Università degli Studi di Padova

Statistical Learning Project

White Wine Quality Analysis

Anna Del Savio, 2097098
Francesco Tomaselli, 2089207
Inês Jesus, 2073570

Department of Mathematics

June 27, 2023

Table of Contents

1 Introduction

In this report, a ”White Wine Quality” dataset (available here) is analysed to determine if it is possible to predict the quality of white wine based on its physicochemical properties and, of these, which are the most influential. This can allow to improve wine production and selling and help with wine quality certification.

The data, collected from May/2004 to February/2007, has 4898 samples and 12 features. The variables are mostly physicochemical but also sensory (quality of the wine) and are all commonly used in wine testing.

Below is reported the list of the available features in the dataset.

1. *quality*: score (discrete) from 0 (very bad) to 10 (excellent) of wine quality, obtained from the median of the assessments of at least three wine experts, using blind tests. This is the variable of interest for this report.
2. *fixed acidity*: continuous variable which indicates the concentration, in g/dm^3, of tartaric acid in the wine, the most important fixed acid (acids that don’t evaporate readily) from a winemaking perspective, maintaining its chemical stability and its color.
3. *volatile acidity*: continuous variable which indicates the concentration, in g/dm^3, of acetic acid in the wine, the primary volatile acid in wine, unpleasant when in high levels.
4. *citric acid*: continuous variable which indicates the concentration, in g/dm^3, of citric acid in the wine, the principal fixed acid responsible for its freshness.
5. *residual sugar*: continuous variable which indicates the concentration, in g/dm^3, of natural grape sugar left in the wine after alcoholic fermentation, also referred to as its sweetness.
6. *chlorides*: continuous variable which indicates the concentration, in g/dm^3, of sodium chloride in the wine, contributing to its saltiness.
7. *free sulfur dioxide*: continuous variable which indicates the concentration, in mg/dm^3, of free (unbound) sulfur dioxide ions, or sulfites, in the wine.
8. *total sulfur dioxide*: continuous variable which indicates the concentration, in mg/dm^3, of sulfur dioxide ions, whether free or bound, in the wine. This chemical is added during the wine’s production to protect it from oxidation and bacterial spoilage.
9. *density*: continuous variable which indicates the density, in g/dm^3, of the wine, determined mainly by the concentration of alcohol and residual sugar.
10. *pH*: continuous variable, ranging from 0 to 14, measuring the pH of the wine.
11. *sulphates*: continuous variable which indicates the concentration, in g/dm^3, of potassium sulphate in the wine.
12. *alcohol*: continuous variable which indicates the alcohol by volume in the wine, in % *vol*.

Throughout this report, for every statistical test, we consider a significance level of 5%.

1.1 Dataset Preprocessing

First, the necessary libraries are imported and then the dataset is read and cleaned.

```r
# Importing the necessary libraries
library(ggplot2)
library(MASS)
library(glmnet)
library(pROC)
library(caret)
library(corrplot)
library(tidyverse)
library(e1071)
```

```r
# Reading the dataset
data <- read.csv2("winequality-white.csv")

colnames(data) <- c("fixed_acidity", "volatile_acidity", "citric_acid",
                    "residual_sugar", "chlorides", "free_sulfur_dioxide",
                    "total_sulfur_dioxide", "density", "pH",
                    "sulphates", "alcohol", "quality")
```

Now, we're ready to start preparing our data. We start by checking if there are any missing values and then transforming the features to appropriate datatypes, that is, **factor** for *quality* and **numeric** for the remaining variables.

```r
# Check if there are missing values (returns TRUE if there are)
anyNA(data)

## [1] FALSE

# Changing independent variables type to numeric
for (i in 1:12) {
  data[,i]<-as.numeric(data[,i]) }

# Changing quality variable type to factor
data$quality <- as.factor(data$quality)
```

Let's see what we get from these transformations.

```r
summary(data)

##  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.: 1.700
##  Median : 6.800   Median :0.2600   Median :0.3200   Median : 5.200
##  Mean   : 6.855   Mean   :0.2782   Mean   :0.3342   Mean   : 6.391
##  3rd Qu.: 7.300   3rd Qu.:0.3200   3rd Qu.:0.3900   3rd Qu.: 9.900
##  Max.   :14.200   Max.   :1.1000   Max.   :1.6600   Max.   :65.800
##
```

```
##    chlorides       free_sulfur_dioxide total_sulfur_dioxide    density
##  Min.   :0.00900   Min.   :  2.00      Min.   :  9.0        Min.   :0.9871
##  1st Qu.:0.03600   1st Qu.: 23.00      1st Qu.:108.0        1st Qu.:0.9917
##  Median :0.04300   Median : 34.00      Median :134.0        Median :0.9937
##  Mean   :0.04577   Mean   : 35.31      Mean   :138.4        Mean   :0.9940
##  3rd Qu.:0.05000   3rd Qu.: 46.00      3rd Qu.:167.0        3rd Qu.:0.9961
##  Max.   :0.34600   Max.   :289.00      Max.   :440.0        Max.   :1.0390
##
##        pH          sulphates         alcohol      quality
##  Min.   :2.720   Min.   :0.2200   Min.   : 8.00   3:  20
##  1st Qu.:3.090   1st Qu.:0.4100   1st Qu.: 9.50   4: 163
##  Median :3.180   Median :0.4700   Median :10.40   5:1457
##  Mean   :3.188   Mean   :0.4898   Mean   :10.51   6:2198
##  3rd Qu.:3.280   3rd Qu.:0.5500   3rd Qu.:11.40   7: 880
##  Max.   :3.820   Max.   :1.0800   Max.   :14.20   8: 175
##                                                   9:   5
```

```
# Calculates the standard deviation for every independent value
sd<-c()
for(i in 1:11){
  sd[i] = sd(data[,i])
}

data.frame(
  "variable" = names(data[,-12]),
  "sd" = sd
)
```

```
##                variable           sd
## 1         fixed_acidity  0.843868228
## 2      volatile_acidity  0.100794548
## 3           citric_acid  0.121019804
## 4        residual_sugar  5.072057784
## 5             chlorides  0.021847968
## 6   free_sulfur_dioxide 17.007137325
## 7  total_sulfur_dioxide 42.498064554
## 8               density  0.002990907
## 9                    pH  0.151000600
## 10            sulphates  0.114125834
## 11              alcohol  1.230620568
```

From the code above, we can see some of the descriptive statistics of the variables. Regarding the *total sulfur dioxide* and *free sulfur dioxide*, we witness a larger standard deviation, specially in the first. This is explained by the concentration units used to obtain these variables, which is mg/dm^3, while the rest of concentration-related variables are represented in g/dm^3.

2 Exploratory Data Analysis

2.1 Univariate exploratory analysis

We start by plotting the histogram of the response variable *quality*.

```
table(data$quality)
```

```
##
##    3    4    5    6    7    8    9
##   20  163 1457 2198  880  175    5
```

```
n <- length(data$quality)
d_q <- data.frame(  Quality <- 3:9,
  Freq <- c(20,  163, 1457, 2198,  880,  175,    5))
ggplot(data=d_q, aes(x=Quality, y=Freq))+
  geom_bar(stat="identity", fill="yellow3")+
  geom_text(aes(label=round(Freq/n*100,2)), vjust=-0.3, size=3.5)+
  theme_minimal()+  scale_x_discrete(limits=Quality)+
  ggtitle(label = "Barplot of quality")
```

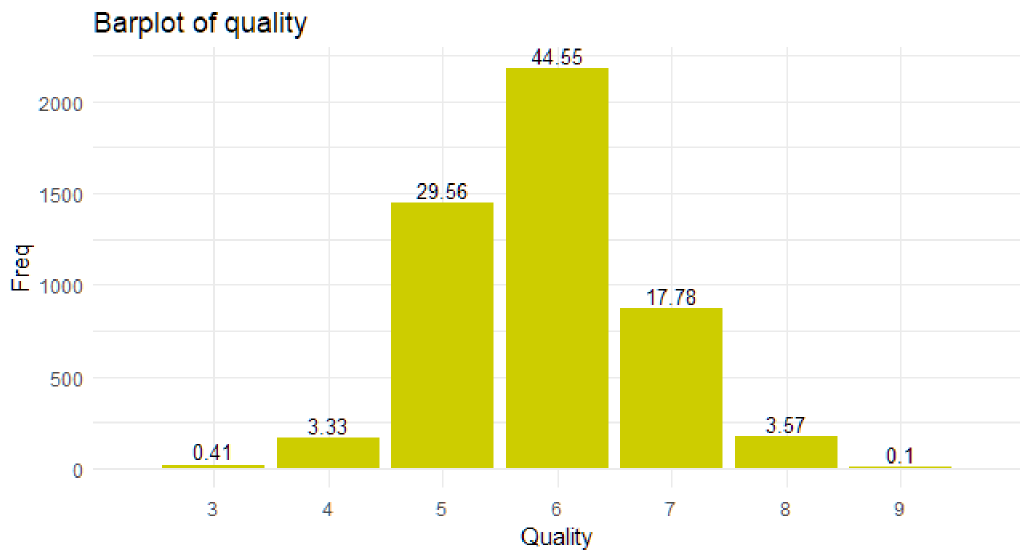


Figure 2.1: Histogram of the quality feature.

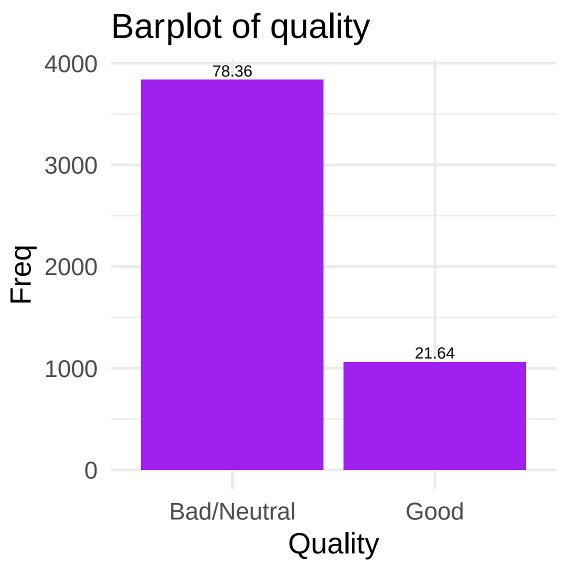
Wine Quality	Absolute Frequency	Percentage Frequency
3	20	0.41%
4	163	3.33%
5	1457	29.56%
6	2198	44.55%
7	880	17.78%
8	175	3.57%
9	5	0.1%
Total	4898	100%

Table 2.1: Frequency table of quality wine.

Since we want to perform binary classification and also to identify the variables that give quality to the wine, we split this variable in two classes:

- with a score from 3 to 6, the wine sample will be labeled as 0, meaning ”bad/neutral” quality;
- with a score above 6, the wine sample will be labeled as 1, meaning ”good” quality”.

```
# Transformation of the quality variable
data$quality <- ifelse(data$quality >= 7, 1, 0)

table(data$quality)

##
##    0    1
## 3838 1060

n <- length(data$quality)
d_q_bin <- data.frame(  Quality <- c("Bad/Neutral","Good"),
  Freq <- c(3838, 1060))
ggplot(data=d_q_bin, aes(x=Quality, y=Freq))+
  geom_bar(stat="identity", fill="purple")+
  geom_text(aes(label=round(Freq/n*100,2)), vjust=-0.3, size=2)+
  theme_minimal()+  scale_x_discrete(limits=Quality)+
  ggtitle(label = "Barplot of quality")
```



We can see that now we have 3838 observations in class 0 and 1060 in class 1: for each good wine we have 3.62 bad/neutral wine and thus we can say our dataset in imbalanced.

Quality Wine	Absolute Frequency	Percentage Frequency
0 (bad/neutral quality)	3838	78.36%
1 (good quality)	1060	21.64%
Total	4898	100%

Table 2.2: Caption

Regarding the evaluation metric, we will not be able to use the accuracy score, because this is not a proper measure when dealing with imbalanced data, so we will use specificity, sensitivity and the AUC.

Now, we wish to analyse the univariate density plots and boxplots.

```r
# Histogram of quality variable
par(mfrow=c(1,2))
for (i in 1:11){
    hist(data[,i],
         main=paste(" Histogram of ", colnames(data)[i]),
         col="pink", xlab="", prob=T)
    legend("topright",legend=c("Median","Mean","Density"), lty=c(2,2,1),
           lwd=2, col=c("black","blue","red"), cex=0.7)
    abline(v=median(data[,i]), lwd=3, lty=2, col="black")
    abline(v=mean(data[,i]), lwd=3, lty=2, col="blue")
    lines(density(data[,i]), col="red", lwd=2)
    boxplot(data[,i],
            main=paste(" Boxplot of ", colnames(data)[i]), col ="pink")
}
```

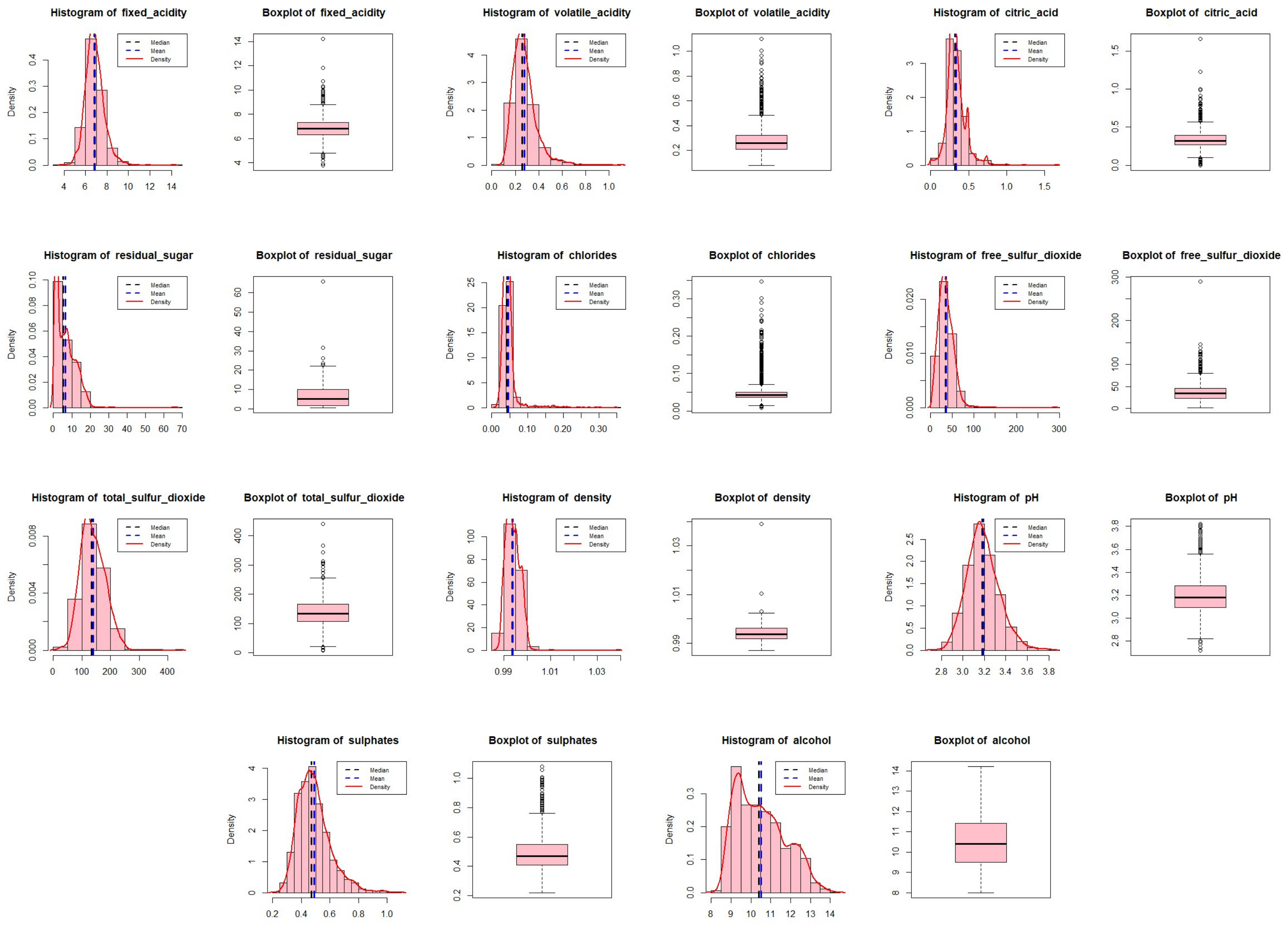


Figure 2.2: Density plots and box plots of the independent variables

From fig. 2.2, we observe that all of the variables are unimodal. With the Shapiro-Wilk test, we see that the variables do not follow a normal distribution. However, since we have a high number of samples, we are able to perform a *t test*, from which we learn that the mean and median are statistically different for all the variables.

```r
# Example of Shapiro-Wilk and t tests for the variable 'fixed_acidity'

shapiro.test(data$fixed_acidity)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  data$fixed_acidity
## W = 0.97656, p-value < 2.2e-16
```

```r
med <- median(data$fixed_acidity)
t.test(data$fixed_acidity, mu= med)
```

```
##
##  One Sample t-test
##
## data:  data$fixed_acidity
## t = 4.5438, df = 4897, p-value = 5.658e-06
## alternative hypothesis: true mean is not equal to 6.8
## 95 percent confidence interval:
##  6.831149 6.878426
## sample estimates:
## mean of x
##  6.854788
```

Furthermore, it is possible to see by the naked eye that the plots are right-skewed. In fact, calculating the asymmetry indexes following this formula:

$$\gamma_1 = \frac{1}{n}\sum_{i=1}^{n}\left(\frac{x_i - \bar{x}}{\hat{\sigma}}\right)^3,$$

we get positive values for all variables, meaning they are, some more than others, positive asymmetric.

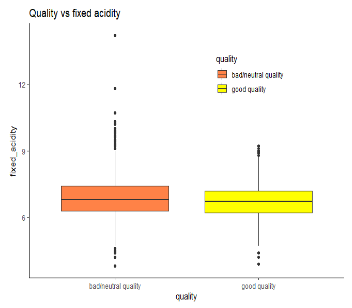
```r
skew = function(x){
  n = length(x)
  s3 = sqrt(var(x)*(n-1)/n)^3
  mx= mean(x)
  sk = sum((x-mx)^3)/s3
  sk/n
}

# Example for the variable 'fixed_acidity'
skew(data$fixed_acidity)
```

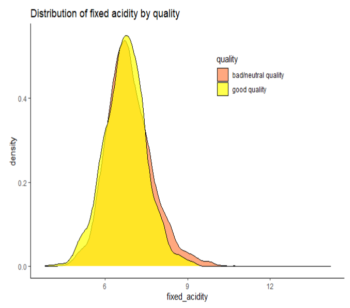
```
## [1] 0.6475531
```

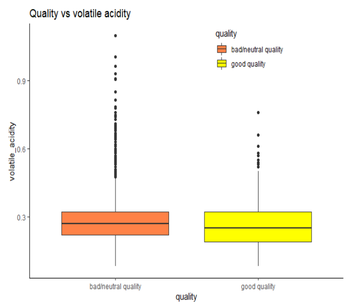
Moreover, from the boxplot figures, we notice that all variables with the exception of *alcohol* have many outliers.

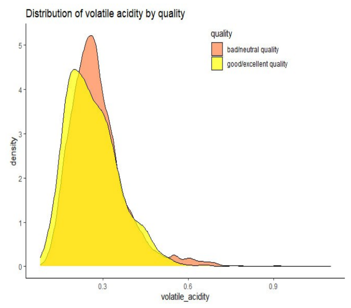
2.2 Bivariate exploratory analysis

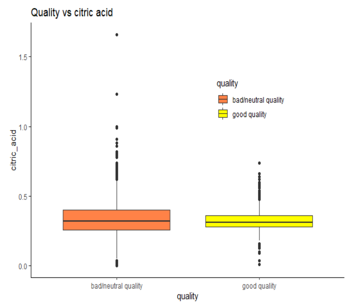
```
# Example for the variable 'fixed_acidity'

p<-ggplot(data, aes(x=quality, y=fixed_acidity, fill=quality)) +
  geom_boxplot() +
  scale_fill_manual(values = c("0" = "sienna1", "1" = "yellow1"),
                    labels = c("bad/neutral quality", "good quality")) +
  scale_x_discrete(labels = c("bad/neutral quality", "good quality")) +
  labs(fill = "quality", title = "Quality vs fixed acidity")
p + theme_classic() + theme(legend.position = c(0.7,0.8))
cols <- c("sienna1", "yellow1")

ggplot(data, aes(x = fixed_acidity, fill = quality)) +
  geom_density(alpha = 0.7) +
  scale_fill_manual(values = cols,
                    labels = c("bad/neutral quality", "good quality")) +
  theme_classic() +
  labs(title = "Distribution of fixed acidity by quality") +
  theme(legend.position = c(0.7,0.8))
```

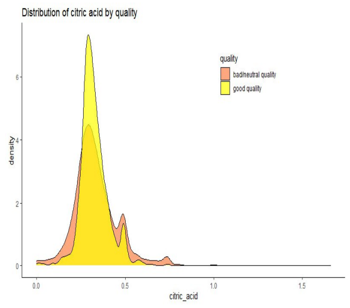


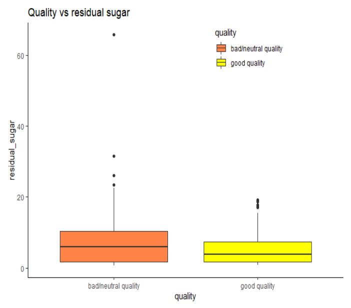


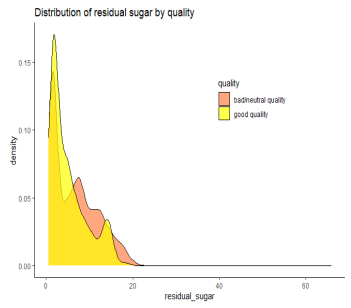












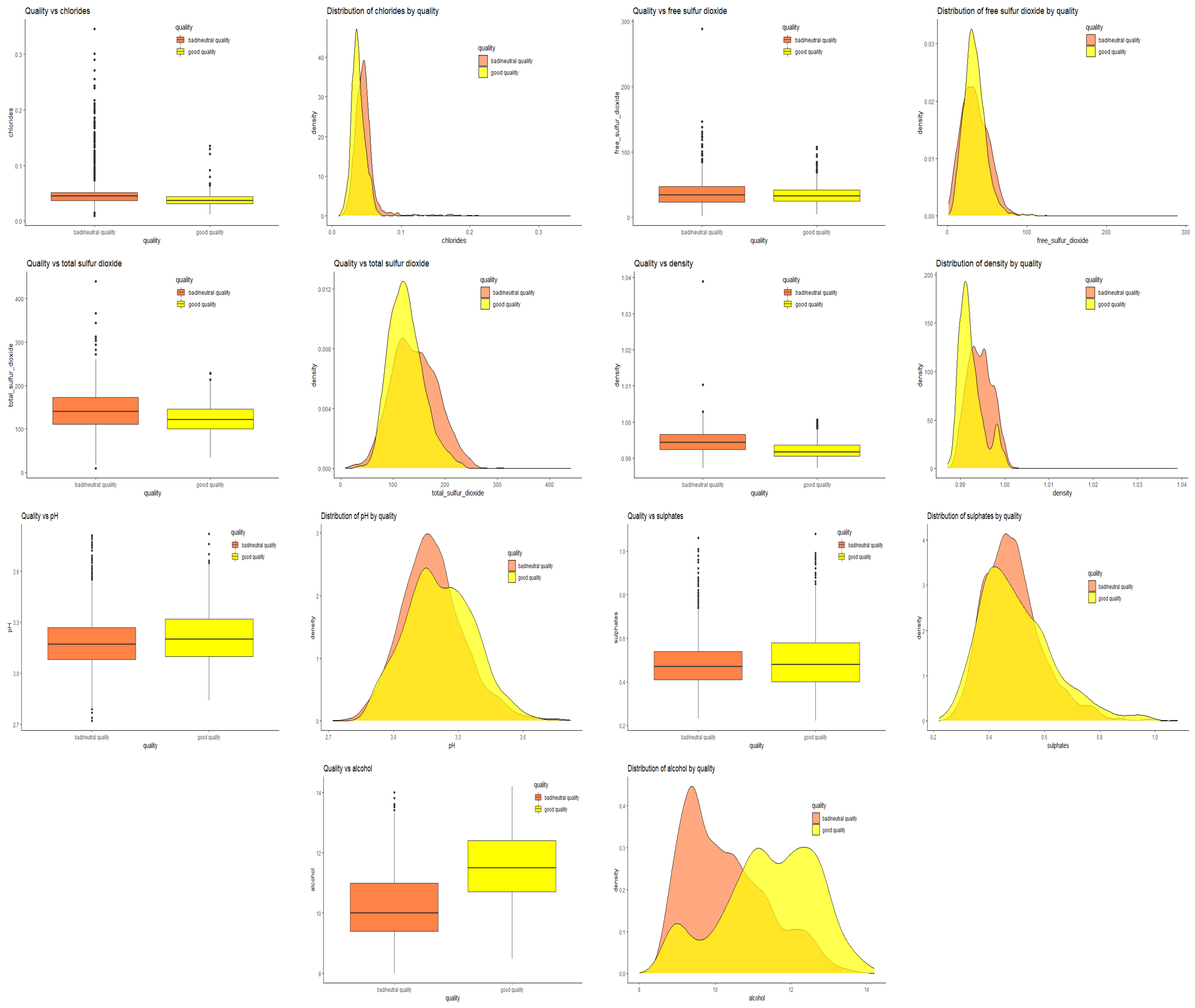


Figure 2.3: Density plots and box plots of the independent variables separated by the quality variable

First of all we can test the normality of the variable divided by quality with the Shapiro-Wilk tests.

```r
# Shapiro-Wilk test example for the variable 'fixed_acidity' and class 1 of quality
shapiro.test(data$fixed_acidity[data$quality=='1'])
```

```
##
##  Shapiro-Wilk normality test
##
## data:  data$fixed_acidity[data$quality == "1"]
## W = 0.99485, p-value = 0.001121
```

```r
# Shapiro-Wilk test example for the variable 'fixed_acidity' and class 0 of quality
shapiro.test(data$fixed_acidity[data$quality=='0'])
```

```
##
##  Shapiro-Wilk normality test
##
## data:  data$fixed_acidity[data$quality == "0"]
## W = 0.97126, p-value < 2.2e-16
```

From the results we conclude that, even when divided by quality, none of the variables follow a normal distribution.

From looking at fig. 2.3, it is possible to observe that we can only see a very small difference in the mean between good and bad/neutral quality samples, except in the case of the variables *alcohol*, *density* and *total sulphur dioxide*, where this difference is more notable.

This might mean that the latter variables, when alone, have a greater impact on the response variable.

In particular, we notice that for *pH*, *sulphates* and *alcohol*, the mean is slightly higher for the good quality class, while for the rest of the variables the case is the opposite.

It is very evident that the more alcohol is present in the wine, the better should be the quality.

We can test the differences between the means by the *t test*, applicable to our case due to the large number of samples.

```r
# t test example for the variable 'free_sulfur_dioxide' split by quality
x1 <-data$free_sulfur_dioxide[data$quality=='0']
x2 <- data$free_sulfur_dioxide[data$quality=='1']
t.test(x1,x2)
```

```
##
##  Welch Two Sample t-test
##
## data:  x1 and x2
## t = 1.8888, df = 2130.5, p-value = 0.05905
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -0.03699207  1.97070214
## sample estimates:
```

```
## mean of x mean of y
##  35.51733  34.55047
```

From the *t tests* we see that only for the variable *free sulfur dioxide* we can accept the null hypothesis of equality of means, meaning that the variable alone doesn't have effects on the response. In the other cases we reject it meaning that in the two class of *quality* there are differences in the mean.

Besides that, we can see that generally we have more outliers for the class of bad/neutral quality wine. This is interesting as it might signify a more volatile physicochemical behaviour of bad/neutral quality wine.

From the pairs plot (fig. 2.4) and the correlation plot (fig. 2.5), it will be possible to understand if some variables are correlated.

```
# Pairs plot
l <- quality == '1'
pairs(data[,-12], col=l+1, pch=l*15+1)
```

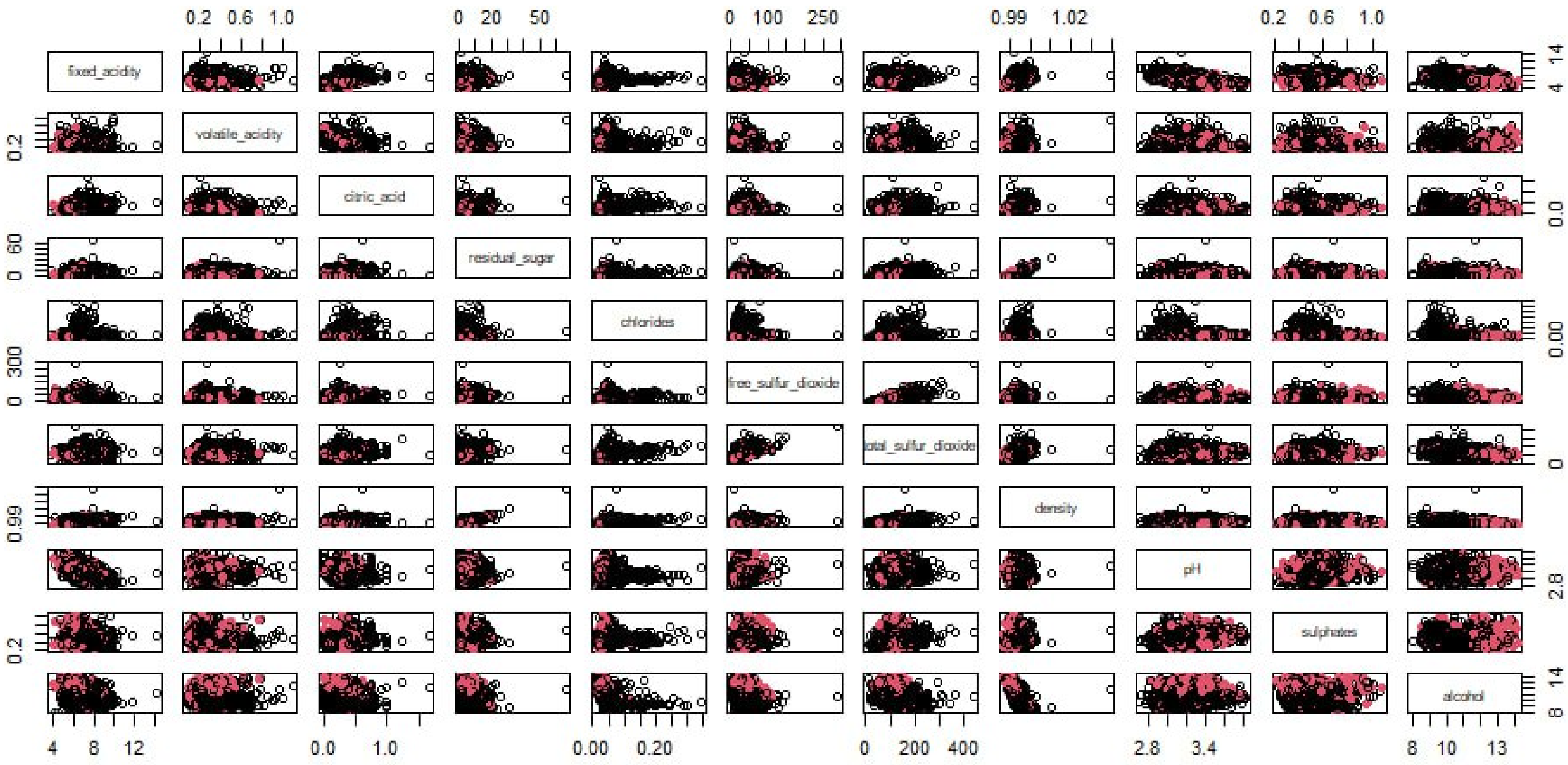


Figure 2.4: Pairs plot of the independent variables where red points represent the good quality class and the black circles the bad/neutral quality class.

```
# Correlation plot
corrplot(
  cor(data[,c(-8,-12)]),
  method = "square",
  type = "upper",
  tl.col = "black",
  tl.cex = 0.7
  )
```

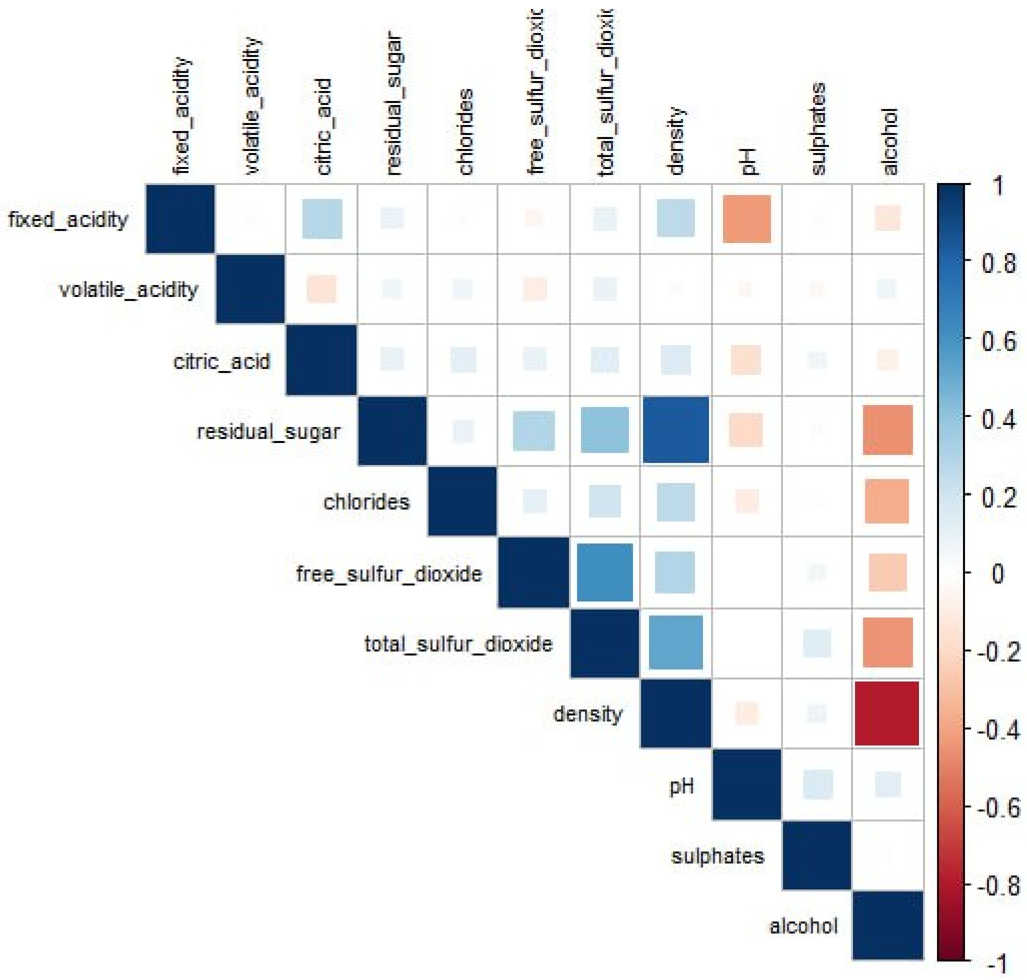


Figure 2.5: Correlation matrix of the independent variables

In fig. 2.5 it is evident that variables *density* and *alcohol* are negatively correlated, with a linear correlation value of around -0.8. In fact, this correlation is also noticeable in fig. 2.4 by the distribution of the points in an almost straight line. Furthermore, we observe a very high positive correlation between *density* and *residual sugar*, with a correlation value of around 0.8.

We are in presence of multicollinearity, and so we decide to remove the variable *density* before our models, because we know that it violates the assumption of independence between variables and, in general, it undermines the statistical significance of an independent variable.

Moreover, in the pairs plot, it is once again visible that the *alcohol* variable is able to mildly separate the two different classes, represented by different colors.
At the end, we add a test for the independence of the features, obtained by Chi-squared test.

```r
Pvalue_corr_matrix <- matrix(rep(0, 11*11), nrow=11)

for(i in 1:11){
  for(j in 1:11){
    if(i==j){
      Pvalue_corr_matrix[i,j]=0
    }
    else{
      Pvalue_corr_matrix[i,j] = cor.test(data[,i], data[,j],
                                           exact = F)$p.value
    }
  }
}
round(Pvalue_corr_matrix,3)
```

	fixed ac.	volatile ac.	citric ac.	res. sugar	chlorides	free s.d.	total s.d.	density	pH	sulphates	alcohol
fixed ac.	0	0.112	0	0	0.106	0.001	0	0	0	0.230	0
volatile ac.	0.112	0	0	0	0	0	0	0.058	0.026	0.012	0
citric ac.	0	0	0	0	0	0	0	0	0	0	0
res. sugar	0	0	0	0	0	0	0	0	0	0.062	0
chlorides	0.106	0	0	0	0	0	0	0	0	0.241	0
free s.d.	0.001	0	0	0	0	0	0	0	0.966	0	0
total s.d.	0	0	0	0	0	0	0	0	0.871	0	0
density	0	0.058	0	0	0	0	0	0	0	0	0
pH	0	0.026	0	0	0	0.966	0.871	0	0	0	0
sulphates	0.230	0.012	0	0.062	0.241	0	0	0	0	0	0.223
alcohol	0	0	0	0	0	0	0	0	0	0.223	0

Table 2.3: Chi-squared test for the independence of the features

From table 2.3 we can see that many correlation are significantly different from zero for a significant level of 95% (*p-value* equal to zero).

3 Modeling

In this section we are going to perform classification and multivariate analysis considering some models. Since the response variable is a binary feature, we are facing a binary classification problem, for which we can use many models. Below are reported the ones we chose to apply.

1. K-Nearest Neighbors (K-NN): non-parametric model that uses feature similarity to make classifications of the data; in other terms, we want to investigate if the examples with similar wine features have similar wine quality.
2. Naive Bayes Classifier: is an alternative model for classification problem based on Bayes theorem, which, under some assumptions, can be a good substitute of logistic regression. Anyway, in section 2.2 we have showed that there is a significant correlation between the qualitative features, while the independence of features is a requirement of this model; for this reason, we are going to carefully apply Naive Bayes Classifier.
3. Logistic Regression: it is a method where the fixed independent features estimates the probability to obtain a class of the response feature, in our case "good quality wine". The main advantage of this model is the parameters interpretability.
4. Penalised Logistic Regression: this model is a variation of logistic regression, where there is an hyperparameter adding. The hyperparameter regularizes the importance of the parameters, preventing overfitting and underfitting problems. We use two types of regularization, which are the Lasso Regression and the Ridge Regression.

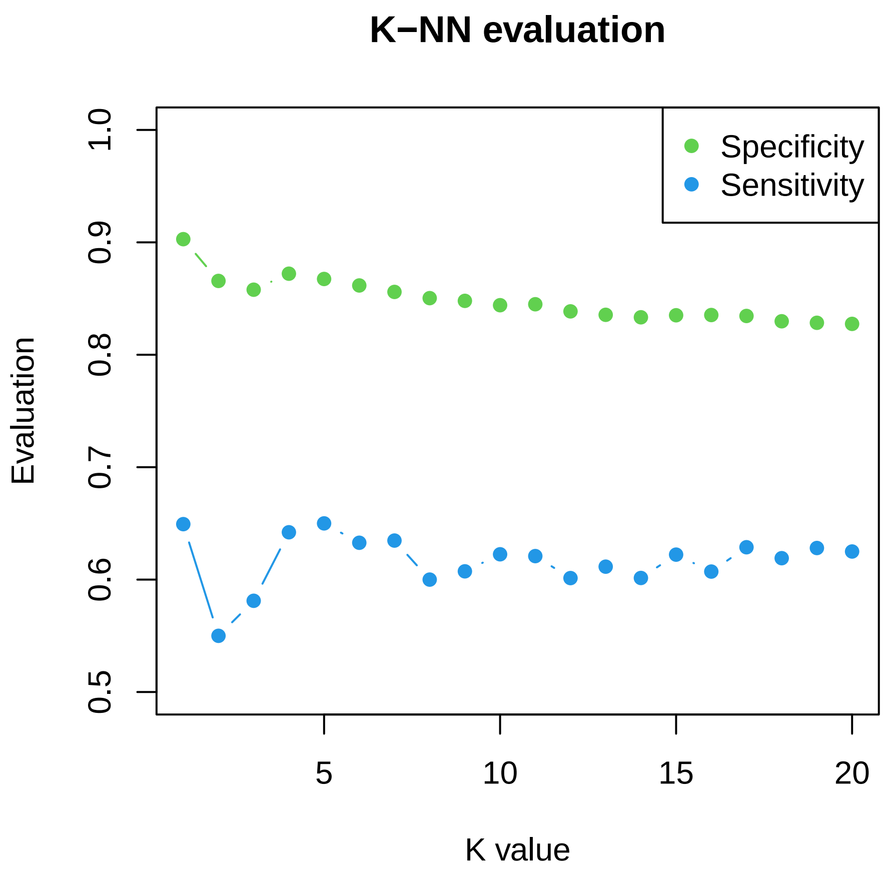
We also considered the Linear Discriminant Analysis and Quadratic Discriminant Analysis models, alternative methods for classification where, under some assumptions, the classification of the example can be more accurate than the logistic regression. Anyway, in section 2.2, we have showed that the explanatory features split by the wine quality are not normally distributed; for this reason, we can not apply either.
Keep in mind that we evaluate our model using AUC for a general model evaluation, specificity and sensibility to detect the model capacity to discriminate between high and low quality wine.

3.1 K-Nearest Neighbors

To apply this method, we first need to scale our feature, since K-NN is a distance based algorithm and the feature scales can influence the "importance" of the features in the algorithm.
Moreover, since the goal of the report is explaining high quality wine in function of wine features, we set the high quality wine as positive and low quality wine as negative. In the following code are shown the steps to implement the K-NN algorithm for different hyperparameter values. For the evaluation metric, we'll use sensitivity and specificity, since the imbalance of wine quality returns untrustable accuracy.

```r
set.seed(123)
data$quality<-as.numeric(data$quality)
```

```r
data_norm <- scale(data, center = T, scale = T)
data$quality<-as.factor(data$quality)
data_norm[,12]<-as.factor(data_norm[,12])

sample <- sample(c(TRUE, FALSE), nrow(data_norm),
                 replace=TRUE, prob=c(0.8,0.2))

train_X <- data_norm[sample, -12]
test_X <- data_norm[!sample, -12]
train_Y <- data_norm[sample, 12]
test_Y <- data_norm[!sample, 12]


spe <- rep(0,20)
sen <- rep(0,20)

for(i in 1:20){

  knn_pred <- as.factor(knn3Train(train_X, test_X, train_Y, k=i))

  TP <- table(knn_pred, test_Y)[2,2]
  TN <- table(knn_pred, test_Y)[1,1]
  FN <- table(knn_pred, test_Y)[2,1]
  FP <- table(knn_pred, test_Y)[1,2]

  spe[i] <- TN/(TN+FP)
  sen[i] <- TP/(TP+FN)

}

plot(1:20, spe, type="b", xlab="K value", ylab="Evaluation" ,
     main="K-NN evaluation", col=3, pch=16, ylim=c(0.5,1))
points(sen,type="b", col=4, pch=16)
legend("topright",legend = c("Specificity",
        "Sensitivity"), pch=16,
        col=c(3:4))
```



Notice from the plot that specificity is much higher than sensitivity, which means that the algorithm is much better at classifying the "bad/neutral" quality wine than the "good" quality wine. The best values of specificity and sensitivity, obtained for K=1, are reported in the following table.

```
# Specificity and sensitivity for K = 1
c(spe[1], sen[1])
```

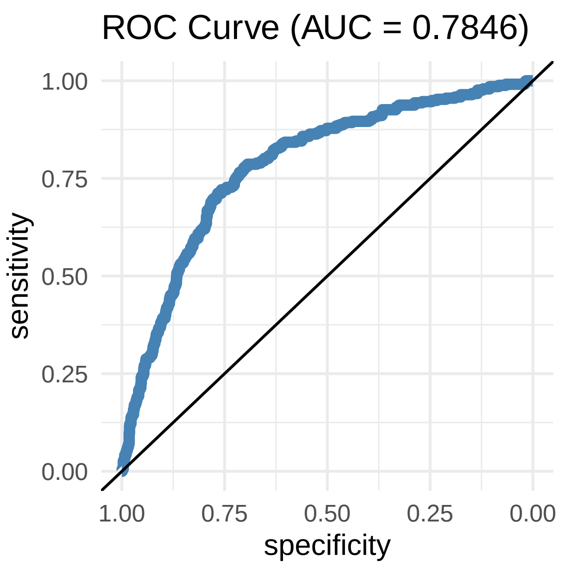
```
## [1] 0.9028340 0.6493506
```

Concluding, the points distance can't explained sufficiently the wine quality, in particular K-NN has a bad capacity to predict good quality wine when the quality wine is truly good: this is a huge limitation for the goal of the report, since we need to modeling the good quality wine.

3.2 Naive Bayes Classifier

In this section we are going to train a Naive Bayes Classifier, knowing that the assumption of independence of explanatory features is violated.

```
set.seed(123)
sample <- sample(c(TRUE, FALSE), nrow(data),
replace=TRUE, prob=c(0.7,0.3))
train <- data[sample, -8]
test <- data[!sample, -8]
nb_model <- naiveBayes(train[,-11], train$quality)
nb_pred <- predict(nb_model, test[,-11], type="raw")

roc_NB <- roc(as.numeric(test$quality), as.numeric(nb_pred[,2]))
auc_NB <- round(auc(as.numeric(test$quality), as.numeric(nb_pred[,2])),4)

ggroc(roc_NB, colour = 'steelblue', size = 2) +
  ggtitle(paste0('ROC Curve ', '(AUC = ', auc_NB, ')'))+
  theme_minimal()+ geom_abline(intercept = 1)
```



```
nb_pred <- ifelse(nb_pred[,2]>=nb_pred[,1], 1, 0)

coords(roc_NB, x="best")

##   threshold specificity sensitivity
## 1  0.391105   0.7032086   0.7774481
```

The validation of the Naive Bayes Classifier reports a specificity of 70.32%, a sensitivity of 77.74% and an area under ROC curve of 0.7846: the model has a good fit to the data.

3.3 Logistic Regression

Since for this project our main goal is to quantify the impact of each feature on wine quality, it is desirable to use models that produce results with an explicit interpretation, like logistic regression with *logit* link function. For this reason, for this model we decide to not split the dataset in train and tests sets, so we can use all of the samples to train the model.

First of all, we need to define the full model, which considers all the features, and the null model, which considers only the intercept (both models are computed with logit link function). In particular, the null model is given by:

$$logit(\mu) = \beta_0 \tag{3.1}$$

```
model_N <- glm(quality ~ 1, data=data[,-8], family = "binomial")
summary(model_N)
```

```
##
## Call:
## glm(formula = quality ~ 1, family = "binomial", data = data[,
##     -8])
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -0.6984  -0.6984  -0.6984  -0.6984   1.7496
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  -1.2867     0.0347  -37.08   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 5116.8  on 4897  degrees of freedom
## Residual deviance: 5116.8  on 4897  degrees of freedom
## AIC: 5118.8
##
## Number of Fisher Scoring iterations: 4
```

Instead, the full model is given by:

$$\begin{aligned} logit(\mu) =& \beta_0 + \beta_1 fixed\_acidity + \beta_2 volatile\_acidity + \beta_3 citric\_acid \\ &+ \beta_4 residual\_sugar + +\beta_5 chlorides + \beta_6 free\_sulfur\_dioxide \\ &+ \beta_7 total\_sulfur\_dioxide + \beta_8 density + \beta_9 pH + \beta_{10} sulphates + \beta_{11} alcohol \end{aligned} \tag{3.2}$$

```
model_F <- glm(quality~.,data=data[,-8], family="binomial")
summary(model_F)
```

```
##
## Call:
## glm(formula = quality ~ ., family = "binomial", data = data[,
##     -8])
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.9830  -0.6676  -0.4285  -0.1864   3.0170
##
## Coefficients:
##                        Estimate Std. Error z value Pr(>|z|)
## (Intercept)          -13.983108   1.311131 -10.665  < 2e-16 ***
## fixed_acidity          0.063796   0.055321   1.153   0.2488
## volatile_acidity      -3.937611   0.481884  -8.171 3.05e-16 ***
## citric_acid           -0.887589   0.397470  -2.233   0.0255 *
## residual_sugar         0.057271   0.009811   5.837 5.31e-09 ***
## chlorides            -18.063619   3.881794  -4.653 3.26e-06 ***
## free_sulfur_dioxide    0.012797   0.003036   4.216 2.49e-05 ***
## total_sulfur_dioxide  -0.003243   0.001427  -2.273   0.0230 *
## pH                     1.225067   0.292611   4.187 2.83e-05 ***
## sulphates              1.273688   0.320098   3.979 6.92e-05 ***
## alcohol                0.873541   0.044521  19.621  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 5116.8  on 4897  degrees of freedom
## Residual deviance: 4196.5  on 4887  degrees of freedom
## AIC: 4218.5
##
## Number of Fisher Scoring iterations: 6
```

From the model summaries we can see that the null model has a higher AIC than the full model, which means the full model has a better fit to the data. Anyway, in the full model there are some features which are not significant at a level of 5%.

At this point, it is useful to apply some methods to determine if there exists a reduced model, between the null and the full model, that has a good fitting to the data and where every feature is relevant. To do this, we use the ”backward” and the ”forward” feature selection procedures. For both, we fix the significant level to 5%.

3.3.1 Backward selection

In this part we compute a backward selection of the features, where the test used is the log-likelihood ratio test.

```
drop1(model_F, test = "LRT")

## Single term deletions
##
## Model:
## quality ~ fixed_acidity + volatile_acidity + citric_acid + residual_sugar +
##     chlorides + free_sulfur_dioxide + total_sulfur_dioxide +
##     pH + sulphates + alcohol
##                      Df Deviance    AIC    LRT  Pr(>Chi)
## <none>                    4196.5 4218.5
## fixed_acidity         1   4197.8 4217.8   1.32   0.24983
## volatile_acidity      1   4269.9 4289.9  73.46 < 2.2e-16 ***
## citric_acid           1   4201.6 4221.6   5.11   0.02379 *
## residual_sugar        1   4229.5 4249.5  33.03 9.061e-09 ***
## chlorides             1   4222.8 4242.8  26.34 2.856e-07 ***
## free_sulfur_dioxide   1   4214.3 4234.3  17.85 2.389e-05 ***
## total_sulfur_dioxide  1   4201.7 4221.7   5.22   0.02233 *
## pH                    1   4213.9 4233.9  17.45 2.950e-05 ***
## sulphates             1   4212.2 4232.2  15.77 7.165e-05 ***
## alcohol               1   4634.7 4654.7 438.23 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

model_R_BW <- update(model_F, .~.-fixed_acidity)

drop1(model_R_BW, test = "LRT")

## Single term deletions
##
## Model:
## quality ~ volatile_acidity + citric_acid + residual_sugar + chlorides +
##     free_sulfur_dioxide + total_sulfur_dioxide + pH + sulphates +
##     alcohol
##                      Df Deviance    AIC    LRT  Pr(>Chi)
## <none>                    4197.8 4217.8
## volatile_acidity      1   4271.4 4289.4  73.57 < 2.2e-16 ***
## citric_acid           1   4201.9 4219.9   4.10   0.04288 *
## residual_sugar        1   4230.1 4248.1  32.27 1.338e-08 ***
## chlorides             1   4224.1 4242.1  26.28 2.949e-07 ***
## free_sulfur_dioxide   1   4214.6 4232.6  16.84 4.074e-05 ***
## total_sulfur_dioxide  1   4202.4 4220.4   4.59   0.03222 *
## pH                    1   4214.7 4232.7  16.91 3.919e-05 ***
## sulphates             1   4213.5 4231.5  15.76 7.200e-05 ***
## alcohol               1   4635.3 4653.3 437.54 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
summary(model_R_BW)

##
## Call:
## glm(formula = quality ~ volatile_acidity + citric_acid + residual_sugar +
##     chlorides + free_sulfur_dioxide + total_sulfur_dioxide +
##     pH + sulphates + alcohol, family = "binomial", data = data[,
##     -8])
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.9994  -0.6665  -0.4297  -0.1898   2.9922
##
## Coefficients:
##                        Estimate Std. Error z value Pr(>|z|)
## (Intercept)          -13.044984   1.024812 -12.729  < 2e-16 ***
## volatile_acidity      -3.941681   0.481910  -8.179 2.86e-16 ***
## citric_acid           -0.763125   0.380412  -2.006   0.0449 *
## residual_sugar         0.056486   0.009789   5.770 7.91e-09 ***
## chlorides            -17.934840   3.863498  -4.642 3.45e-06 ***
## free_sulfur_dioxide    0.012299   0.003003   4.096 4.20e-05 ***
## total_sulfur_dioxide  -0.003010   0.001413  -2.131   0.0331 *
## pH                     1.069107   0.259484   4.120 3.79e-05 ***
## sulphates              1.274863   0.320481   3.978 6.95e-05 ***
## alcohol                0.868258   0.044225  19.633  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 5116.8  on 4897  degrees of freedom
## Residual deviance: 4197.8  on 4888  degrees of freedom
## AIC: 4217.8
##
## Number of Fisher Scoring iterations: 6
```

From the backward selection *fixed_acidity* results not significant at a level of 5% to describe the wine quality. The estimated model is given by:

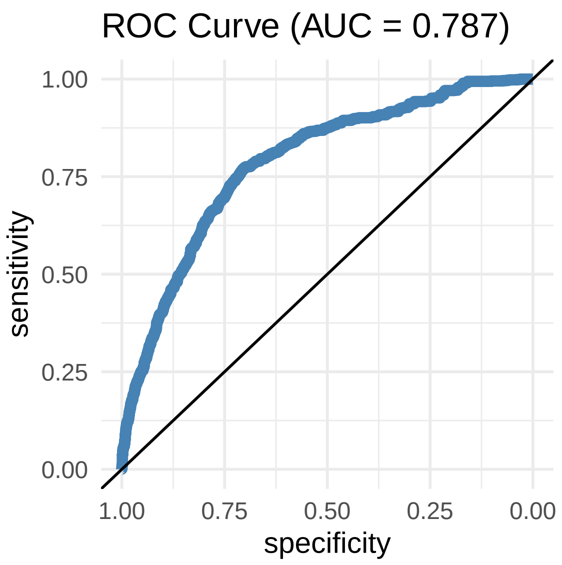
$$\begin{aligned} logit(\hat{\mu}) = &- 13.04 - 3.94 * volatile\_acidity - 0.76 * citric\_acid + 0.06 * residual\_sugar \\ &- 17.93 * chlorides + 0.01 * free\_sulfur\_dioxide - 0.003 * total\_sulfure\_dioxide \\ &+ 1.07 * pH + 1.27 * sulphates + 0.87 * alcohol \end{aligned} \tag{3.3}$$

Moreover, if we apply the exponential function to the model parameters, we obtain the increasing probability to obtain a good quality wine for each unitary increase of explanatory variables. For example, for each unitary increasing of pH the quality wine increases of 2.91 times; for each unitary increasing of sulphates, the quality wine increases by 3.56 times[1].

Model Validation

Below is reported the evaluation of model obtained with backward selection.

```r
logistic_prob_BW <- predict(model_R_BW, type="response")

roc_BW <- roc(data$quality, logistic_prob_BW)

auc_BW <- round(auc(data$quality, logistic_prob_BW),4)

ggroc(roc_BW, colour = 'steelblue', size = 2) +
  ggtitle(paste0('ROC Curve ', '(AUC = ', auc_BW, ')'))+
  theme_minimal()+ geom_abline(intercept = 1)
```



```r
coords(roc_BW, x="best")
```

```
##   threshold specificity sensitivity
## 1 0.2106335   0.7024492   0.7726415
```

The validation for logistic model obtained with a forward features selection reports a specificity of 70.24%, a sensitivity of 77.26% and an area under ROC curve of 0.787: the model has a good fit to the data.

[1]Pay attention to the scale of pH and sulphates, because an unitary increment is a huge increment for the feature scales.

3.3.2 Forward selection

In this part we compute a forward selection of the features, at a significant level of 5%, where the test we used is log-likelihood ratio test, starting from the null model.

```
add1(model_N, scope = model_F, test="LRT")
```

```
## Single term additions
##
## Model:
## quality ~ 1
##                      Df Deviance    AIC    LRT  Pr(>Chi)
## <none>                    5116.8 5118.8
## fixed_acidity         1   5083.9 5087.9  32.84 1.003e-08 ***
## volatile_acidity      1   5093.4 5097.4  23.38 1.328e-06 ***
## citric_acid           1   5110.5 5114.5   6.25  0.012420 *
## residual_sugar        1   5045.0 5049.0  71.77 < 2.2e-16 ***
## chlorides             1   4773.4 4777.4 343.36 < 2.2e-16 ***
## free_sulfur_dioxide   1   5114.0 5118.0   2.73  0.098651 .
## total_sulfur_dioxide  1   4982.3 4986.3 134.51 < 2.2e-16 ***
## pH                    1   5074.6 5078.6  42.22 8.173e-11 ***
## sulphates             1   5106.0 5110.0  10.80  0.001016 **
## alcohol               1   4392.8 4396.8 723.94 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
which.min(add1(model_N, scope = model_F, test="LRT")$`Pr(>Chi)`)
```

```
## [1] 11
```

```
model_R_FW <- update(model_N, .~.+alcohol)
add1(model_R_FW, scope = model_F, test="LRT")
```

```
## Single term additions
##
## Model:
## quality ~ alcohol
##                      Df Deviance    AIC    LRT  Pr(>Chi)
## <none>                    4392.8 4396.8
## fixed_acidity         1   4388.4 4394.4  4.453  0.034841 *
## volatile_acidity      1   4314.0 4320.0 78.860 < 2.2e-16 ***
## citric_acid           1   4392.0 4398.0  0.863  0.352883
## residual_sugar        1   4382.8 4388.8 10.024  0.001546 **
## chlorides             1   4360.1 4366.1 32.725 1.062e-08 ***
## free_sulfur_dioxide   1   4365.3 4371.3 27.518 1.556e-07 ***
## total_sulfur_dioxide  1   4392.8 4398.8  0.029  0.865501
## pH                    1   4372.9 4378.9 19.921 8.071e-06 ***
## sulphates             1   4374.8 4380.8 18.037 2.167e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
model_R_FW <- update(model_R_FW, .~.+volatile_acidity)
add1(model_R_FW, scope = model_F, test="LRT")
```

```
## Single term additions
##
## Model:
## quality ~ alcohol + volatile_acidity
##                      Df Deviance    AIC     LRT  Pr(>Chi)
## <none>                    4314.0 4320.0
## fixed_acidity         1   4308.9 4316.9  5.0986 0.0239450 *
## citric_acid           1   4307.4 4315.4  6.6110 0.0101350 *
## residual_sugar        1   4287.0 4295.0 26.9375 2.101e-07 ***
## chlorides             1   4286.2 4294.2 27.8110 1.338e-07 ***
## free_sulfur_dioxide   1   4292.1 4300.1 21.9012 2.870e-06 ***
## total_sulfur_dioxide  1   4312.0 4320.0  2.0143 0.1558251
## pH                    1   4297.2 4305.2 16.7357 4.296e-05 ***
## sulphates             1   4299.2 4307.2 14.7796 0.0001208 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
model_R_FW <- update(model_R_FW, .~.+chlorides)
add1(model_R_FW, scope = model_F, test="LRT")
```

```
## Single term additions
##
## Model:
## quality ~ alcohol + volatile_acidity + chlorides
##                      Df Deviance    AIC     LRT  Pr(>Chi)
## <none>                    4286.2 4294.2
## fixed_acidity         1   4281.2 4291.2  4.9710   0.02578 *
## citric_acid           1   4281.5 4291.5  4.6432   0.03118 *
## residual_sugar        1   4260.3 4270.3 25.8737 3.645e-07 ***
## free_sulfur_dioxide   1   4262.4 4272.4 23.7815 1.079e-06 ***
## total_sulfur_dioxide  1   4282.2 4292.2  3.9989   0.04553 *
## pH                    1   4268.8 4278.8 17.3741 3.070e-05 ***
## sulphates             1   4269.6 4279.6 16.5554 4.725e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
model_R_FW <- update(model_R_FW, .~.+residual_sugar)
add1(model_R_FW, scope = model_F, test="LRT")
```

```
## Single term additions
##
## Model:
## quality ~ alcohol + volatile_acidity + chlorides + residual_sugar
##                      Df Deviance    AIC     LRT  Pr(>Chi)
## <none>                      4260.3 4270.3
## fixed_acidity         1   4254.6 4266.6  5.6450   0.01751 *
## citric_acid           1   4253.8 4265.8  6.4792   0.01091 *
## free_sulfur_dioxide   1   4244.1 4256.1 16.1560 5.833e-05 ***
## total_sulfur_dioxide  1   4259.3 4271.3  0.9713   0.32437
## pH                    1   4235.0 4247.0 25.3219 4.852e-07 ***
## sulphates             1   4239.4 4251.4 20.8422 4.987e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
model_R_FW <- update(model_R_FW, .~.+pH)
add1(model_R_FW, scope = model_F, test="LRT")
```

```
## Single term additions
##
## Model:
## quality ~ alcohol + volatile_acidity + chlorides + residual_sugar +
##     pH
##                      Df Deviance    AIC     LRT  Pr(>Chi)
## <none>                      4235.0 4247.0
## fixed_acidity         1   4235.0 4249.0  0.0003 0.9872509
## citric_acid           1   4230.9 4244.9  4.0296 0.0447072 *
## free_sulfur_dioxide   1   4221.3 4235.3 13.6662 0.0002183 ***
## total_sulfur_dioxide  1   4234.8 4248.8  0.1563 0.6925458
## sulphates             1   4219.9 4233.9 15.0690 0.0001037 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
model_R_FW <- update(model_R_FW, .~.+sulphates)
add1(model_R_FW, scope = model_F, test="LRT")
```

```
## Single term additions
##
## Model:
## quality ~ alcohol + volatile_acidity + chlorides + residual_sugar +
##     pH + sulphates
##                      Df Deviance    AIC     LRT Pr(>Chi)
## <none>                      4219.9 4233.9
## fixed_acidity         1   4219.9 4235.9  0.0106 0.917880
## citric_acid           1   4214.7 4230.7  5.1951 0.022650 *
## free_sulfur_dioxide   1   4207.8 4223.8 12.1267 0.000497 ***
## total_sulfur_dioxide  1   4219.9 4235.9  0.0004 0.983664
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
model_R_FW <- update(model_R_FW, .~.+free_sulfur_dioxide)
add1(model_R_FW, scope = model_F, test="LRT")
```

```
## Single term additions
```

```
##
## Model:
## quality ~ alcohol + volatile_acidity + chlorides + residual_sugar +
##     pH + sulphates + free_sulfur_dioxide
##                      Df Deviance    AIC    LRT Pr(>Chi)
## <none>                    4207.8 4223.8
## fixed_acidity         1   4207.8 4225.8 0.0184  0.89196
## citric_acid           1   4202.4 4220.4 5.3986  0.02015 *
## total_sulfur_dioxide  1   4201.9 4219.9 5.8854  0.01527 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
model_R_FW <- update(model_R_FW, .~.+total_sulfur_dioxide)
add1(model_R_FW, scope = model_F, test="LRT")
```

```
## Single term additions
##
## Model:
## quality ~ alcohol + volatile_acidity + chlorides + residual_sugar +
##     pH + sulphates + free_sulfur_dioxide + total_sulfur_dioxide
##               Df Deviance    AIC    LRT Pr(>Chi)
## <none>             4201.9 4219.9
## fixed_acidity  1   4201.6 4221.6 0.3146  0.57487
## citric_acid    1   4197.8 4217.8 4.1002  0.04288 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
model_R_FW <- update(model_R_FW, .~.+citric_acid)
add1(model_R_FW, scope = model_F, test="LRT")
```

```
## Single term additions
##
## Model:
## quality ~ alcohol + volatile_acidity + chlorides + residual_sugar +
##     pH + sulphates + free_sulfur_dioxide + total_sulfur_dioxide +
##     citric_acid
##               Df Deviance    AIC    LRT Pr(>Chi)
## <none>             4197.8 4217.8
## fixed_acidity  1   4196.5 4218.5 1.3243   0.2498
```

```
summary(model_R_FW)
```

```
##
## Call:
## glm(formula = quality ~ alcohol + volatile_acidity + chlorides +
##     residual_sugar + pH + sulphates + free_sulfur_dioxide + total_sulfur_dioxide +
##     citric_acid, family = "binomial", data = data[, -8])
##
## Deviance Residuals:
```

```
##     Min       1Q   Median       3Q      Max
## -1.9994  -0.6665  -0.4297  -0.1898   2.9922
##
## Coefficients:
##                        Estimate Std. Error z value Pr(>|z|)
## (Intercept)          -13.044984   1.024812 -12.729  < 2e-16 ***
## alcohol                0.868258   0.044225  19.633  < 2e-16 ***
## volatile_acidity      -3.941681   0.481910  -8.179 2.86e-16 ***
## chlorides            -17.934840   3.863498  -4.642 3.45e-06 ***
## residual_sugar         0.056486   0.009789   5.770 7.91e-09 ***
## pH                     1.069107   0.259484   4.120 3.79e-05 ***
## sulphates              1.274863   0.320481   3.978 6.95e-05 ***
## free_sulfur_dioxide    0.012299   0.003003   4.096 4.20e-05 ***
## total_sulfur_dioxide  -0.003010   0.001413  -2.131   0.0331 *
## citric_acid           -0.763125   0.380412  -2.006   0.0449 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 5116.8  on 4897  degrees of freedom
## Residual deviance: 4197.8  on 4888  degrees of freedom
## AIC: 4217.8
##
## Number of Fisher Scoring iterations: 6
```

From the forward selection, in order, the features added at a significant level of 5% are: *alcohol*, *volatile acidity*, *chlorides*, *residual sugar*, *pH*, *sulphates*, *free sulfure dioxide*, *total sulfure dioxide* and *citric acid*, leaving out *fixed acidity*. The estimated model is given by:

$$\begin{aligned} logit(\hat{\mu}) = & -13.04 + 0.87 * alcohol - 3.94 * volatile\_acidity - 17.93 * chlorides \\ & + 0.05 * residual\_sugar + 1.07 * pH + 1.27 * sulphates \\ & + 0.01 * free\_sulfur\_dioxide - 0.003 * total\_sulfur\_dioxide - 0.76 * citric\_acid \end{aligned} \tag{3.4}$$

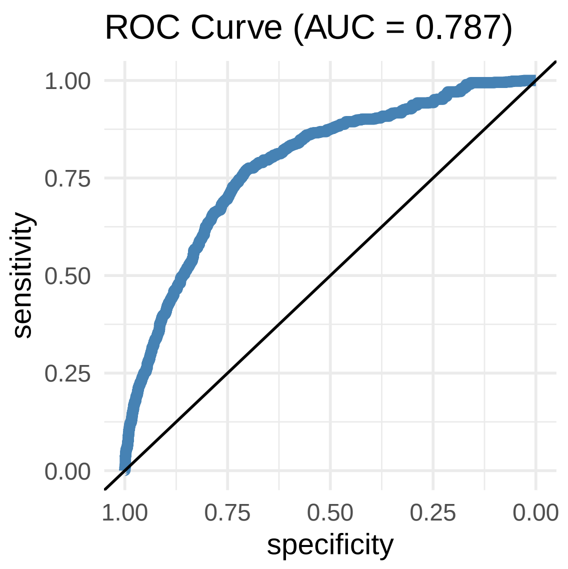
With the forward selection method we obtained the same model as backward selection.

Model Validation

Now we are going to evaluate the model obtained with forward selection.

```
logistic_prob_FW <- predict(model_R_FW, type="response")

roc_FW <- roc(data$quality, logistic_prob_FW)

auc_FW <- round(auc(data$quality, logistic_prob_FW),4)
```

```
ggroc(roc_FW, colour = 'steelblue', size = 2) +
  ggtitle(paste0('ROC Curve ', '(AUC = ', auc_FW, ')'))+
  theme_minimal()+ geom_abline(intercept = 1)
```



```
coords(roc_FW, x="best")

##   threshold specificity sensitivity
## 1 0.2106335   0.7024492   0.7726415
```

Notice from the output that the model diagnostic is the same obtained from the logistic regression model given by backward feature selection.

3.3.3 AIC evaluation

Since we are facing multicollinearity, we have removed the *density* feature from the modeling phase; however, it still holds that there is a significant correlation between the features, even if not very high. For this reason, we apply a selection of features based on the AIC criterion, with both forward and backward directions, to analyse the eventual differences from the previous models.

```
stepAIC(model_F ,direction = "both")

## Start:  AIC=4218.46
## quality ~ fixed_acidity + volatile_acidity + citric_acid + residual_sugar +
##     chlorides + free_sulfur_dioxide + total_sulfur_dioxide +
##     pH + sulphates + alcohol
##
##                        Df Deviance    AIC
## - fixed_acidity         1   4197.8 4217.8
## <none>                        4196.5 4218.5
## - citric_acid           1   4201.6 4221.6
## - total_sulfur_dioxide  1   4201.7 4221.7
## - sulphates             1   4212.2 4232.2
## - pH                    1   4213.9 4233.9
## - free_sulfur_dioxide   1   4214.3 4234.3
## - chlorides             1   4222.8 4242.8
## - residual_sugar        1   4229.5 4249.5
## - volatile_acidity      1   4269.9 4289.9
## - alcohol               1   4634.7 4654.7
##
## Step:  AIC=4217.78
## quality ~ volatile_acidity + citric_acid + residual_sugar + chlorides +
##     free_sulfur_dioxide + total_sulfur_dioxide + pH + sulphates +
##     alcohol
##
##                        Df Deviance    AIC
## <none>                        4197.8 4217.8
## + fixed_acidity         1   4196.5 4218.5
## - citric_acid           1   4201.9 4219.9
## - total_sulfur_dioxide  1   4202.4 4220.4
## - sulphates             1   4213.5 4231.5
## - free_sulfur_dioxide   1   4214.6 4232.6
## - pH                    1   4214.7 4232.7
## - chlorides             1   4224.1 4242.1
## - residual_sugar        1   4230.1 4248.1
## - volatile_acidity      1   4271.4 4289.4
## - alcohol               1   4635.3 4653.3
##
## Call:  glm(formula = quality ~ volatile_acidity + citric_acid + residual_sugar +
##     chlorides + free_sulfur_dioxide + total_sulfur_dioxide +
##     pH + sulphates + alcohol, family = "binomial", data = data[,
##     -8])
##
## Coefficients:
##          (Intercept)      volatile_acidity           citric_acid
##            -13.04498              -3.94168              -0.76312
##       residual_sugar             chlorides   free_sulfur_dioxide
##              0.05649             -17.93484               0.01230
## total_sulfur_dioxide                    pH             sulphates
##             -0.00301               1.06911               1.27486
##              alcohol
##              0.86826
##
## Degrees of Freedom: 4897 Total (i.e. Null);  4888 Residual
## Null Deviance:     5117
## Residual Deviance: 4198  AIC: 4218
```

```
model_AIC <- glm(formula = quality ~ volatile_acidity +
                   citric_acid +residual_sugar +
                   chlorides + free_sulfur_dioxide +
                   total_sulfur_dioxide + pH + sulphates +
                   alcohol, family = "binomial", data = data[,-8])
summary(model_AIC)
```

```
##
## Call:
## glm(formula = quality ~ volatile_acidity + citric_acid + residual_sugar +
##     chlorides + free_sulfur_dioxide + total_sulfur_dioxide +
##     pH + sulphates + alcohol, family = "binomial", data = data[,
##     -8])
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.9994  -0.6665  -0.4297  -0.1898   2.9922
##
## Coefficients:
##                        Estimate Std. Error z value Pr(>|z|)
## (Intercept)          -13.044984   1.024812 -12.729  < 2e-16 ***
## volatile_acidity      -3.941681   0.481910  -8.179 2.86e-16 ***
## citric_acid           -0.763125   0.380412  -2.006   0.0449 *
## residual_sugar         0.056486   0.009789   5.770 7.91e-09 ***
## chlorides            -17.934840   3.863498  -4.642 3.45e-06 ***
## free_sulfur_dioxide    0.012299   0.003003   4.096 4.20e-05 ***
## total_sulfur_dioxide  -0.003010   0.001413  -2.131   0.0331 *
## pH                     1.069107   0.259484   4.120 3.79e-05 ***
## sulphates              1.274863   0.320481   3.978 6.95e-05 ***
## alcohol                0.868258   0.044225  19.633  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 5116.8  on 4897  degrees of freedom
## Residual deviance: 4197.8  on 4888  degrees of freedom
## AIC: 4217.8
##
## Number of Fisher Scoring iterations: 6
```

The output of automatic selection of features estimates the same model we estimated with backward and forward feature selection based on log-likelihood ratio test, and for this reason the model selection based on log-likelihood ratio test is also trustable.

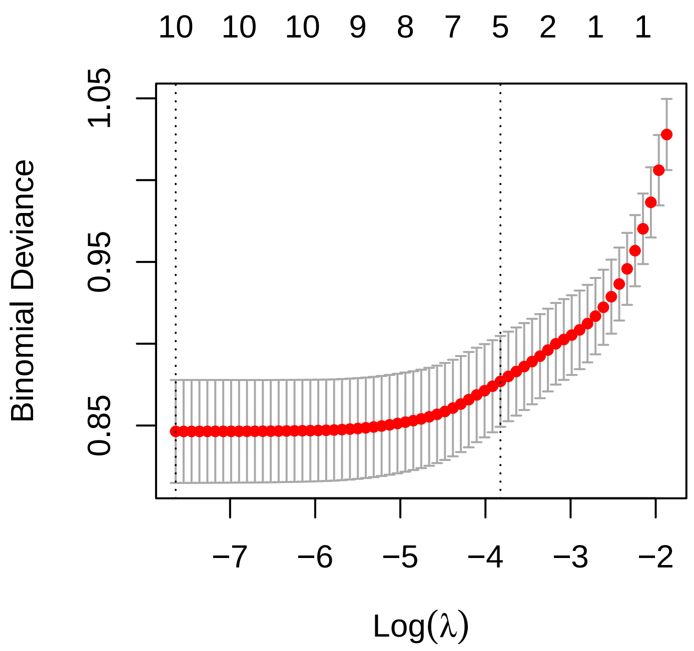
3.4 Penalised Logistic Regression

In this section we are going to compute penalised (or regularised) logistic regression models with l_1 and l_2 norms (respectively, Lasso and Ridge regressions), where an hyperparameter (λ) regularises the importance of the parameters, determining the smallest model with the lowest loss. First of all, we need to split our data in train and test sets.

```
set.seed(123)
sample <- sample(c(TRUE, FALSE), nrow(data),
                 replace=TRUE, prob=c(0.7,0.3))

train <- data[sample, ]
test <- data[!sample, ]
```

3.4.1 Lasso Regression

```
cv_lasso <- cv.glmnet(as.matrix(train[,c(-8,-12)]), train[,12],
                      family = "binomial", alpha = 1)
plot(cv_lasso)
```



```
# Model selection
coef(cv_lasso)
```

```
## 11 x 1 sparse Matrix of class "dgCMatrix"
##                                s1
## (Intercept)          -7.9540419132
## fixed_acidity          .
## volatile_acidity     -1.5897125448
## citric_acid            .
## residual_sugar         .
## chlorides            -3.2501131385
## free_sulfur_dioxide   0.0005987363
## total_sulfur_dioxide  .
## pH                     .
## sulphates             0.2978208406
## alcohol               0.6534046354
```

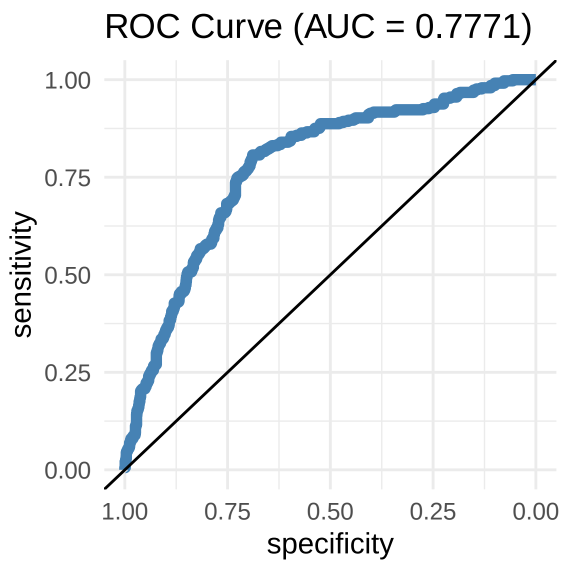
This output return the model selection obtained with L^1 no

```
#Best lambda
cv_lasso$lambda.1se
```

```
## [1] 0.02183489
```

The plot shows the best value of $log(\lambda)$ that minimizes the cross-validation error is approximately -4, and the value of λ is equal to 0.02183489.

```
lasso.model <- glmnet(as.matrix(train[,c(-8,-12)]), train[,12],
                      alpha = 1, family = "binomial",
                      lambda = cv_lasso$lambda.1se)

probabilities <- lasso.model %>% predict(newx = as.matrix(test[,c(-8,-12)]),
type="response")

predicted.classes <- ifelse(probabilities > 0.5, 1,0)

observed.classes <- test$quality

roc_PL <- roc(observed.classes,
              as.numeric(probabilities))

auc_PL <- round(auc(as.numeric(observed.classes),
                    as.numeric(probabilities)),4)

ggroc(roc_PL, colour = 'steelblue', size = 2) +
  ggtitle(paste0('ROC Curve ', '(AUC = ', auc_PL, ')'))+
  theme_minimal()+ geom_abline(intercept = 1)
```



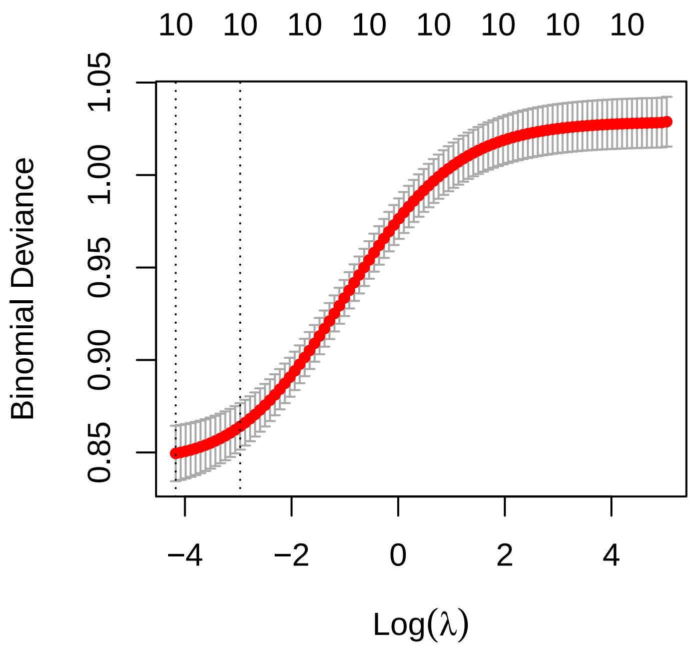
```
coords(roc_PL, x="best")

##   threshold specificity sensitivity
## 1 0.2065395   0.6889483   0.8071217
```

Unlike the logistic model, we only keep the variables: *volatile acidity*, *chlorides*, *free sulfur dioxide*, *sulphates* and *alcohol*. The model evaluation outputs show how the Lasso model has specificity of 69.89%, sensibility of 80.71% and an area under ROC curve equal to 0.7771, which is a good value. For this reason, the evaluation phase of Lasso model suggests a good fit to data.

3.4.2 Ridge Regression

Although, in general, one might expect Lasso Regression to perform better and Ridge Regression does not perform feature selection, we decided to also try to fit this model to our data.

```
cv_ridge <- cv.glmnet(as.matrix(train[,c(-8,-12)]), train[,12],
                      family = "binomial", alpha = 0)
plot(cv_ridge)
```

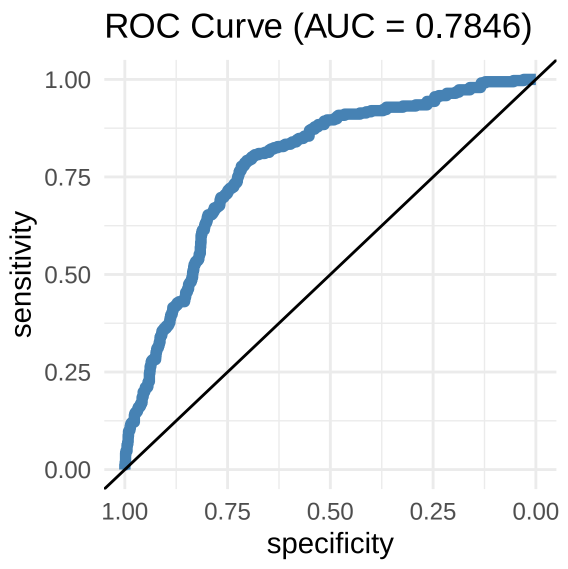


```
cv_ridge$lambda.1se

## [1] 0.05162837
```

The plot shows the best value of $log(\lambda)$ that minimizes the cross-validation error is approximately -3, and the value of λ is equal to 0.05162837.

```
ridge.model <- glmnet(as.matrix(train[,c(-8,-12)]), train[,12],
                      alpha = 0, family = "binomial",
                      lambda = cv_lasso$lambda.1se)

probabilities <- ridge.model %>% predict(newx = as.matrix(test[,c(-8,-12)]),
type="response")

predicted.classes <- ifelse(probabilities > 0.5, 1,0)

observed.classes <- test$quality

roc_Rg <- roc(observed.classes, as.numeric(probabilities))

auc_Rg <- round(auc(as.numeric(observed.classes),
                    as.numeric(probabilities)),4)
```

```r
ggroc(roc_Rg, colour = 'steelblue', size = 2) +
  ggtitle(paste0('ROC Curve ', '(AUC = ', auc_Rg, ')'))+
  theme_minimal()+ geom_abline(intercept = 1)
```



```r
coords(roc_Rg, x="best")
```

```
##   threshold specificity sensitivity
## 1 0.2163591   0.7032086   0.7922849
```

The diagnostic of Ridge model reports a good fit to data, in fact we have 70.32% of specificity, 79.22% of sensitivity and 0.7846 of AUC.

3.5 Modeling summary

In the previous sections, we fitted several models to our data, in order to find the one which fits best to the data. The following table summarises the evaluation model phase for each of the tested models.

Model	AUC	Specificity (best)	Sensitivity (best)
K-NN	-	0.903	0.649
Naive Bayes	0.785	0.703	0.777
Logistic	0.787	0.702	0.773
Lasso	0.777	0.689	0.807
Ridge	0.785	0.703	0.792

Table 3.1: Summary of model evaluation for models tested in chapter 3.

Notice from the table 3.1 that all the tested models with the exception of the K-NN have a good and similar fit to the data: the decisive criterion with which we choose to use one model over another has to be based on the usage goal. In fact, if we need to detect the impact of features on quality wine, before the wine being product, we should use the logistic model[2] for the parameters interpretability; instead, if we need to classify a wine as having good or neutral/bad quality once we produce it, we can use the Lasso Regression, Ridge Regression or the Naive Bayes Classifier.

For a classification problem, the validation phase suggests that we should use the Naive Bayes or Ridge models if we are interested to have a lower probability that a predicted good quality wine is not really good quality (since NB and Ridge have 0.703 of specificity, the highest of all models), and we can use Lasso model if we need a higher probability that a predicted good quality wine really is good quality wine (since Lasso has 0.807 of sensitivity, again the highest among all the models).

[2]Third row of the table 3.1 contains the metrics obtained with the logistic model, with forward selection, backward selection and AIC for both directions, since the final model is the same in all cases.

4 Conclusion

The dataset is a collection of 4898 white wines, for each of whom are collected 11 features and the wine quality.

Since we transformed the *quality* wine into a binary feature, the new dataset contains 3838 bad and neutral quality wine (score between 0 and 6) and 1060 good quality wine (score between 7 and 10).

In bivariate analysis, the correlation plot shows a high correlation between *density* with *alcohol* and with *residual sugar*, and for this reason we are not going to consider *density* in the modeling phase.

In order to solve the classification problem, so to define if the wine features can describe the quality of the wine, we built a K-NN model, a Logistic model, two Penalised Logistic models and a Naive Bayes Classier.

1. K-Nearest Neighbors: has a specificity much higher than its sensitivity, which translates in a good behaviour when classifying "bad/neutral" quality wine, but difficulties classifying "good" quality wine.
2. Naive Bayes Classifier: despite the independence between predictor variables assumption violation, the model still has a good fit to the data, reporting an AUC equal to 0.785.
3. Logistic Model: we computed a forward and backward and AIC with both directions selection of the features. All three procedures produce the same result, suggesting that *fixed acidity* is not relevant to describe the wine quality, at a significance level of 5%. Overall, the Logistic model presents a good fit to the data, with an AUC equal to 0.787.
4. Penalised Logistic Model: we performed Lasso and Ridge regressions and, for both, we determined the λ value that gives the most simplicity with lowest loss. Like the logistic regression and Naive Bayes, we again have models that fit relatively well to the data, with AUC of 0.777 for the Lasso and 0.785 for the Ridge.

The multivariate analysis suggests that the Logistic Model is the best one for our problem, because it fits well to the data while also providing an interpretation and some suggestions for producing wine with good quality. So, we can take our conclusion from the Logistic Model and say that when the coefficient of a variable is positive than the Odds Ratio is larger than one, meaning that the probability to have a good wine, keeping invariant the other variables, increases with the increasing of that variable; this is the case of *alcohol*, *residual sugar*, *pH*, *sulphates* and *free sulfur dioxide*. On the other side, if the coefficient is negative then the Odds Ratio is less than 1, meaning that the probability to have a good wine decreases if the variables *volatile acidity*, *chlorides*, *total sulfur dioxide* and *citric acid* increase (always just one at a time keeping invariant the others).

Since the quality tests are sensory, they are susceptible to the taste preferences of the wine expert, making it harder to predict the quality of the wine. The quality might also depend on factors that are not physicochemical, like year of production, grape types, wine brand and so on, so having access to this type of features might bring better classification and prediction results.

References

[1] Paulo Cortez, António Cerdeira, Fernando Almeida, Telmo Matos, and José Reis. Modeling wine preferences by data mining from physicochemical properties. *Decision Support Systems*, 47(4):547–553, 2009. Smart Business Networks: Concepts and Empirical Evidence.

[2] A. Salvan, N. Sartori, and L. Pace. *Modelli Lineari Generalizzati*. UNITEXT. Springer Milan, 2020.

[3] L. Ventura and W. Racugno. *Biostatistica. Casi di studio in R*. I Manuali. EGEA, 2017.

[4] M. Grigoletto, L. Ventura, and F. Pauli. *Modello Lineare: Teoria e Applicazioni con R*. Giappichelli, 2017.

[5] Alberto Roverato. Statistical Learning, 2023.