acidity and volatile acidity

Correlation matrix, correlation network, boxplot, scatter plot and scatter plot with fitting regression were used to examine the correlation between these properties.

The quality variable had a meaningful correlation with alcohol and volatile acidity. There is a positive trend between quality and alcohol, so the level of alcohol increases with a better quality ranking. The opposite occurs with quality and volatile acidity, and the level of volatile acidity decreases with a better quality ranking. This completely makes sense because high levels of volatile acidity lead to an unpleasant, vinegar taste.

Besides that, we saw how alcohol and volatile acidity (the attributes correlated with quality) were influenced or influence other variables, such as density.

In the multivariate analysis, we analyzed alcohol vs density, volatile acidity vs citric acid and alcohol vs volatile acidity interpreted by quality with scatter plot and regression model. We found that there is a strong positive correlation between density vs fixed acidity, alcohol, and sugar, i.e. 0.98, 0.97 and 0.95 respectively. Weak positive correlation with citric acid and fixed acid. While there's a strong negative correlation between density and alcohol

Finally, four classification models to predict whether wine is high, medium or low quality. The best model was developed with random forest method, this model achieves 100% accuracy in our test with the validation data set. As our validation data set contains only 20% observations, but we strongly recommend testing this model in other wine data sets to evaluate if it is a reliable model.