

Project 1: Phân loại chất lượng rượu vang

Final project - Xử lý số liệu thống kê - Nhóm L

Bản đề xuất phân tích

1. Mục tiêu phân tích cần đạt được

Mục tiêu 1: Xác định các feature quan trọng có ảnh hưởng đáng kể đến chất lượng rượu vang.

Để xác định chính xác những thuộc tính hóa lý nào của rượu vang, có thể kể đến như độ axit, lượng đường, hàm lượng cồn, sulphates, độ pH và các thuộc tính khác nữa có ảnh hưởng như thế nào đến hương vị rượu vang.

Mục tiêu 2: Xây dựng mô hình hồi quy dự đoán chất lượng rượu vang.

Mục tiêu là tạo ra và tối ưu hóa mô hình hồi quy có thể dự đoán điểm chất lượng của rượu vang dựa trên các thuộc tính hóa lý cho trước, từ đó xác định được các features nào là quan trọng và có ảnh hưởng đến chất lượng rượu vang.

Mục tiêu 3: Xây dựng mô hình phân loại dự đoán chất lượng rượu vang.

Mục tiêu là tạo ra một mô hình phân loại chất lượng rượu vang một cách đáng tin cậy dựa trên các đặc tính định lượng (độ axit, nồng độ cồn, lượng đường, ...).

Mục tiêu 4: Đề ra các chiến lược cải tiến sản phẩm và chiến lược kinh doanh dựa trên các insight có được từ quá trình phân tích.

2. Đề xuất các phương pháp và chiến lược phân tích tương ứng

Phương pháp	Chiến lược phân tích	Chi tiết
EDA	Thống kê mô tả toàn bộ các biến trên toàn bộ tập dữ liệu tổng hợp của 2 file red-wine và white-wine	+ Xu hướng trung tâm: mean, median, ... + Độ biến thiên: std, min-max, IQR, ... + Phân phối dữ liệu. + Dùng Bootstrap để khảo sát phân phối mẫu, khoảng tin cậy, ... cho các thống kê hoặc các thông số cần quan tâm của dữ liệu.
	Trực quan hóa dữ liệu	+ Sử dụng các biểu đồ: Histogram, Boxplot, Density, Scatter Plot để phát hiện patterns cũng như các điểm bất thường, điểm ngoại lai trong bộ dữ liệu. + Sử dụng Heatmap trực quan hóa ma trận tương quan để xác định mối quan hệ, sự tương quan và tương tác giữa các biến với nhau
Preprocessing	Xử lý missing values và duplicates	
	Xử lý imbalanced data	Xử lý dữ liệu mất cân bằng sử dụng các phương pháp Over-sampling và Under-sampling
Kiểm định giả thuyết	Kiểm định trung bình	Kiểm định trung bình cho các đặc trưng giữa hai loại rượu xem chúng có khác nhau hay không bằng Bootstrap Permutation Test
	Kiểm định ANOVA	Kiểm định ANOVA cho các đặc trưng của rượu giữa rượu có chất lượng khác nhau, xem chúng có khác nhau không
Xây dựng mô hình	Xây dựng mô hình	+ Train set: để huấn luyện mô hình. + Test set: để đánh giá mô hình.
Regression Models	Xây dựng các mô hình hồi quy dự đoán chất lượng rượu vang theo các biến thu thập được trong dữ liệu	+ Linear Regression + Ridge Regression + Hồi quy đa thức
	Sử dụng các phương pháp lựa chọn mô hình để tìm ra tập con tốt nhất cho các mô hình hồi quy	+ Hồi quy từng bước cho Linear Regression. + Phương pháp co hệ số cho Ridge Regression
	Thực hiện các thống kê suy luận cho các mô hình vừa xây dựng	(khoảng tin cậy cho hệ số của mô hình, khoảng tiên đoán cho chất lượng rượu vang, ...) sử dụng phương pháp Bootstrap
	Đánh giá	Sử dụng các chỉ số: r-squared-adj, r-squared, MAE, MSE, RMSE, accuracy, ...

Chuẩn đoán thặng dư mô hình

- +) Kiểm tra tính tuyến tính của mô hình sử dụng biểu đồ Residuals vs Fitted.
- +) Kiểm tra tính tuyến tính từng phần của mô hình sử dụng biểu đồ thặng dư từng phần (Partial Residual Plots).
- +) Kiểm tra tính đồng nhất phương sai sử dụng biểu đồ Scale-Location.
- +) Kiểm tra điểm ngoại lai trong mô hình

Nhận xét và rút ra kết luận cho từng mô hình

- + Xác định các features quan trọng có ảnh hưởng đến chất lượng rượu vang.
- + So sánh các mô hình

Classification Models (phân loại chất lượng rượu)

Xây dựng và huấn luyện mô hình

- + Naive Bayes
- + LDA
- + Multinomial Logistic

Nhận xét và rút ra kết luận cho từng mô hình

Thực hiện phân tích với R

Đọc dữ liệu và import thư viện cần sử dụng

library(janitor)

```
##  
## Attaching package: 'janitor'
```

```
## The following objects are masked from 'package:stats':  
##  
##   chisq.test, fisher.test
```

library(tidyverse)

```
## Warning: package 'ggplot2' was built under R version 4.3.1
```

```
## Warning: package 'tidyverse' was built under R version 4.3.1
```

```
## Warning: package 'readr' was built under R version 4.3.1
```

```
## Warning: package 'dplyr' was built under R version 4.3.1
```

```
## Warning: package 'stringr' was built under R version 4.3.1
```

```
## Warning: package 'lubridate' was built under R version 4.3.1
```

```
## —— Attaching core tidyverse packages —— tidyverse 2.0.0 ——  
## ✓ dplyr 1.1.4 ✓ readr 2.1.5  
## ✓forcats 1.0.0 ✓ stringr 1.5.1  
## ✓ ggplot2 3.5.0 ✓ tibble 3.2.1  
## ✓ lubridate 1.9.3 ✓ tidyverse 1.3.1  
## ✓ purrr 1.0.2
```

```
## —— Conflicts —— tidyverse_conflicts() ——  
## ✘ dplyr::filter() masks stats::filter()  
## ✘ dplyr::lag() masks stats::lag()  
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

library(dplyr)
library(corrplot)

```
## corrplot 0.92 loaded
```

```
library(ROSE)
```

```
## Loaded ROSE 0.0-4
```

```
library(leaps)
library(boot)
library(ggplot2)
library(nnet)
library(caret)
```

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

```
library(gridExtra)
```

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

```
library(klaR)
```

```
## Warning: package 'klaR' was built under R version 4.3.1
```

```
## Loading required package: MASS
##
## Attaching package: 'MASS'
##
## The following object is masked from 'package:dplyr':
##
##     select
```

```
library(caTools)
library(mgcv)
```

```
## Warning: package 'mgcv' was built under R version 4.3.1
```

```
## Loading required package: nlme
##
## Attaching package: 'nlme'
##
## The following object is masked from 'package:dplyr':
##
##     collapse
##
## This is mgcv 1.9-1. For overview type 'help("mgcv-package")'.
##
## Attaching package: 'mgcv'
##
## The following object is masked from 'package:nnet':
##
##     multinom
```

```
library(glmnet)
```

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

```
library(Hmisc)
```

```
## Warning: package 'Hmisc' was built under R version 4.3.3
```

```
##
## Attaching package: 'Hmisc'
##
## The following objects are masked from 'package:dplyr':
##   src, summarize
##
## The following objects are masked from 'package:base':
##   format.pval, units
```

```
library(psych)
```

```
## Warning: package 'psych' was built under R version 4.3.3
```

```
##
## Attaching package: 'psych'
##
## The following object is masked from 'package:Hmisc':
##   describe
##
## The following object is masked from 'package:boot':
##   logit
##
## The following objects are masked from 'package:ggplot2':
##   %+%, alpha
```

```
set.seed(21)
```

```
red_data <- read.csv(file = "winequality-red.csv", sep=";") |> janitor::clean_names()
white_data <- read.csv(file = "winequality-white.csv") |> janitor::clean_names()

# add categorical variables to both sets
red_data[["color"]] <- 'red'
white_data[["color"]] <- 'white'

# merge red wine and white wine datasets
data <- rbind(red_data, white_data)
```

```
head(data)
```

```

## fixed_acidity volatile_acidity citric_acid residual_sugar chlorides
## 1      7.4      0.70     0.00     1.9   0.076
## 2      7.8      0.88     0.00     2.6   0.098
## 3      7.8      0.76     0.04     2.3   0.092
## 4     11.2      0.28     0.56     1.9   0.075
## 5      7.4      0.70     0.00     1.9   0.076
## 6      7.4      0.66     0.00     1.8   0.075
## free_sulfur_dioxide total_sulfur_dioxide density p_h sulphates alcohol
## 1        11       34 0.9978 3.51   0.56   9.4
## 2        25       67 0.9968 3.20   0.68   9.8
## 3        15       54 0.9970 3.26   0.65   9.8
## 4        17       60 0.9980 3.16   0.58   9.8
## 5        11       34 0.9978 3.51   0.56   9.4
## 6        13       40 0.9978 3.51   0.56   9.4
## quality color
## 1      5 red
## 2      5 red
## 3      5 red
## 4      6 red
## 5      5 red
## 6      5 red

```

I. EDA

I.1. Thống kê mô tả

```
glimpse(data)
```

```

## Rows: 6,497
## Columns: 13
## $ fixed_acidity    <dbl> 7.4, 7.8, 7.8, 11.2, 7.4, 7.4, 7.9, 7.3, 7.8, 7.5...
## $ volatile_acidity <dbl> 0.700, 0.880, 0.760, 0.280, 0.700, 0.660, 0.600, ...
## $ citric_acid      <dbl> 0.00, 0.00, 0.04, 0.56, 0.00, 0.00, 0.06, 0.00, 0...
## $ residual_sugar   <dbl> 1.9, 2.6, 2.3, 1.9, 1.9, 1.8, 1.6, 1.2, 2.0, 6.1, ...
## $ chlorides         <dbl> 0.076, 0.098, 0.092, 0.075, 0.076, 0.075, 0.069, ...
## $ free_sulfur_dioxide <dbl> 11, 25, 15, 17, 11, 13, 15, 15, 9, 17, 15, 17, 16...
## $ total_sulfur_dioxide <dbl> 34, 67, 54, 60, 34, 40, 59, 21, 18, 102, 65, 102, ...
## $ density           <dbl> 0.9978, 0.9968, 0.9970, 0.9980, 0.9978, 0.9978, 0...
## $ p_h                <dbl> 3.51, 3.20, 3.26, 3.16, 3.51, 3.51, 3.30, 3.39, 3...
## $ sulphates          <dbl> 0.56, 0.68, 0.65, 0.58, 0.56, 0.56, 0.46, 0.47, 0...
## $ alcohol             <dbl> 9.4, 9.8, 9.8, 9.8, 9.4, 9.4, 9.4, 10.0, 9.5, 10....
## $ quality              <int> 5, 5, 5, 6, 5, 5, 5, 7, 7, 5, 5, 5, 5, 5, 5, 5, 7...
## $ color               <chr> "red", "red", "red", "red", "red", "red", "red", ...

```

Dữ liệu gồm mẫu thử của 1599 mẫu rượu vang đỏ và 4898 mẫu rượu vang trắng, được tổng hợp từ 02 file dữ liệu winequality-red.csv và winequality-white.csv, tương ứng cho hai loại rượu vang đỏ và vang trắng, các biến bao gồm:

- fixed acidity - g(tartaric acid)/dm3;
- volatile acidity - g(acetic acid)/dm3;
- citric acid - g/dm3;
- residual sugar - g/dm3;
- chlorides - g(sodium chloride)/dm3;
- free sulfur dioxide - mg/dm3;
- total sulfur dioxide - mg/dm3;
- density - mg/dm3;
- pH;
- sulphates - g(potassium sulphate)/dm3;
- alcohol - % nồng độ cồn;
- quality - điểm chất lượng.

Lập bảng thống kê

```
#sử dụng thư viện
describe(data)
```

```

##          vars   n  mean   sd median trimmed  mad min  max
## fixed_acidity     1 6497  7.22 1.30  7.00  7.06  0.89 3.80 15.90
## volatile_acidity  2 6497  0.34 0.16  0.29  0.32  0.12 0.08  1.58
## citric_acid      3 6497  0.32 0.15  0.31  0.32  0.10 0.00  1.66
## residual_sugar    4 6497  5.44 4.76  3.00  4.70  2.52 0.60 65.80
## chlorides         5 6497  0.06 0.04  0.05  0.05  0.02 0.01  0.61
## free_sulfur_dioxide 6 6497 30.53 17.75 29.00 29.32 17.79 1.00 289.00
## total_sulfur_dioxide 7 6497 115.74 56.52 118.00 115.92 57.82 6.00 440.00
## density           8 6497  0.99 0.00  0.99  0.99  0.00 0.99  1.04
## p_h                9 6497  3.22 0.16  3.21  3.21  0.16 2.72  4.01
## sulphates        10 6497  0.53 0.15  0.51  0.52  0.12 0.22  2.00
## alcohol          11 6497  10.49 1.19 10.30 10.40  1.33 8.00 14.90
## quality           12 6497  5.82 0.87  6.00  5.79  1.48 3.00  9.00
## color*            13 6497  1.75 0.43  2.00  1.82  0.00 1.00  2.00
## range      skew kurtosis se
## fixed_acidity    12.10 1.72  5.05 0.02
## volatile_acidity 1.50 1.49  2.82 0.00
## citric_acid     1.66 0.47  2.39 0.00
## residual_sugar   65.20 1.43  4.35 0.06
## chlorides        0.60 5.40  50.84 0.00
## free_sulfur_dioxide 288.00 1.22  7.90 0.22
## total_sulfur_dioxide 434.00 0.00 -0.37 0.70
## density          0.05 0.50  6.60 0.00
## p_h              1.29 0.39  0.37 0.00
## sulphates       1.78 1.80  8.64 0.00
## alcohol          6.90 0.57 -0.53 0.01
## quality          6.00 0.19  0.23 0.01
## color*          1.00 -1.18 -0.61 0.01

```

```

#tự code tay
descriptive_statistics <- function (data) {
  data_stats <- data |>
    summarise(across(where(is.numeric), list(
      mean = mean,
      median = median,
      sd = sd,
      min = min,
      max = max,
      iqr = IQR,
      n = ~n()
    )))

  data_stats_tidy <- data_stats |>
    gather(TEN, GIA_TRI) |>
    extract(TEN, into = c("BIEN", "THONG_KE"), regex = "^(.*)(.*$)") |>
    spread(THONG_KE, GIA_TRI) |>
    dplyr::select(BIEN,n, mean, median, sd, min, max, iqr)

  return (data_stats_tidy)
}

data_summary <- descriptive_statistics(data)
print.data.frame(data_summary, digits = 2)

```

```

##          BIEN   n  mean median   sd min  max   iqr
## 1      alcohol 6497 10.492 10.300 1.193 8.000 14.90 1.8000
## 2      chlorides 6497  0.056 0.047 0.035 0.009 0.61 0.0270
## 3      citric_acid 6497  0.319 0.310 0.145 0.000 1.66 0.1400
## 4      density 6497  0.995 0.995 0.003 0.987 1.04 0.0047
## 5      fixed_acidity 6497  7.215 7.000 1.296 3.800 15.90 1.3000
## 6      free_sulfur_dioxide 6497 30.525 29.000 17.749 1.000 289.00 24.0000
## 7      p_h 6497  3.219 3.210 0.161 2.720 4.01 0.2100
## 8      quality 6497  5.818 6.000 0.873 3.000 9.00 1.0000
## 9      residual_sugar 6497  5.443 3.000 4.758 0.600 65.80 6.3000
## 10     sulphates 6497  0.531 0.510 0.149 0.220 2.00 0.1700
## 11     total_sulfur_dioxide 6497 115.745 118.000 56.522 6.000 440.00 79.0000
## 12     volatile_acidity 6497  0.340 0.290 0.165 0.080 1.58 0.1700

```

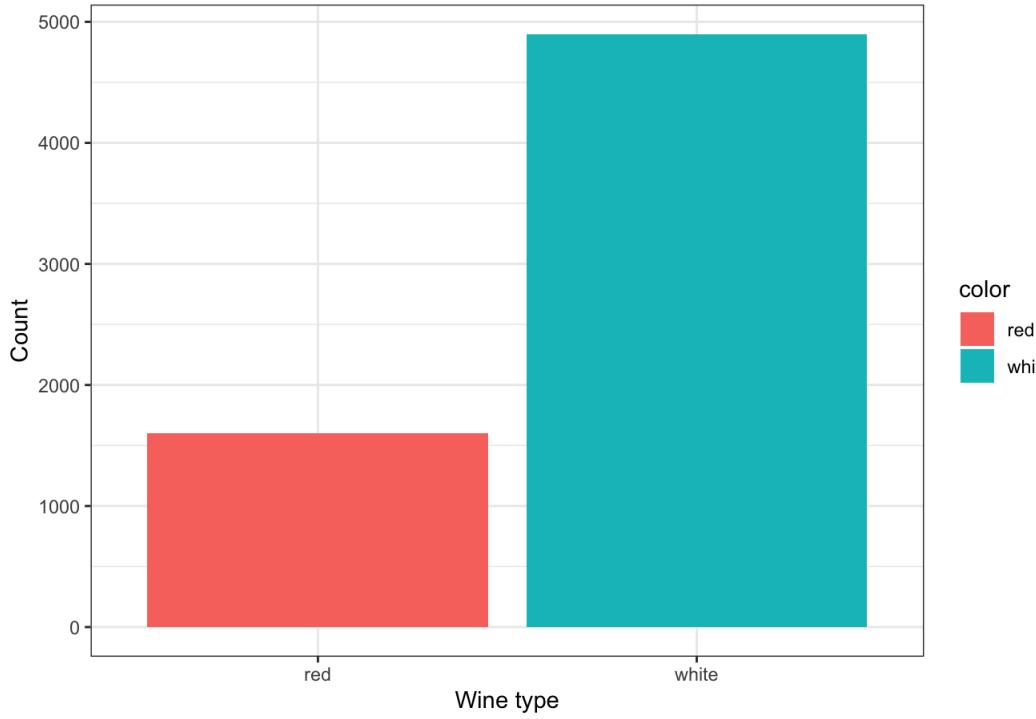
I.2. Trực quan hóa dữ liệu

I.2.1. Khảo sát phân phối của dữ liệu

a. Kiểm tra sự cân bằng của dữ liệu

```
ggplot(data, aes(x = color, fill = color)) +
  geom_bar() +
  labs(
    title = "Number of red/white wine in dataset",
    x = "Wine type",
    y = "Count"
  ) +
  theme_bw()
```

Number of red/white wine in dataset



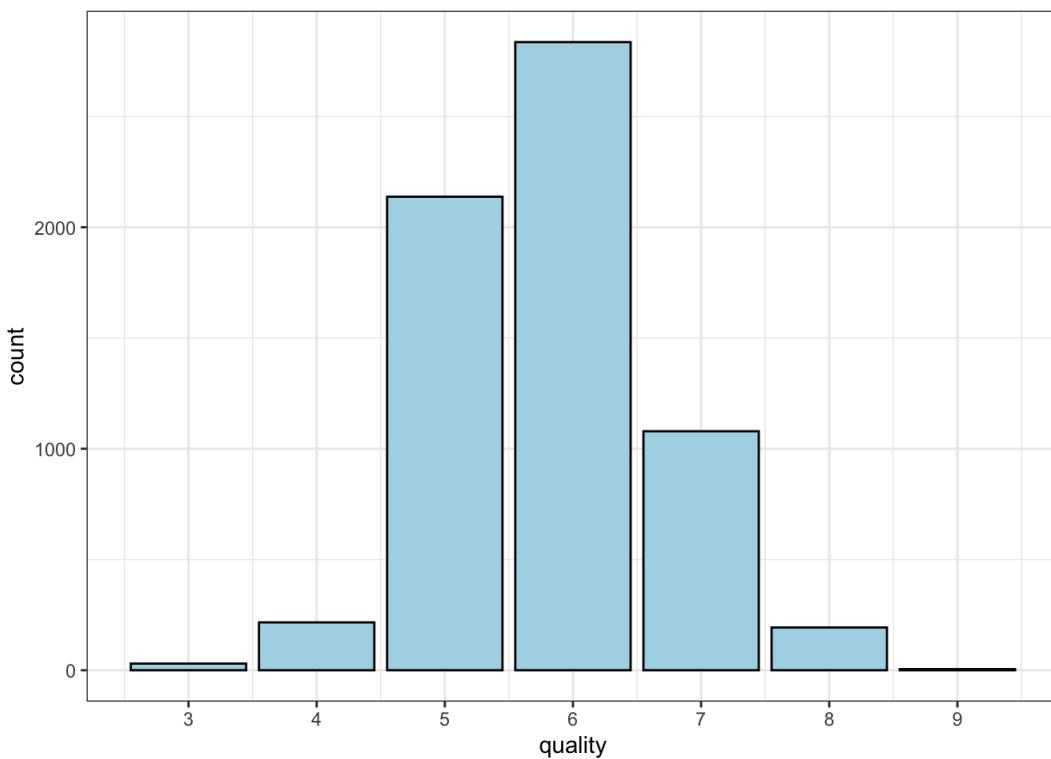
```
table(data$color)
```

```
##  
##  red white  
## 1599 4898
```

Có thể thấy số lượng mẫu lớp white nhiều gấp ba lần số lượng mẫu lớp red, nên đây là một bộ dữ liệu không cân bằng (imbalanced data) khi lượng mẫu trong các lớp không bằng nhau.

b. Kiểm tra sự phân bố của chất lượng rượu

```
ggplot(data, aes(x = quality)) +
  scale_x_continuous(breaks = seq(min(data$quality), max(data$quality), by = 1)) +
  geom_bar(fill = "lightblue", color = "black") +
  theme_bw()
```



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

```
##  
## 3 4 5 6 7 8 9  
## 30 216 2138 2836 1079 193 5
```

Đa số dữ liệu có quality trong khoảng [5,7], cực ít dữ liệu có giá trị quality bé hơn 3 và có một số ít dữ liệu có giá trị quality lớn hơn 8.

=> Để kiểm tra, ta sử dụng bootstrap để ước lượng phân phối mẫu và khoảng tin cậy cho quality và các biến khác trong tập dữ liệu.

c. Bootstrap

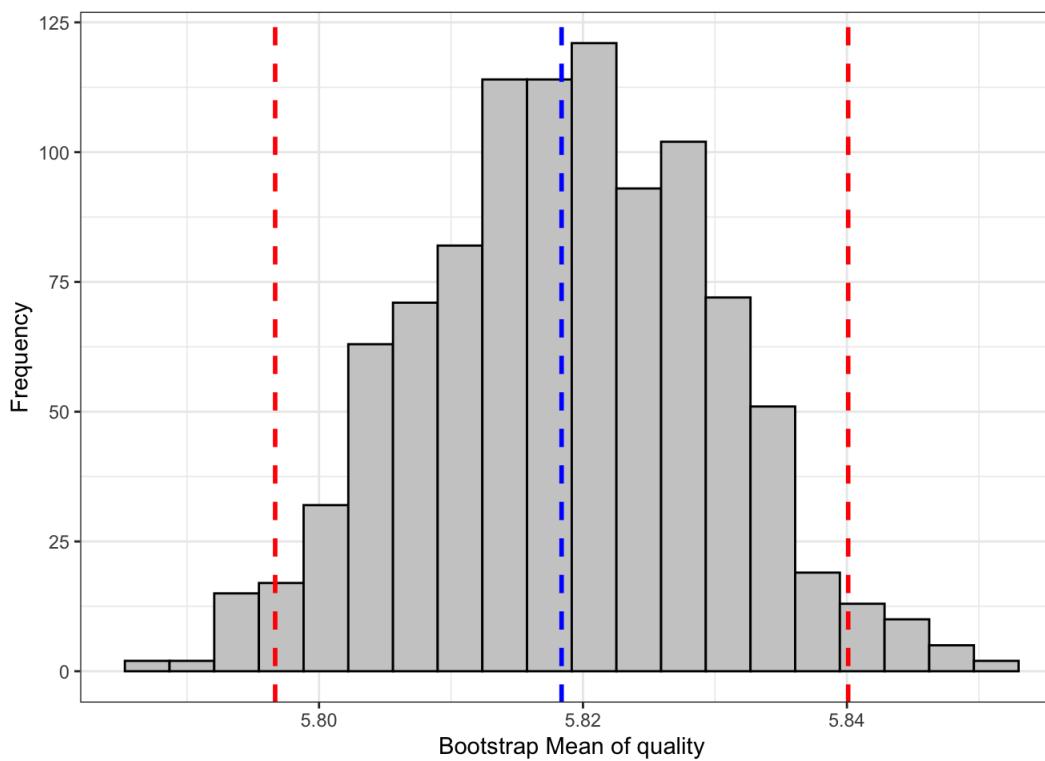
```
# Hàm thực hiện bootstrap, vẽ histogram và tính khoảng tin cậy  
plot_bootstrap_hist <- function(data, feature, R = 1000, conf = 0.95) {  
  
  boot_mu_fun <- function(data, ind) {  
    data_new <- data[ind]  
    out <- mean(data_new)  
    return(out)  
  }  
  
  # Thực hiện bootstrap  
  set.seed(34)  
  out <- boot(data[[feature]], statistic = boot_mu_fun, R = R)  
  
  # Vẽ histogram  
  p <- ggplot(data = data.frame(t = out$t), mapping = aes(x = t)) +  
    geom_histogram(fill = "gray80", color = "black", bins = 20) +  
    geom_vline(xintercept = out$0, color = "blue", linetype = "dashed", linewidth = 1) +  
    geom_vline(xintercept = quantile(out$, c((1 - conf) / 2, 1 - (1 - conf) / 2)), color = "red", linetype = "dashed", linewidth = 1) +  
    xlab(paste("Bootstrap Mean of", feature)) + ylab("Frequency") +  
    theme_bw()  
  
  boot_ci <- boot.ci(out, type = "perc", conf = conf)  
  
  return(list(plot = p, boot_ci = boot_ci))  
}
```

```
features <- colnames(data)  
features <- setdiff(features, "color")  
features
```

```
## [1] "fixed_acidity"      "volatile_acidity"   "citric_acid"  
## [4] "residual_sugar"     "chlorides"          "free_sulfur_dioxide"  
## [7] "total_sulfur_dioxide" "density"           "p_h"  
## [10] "sulphates"         "alcohol"          "quality"
```

Bootstrap cho quality

```
result <- plot_bootstrap_hist(data, "quality")
result$plot
```



```
result$boot_ci
```

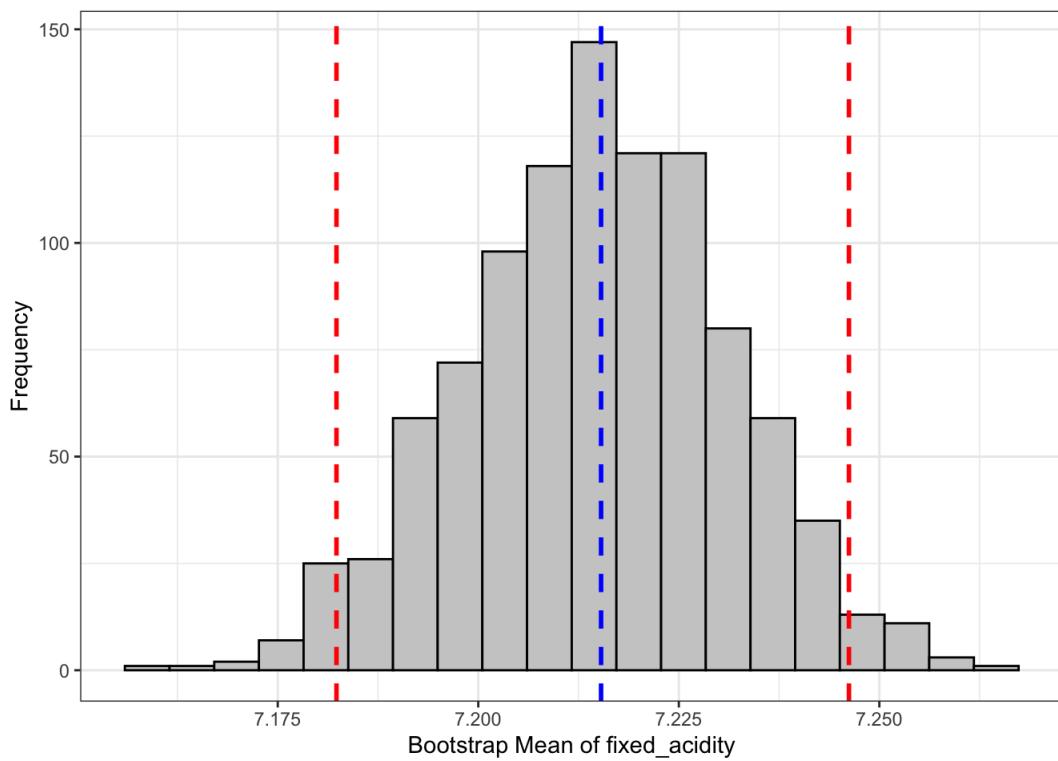
```
## BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
## Based on 1000 bootstrap replicates
##
## CALL :
## boot.ci(boot.out = out, conf = conf, type = "perc")
##
## Intervals :
## Level   Percentile
## 95%   ( 5.797, 5.841 )
## Calculations and Intervals on Original Scale
```

Có thể thấy, phân phối mẫu của trung bình mẫu của quality có dạng phân phối chuẩn, với trung bình là 5.82, và độ tin cậy 95% cho khoảng tin cậy từ 5.797 -> 5.841 của chất lượng.

Tương tự, ta cũng có thể kiểm tra phân phối mẫu và khoảng tin cậy cho các biến còn lại.

Bootstrap cho fixed_acidity

```
result <- plot_bootstrap_hist(data, "fixed_acidity")
result$plot
```

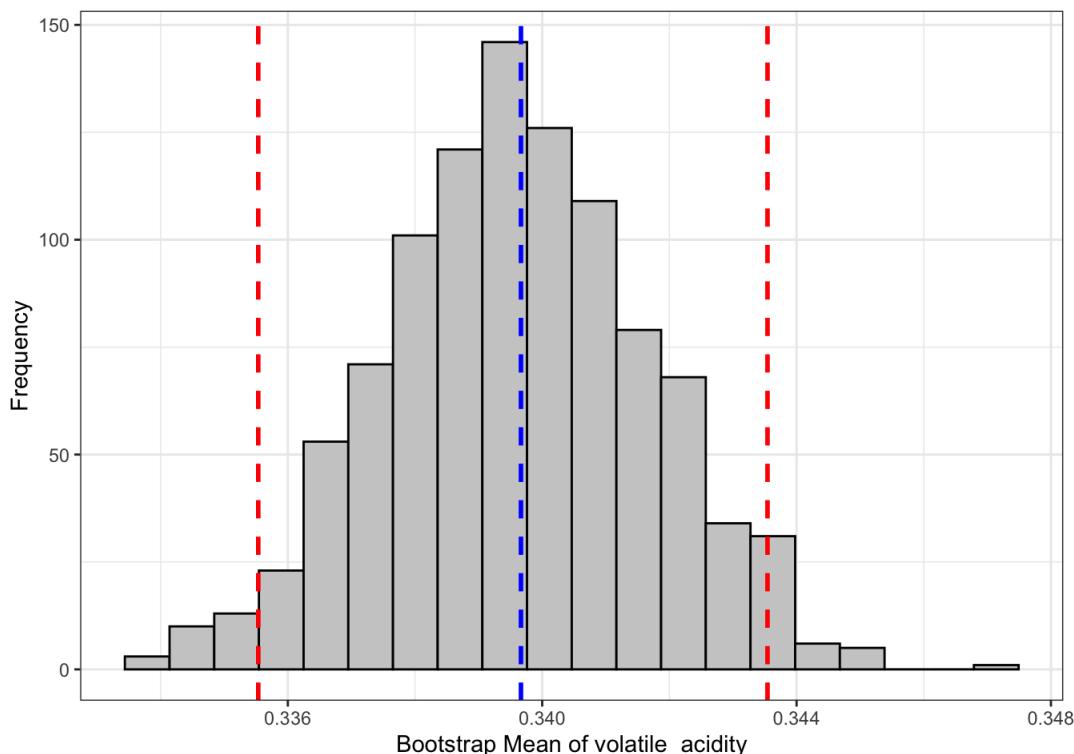


```
result$boot_ci
```

```
## BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
## Based on 1000 bootstrap replicates
##
## CALL :
## boot.ci(boot.out = out, conf = conf, type = "perc")
##
## Intervals :
## Level Percentile
## 95%  ( 7.182, 7.247 )
## Calculations and Intervals on Original Scale
```

Bootstrap cho volatile_acidity

```
result <- plot_bootstrap_hist(data, "volatile_acidity")
result$plot
```



```
result$boot_ci
```

```

## BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
## Based on 1000 bootstrap replicates
##
## CALL :
## boot.ci(boot.out = out, conf = conf, type = "perc")
##
## Intervals :
## Level Percentile
## 95%  ( 0.3355, 0.3436 )
## Calculations and Intervals on Original Scale

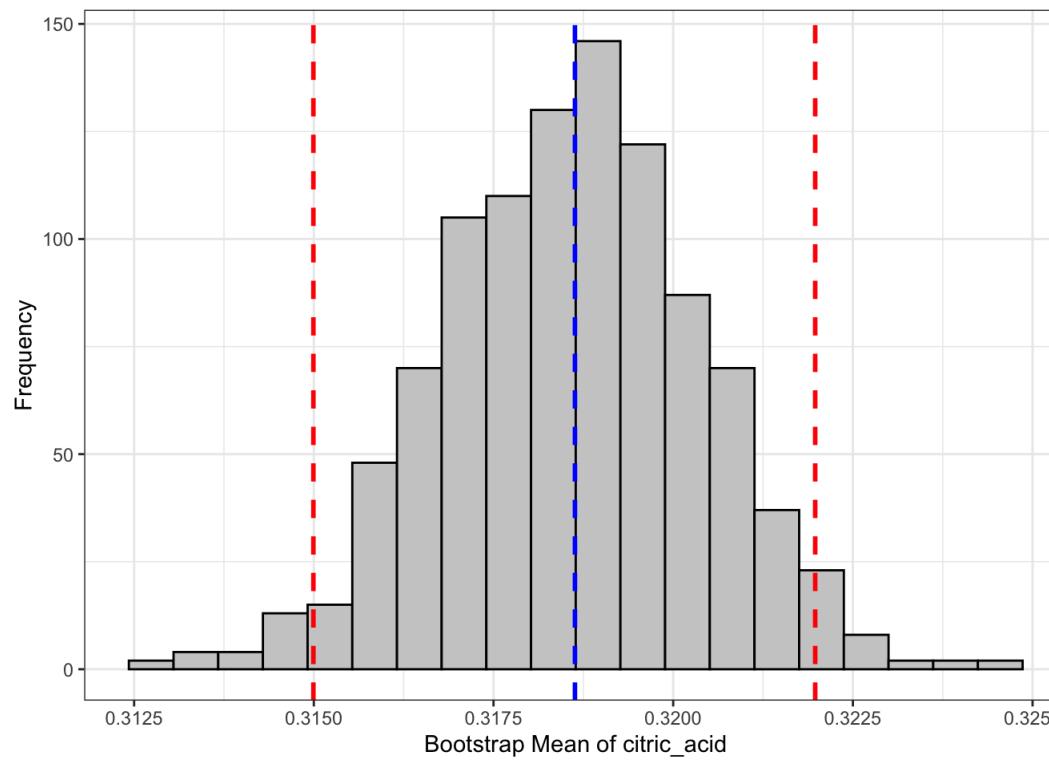
```

Bootstrap cho citric_acid

```

result <- plot_bootstrap_hist(data, "citric_acid")
result$plot

```



```
result$boot_ci
```

```

## BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
## Based on 1000 bootstrap replicates
##
## CALL :
## boot.ci(boot.out = out, conf = conf, type = "perc")
##
## Intervals :
## Level Percentile
## 95%  ( 0.315, 0.322 )
## Calculations and Intervals on Original Scale

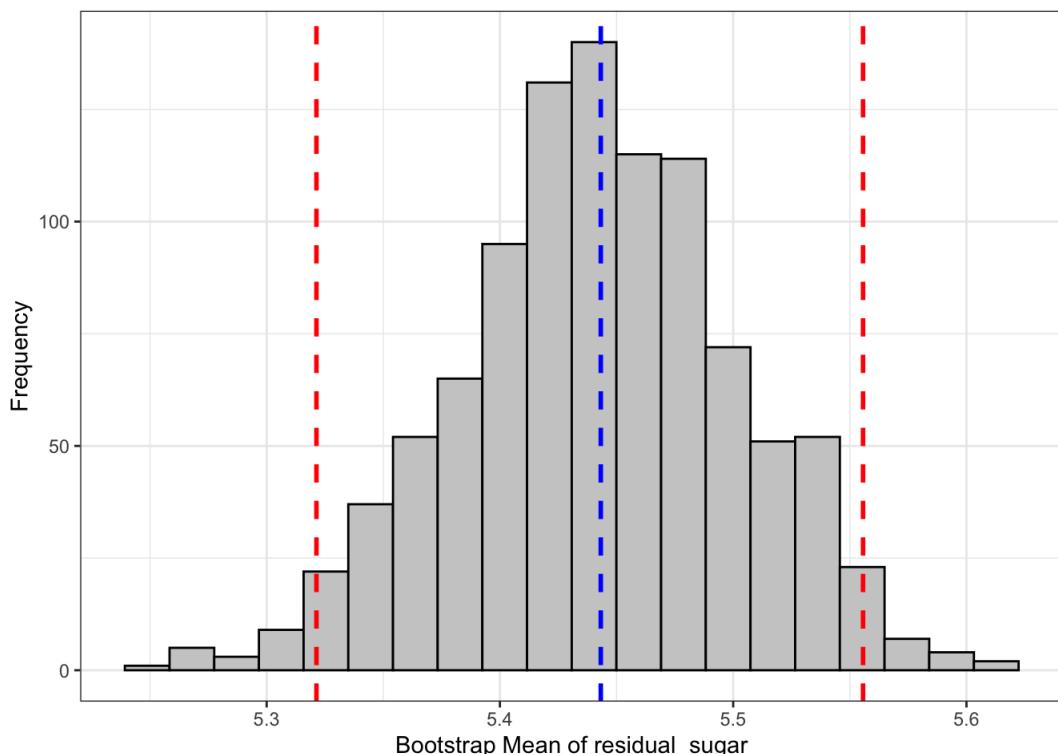
```

Bootstrap cho residual_sugar

```

result <- plot_bootstrap_hist(data, "residual_sugar")
result$plot

```

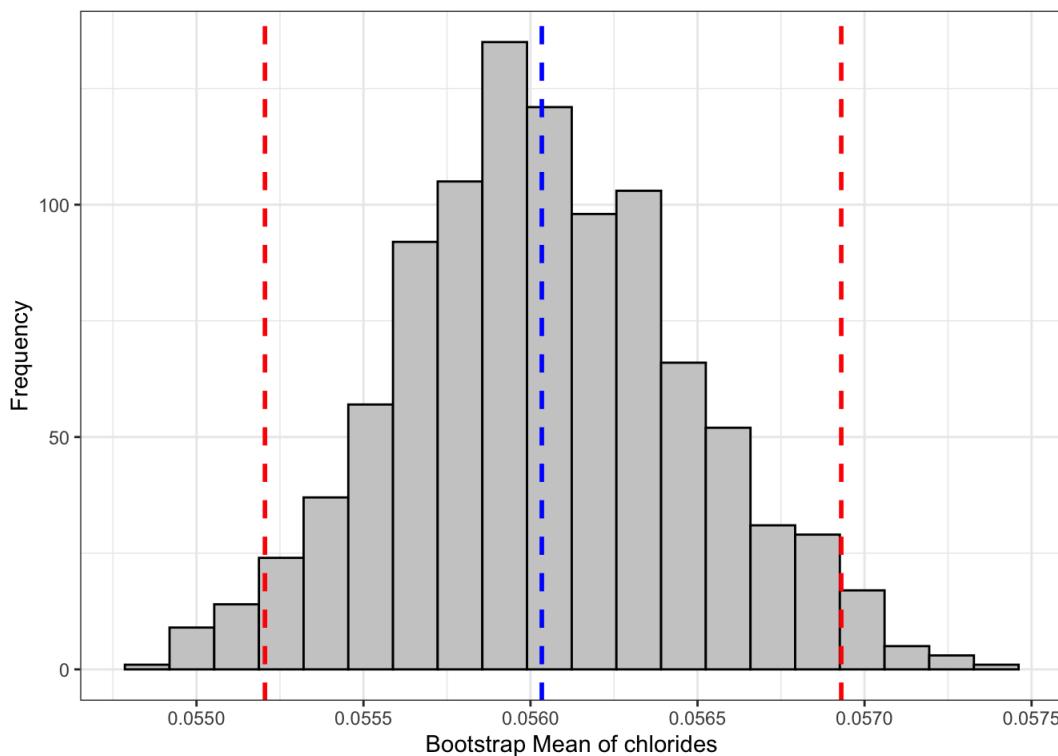


```
result$boot_ci
```

```
## BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
## Based on 1000 bootstrap replicates
##
## CALL :
## boot.ci(boot.out = out, conf = conf, type = "perc")
##
## Intervals :
## Level Percentile
## 95%  ( 5.321, 5.556 )
## Calculations and Intervals on Original Scale
```

Bootstrap cho chlorides

```
result <- plot_bootstrap_hist(data, "chlorides")
result$plot
```



```
result$boot_ci
```

```

## BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
## Based on 1000 bootstrap replicates
##
## CALL :
## boot.ci(boot.out = out, conf = conf, type = "perc")
##
## Intervals :
## Level Percentile
## 95% ( 0.0552, 0.0569 )
## Calculations and Intervals on Original Scale

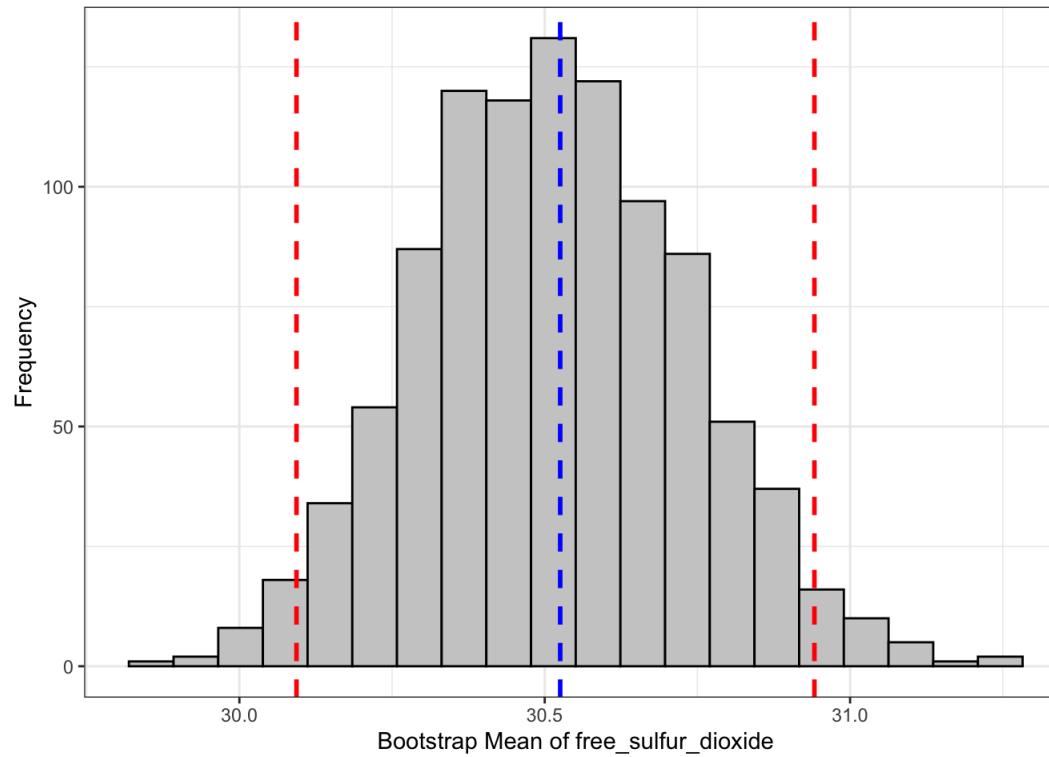
```

Bootstrap cho free_sulfur_dioxide

```

result <- plot_bootstrap_hist(data, "free_sulfur_dioxide")
result$plot

```



```
result$boot_ci
```

```

## BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
## Based on 1000 bootstrap replicates
##
## CALL :
## boot.ci(boot.out = out, conf = conf, type = "perc")
##
## Intervals :
## Level Percentile
## 95% ( 30.09, 30.94 )
## Calculations and Intervals on Original Scale

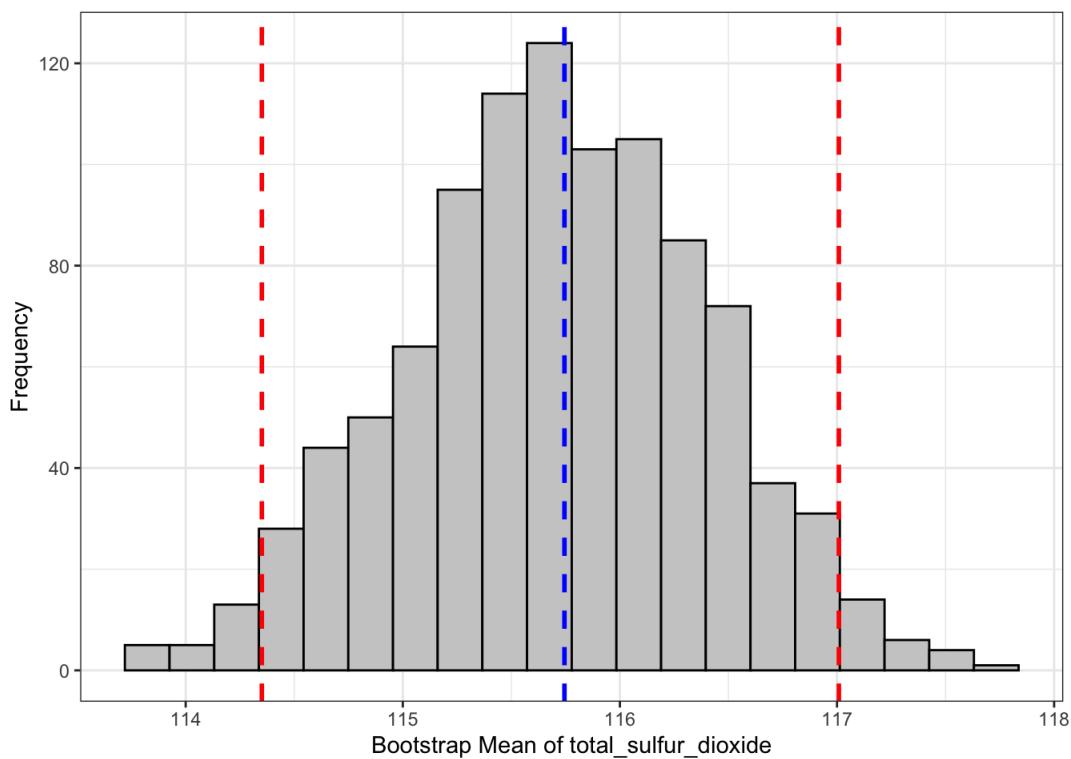
```

Bootstrap cho total_sulfur_dioxide

```

result <- plot_bootstrap_hist(data, "total_sulfur_dioxide")
result$plot

```

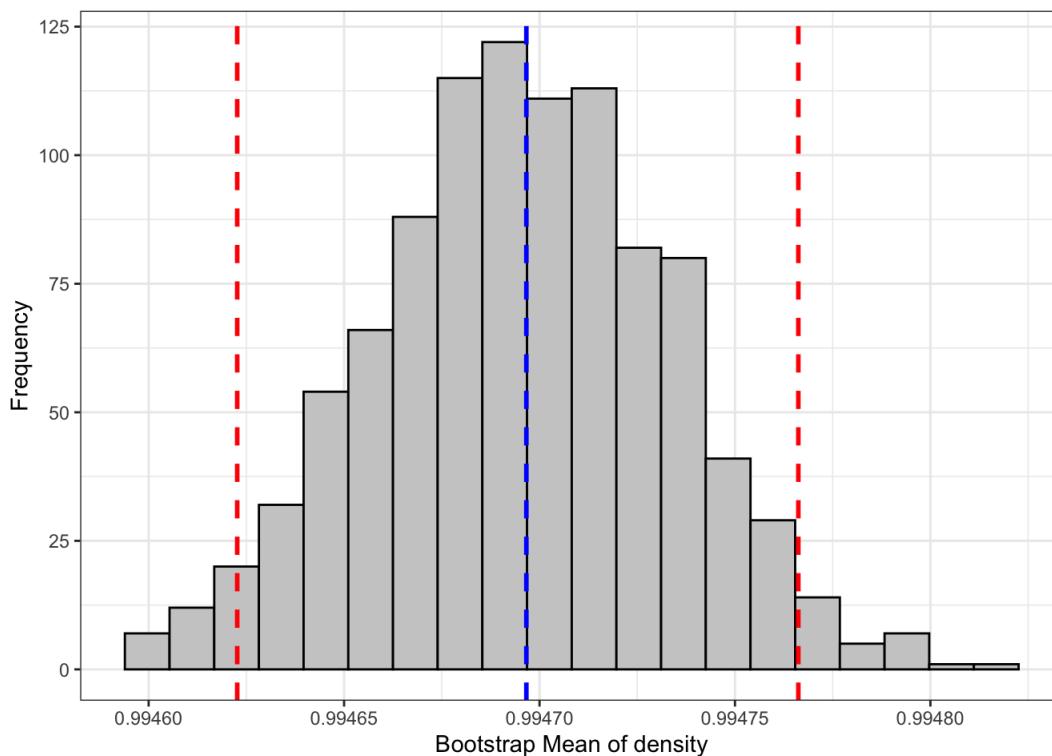


```
result$boot_ci
```

```
## BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
## Based on 1000 bootstrap replicates
##
## CALL :
## boot.ci(boot.out = out, conf = conf, type = "perc")
##
## Intervals :
## Level Percentile
## 95% (114.3, 117.0 )
## Calculations and Intervals on Original Scale
```

Bootstrap cho density

```
result <- plot_bootstrap_hist(data, "density")
result$plot
```



```
result$boot_ci
```

```

## BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
## Based on 1000 bootstrap replicates
##
## CALL :
## boot.ci(boot.out = out, conf = conf, type = "perc")
##
## Intervals :
## Level Percentile
## 95%  ( 0.9946, 0.9948 )
## Calculations and Intervals on Original Scale

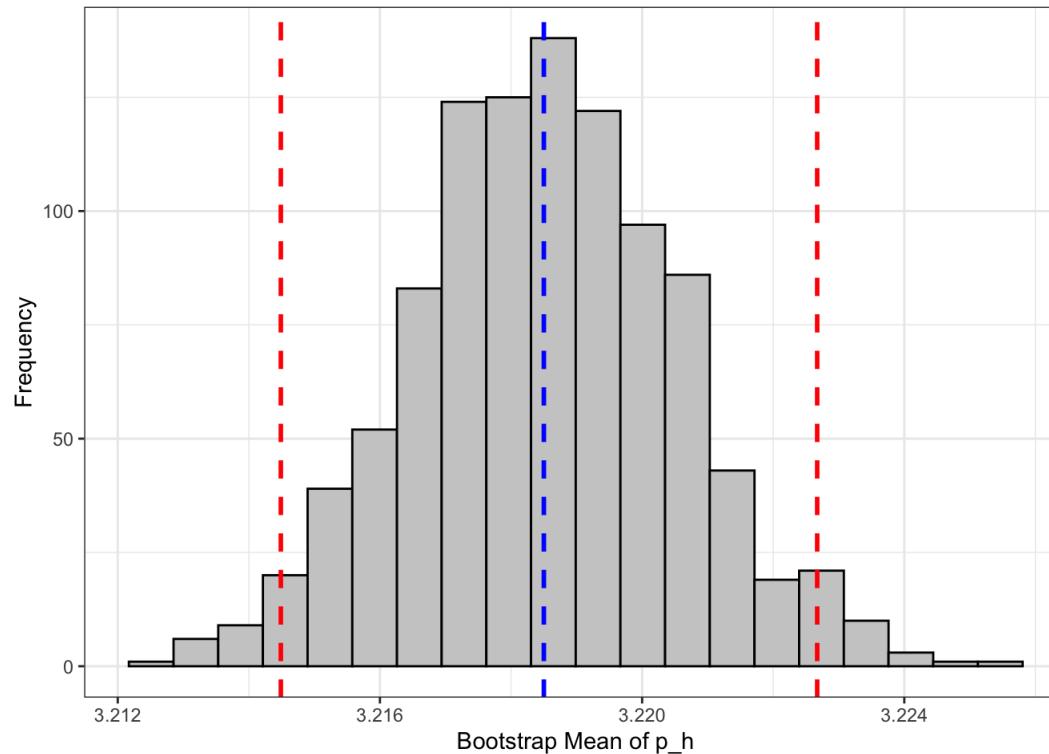
```

Bootstrap cho p_h

```

result <- plot_bootstrap_hist(data, "p_h")
result$plot

```



```
result$boot_ci
```

```

## BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
## Based on 1000 bootstrap replicates
##
## CALL :
## boot.ci(boot.out = out, conf = conf, type = "perc")
##
## Intervals :
## Level Percentile
## 95%  ( 3.214, 3.223 )
## Calculations and Intervals on Original Scale

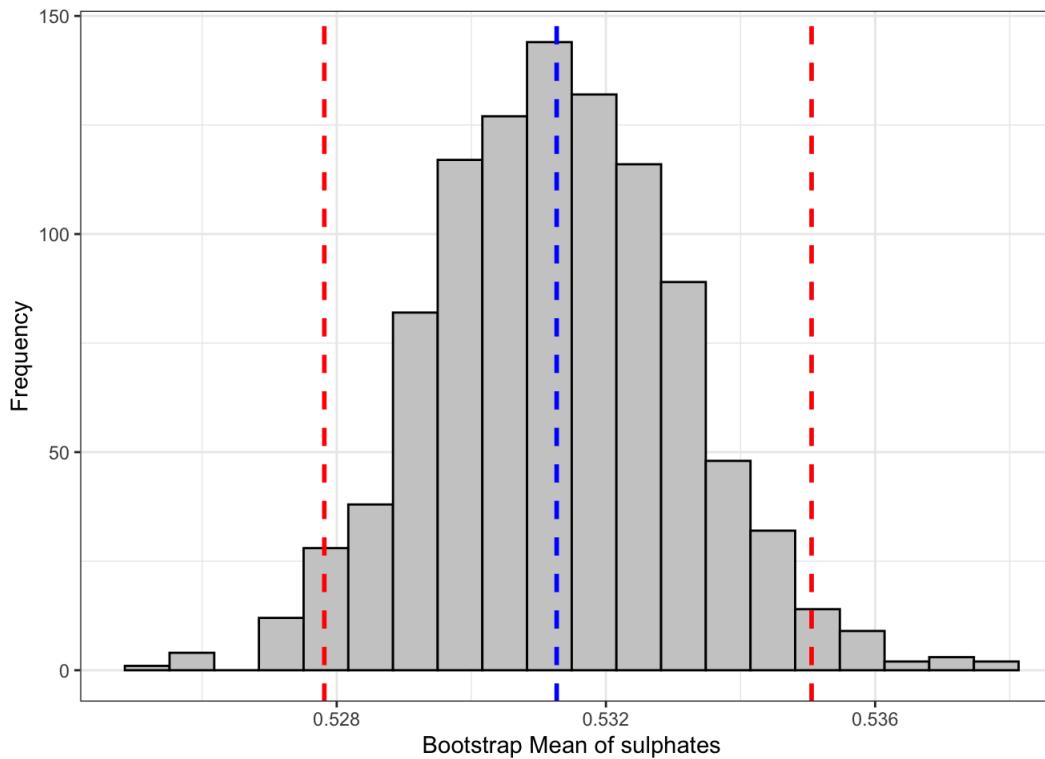
```

Bootstrap cho sulphates

```

result <- plot_bootstrap_hist(data, "sulphates")
result$plot

```

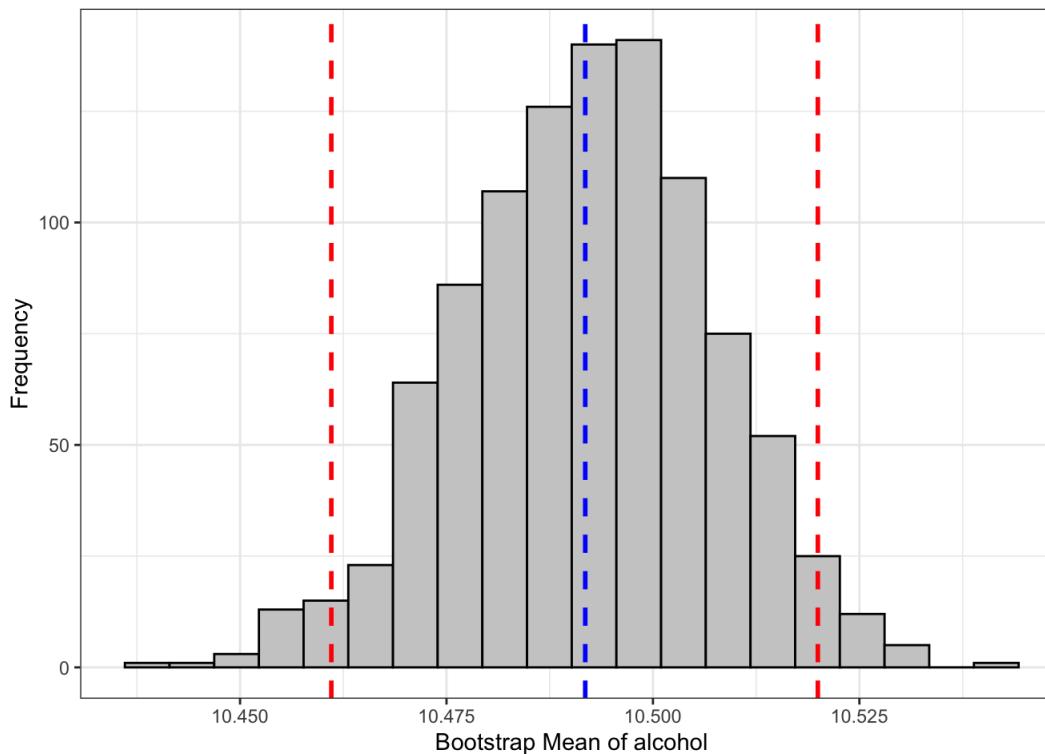


```
result$boot_ci
```

```
## BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
## Based on 1000 bootstrap replicates
##
## CALL :
## boot.ci(boot.out = out, conf = conf, type = "perc")
##
## Intervals :
## Level Percentile
## 95%  ( 0.5278, 0.5351 )
## Calculations and Intervals on Original Scale
```

Bootstrap cho alcohol

```
result <- plot_bootstrap_hist(data, "alcohol")
result$plot
```



```
result$boot_ci
```

```

## BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
## Based on 1000 bootstrap replicates
##
## CALL :
## boot.ci(boot.out = out, conf = conf, type = "perc")
##
## Intervals :
## Level Percentile
## 95% (10.46, 10.52)
## Calculations and Intervals on Original Scale

```

d. Histogram và Density

```

plot_distribution <- function(data, feature) {
  mean_value <- mean(data[[feature]], na.rm = TRUE)
  
  ggplot(data, aes_string(x = feature)) +
    geom_histogram(aes(y = ..density.., fill = "Histogram"), color = "black", alpha = 0.7, bins = 65) +
    geom_density(aes(color = "Density"), bw = "nrd0", kernel = "gaussian") +
    geom_vline(aes(xintercept = mean_value, color = "Mean Value"), linetype = "dashed", size = 1) +
    scale_fill_manual(name = "Legend", values = c("Histogram" = "white")) +
    scale_color_manual(name = "Legend", values = c("Density" = "blue", "Mean Value" = "red")) +
    labs(title = paste("Explore the distribution for", feature),
        x = feature,
        y = "Density") +
    theme(legend.position = "top")
}

```

```
plot_distribution(data, "fixed_acidity")
```

```

## Warning: `aes_string()` was deprecated in ggplot2 3.0.0.
## i Please use tidy evaluation idioms with `aes()`.

## i See also `vignette("ggplot2-in-packages")` for more information.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.

```

```

## Warning: Using `size` aesthetic for lines was deprecated in ggplot2 3.4.0.
## i Please use `linewidth` instead.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.

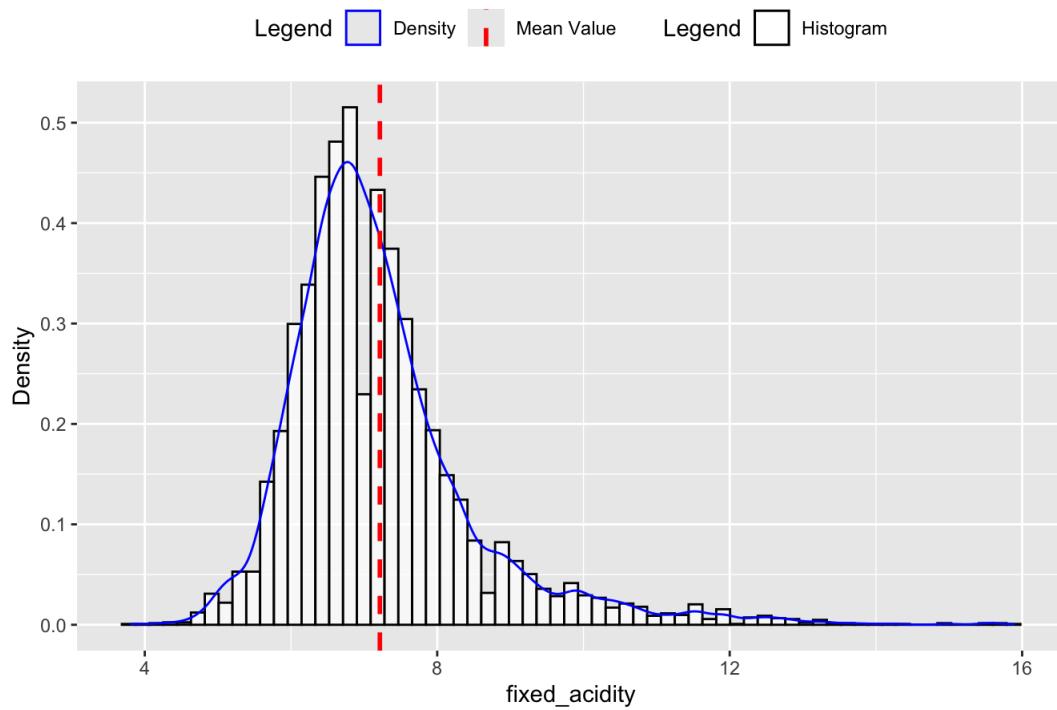
```

```

## Warning: The dot-dot notation (`..density..`) was deprecated in ggplot2 3.4.0.
## i Please use `after_stat(density)` instead.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.

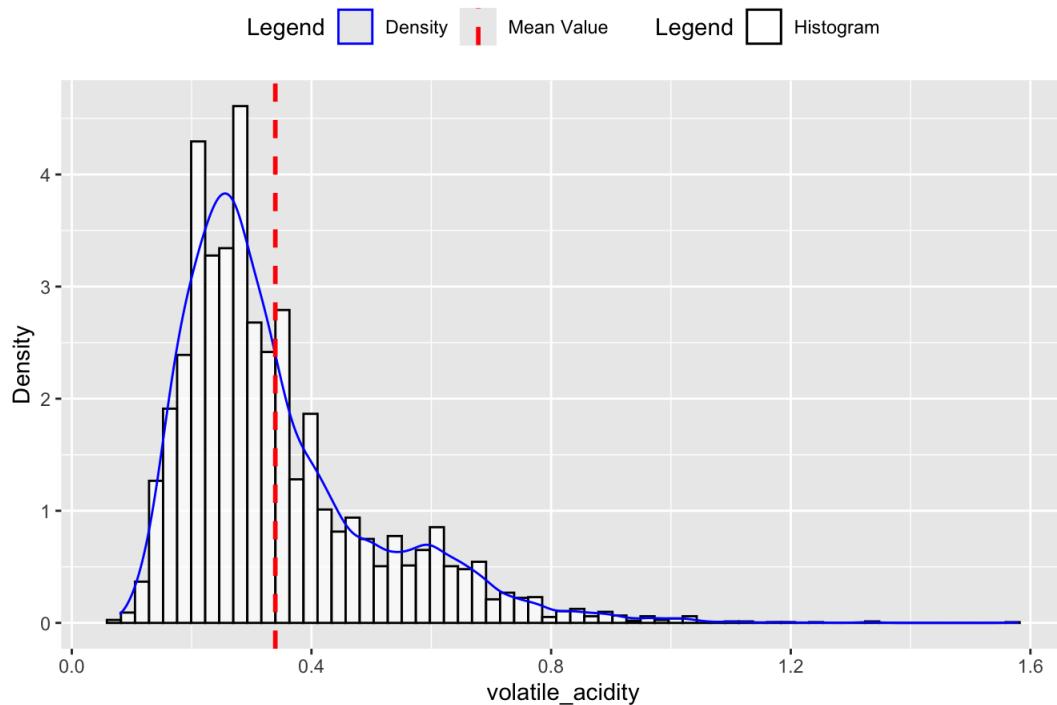
```

Explore the distribution for fixed_acidity



```
plot_distribution(data, "volatile_acidity")
```

Explore the distribution for volatile_acidity

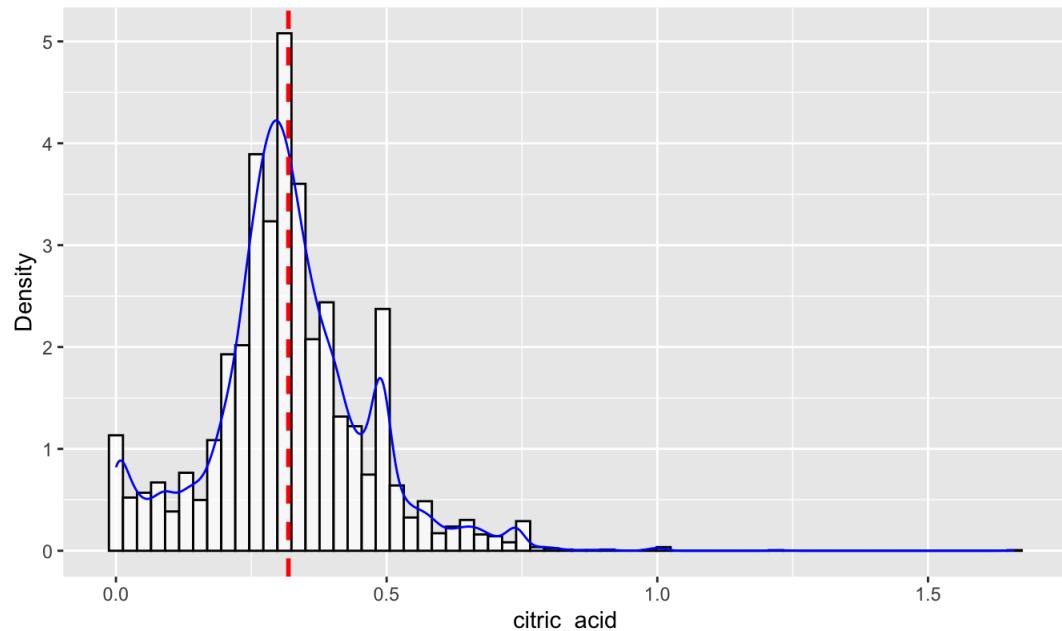


Hai biến fixed và volatile acidity có phân phối lệch phải với đuôi dài.

```
plot_distribution(data, "citric_acid")
```

Explore the distribution for citric_acid

Legend █ Density █ Mean Value Legend █ Histogram

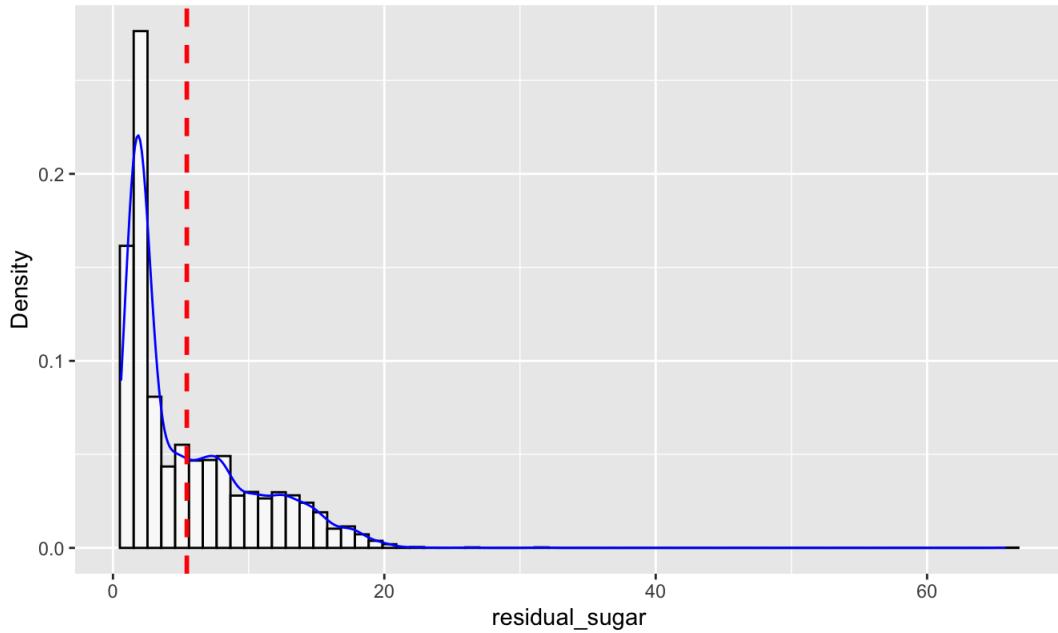


Trong khi đó, phân phối của citric acid có hình dạng 2 đỉnh (có thể do hai loại rượu có nồng độ círc khác nhau) và một số outliers ở rìa. Một vài nồng độ cao bất thường từ khoảng 0.0 -> 0.5 thể hiện có một vài nồng độ cụ thể phổ biến hơn các nồng độ khác.

```
plot_distribution(data, "residual_sugar")
```

Explore the distribution for residual_sugar

Legend █ Density █ Mean Value Legend █ Histogram

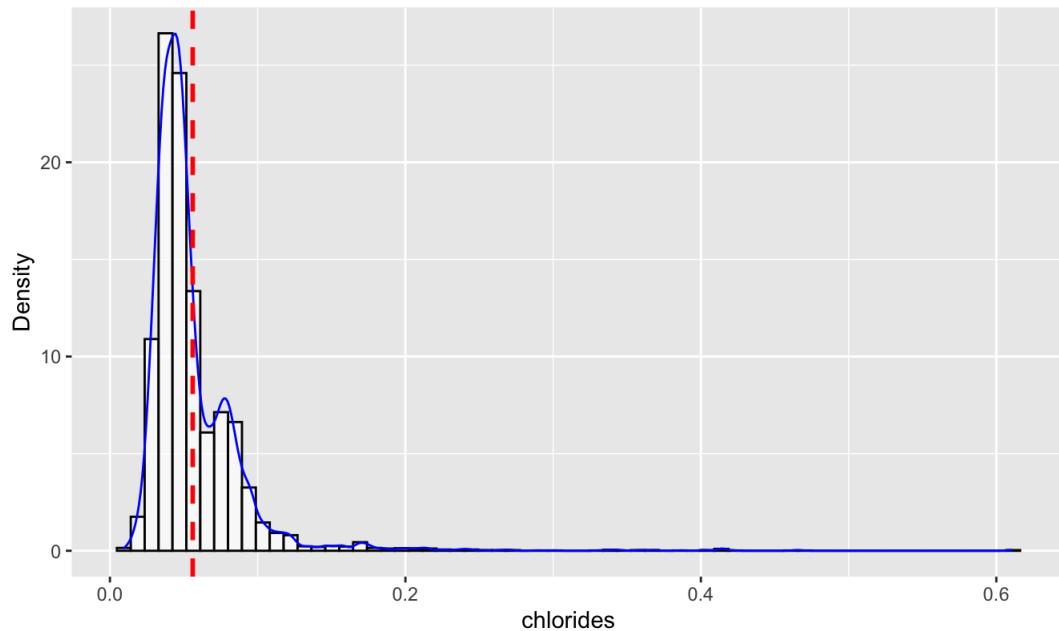


Phân phối của residual sugar bị lệch phải nặng.

```
plot_distribution(data, "chlorides")
```

Explore the distribution for chlorides

Legend Density Mean Value Legend Histogram

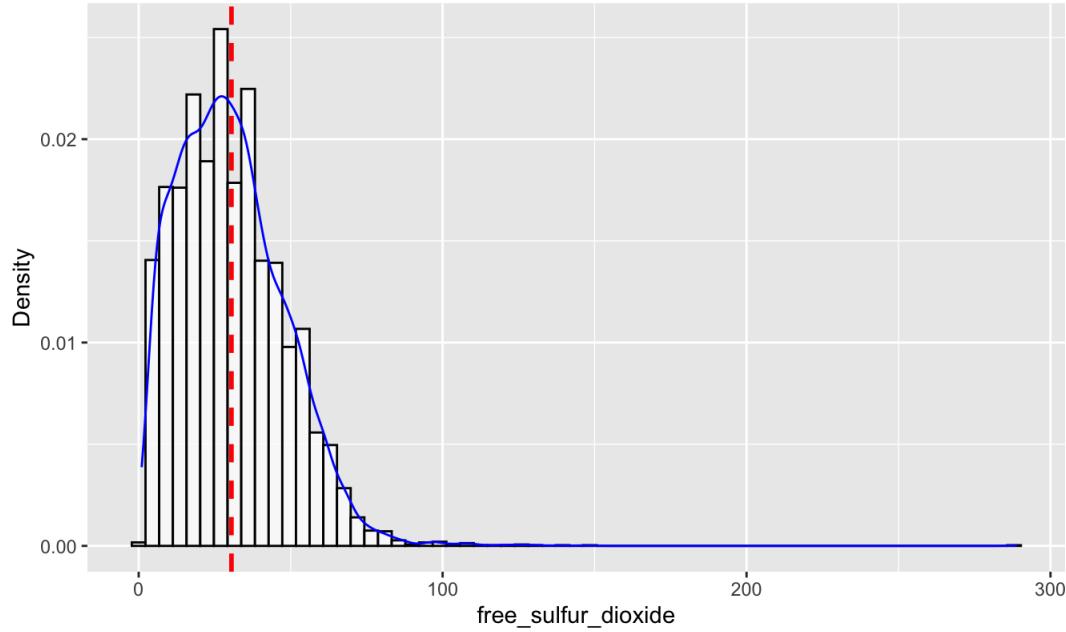


Phân phối của chlorides có 2 đỉnh có thể ứng với xu hướng của hai loại rượu, có đuôi dài từ 0.3 -> 10, có thể có sự xuất hiện của outliers.

```
plot_distribution(data, "free_sulfur_dioxide")
```

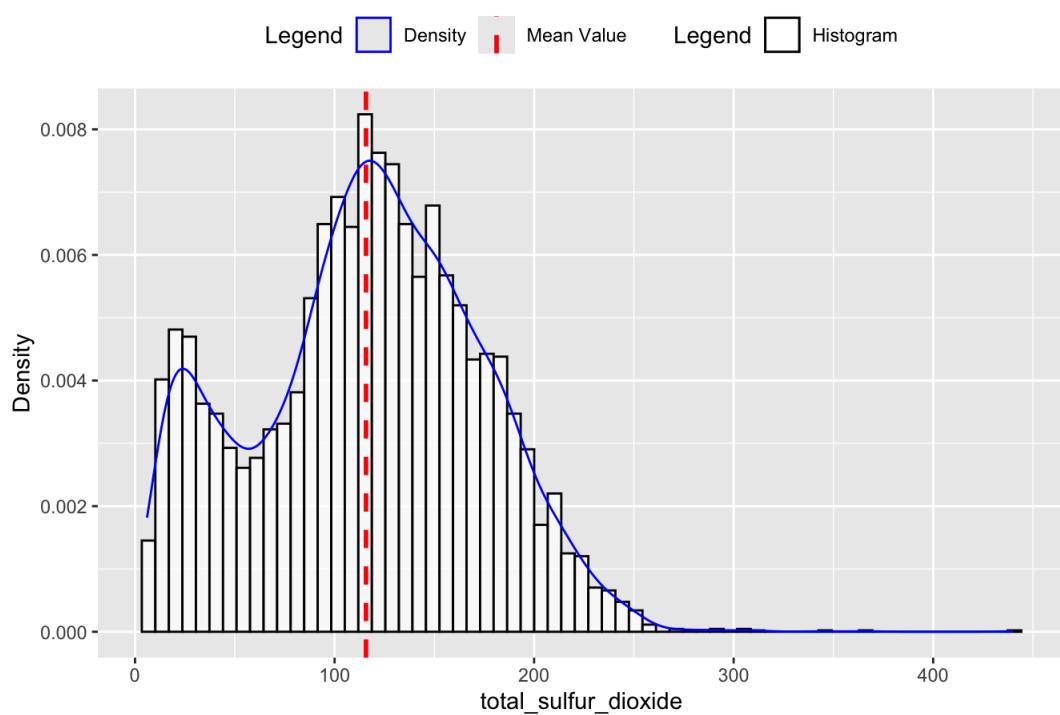
Explore the distribution for free_sulfur_dioxide

Legend Density Mean Value Legend Histogram



```
plot_distribution(data, "total_sulfur_dioxide")
```

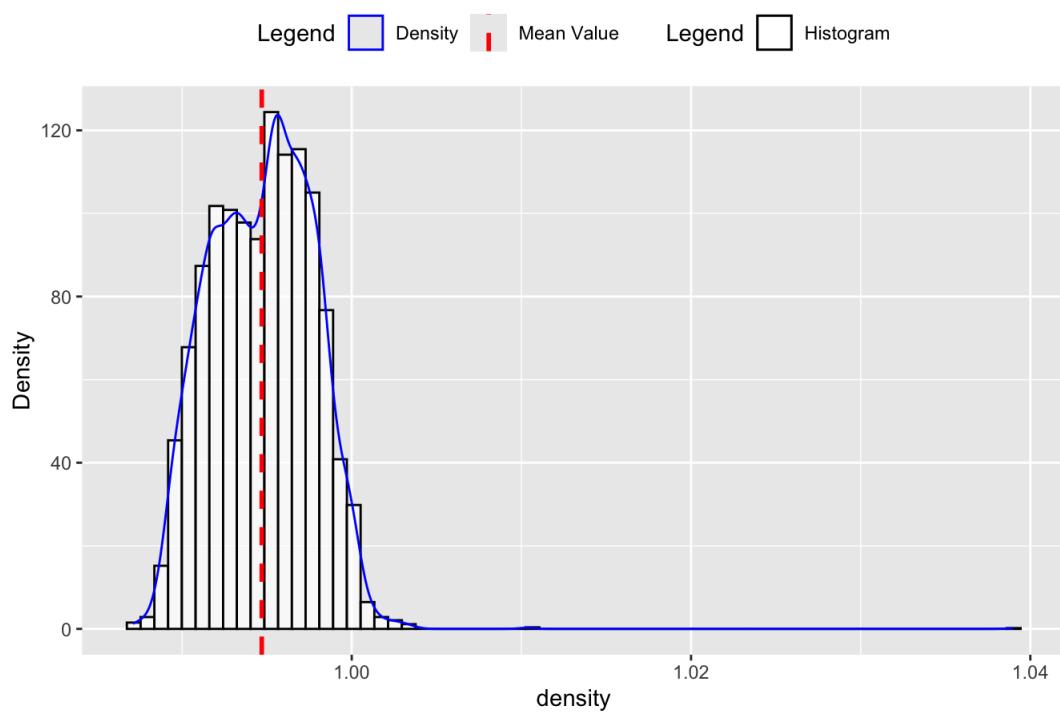
Explore the distribution for total_sulfur_dioxide



Cả hai phân phối free và total SO₂ đều bị lệch phải. Trong đó: - free SO₂ có nhiều dạng gai (dĩnh con) thể hiện một số nồng độ SO₂ phổ biến hơn phần còn lại. - total SO₂ có hai dĩnh rõ rệt ứng với hai loại rượu.

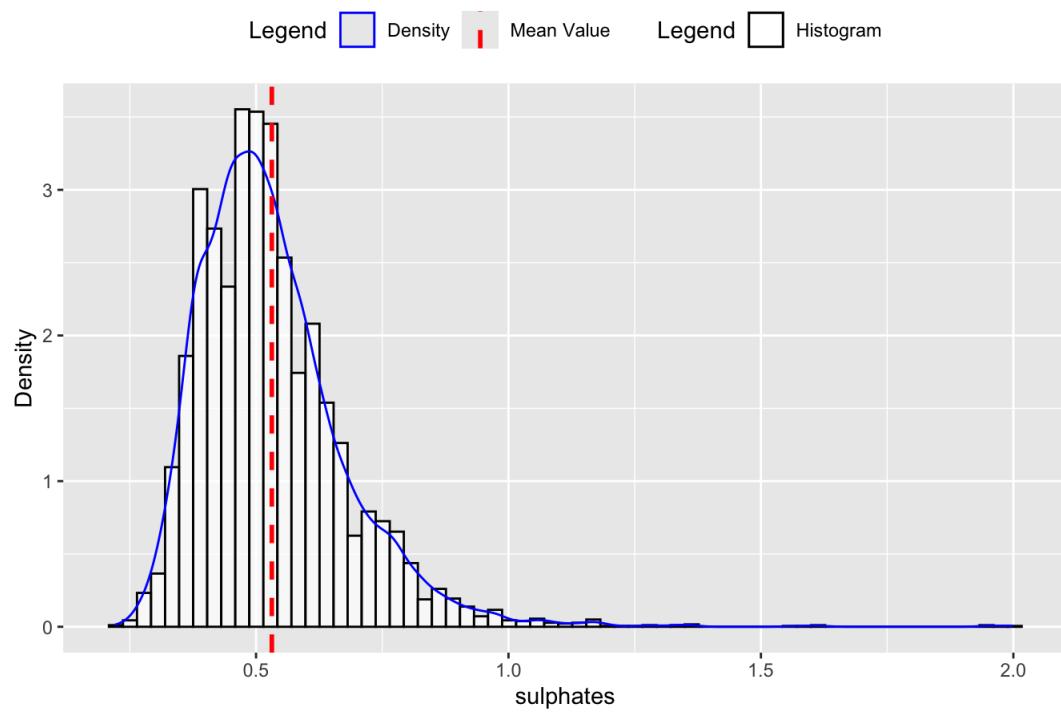
```
plot_distribution(data, "density")
```

Explore the distribution for density



```
plot_distribution(data, "sulphates")
```

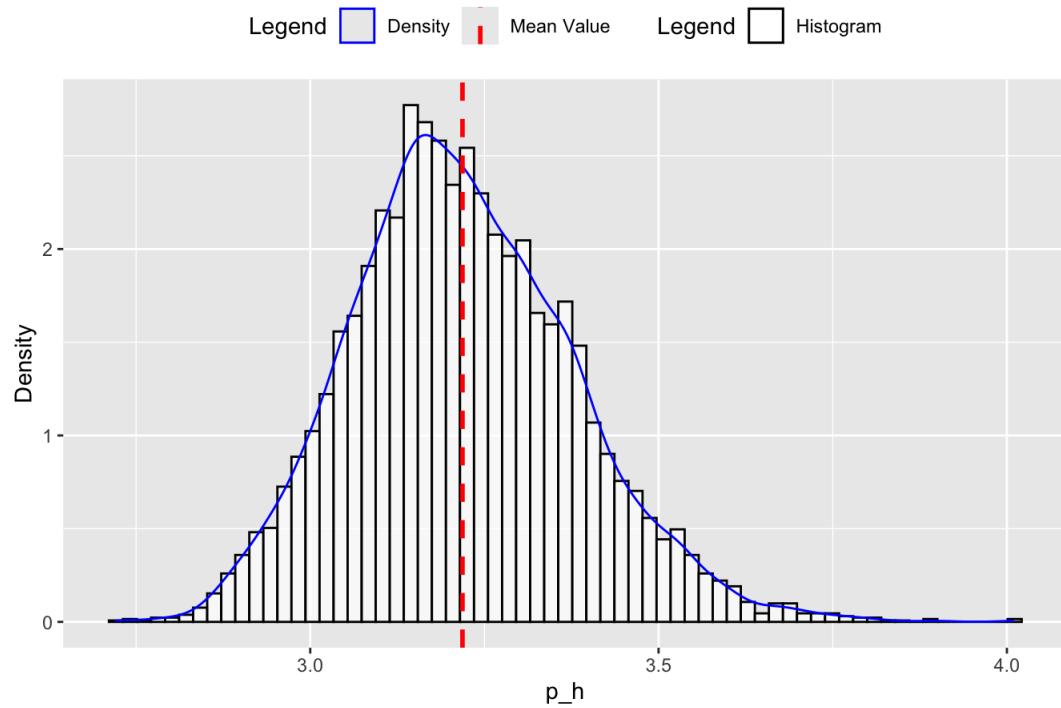
Explore the distribution for sulphates



Density và Sulphates có đuôi dài (có thể là outliers).

```
plot_distribution(data, "p_h")
```

Explore the distribution for p_h

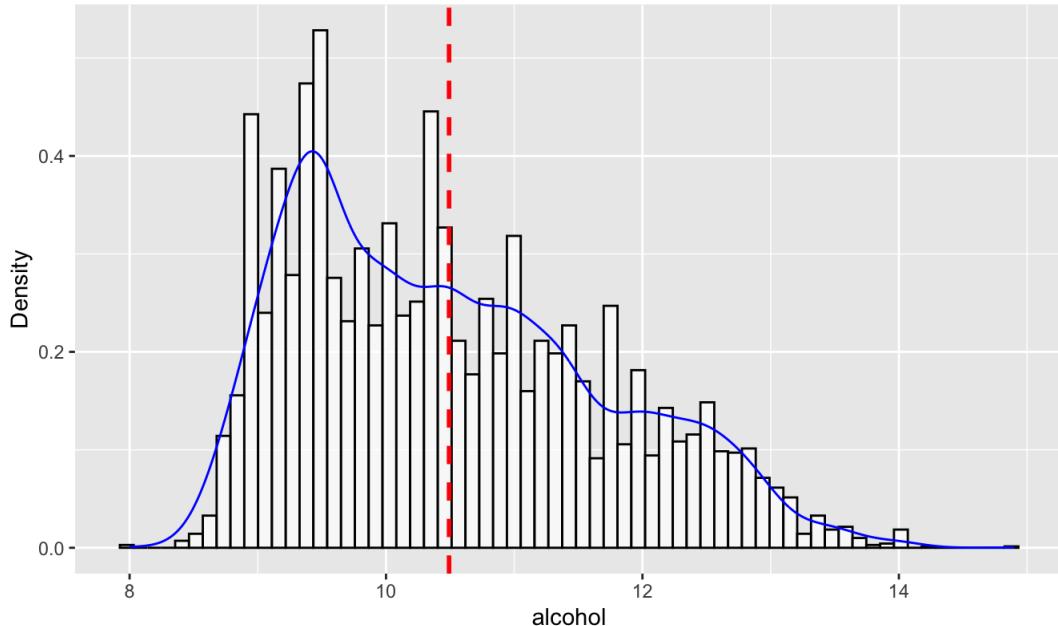


pH có phân phối gần giống phân phối chuẩn, cho thấy hai loại rượu có nồng độ pH gần giống nhau.

```
plot_distribution(data, "alcohol")
```

Explore the distribution for alcohol

Legend █ Density █ Mean Value Legend █ Histogram

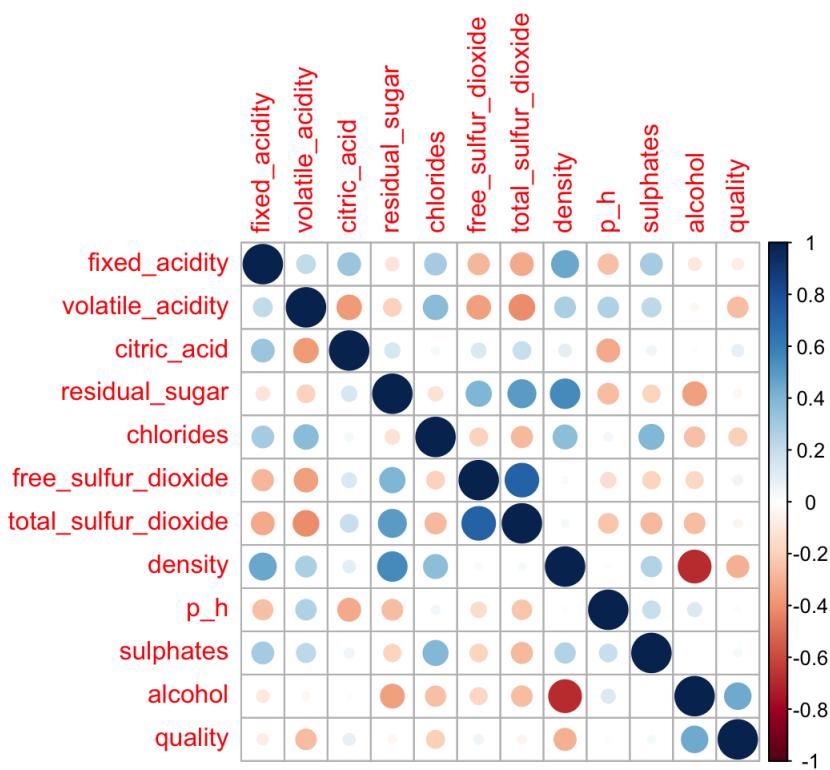


Alcohol bị lệch phải nhẹ, với một số nồng độ cao bất thường so với phần còn lại

e. Sự tương quan giữa các biến

```
numeric_columns <- data |> dplyr::select(where(is.numeric))

corr_matrix <- cor(numeric_columns, method = "pearson")
corplot(corr_matrix)
```



Một số nhận xét có thể rút ra như sau:

- Chất lượng rượu tương quan mạnh với nồng độ cồn (alcohol) và mật độ (density). Tuy nhiên, cồn và mật độ lại có mối tương quan âm cao với nhau, vì vậy chỉ cần giữ lại một trong hai biến này để dự đoán chất lượng rượu. Theo góc nhìn hóa học, việc giữ lại lượng cồn là lựa chọn hợp lý hơn để dự đoán chất lượng rượu.
- Chất lượng rượu có mối tương quan âm với volatile acidity, do nồng độ cao của nó có thể dẫn đến hương vị giấm trong rượu.
- Free SO₂ và total SO₂ có mối tương quan dương mạnh với nhau và có tương quan âm với volatile acidity, phản ánh việc SO₂ được thêm vào rượu để ngăn chặn sự hình thành axit axetic.
- pH tương quan âm với fixed acidity, citric acid, total SO₂ và residual sugar (vì đường làm giảm độ axit).
- pH tương quan dương với volatile acidity (điều này rất lạ vì thông thường pH càng cao thì độ axit phải càng giảm).
- Residual sugar và density có mối tương quan dương, điều này có thể giải thích bởi việc thêm đường vào rượu làm tăng mật độ của nó.
- Sulphates được thêm vào rượu để sản xuất SO₂, nhưng tổng lượng SO₂ và sulphates lại có mối tương quan âm, có thể là do quá trình chuyển đổi giữa hai hợp chất này.

- Nhìn chung, đa số các feature trong tập dữ liệu không có tương quan mạnh với chất lượng rượu. Tuy nhiên, có một số biến có thể xem là có ảnh hưởng đáng kể ($corr > 0.3$) lên quality so với các biến còn lại, có thể kể đến như: mật độ (density), hàm lượng cồn (alcohol), nồng độ clorua (chlorides) và độ axit dễ bay hơi (volatile_acidity). Trong số các feature này, hàm lượng và mật độ cồn cho thấy mối tương quan mạnh nhất với chất lượng rượu, điều này khá ngạc nhiên vì các thành phần hóa học thường mới là các yếu tố được xem xét kỹ lưỡng trong quá trình sản xuất rượu vang.

Ngoài ra, ta cũng có thể nhận thấy sự khác biệt thú vị giữa màu rượu và hàm lượng sulfur dioxide cũng như giữa màu rượu và độ axit dễ bay hơi. Vì độ axit dễ bay hơi cũng ảnh hưởng đến chất lượng rượu vang, nên ta phải kiểm tra xem màu sắc ảnh hưởng đến mối quan hệ đó như thế nào và liệu chất lượng rượu có bị ảnh hưởng bởi các thông số tương tự nhau ở cả hai loại rượu vang đỏ và vang trắng hay không.

f. Phân tích sâu hơn về mối tương quan giữa các biến với quality

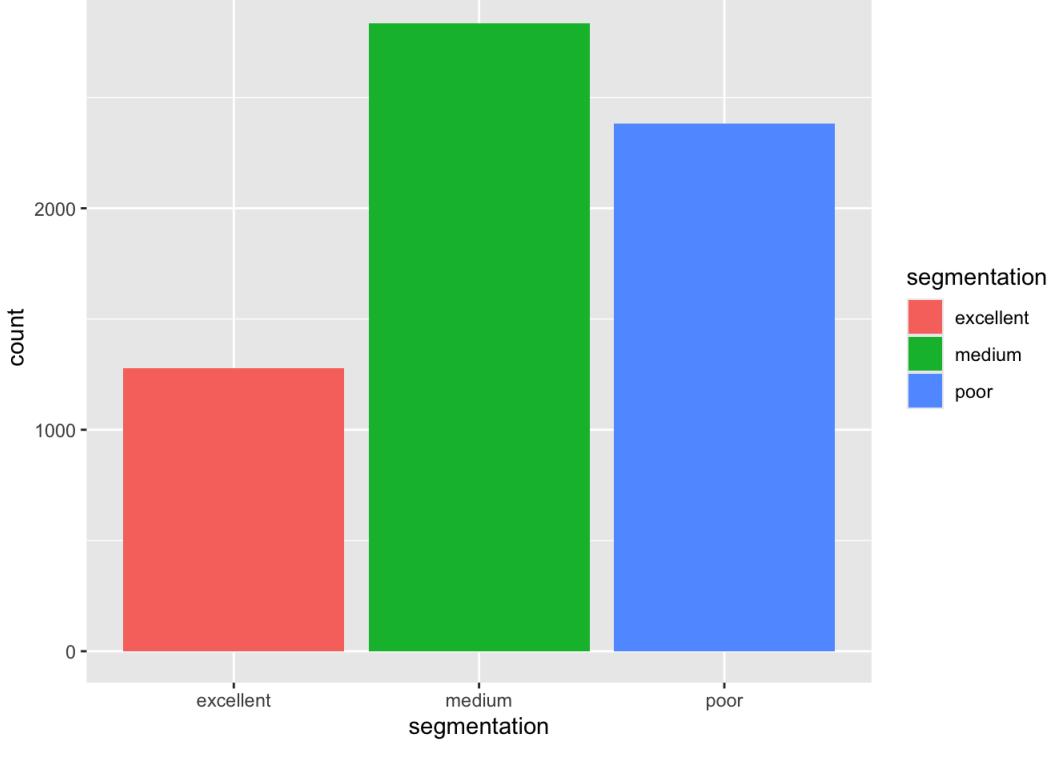
Tạo một feature mới rời rạc hóa quality, nhận vào ba giá trị tương ứng với 3 phân khúc, được chia dựa vào điểm đánh giá chất lượng (quality) như sau: - poor (phân khúc tầm thấp): có điểm đánh giá từ 3 - 5. - medium (phân khúc tầm trung): có điểm đánh giá bằng 6. - excellent (phân khúc cao cấp): có điểm đánh giá từ 7 trở lên.

```
# Tạo các nhóm mới
data$segmentation <- ifelse(data$quality > 6, 'excellent', 'poor')
data$segmentation[data$quality==6] <- 'medium'
data$segmentation <- as.factor(data$segmentation)
```

```
# Kiểm tra kết quả
table(data$segmentation)
```

```
##
## excellent   medium   poor
##    1277     2836    2384
```

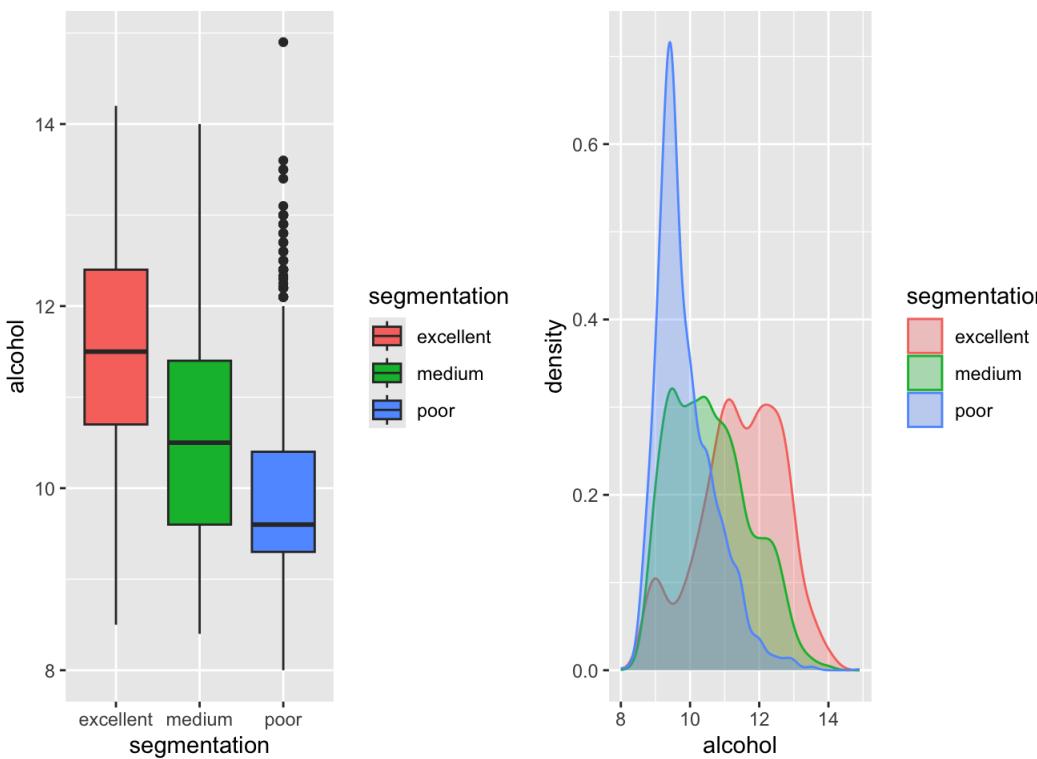
```
ggplot(data, aes(x = segmentation, fill = segmentation)) + geom_bar()
```



Phần lớn rượu có chất lượng trung bình (normal) Ta tiến hành so sánh các đặc trưng của từng phân khúc chất lượng rượu:

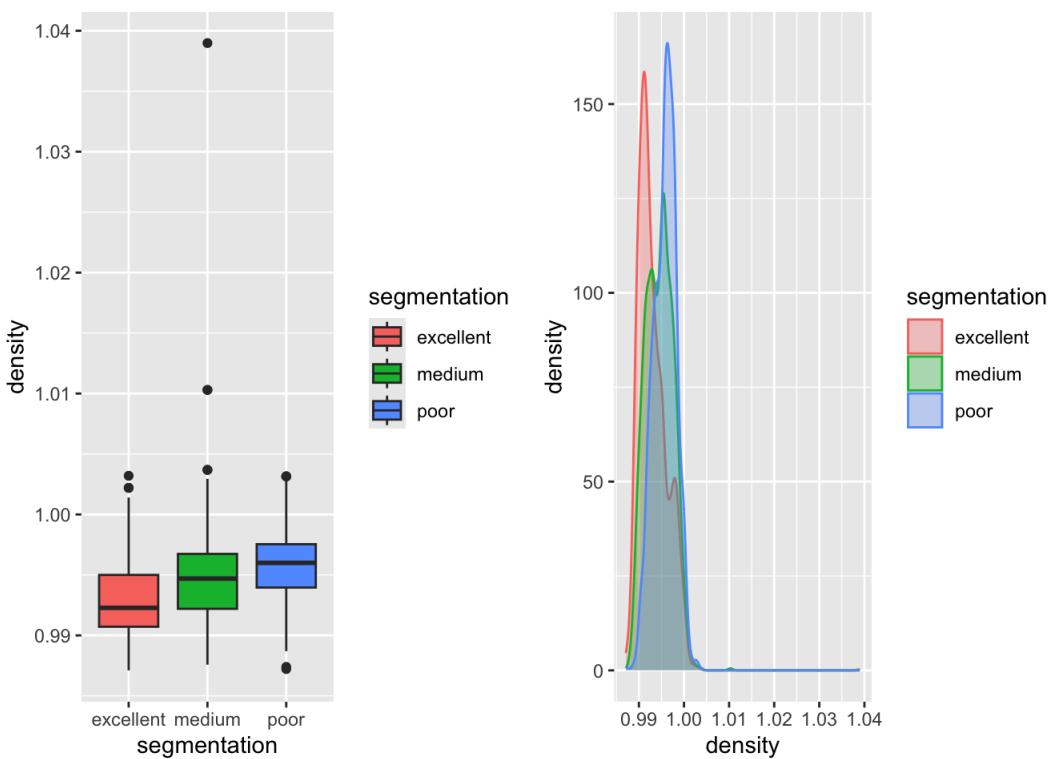
```
plot_feature_in_quality_cat <- function(data, feature) {
  # Create the boxplot
  boxplot <- ggplot(data, aes(x = segmentation, y = .data[[feature]], fill = segmentation)) +
    geom_boxplot()
  # Create the density plot
  density_plot <- ggplot(data, aes(x = .data[[feature]], fill = segmentation)) +
    geom_density(aes(color = segmentation), bw = "nrdd", kernel = "gaussian", alpha = 0.3)
  # Arrange the plots side by side
  grid.arrange(boxplot, density_plot, ncol = 2)
}
```

```
plot_feature_in_quality_cat(data, "alcohol")
```



Quan sát biểu đồ hộp cho thấy rượu vang có chất lượng cao hơn có xu hướng có hàm lượng cồn cao hơn so với rượu vang chất lượng thấp hơn. Điều này gợi ý rằng hàm lượng cồn (alcohol) có thể là một yếu tố quan trọng ảnh hưởng đến chất lượng rượu vang.

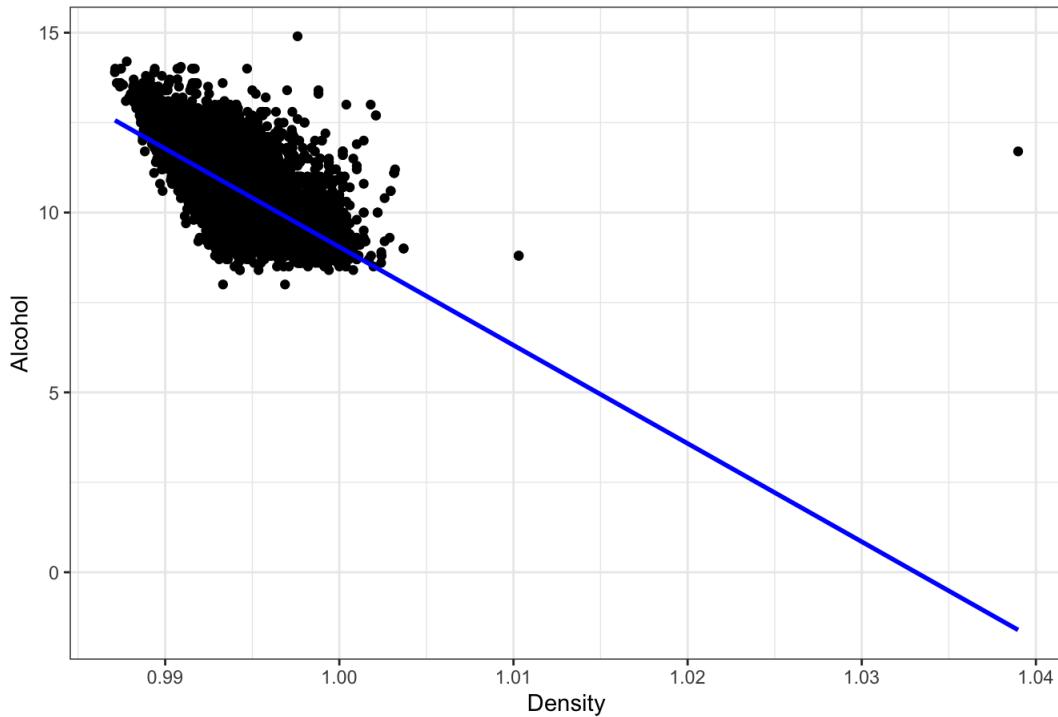
```
plot_feature_in_quality_cat(data, "density")
```



Quan sát biểu đồ hộp cho thấy, rượu vang có chất lượng cao hơn có mật độ (density) thấp hơn so với rượu vang chất lượng thấp hơn. Điều này có nghĩa rằng rượu vang chất lượng cao có xu hướng có mật độ thấp hơn (điều này có thể liên quan đến tỷ lệ nước và cồn trong rượu).

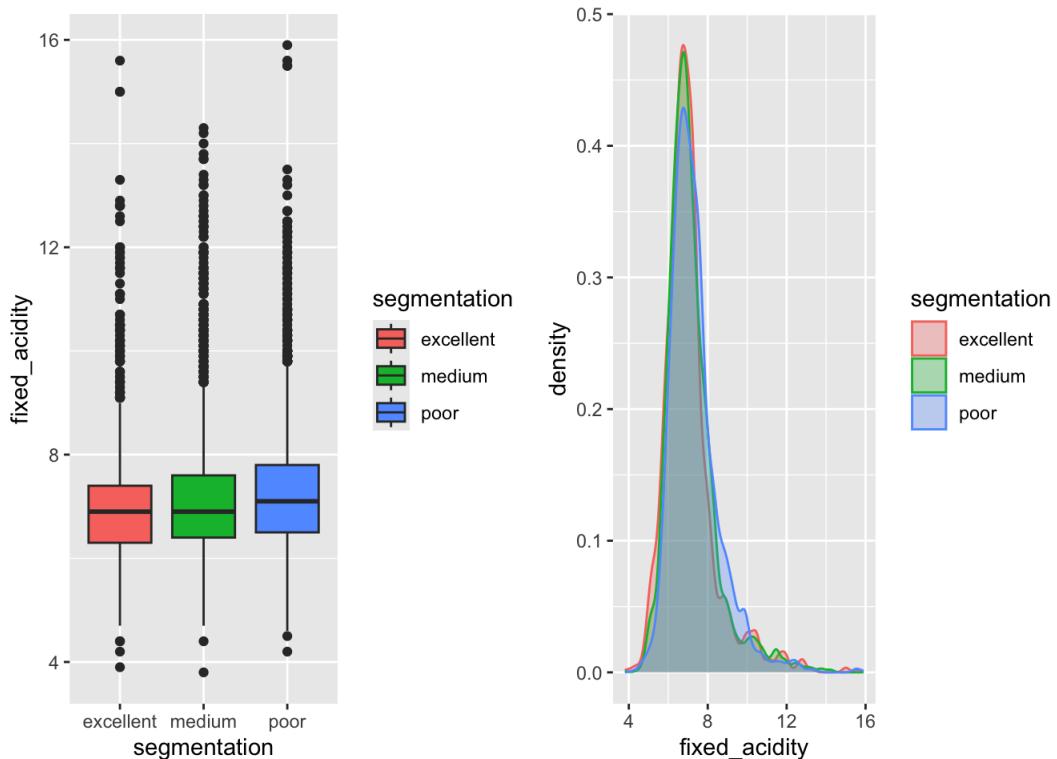
```
ggplot(data, aes(x = density, y = alcohol)) +
  geom_point() +
  geom_smooth(method = "lm", formula = y ~ x, se = FALSE, color = "blue") + # Linear regression without confidence interval
  labs(
    title = "Scatter Plot of Density vs Alcohol",
    x = "Density",
    y = "Alcohol",
  ) +
  theme_bw()
```

Scatter Plot of Density vs Alcohol



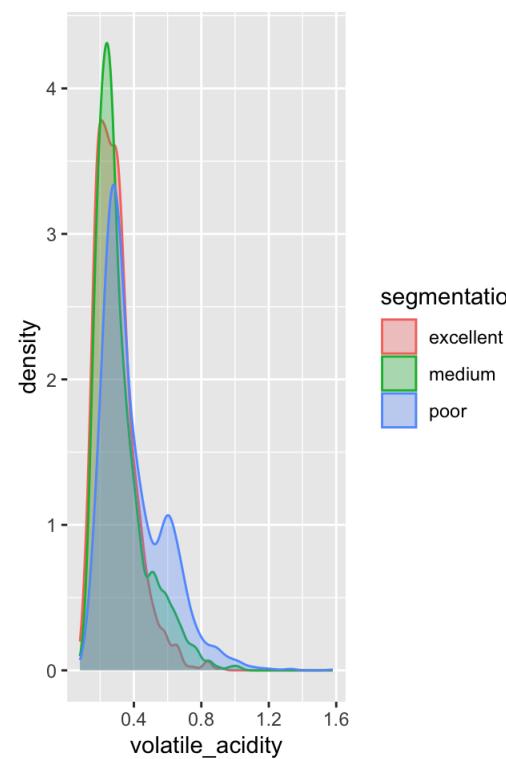
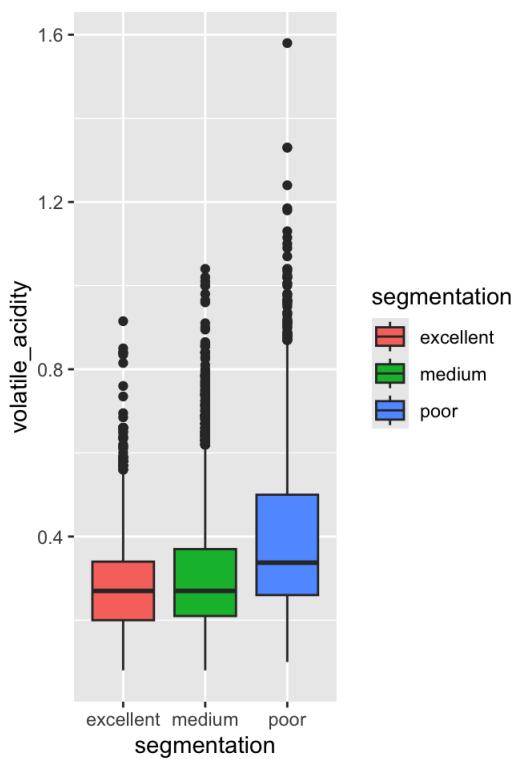
Như đã nói, nồng độ cồn và mật độ có tương quan âm mạnh với nhau.

```
plot_feature_in_quality_cat(data, "fixed_acidity")
```



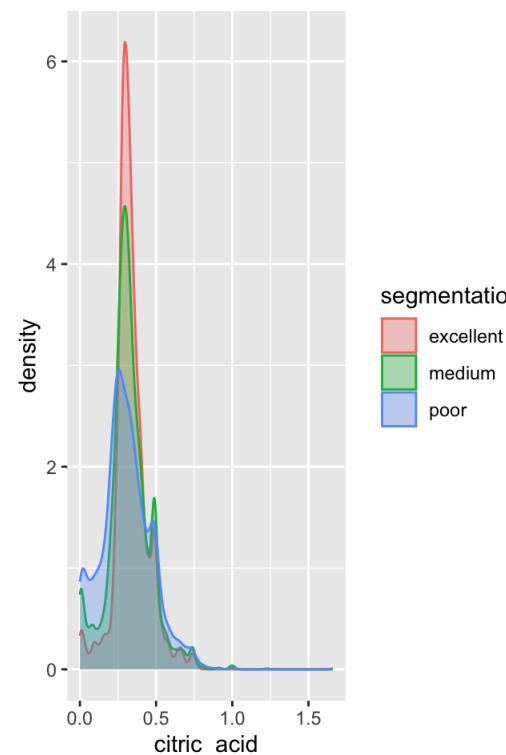
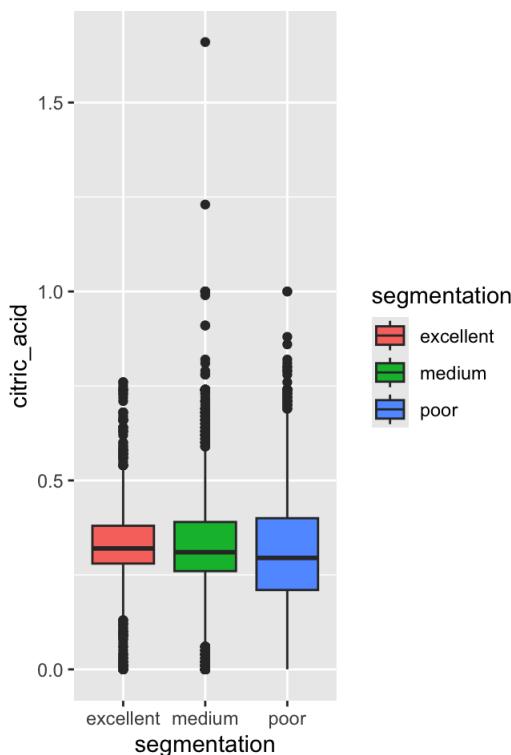
Độ acid không bay hơi ở các loại rượu vang chất lượng khác nhau là tương tự nhau, có khá nhiều điểm mà có khả năng là outliers ở phân khúc rượu trung bình và cao.

```
plot_feature_in_quality_cat(data, "volatile_acidity")
```



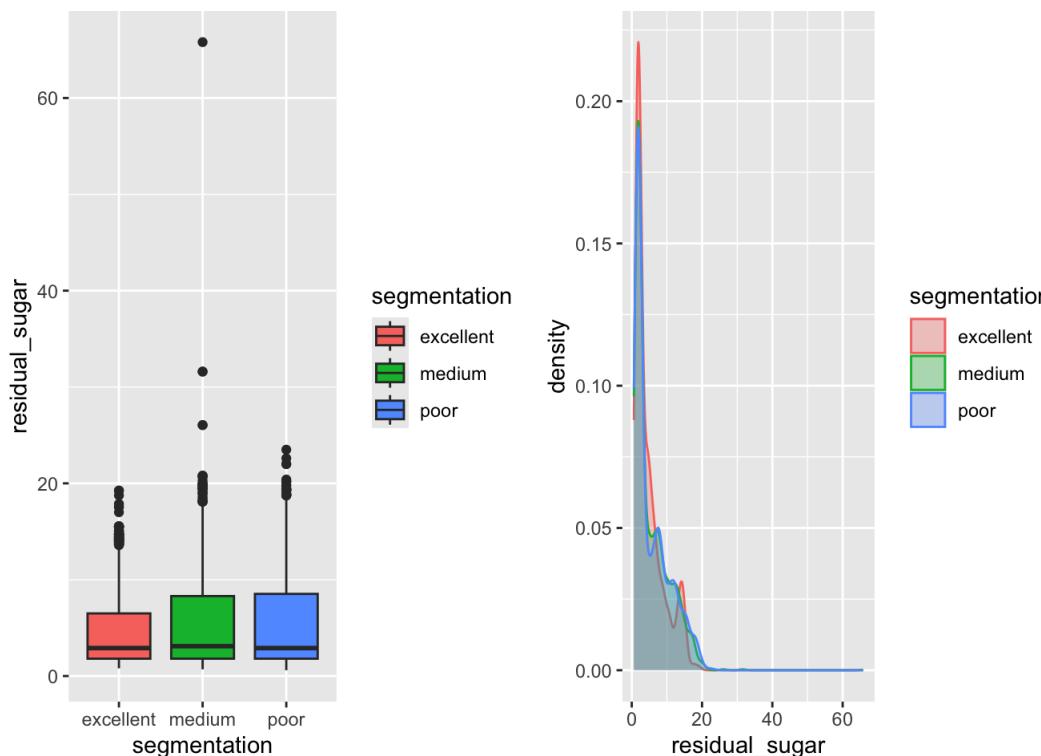
Rượu vang chất lượng cao có phạm vi IQR hẹp hơn, biểu thị sự đồng nhất cao hơn trong độ axit bay hơi so với rượu vang chất lượng thấp, nơi IQR rộng hơn (biểu hiện sự biến động lớn hơn). Đồng thời độ axit bay hơi tỷ lệ nghịch với chất lượng rượu. Ngoài ra, độ axit bay hơi cũng tồn tại các điểm có khả năng cao là outliers.

```
plot_feature_in_quality_cat(data, "citric_acid")
```



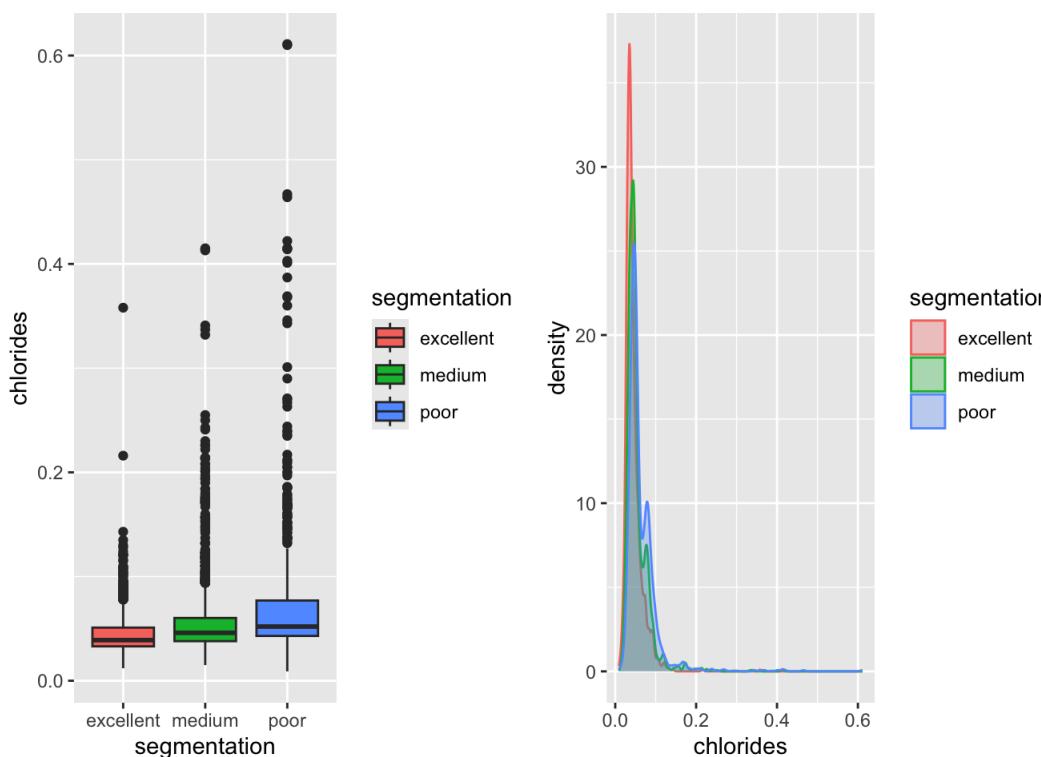
Sự khác biệt nhỏ trong chiều dài của râu giữa các phân khúc chất lượng cho thấy rằng, mặc dù có sự biến động trong hàm lượng axit citric, sự phân bố tổng thể là tương đối nhất quán trong mỗi nhóm chất lượng rượu. hàm lượng axit citric cũng cho thấy các điểm có khả năng cao là outliers.

```
plot_feature_in_quality_cat(data, "residual_sugar")
```



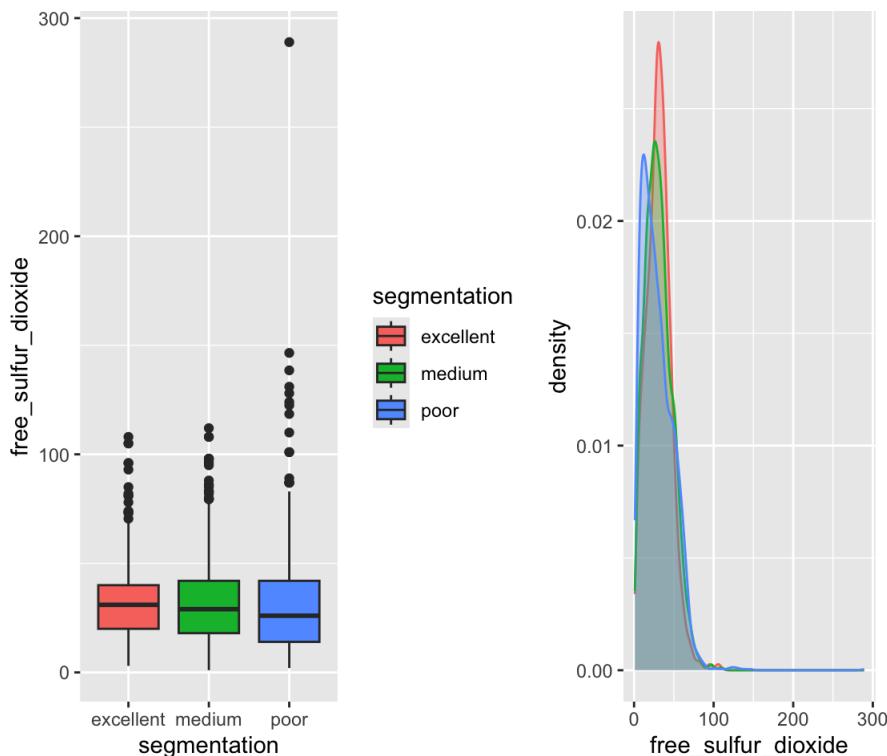
Biểu đồ mật độ cho thấy đỉnh phân bố của hàm lượng đường dư có xu hướng không thay đổi nhiều giữa các phân khúc chất lượng khác nhau. Điều này cho thấy rằng hàm lượng đường dư không phải là yếu tố phân biệt rõ ràng giữa các nhóm chất lượng của rượu vang. Hàm lượng đường dư cũng tồn tại các điểm có khả năng là outliers.

```
plot_feature_in_quality_cat(data, "chlorides")
```



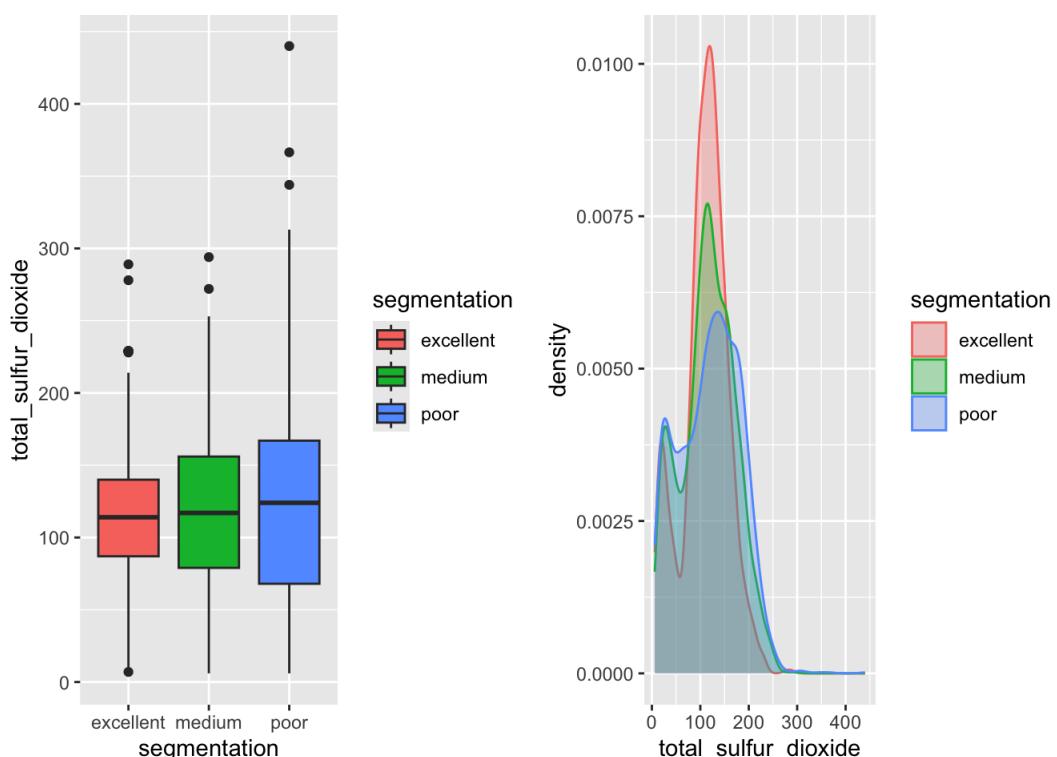
Sự đồng nhất cao hơn và xu hướng hàm lượng chlorides thấp hơn trong rượu vang chất lượng cao gợi ý rằng kiểm soát hàm lượng chlorides có thể là một yếu tố quan trọng trong sản xuất rượu vang chất lượng cao. Chlorides cũng tồn tại các điểm có khả năng là outliers.

```
plot_feature_in_quality_cat(data, "free_sulfur_dioxide")
```



Hình dạng của biểu đồ mật độ biểu hiện rõ ràng sự khác biệt trong phân bố hàm lượng sulfur dioxide giữa các phân khúc chất lượng rượu. Rượu vang chất lượng cao có đỉnh phân bố rõ ràng và hẹp hơn, trong khi rượu vang chất lượng thấp và trung bình có đỉnh phân bố thấp hơn và rộng hơn, chỉ ra sự biến động lớn hơn trong hàm lượng sulfur dioxide. Hàm lượng sulfur dioxide cũng tồn tại các điểm có khả năng là outliers.

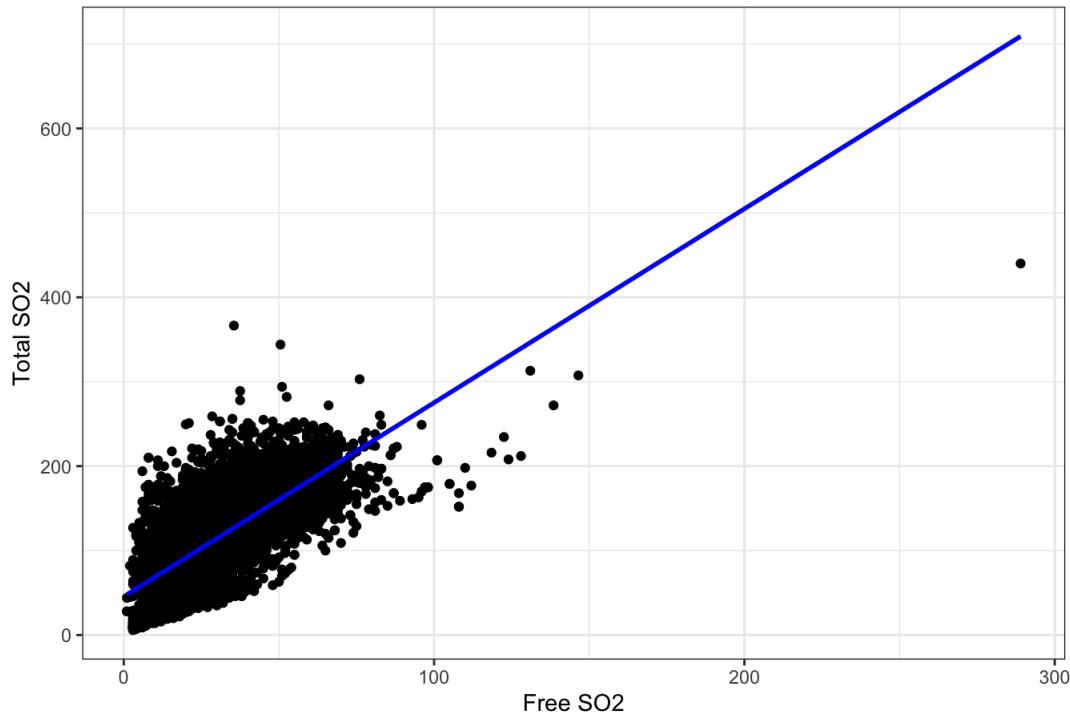
```
plot_feature_in_quality_cat(data, "total_sulfur_dioxide")
```



Sự đồng nhất cao hơn và xu hướng hàm lượng tổng sulfur dioxide thấp hơn trong rượu vang chất lượng cao gợi ý rằng kiểm soát hàm lượng sulfur dioxide có thể là một yếu tố quan trọng trong sản xuất rượu vang chất lượng cao. Ngược lại, sự biến động lớn hơn trong rượu vang chất lượng thấp cho thấy cần có những cải tiến trong quy trình sản xuất để đạt được sự đồng nhất cao hơn.

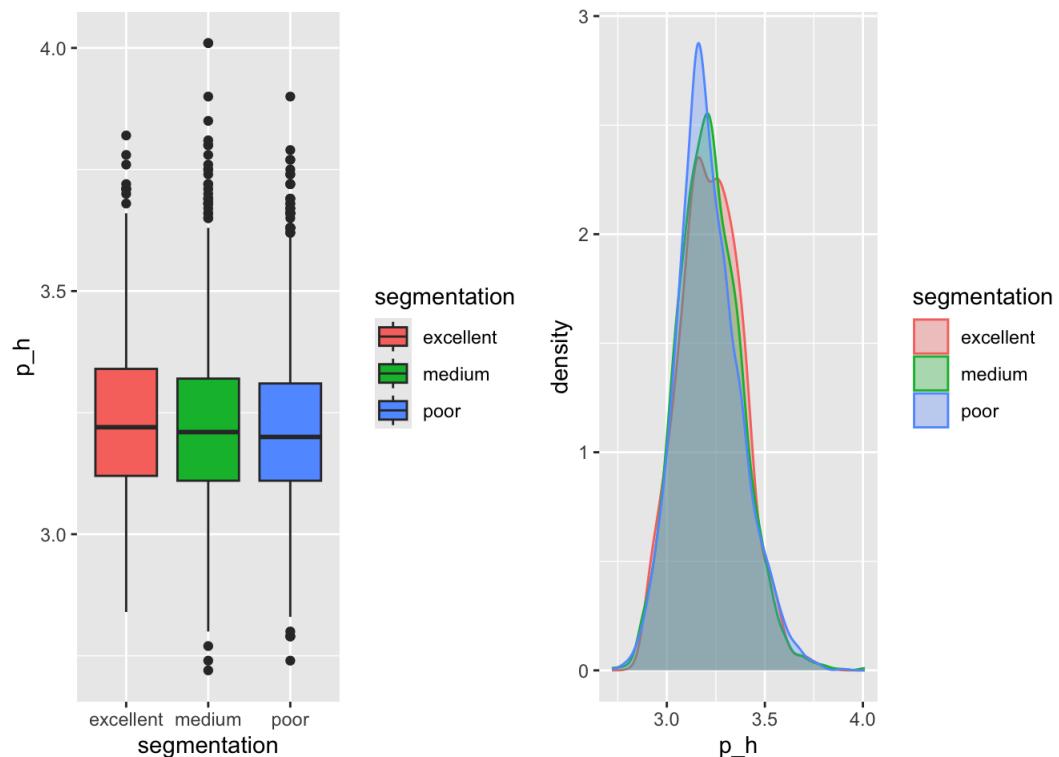
```
ggplot(data, aes(x = free_sulfur_dioxide, y = total_sulfur_dioxide)) +
  geom_point() +
  geom_smooth(method = "lm", formula = y ~ x, se = FALSE, color = "blue") + # Linear regression without confidence interval
  labs(
    title = "Scatter Plot of free vs total SO2",
    x = "Free SO2",
    y = "Total SO2",
  ) +
  theme_bw()
```

Scatter Plot of free vs total SO₂

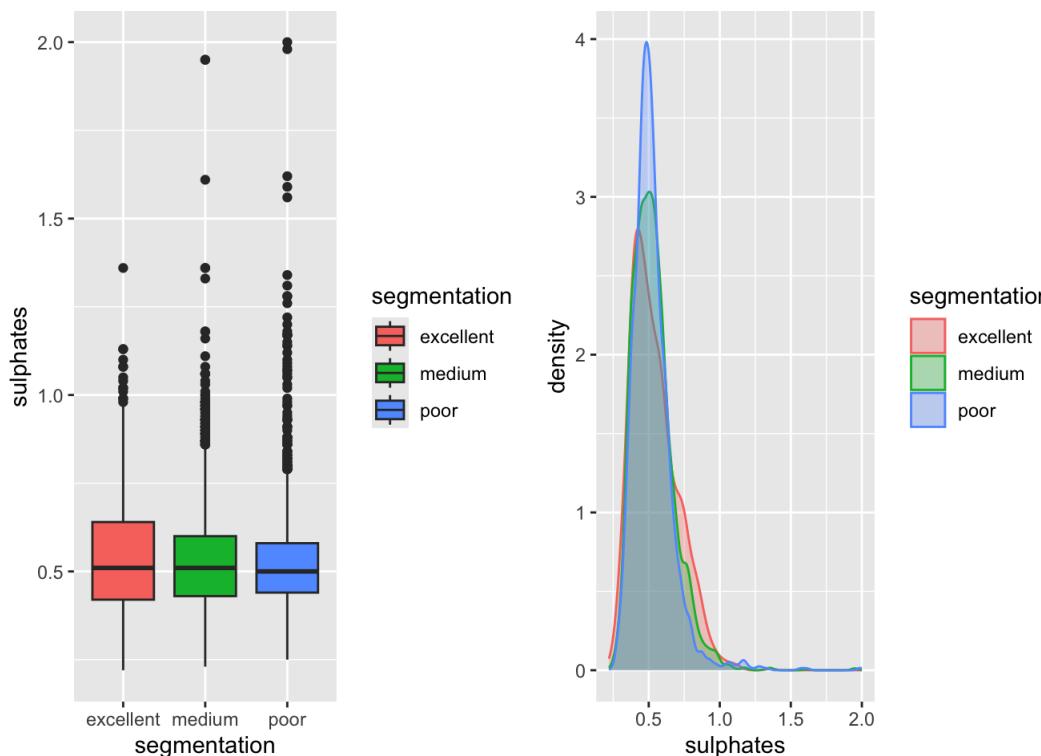


Dễ dàng thấy được Free SO₂ và Total SO₂ có mối tương quan tuyến tính mạnh với nhau.

```
plot_feature_in_quality_cat(data, "p_h")
```



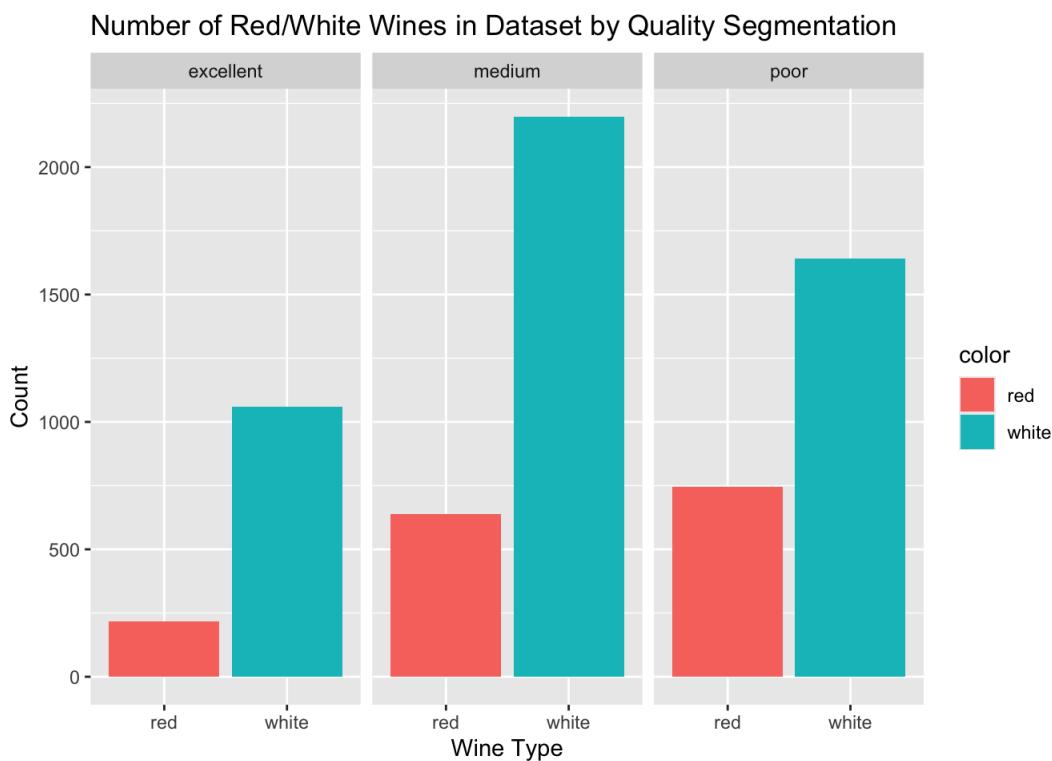
```
plot_feature_in_quality_cat(data, "sulphates")
```



Độ pH và sulphates của các phân khúc chất lượng rượu khác nhau là tương tự nhau.

g. Red Wine và White Wine

```
ggplot(data, aes(x = color, fill = color)) +
  geom_bar() +
  facet_wrap(~ segmentation) + # Facet by quality category
  labs(
    title = "Number of Red/White Wines in Dataset by Quality Segmentation",
    x = "Wine Type",
    y = "Count"
  )
```



Vì đây là bộ data imbalanced giữa red và white nên nếu dùng số lượng để đánh giá sẽ không chính xác, ta sẽ tính tỷ lệ segmentation trên mỗi class red và white

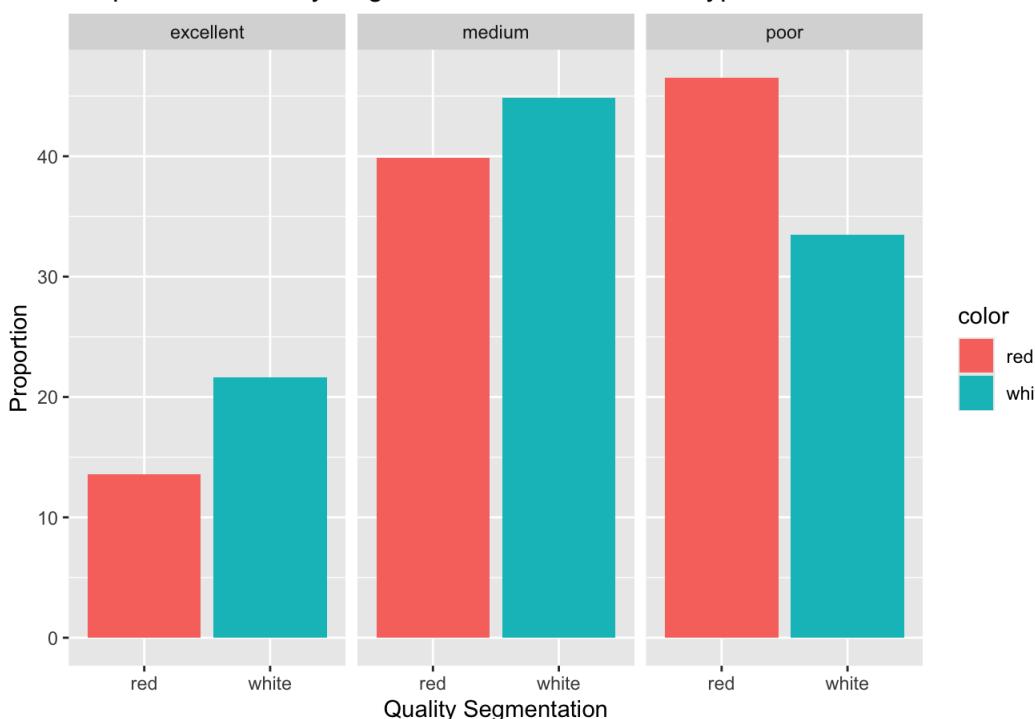
```
wine_summary <- data |>
  count(color, segmentation) |>
  group_by(color) |>
  mutate(total = sum(n),
    percentage = n / total*100) |>
  ungroup()
```

wine_summary

```
## # A tibble: 6 × 5
##   color segmentation  n  total percentage
##   <chr>  <fct>     <int>  <dbl>
## 1 red    excellent  217  1599    13.6
## 2 red    medium    638  1599    39.9
## 3 red    poor      744  1599    46.5
## 4 white  excellent 1060 4898    21.6
## 5 white  medium   2198 4898    44.9
## 6 white  poor     1640 4898    33.5
```

```
ggplot(wine_summary, aes(x = color, y = percentage, fill = color)) +
  geom_bar(stat = "identity", position = "dodge") +
  facet_wrap(~ segmentation) +
  labs(
    title = "Proportion of Quality Segmentation in Each Wine Type",
    x = "Quality Segmentation",
    y = "Proportion"
  )
```

Proportion of Quality Segmentation in Each Wine Type

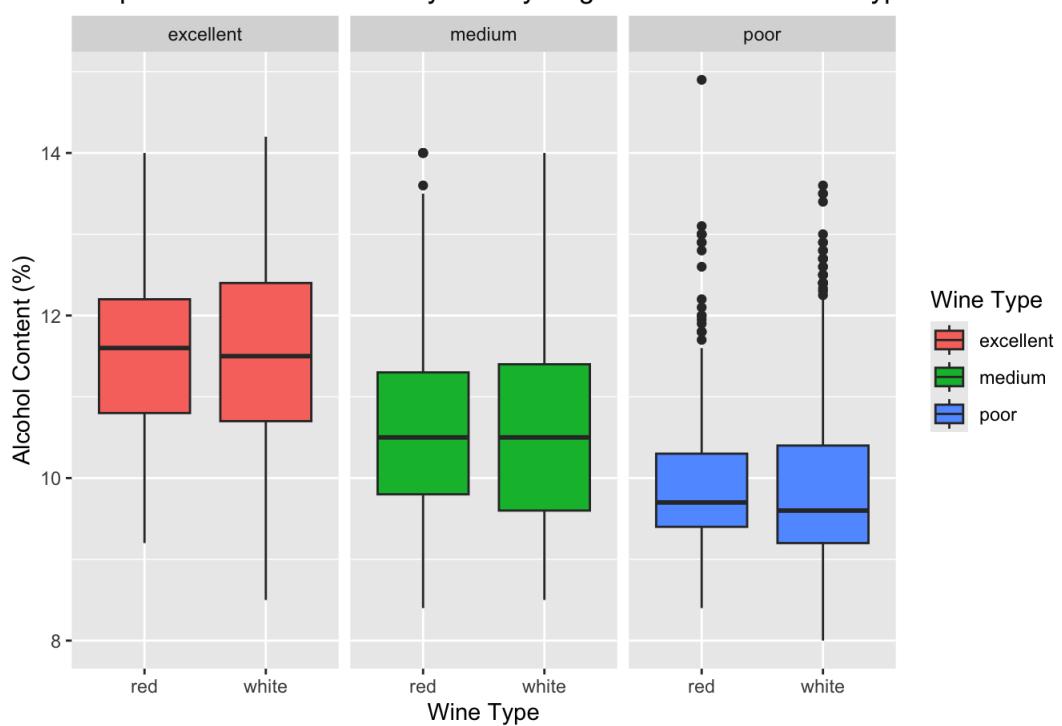


Có thể thấy sự phân phối của chất lượng ở cả hai loại rượu vang khá tương đồng nhau khi phân khúc medium và poor chiếm đa số. Tuy nhiên, tỷ lệ rượu chất lượng cao và trung bình ở rượu vang trắng là nhiều hơn so với rượu vang đỏ, đồng thời tỷ lệ rượu chất lượng thấp ở rượu vang trắng cũng thấp hơn rượu vang đỏ. => color có thể là một feature quan trọng (tuy nhiên điều này cũng có thể là do sự mất cân bằng về dữ liệu giữa hai loại rượu vang này).

h. Wine Type và Wine Quality

```
ggplot(data, aes(x = color, y = alcohol, fill = segmentation)) +
  geom_boxplot() +
  facet_wrap(~ segmentation) +
  labs(
    title = "Boxplot of Alcohol Content by Quality Segmentation and Wine Type",
    x = "Wine Type",
    y = "Alcohol Content (%)",
    fill = "Wine Type"
  )
```

Boxplot of Alcohol Content by Quality Segmentation and Wine Type



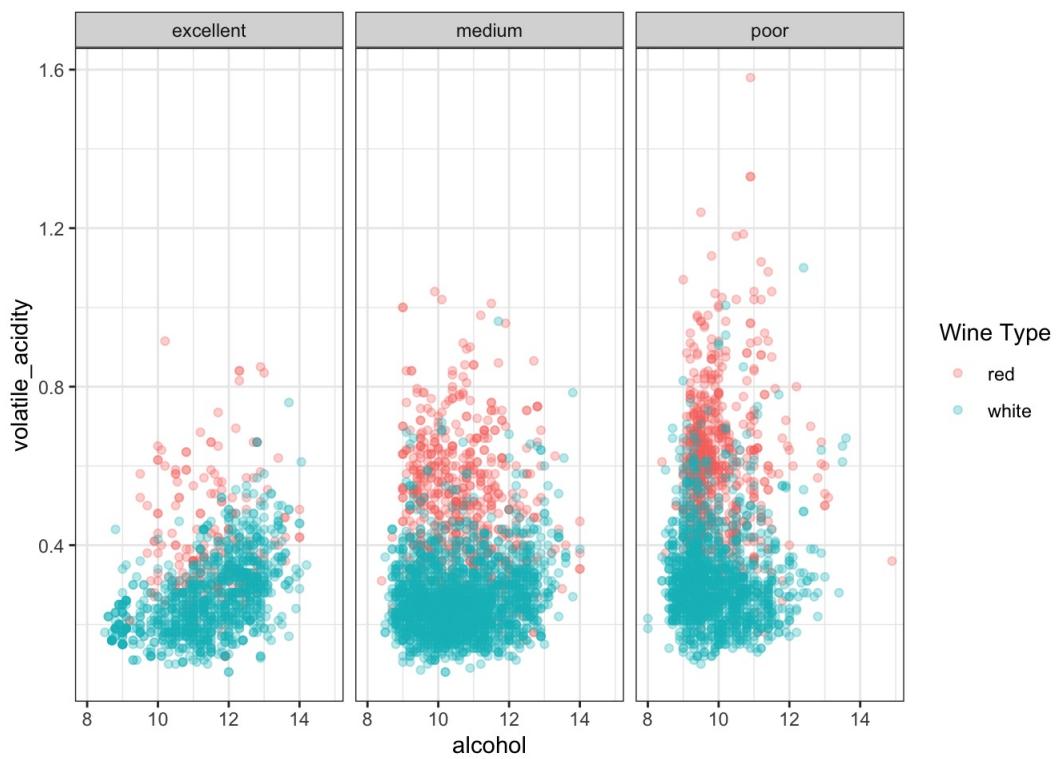
Không có sự khác nhau đáng kể về nồng độ cồn giữa hai loại rượu vang trắng và vang đỏ ở các phân khúc khác nhau.

```
plot_scatter_in_wine_type <- function(data, feature1, feature2) {
  ggplot(data = data, aes_string(x = feature1, y = feature2, color = "color")) +
    geom_point(alpha = 0.3) +
    facet_wrap(~ segmentation) +
    labs(
      x = feature1,
      y = feature2,
      color = "Wine Type"
    ) +
    theme_bw()
}
```

features

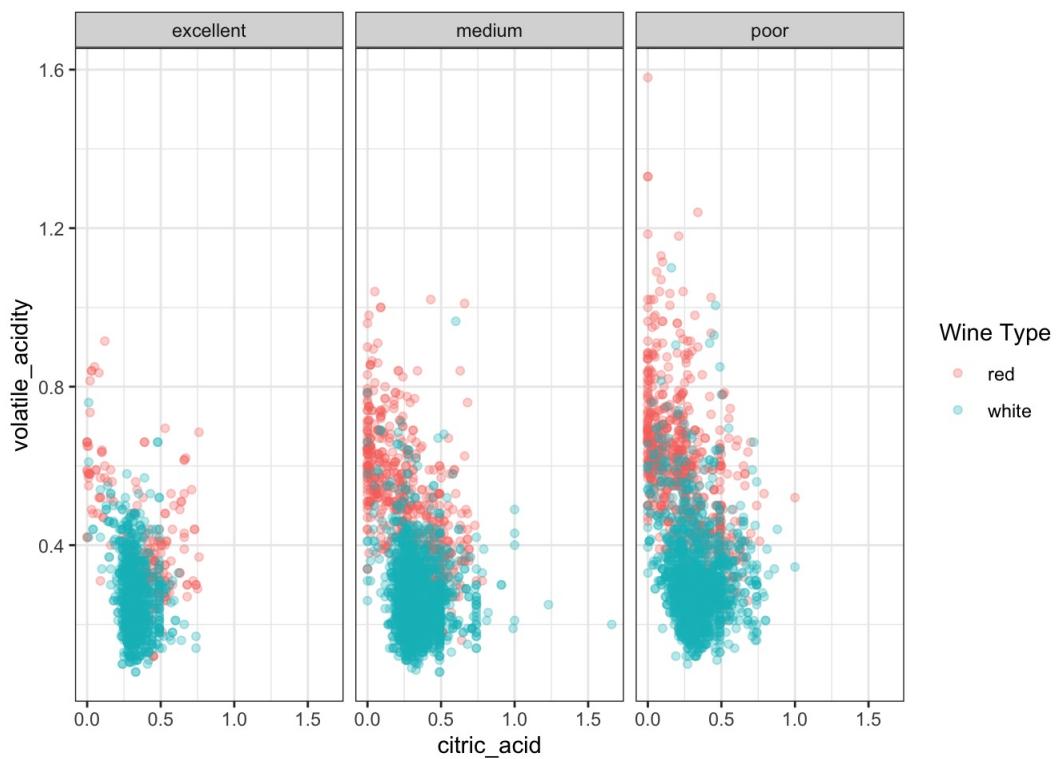
```
## [1] "fixed_acidity"     "volatile_acidity"   "citric_acid"
## [4] "residual_sugar"   "chlorides"        "free_sulfur_dioxide"
## [7] "total_sulfur_dioxide" "density"        "p_h"
## [10] "sulphates"        "alcohol"         "quality"
```

```
plot_scatter_in_wine_type(data,"alcohol","volatile_acidity")
```



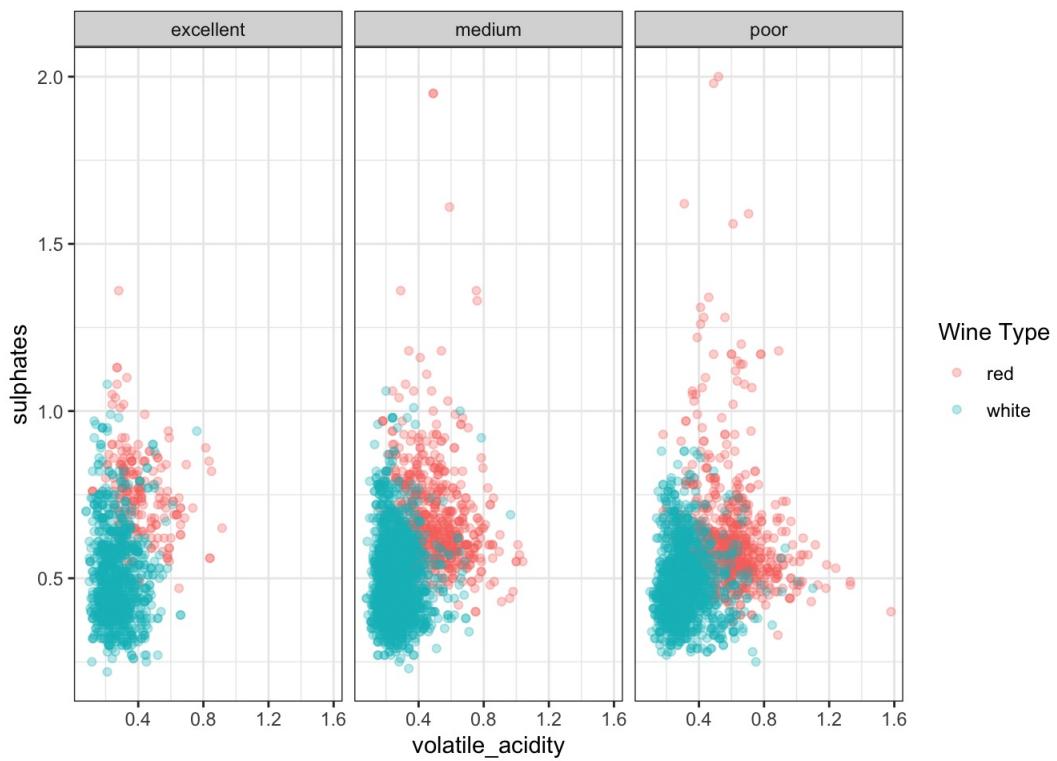
Dựa vào alcohol và volatile acidity, ta không cho thấy có sự phân biệt rõ rệt nào giữa hai tập rượu vang đỏ và vang trắng.

```
plot_scatter_in_wine_type(data, "citric_acid", "volatile_acidity")
```

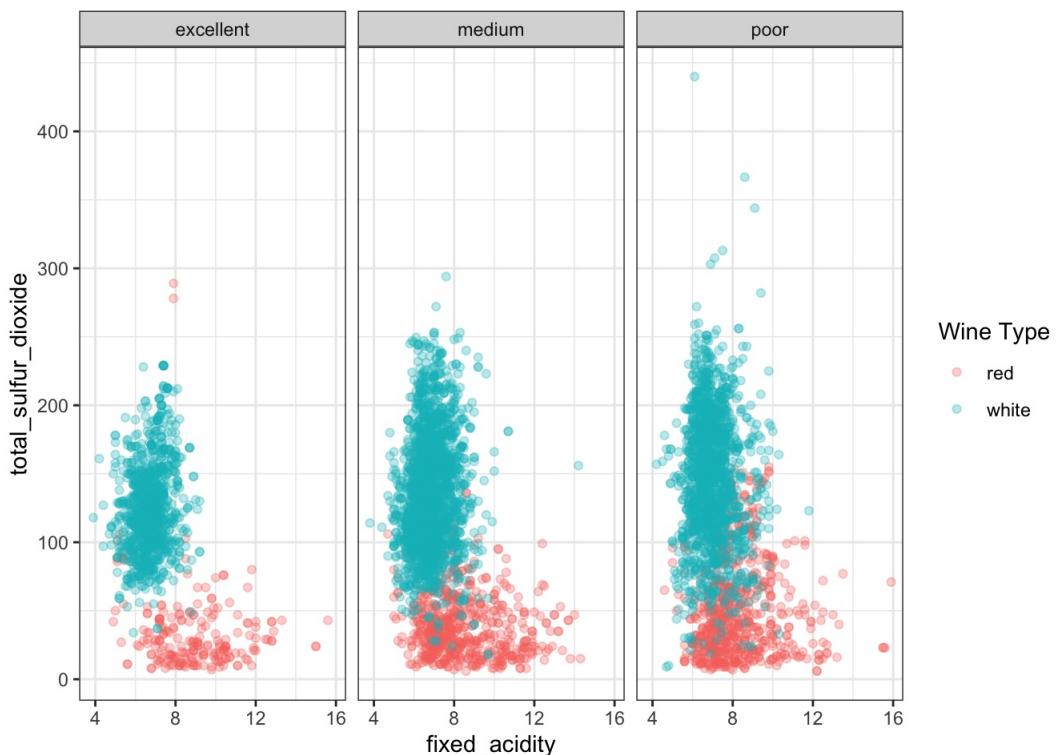


Tương tự, volatile_acidity và citric_acid cũng không cho thấy rõ sự phân biệt nào giữa hai loại rượu.

```
plot_scatter_in_wine_type(data, "volatile_acidity", "sulphates")
```



```
plot_scatter_in_wine_type(data,"fixed_acidity","total_sulfur_dioxide")
```



Tuy nhiên, có thể thấy rõ sự phân biệt tách lớp này giữa red/white ở các cặp feature total SO₂ - fixed_acidity và sulphates - volatile_acidity.

i. Phân tích đa biến (Multivariate Analysis)

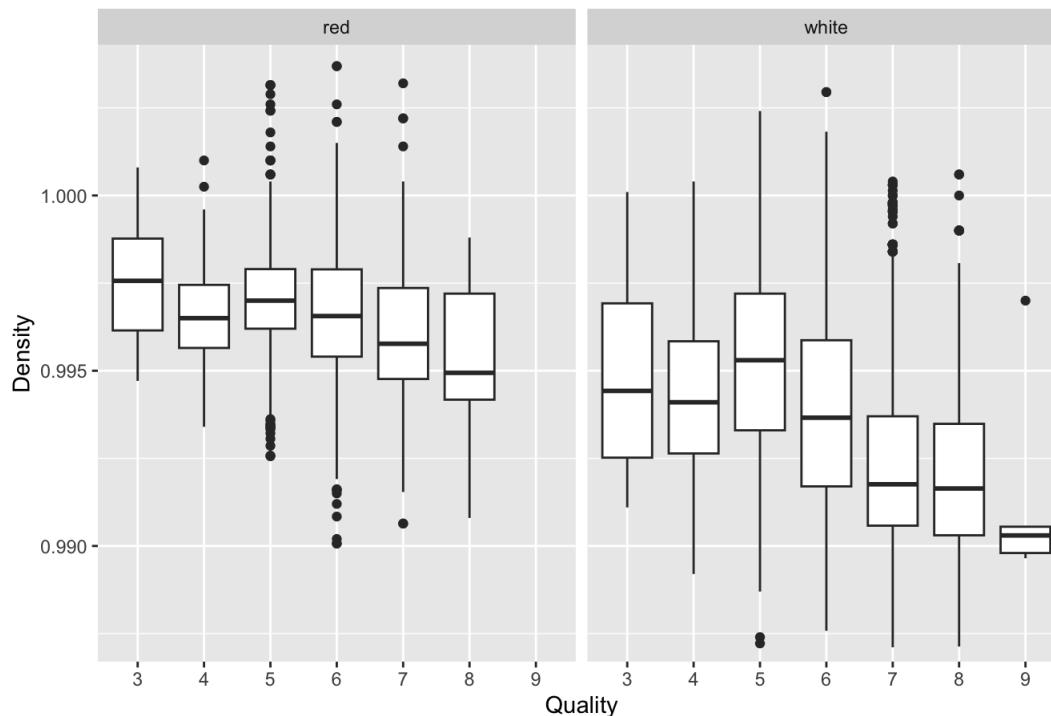
Color và Wine Quality

Trước hết, cần kiểm tra xem liệu tất cả các feature đã được xác định là có ảnh hưởng đáng kể đến chất lượng rượu vang, có độc lập với màu rượu hay không. Hay nói cách khác là kiểm tra xem các yếu tố ảnh hưởng đến chất lượng rượu vang có khác nhau ở hai loại rượu này hay không.

Density vs. Quality

```
ggplot(data = data, aes(y = density, x = as.factor(quality))) +
  geom_boxplot(outlier.shape = 19) +
  coord_cartesian(ylim=c(0.9875, 1.0035)) +
  facet_wrap(~ color) +
  xlab('Quality') +
  ylab('Density') +
  ggtitle('How Density Level Affects Wine Quality')
```

How Density Level Affects Wine Quality

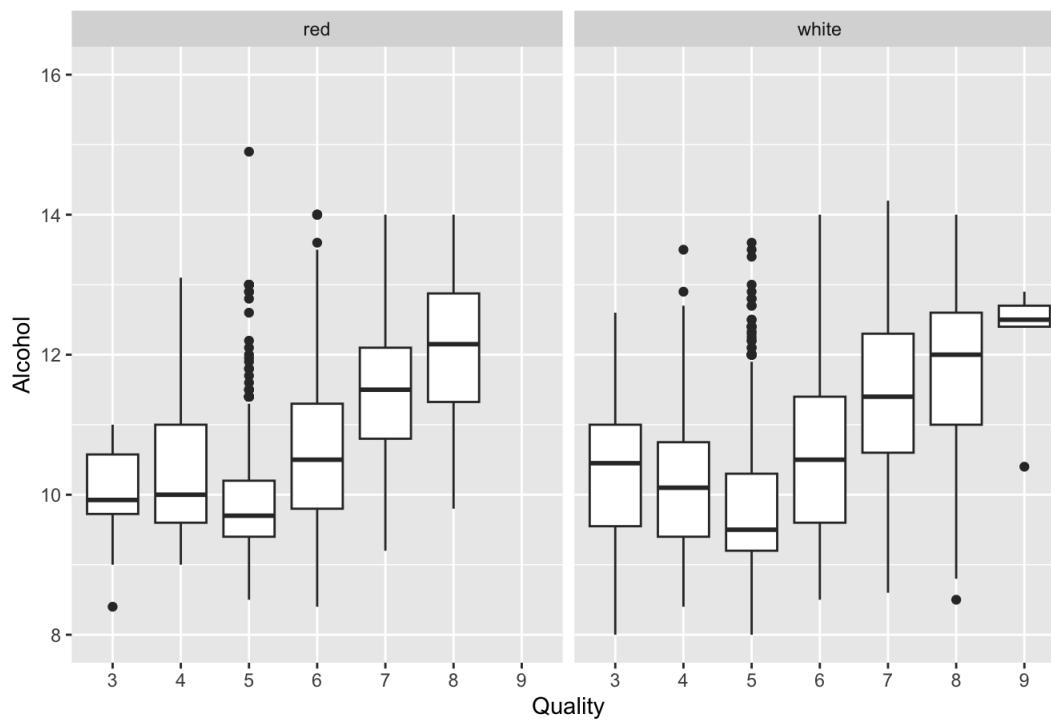


Đối với cả hai loại rượu, chất lượng có mối tương quan nghịch với mật độ. Tuy nhiên, mối tương quan này có vẻ mạnh hơn một chút đối với rượu vang trắng.

Alcohol vs. Quality

```
ggplot(data = data, aes(y = alcohol, x = as.factor(quality))) +  
  geom_boxplot(outlier.shape = 19) +  
  coord_cartesian(ylim=c(8, 16)) +  
  facet_wrap(~ color) +  
  xlab('Quality') +  
  ylab('Alcohol') +  
  ggtitle('How Alcohol Level Affects Wine Quality')
```

How Alcohol Level Affects Wine Quality

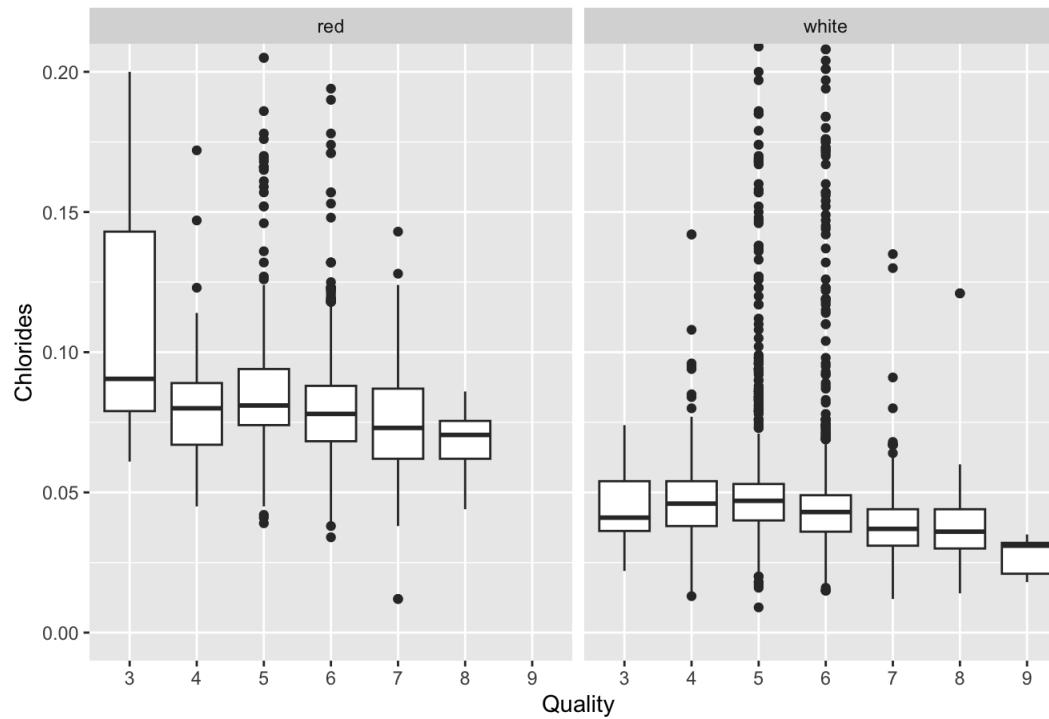


Với hàm lượng cồn, ta cũng nhận thấy hai xu hướng tương tự nhau giữa hai loại rượu.

Chlorides vs. Quality

```
ggplot(data = data, aes(y = chlorides, x = as.factor(quality))) +
  geom_boxplot(outlier.shape = 19) +
  coord_cartesian(ylim=c(0, 0.2)) +
  facet_wrap(~ color) +
  xlab('Quality') +
  ylab('Chlorides') +
  ggtitle('How Chlorides Affects Wine Quality')
```

How Chlorides Affects Wine Quality

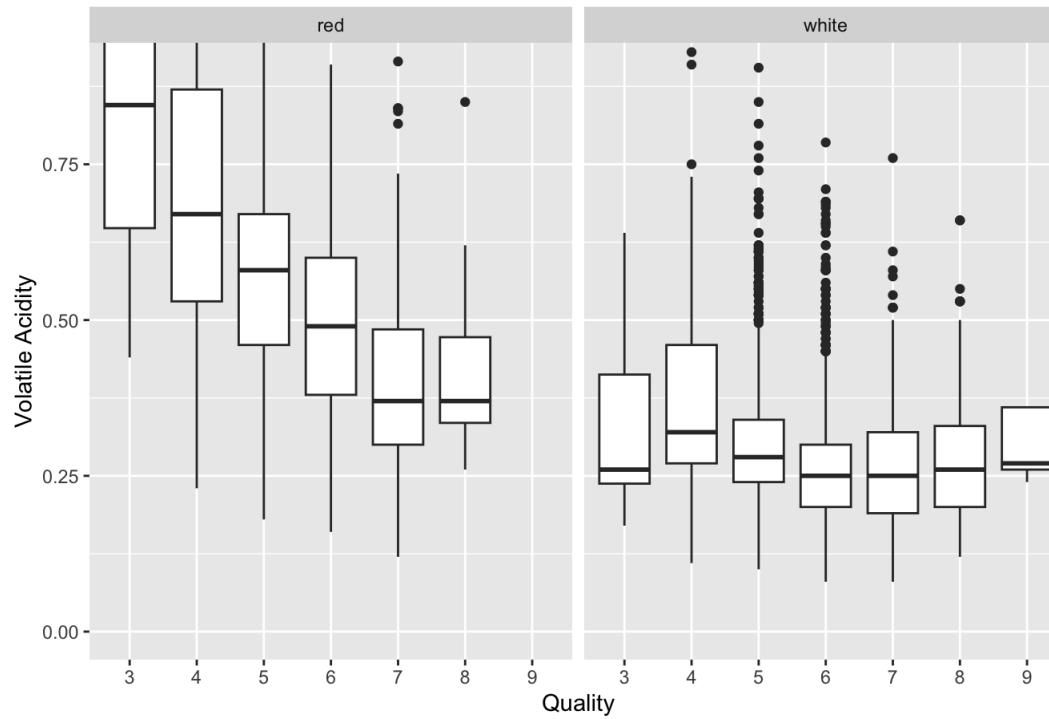


Rượu vang trắng về tổng quan có nồng độ clorua thấp hơn rượu vang đỏ. Tuy nhiên, cả hai loại rượu đều cho thấy một xu hướng tương quan nghịch giữa nồng độ clorua và chất lượng rượu, nhưng nhìn chung tác động này rất yếu và có rất nhiều điểm ngoại lai ở phân khúc rượu trung bình.

Volatile Acidity vs. Quality

```
ggplot(data = data, aes(y = volatile_acidity, x = as.factor(quality))) +
  geom_boxplot(outlier.shape = 19) +
  coord_cartesian(ylim=c(0, 0.9)) +
  facet_wrap(~ color) +
  xlab('Quality') +
  ylab('Volatile Acidity') +
  ggtitle('How Volatile Acidity Affects Wine Quality')
```

How Volatile Acidity Affects Wine Quality



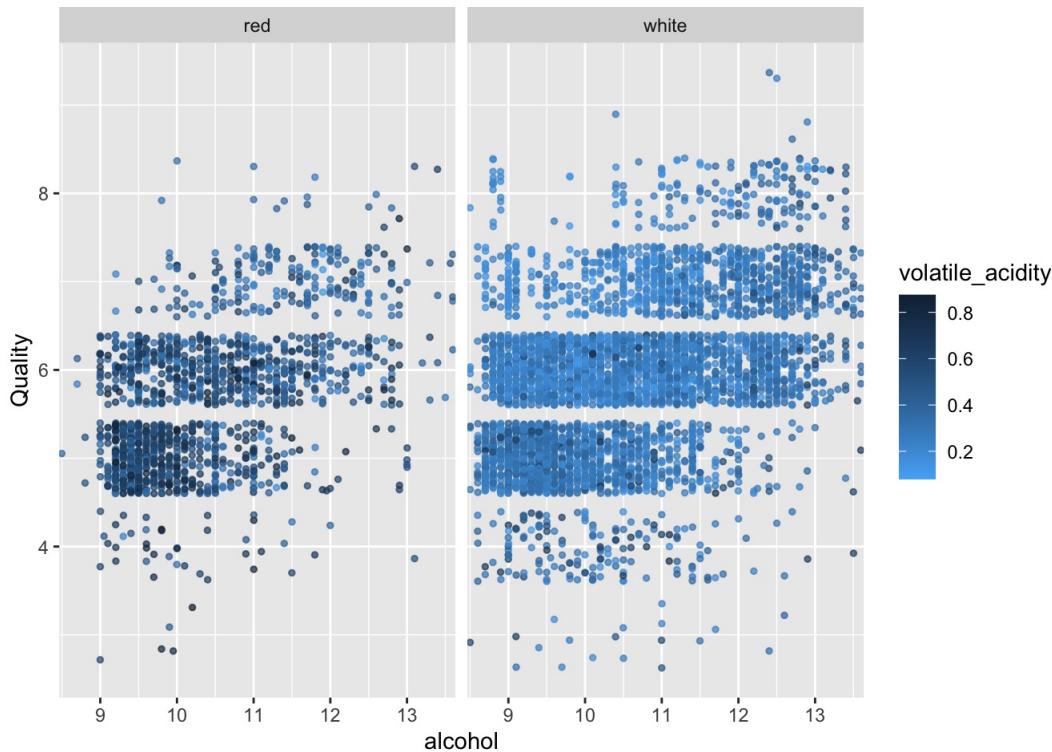
Độ axit dễ bay hơi có ảnh hưởng mạnh mẽ đến chất lượng của rượu vang đỏ nhưng dường như không có mối tương quan đáng kể nào với chất lượng của rượu vang trắng.

Xét các yếu tố chính ảnh hưởng đến chất lượng rượu vang

Trong phần này, ta sẽ tóm tắt sâu hơn về ba thông số ảnh hưởng đến chất lượng rượu một cách trực quan nhất.

Alcohol và Volatile Acidity

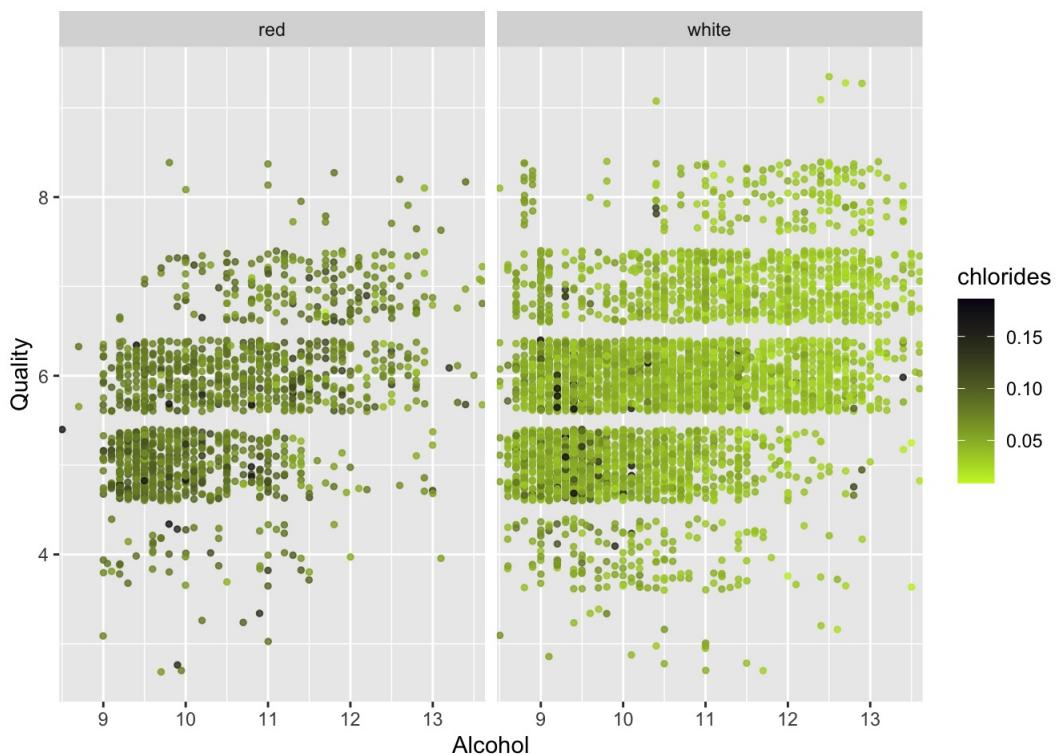
```
ggplot(aes(x = alcohol, y = quality, color = volatile_acidity),  
       data = subset(data, data$volatile_acidity < quantile(data$volatile_acidity, 0.99))) +  
       geom_jitter(size = 1, alpha = 0.7) +  
       scale_colour_gradient(high = "#132B43", low = "#56B1F7") +  
       facet_wrap(~color) +  
       coord_cartesian(xlim = c(quantile(data$alcohol, 0.01), quantile(data$alcohol, 0.99))) +  
       labs(y = 'Quality')
```



Đồ thị trên cho thấy mối tương quan dương giữa hàm lượng cồn và chất lượng của rượu vang đỏ và trắng. Ngoài ra, còn có thể thấy rõ mối tương quan nghịch nhẹ giữa chất lượng rượu và nồng độ axit dễ bay hơi ở rượu vang đỏ. Nhưng mối tương quan này lại hơi khó xác định ở rượu vang trắng vì rượu vang trắng có xu hướng có hàm lượng axit axetic thấp hơn.

Alcohol và Chlorides

```
ggplot(aes(x = alcohol, y = quality, color = chlorides),  
       data = subset(data, data$chlorides < quantile(data$chlorides, 0.99))) +  
       geom_jitter(size = 1, alpha = 0.8) +  
       scale_colour_gradient(low = '#c9f42d', high = '#080118') +  
       facet_wrap(~color) +  
       coord_cartesian(xlim = c(quantile(data$alcohol, 0.01), quantile(data$alcohol, 0.99))) +  
       labs(x = 'Alcohol', y = 'Quality')
```



Hàm lượng cồn cao làm tăng chất lượng rượu trong khi nồng độ clorua làm giảm chất lượng rượu. Một lần nữa, rượu vang trắng chứa nồng độ thành phần clorua thấp hơn, và do đó, mối tương quan này chủ yếu thấy rõ ở rượu vang đỏ.

Nhận xét tổng quan phần EDA

Trước hết, ta có thể nhận thấy rằng mật độ (density), độ cồn (alcohol), hàm lượng clorua (chlorides) và độ axit dễ bay hơi (volatile_acidity) ảnh hưởng đáng kể đến chất lượng rượu vang một cách độc lập với màu sắc. Nhưng những tác động này nhìn chung mạnh hơn đối với rượu vang đỏ.

Tiếp theo, ta xem xét sự kết hợp giữa các thành phần và ảnh hưởng của chúng đến chất lượng rượu vang. Các sự kết hợp khác nhau cho thấy rõ ràng sự khác biệt giữa thành phần rượu vang đỏ và trắng. Ví dụ, rượu vang đỏ chứa nồng độ clorua và sunfat cao, trong khi, rượu vang trắng có đặc điểm là hàm lượng axit dễ bay hơi thấp. Tuy nhiên, không có sự kết hợp nào giữa các thành phần này ảnh hưởng đến chất lượng rượu một cách đáng kể.

II. Preprocessing

```
# Print total missing values
print(paste("Total NA:", sum(is.na(data))))
```

```
## [1] "Total NA: 0"
```

```
# Print total duplicates
print(paste("Total Duplicates:", sum(duplicated(data))))
```

```
## [1] "Total Duplicates: 1177"
```

```
data <- data |> distinct()
print(paste("Total Duplicates:", sum(duplicated(data))))
```

```
## [1] "Total Duplicates: 0"
```

```
# Xử lý imbalanced bằng kết hợp over và under sampling
data_balanced <- ovun.sample(color~., data = data,
                               N=nrow(data), p=0.5,
                               seed=1, method="both")$data
data_balanced <- data_balanced |> mutate(color = factor(color, levels = c("red", "white")))
table(data_balanced$color)
```

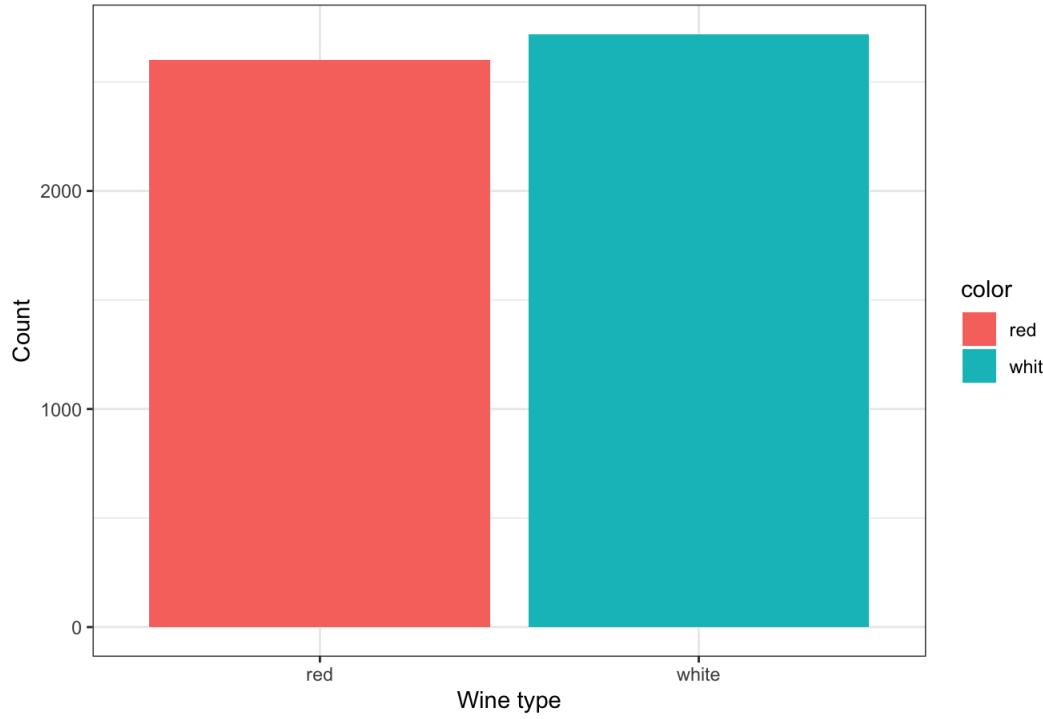
```
##
## red white
## 2601 2719
```

```

ggplot(data_balanced, aes(x = color, fill = color)) +
  geom_bar() +
  labs(
    title = "Number of red/white wine in dataset",
    x = "Wine type",
    y = "Count"
  ) +
  theme_bw()

```

Number of red/white wine in dataset



III. Kiểm Định

Xây dựng hàm vẽ biểu đồ Violin:

```

plot_violin <- function(data, feature, label) {
  p<- ggplot(data, aes_string(x = label, y = feature, fill = label)) +
    geom_violin() +
    geom_boxplot(width = 0.15) +
    labs(
      x = label,
      y = feature
    ) +
    ggtitle(paste("Violin plot of", feature, "by", label)) +
    theme_bw() +
    theme(legend.position = "none")
  return(p)
}

```

Xây dựng hàm thực hiện Permutation Test:

```

perm_test <- function(x, y, R = 1000, alter = "two.sided") {
  n <- length(x)
  a <- split(x, y)
  res_perm <- numeric(R)
  mean_A <- mean(a[[1]])
  mean_B <- mean(a[[2]])

  for (i in 1:R) {
    idx_a <- sample(x = 1:n, size = length(a[[1]]))
    idx_b <- setdiff(x = 1:n, y = idx_a)
    res_perm[i] <- mean(x[idx_a]) - mean(x[idx_b])
  }

  if (alter == 'left') {
    p_value <- mean(res_perm < (mean_A - mean_B))
  } else if (alter == 'right') {
    p_value <- mean(res_perm > (mean_A - mean_B))
  } else if (alter == 'two.sided') {
    p_value <- mean(abs(res_perm) > abs(mean_A - mean_B))
  } else {
    stop("Invalid alternative hypothesis. Choose 'left', 'right', or 'two.sided'.")
  }

  return(list(mean_A = mean_A, mean_B = mean_B, p_value = p_value))
}

```

Xây dựng hàm thực hiện Bootstrap Permutation Test:

```

boot_perm_test <- function(x, y, R, alter) {
  n <- length(x)
  a <- split(x, y)
  res_perm <- numeric(R)
  mean_A <- mean(a[[1]])
  mean_B <- mean(a[[2]])
  set.seed(32)

  for (i in 1:R){
    # Lấy mẫu từ x có hoàn lại
    idx_a <- sample(x = 1:n,replace = TRUE,size = length(a[1]))
    idx_b <- sample(x = 1:n,replace = TRUE,size = length(a[2]))

    res_perm[i] <- mean(x[idx_a]) - mean(x[idx_b])
  }

  if (alter == 'left') {
    p_value <- mean(res_perm < (mean_A - mean_B))
  } else if (alter == 'right') {
    p_value <- mean(res_perm > (mean_A - mean_B))
  } else if (alter == 'two.sided') {
    p_value <- mean(abs(res_perm) > abs(mean_A - mean_B))
  } else {
    stop("Invalid alternative hypothesis. Choose 'left', 'right', or 'two.sided'.")
  }

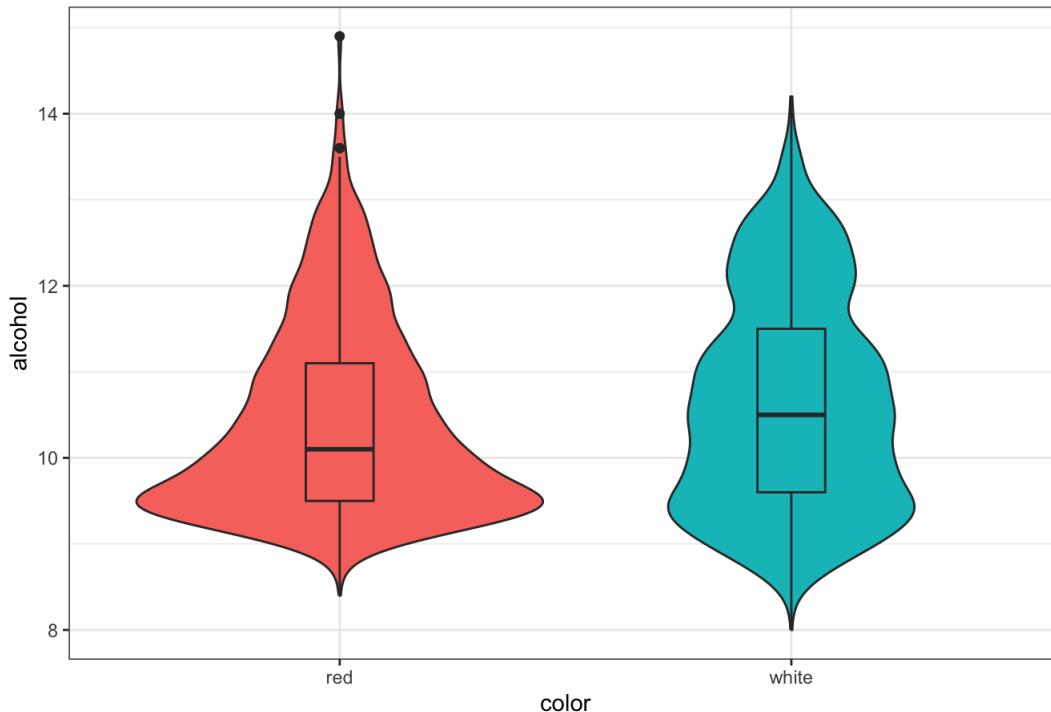
  return(list(mean_A = mean_A, mean_B = mean_B, p_value = p_value))
}

```

III.1. Kiểm định trung bình

```
plot_violin(data_balanced,"alcohol","color")
```

Violin plot of alcohol by color



Thực hiện kiểm định trung bình cho alcohol ở 2 loại rượu để kiểm tra có sự khác biệt giữa nồng độ cồn ở rượu vang đỏ so với rượu vang trắng hay không.

Đặt giả thuyết:

H0: $\mu_{\text{alcohol_red}} = \mu_{\text{alcohol_white}}$

H1: $\mu_{\text{alcohol_red}} \neq \mu_{\text{alcohol_white}}$

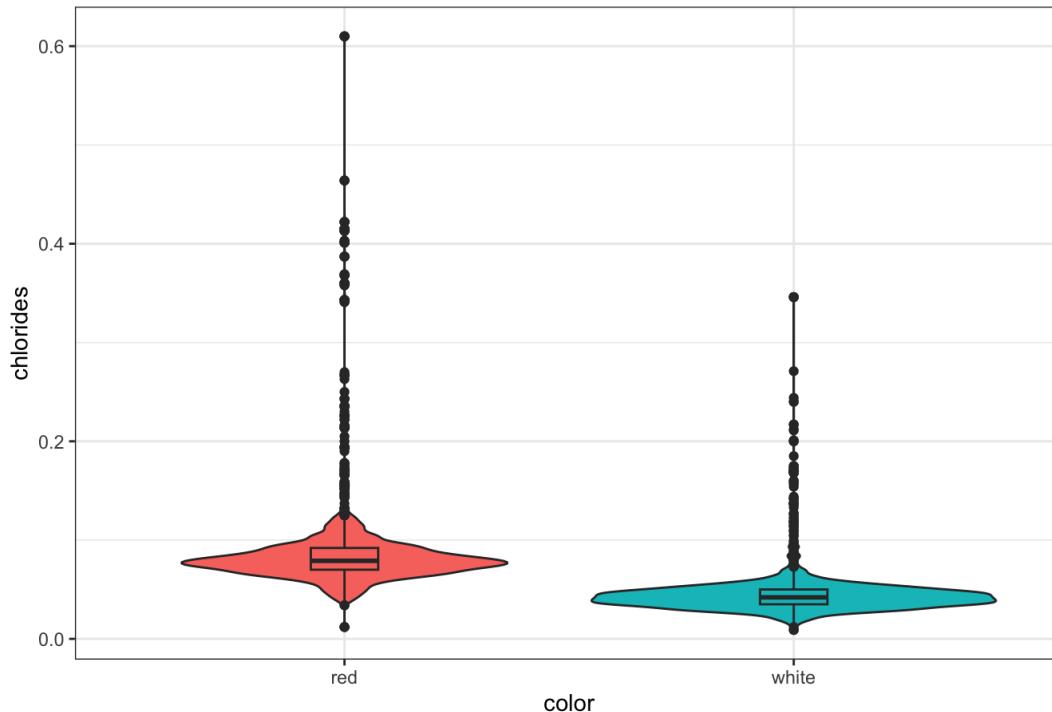
```
boot_perm_test(data_balanced$alcohol, data$color, R=1000, alter = "two.sided")
```

```
## $mean_A  
## [1] 10.62329  
##  
## $mean_B  
## [1] 10.4976  
##  
## $p_value  
## [1] 0.922
```

Với mức ý nghĩa alpha = 0.05, p-value = 0.922 > 0.05, không đủ cơ sở để bác bỏ giả thuyết H0, hay nói cách khác, nồng độ cồn vang đỏ lớn hơn vang trắng chỉ là sự trùng hợp ngẫu nhiên và không mang ý nghĩa thống kê.

```
plot_violin(data_balanced, "chlorides", "color")
```

Violin plot of chlorides by color



Thực hiện kiểm định trung bình cho chlorides ở 2 loại rượu để kiểm tra xem việc hàm lượng clorua ở rượu vang đỏ lớn hơn so với rượu vang trắng có ý nghĩa thống kê hay không.

Đặt giả thuyết:

H0: $\mu_{\text{chlorides_red}} = \mu_{\text{chlorides_white}}$

H1: $\mu_{\text{chlorides_red}} > \mu_{\text{chlorides_white}}$

```
boot_perm_test(data_balanced$chlorides, data$color, R=1000, alter = "right")
```

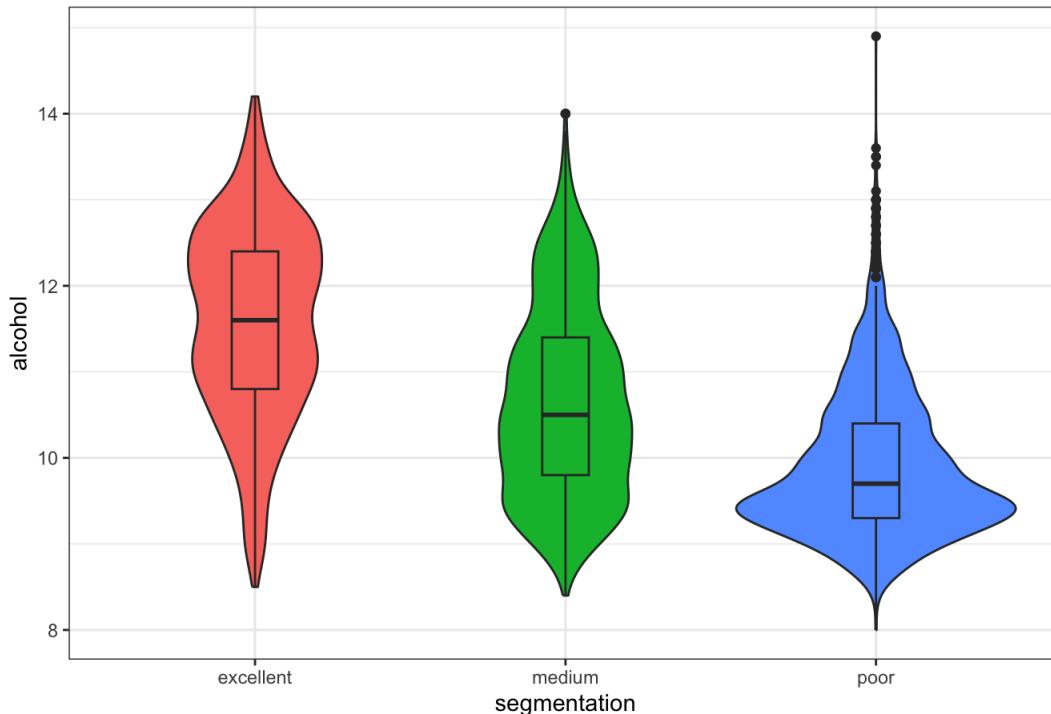
```
## $mean_A  
## [1] 0.04521045  
##  
## $mean_B  
## [1] 0.0749235  
##  
## $p_value  
## [1] 0.836
```

Với mức ý nghĩa alpha = 0.05, p-value = 0.836 > 0.05, không đủ cơ sở để bác bỏ giả thuyết H0, hay nói cách khác, lượng chlorides trong vang đỏ lớn hơn vang trắng chỉ là kết quả của sự trùng hợp ngẫu nhiên và không mang ý nghĩa thống kê.

III.2. Kiểm định ANOVA

```
plot_violin(data, "alcohol", "segmentation")
```

Violin plot of alcohol by segmentation



Kiểm định giả thuyết nồng độ cồn ở các phân khúc chất lượng rượu khác nhau thì khác nhau.

Đặt giả thuyết:

H0: $u_{\text{alcohol_poor}} = u_{\text{alcohol_medium}} = u_{\text{alcohol_excellent}}$

H1: $u_{\text{alcohol_poor}} \neq u_{\text{alcohol_medium}} \neq u_{\text{alcohol_excellent}}$

```
library(lmPerm)
set.seed(56)
out_aov_1 <- aovp(formula = alcohol ~ segmentation, data = data, perm = "Prob")
```

```
## [1] "Settings: unique SS "
```

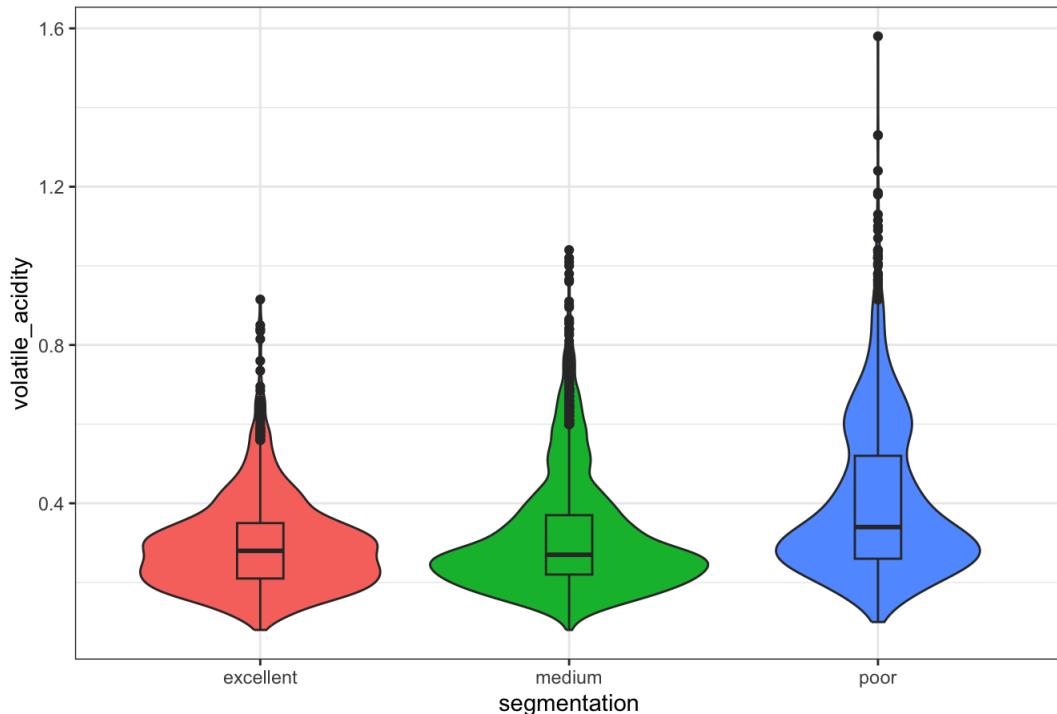
```
summary(out_aov_1)
```

```
## Component 1 :
##             Df  Sum Sq R Mean Sq Iter Pr(Prob)
## segmentation    2   1886.0   943.00 5000 < 2.2e-16 ***
## Residuals   5317   5594.8     1.05
## ---
## Signif. codes: 0 '****' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Với p-value rất nhỏ ~0, ta bác bỏ giả thiết H0, hay nói cách khác là có sự khác nhau về nồng độ cồn ở các phân khúc chất lượng rượu khác nhau.

```
plot_violin(data, "volatile_acidity", "segmentation")
```

Violin plot of volatile_acidity by segmentation



Kiểm định giả thuyết lượng axit dễ bay hơi ở các phân khúc chất lượng rượu khác nhau thì khác nhau.

Đặt giả thuyết:

H0: $u_{volacid_poor} = u_{volacid_medium} = u_{volacid_excellent}$

H1: $u_{volacid_poor} \neq u_{volacid_medium} \neq u_{volacid_excellent}$

```
set.seed(56)
out_aov_2 <- aovp(formula = volatile_acidity ~ segmentation, data = data, perm = "Prob")
```

```
## [1] "Settings: unique SS "
```

```
summary(out_aov_2)
```

```
## Component 1 :
##              Df Sum Sq R Mean Sq Iter Pr(Prob)
## segmentation  2   11.30 5.6500 5000 < 2.2e-16 ***
## Residuals   5317 139.27  0.0262
## ---
## Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Với p-value rất nhỏ ~0, ta bác bỏ giả thiết H0, hay nói cách khác là có sự khác nhau về lượng axit dễ bay hơi ở các phân khúc chất lượng rượu khác nhau.

IV. Building Models

```
# add categorical variables to both sets
data$color <- ifelse(data$color == 'red', 1, 2)
```

```
# merge red wine and white wine datasets
data$color = factor(data$color)
```

```
data <- subset(data, select = -segmentation)
head(data)
```

```

## fixed_acidity volatile_acidity citric_acid residual_sugar chlorides
## 1      7.4      0.70     0.00     1.9   0.076
## 2      7.8      0.88     0.00     2.6   0.098
## 3      7.8      0.76     0.04     2.3   0.092
## 4     11.2      0.28     0.56     1.9   0.075
## 5      7.4      0.66     0.00     1.8   0.075
## 6      7.9      0.60     0.06     1.6   0.069
## free_sulfur_dioxide total_sulfur_dioxide density p_h sulphates alcohol
## 1        11       34 0.9978 3.51   0.56   9.4
## 2        25       67 0.9968 3.20   0.68   9.8
## 3        15       54 0.9970 3.26   0.65   9.8
## 4        17       60 0.9980 3.16   0.58   9.8
## 5        13       40 0.9978 3.51   0.56   9.4
## 6        15       59 0.9964 3.30   0.46   9.4
## quality color
## 1      5   1
## 2      5   1
## 3      5   1
## 4      6   1
## 5      5   1
## 6      5   1

```

IV.1. Chuẩn bị dữ liệu

IV.1.1. Train Test split

```

# get features and target
features = colnames(data)
features = subset(features, features != "quality")
target = "quality"

# function for train test split
train_test_split = function(data, test_size) {
  set.seed(48)
  ind = sample(nrow(data), size = nrow(data) * test_size, replace = FALSE, prob = NULL)
  train_set = data[-ind,]
  test_set = data[ind,]

  return(list("train_set" = train_set, "test_set" = test_set))
}

# split datasets with test_size = 0.25
traintest_data = train_test_split(data, test_size = 0.25)
datatrain = traintest_data$train_set
datatest = traintest_data$test_set

```

Plot the distribution of the quality variable for each dataset

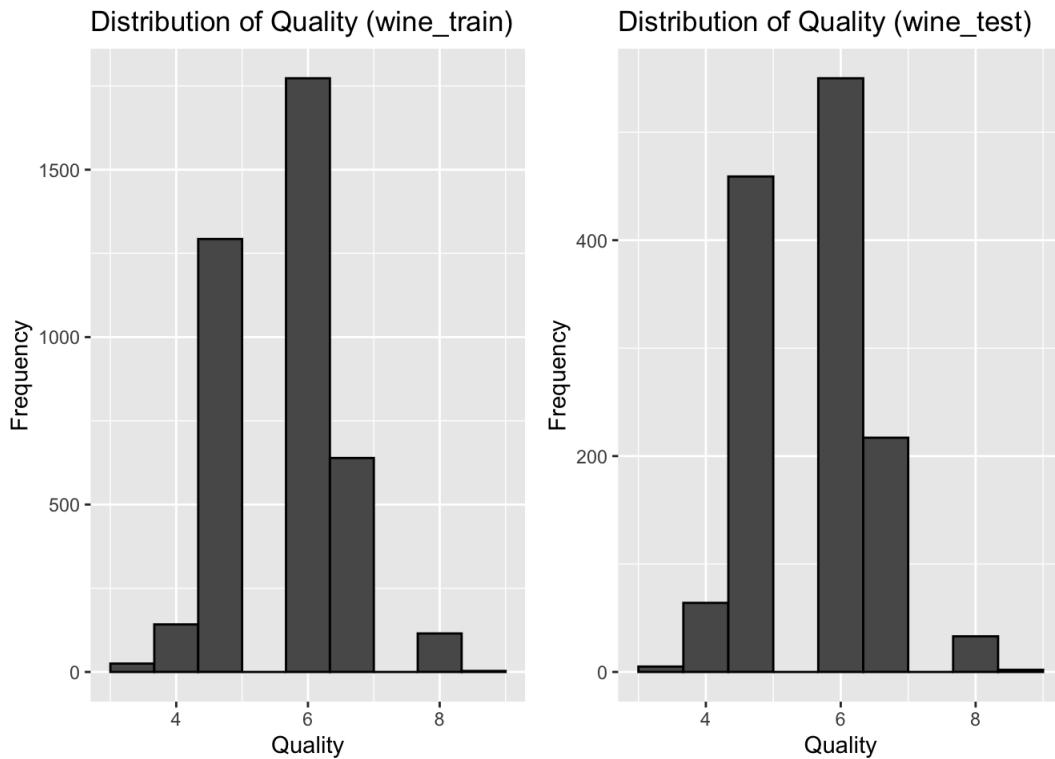
```

# Create the plots
plot1 <- ggplot(datatrain, aes(x = quality)) +
  geom_histogram(bins = 10, color = "black") +
  labs(x = "Quality", y = "Frequency", title = "Distribution of Quality (wine_train)")

plot2 <- ggplot(datatest, aes(x = quality)) +
  geom_histogram(bins = 10, color = "black") +
  labs(x = "Quality", y = "Frequency", title = "Distribution of Quality (wine_test)")

# Arrange the plots in a grid
grid.arrange(grobs = list(plot1, plot2), nrow = 1)

```



IV.1.2. Scale data

```
# write function scaling Testing data with respect to Training data
scale_data = function(traindata, testdata){
  train_scaled <- scale(traindata[,(names(traindata) %in% c("color", "quality"))])
  test_scaled <- scale(testdata[,(names(testdata) %in% c("color", "quality"))]),
    center = attr(train_scaled, "scaled:center"),
    scale = attr(train_scaled, "scaled:scale"))

  # Add the quality column back to the scaled datasets
  train_scaled <- as.data.frame(cbind(train_scaled, color = traindata$color, quality = traindata$quality))
  test_scaled <- as.data.frame(cbind(test_scaled, color = testdata$color, quality = testdata$quality))

  return(list("train_scaled" = train_scaled, "test_scaled" = test_scaled))
}

# scale on data
data_scaled = scale_data(datatrain, datatest)
datatrain_scaled = data_scaled$train_scaled
datatest_scaled = data_scaled$test_scaled
```

IV.2. Regression Model

IV.2.1. Linear Regression

IV.2.1.1. Lựa chọn mô hình

```
#specify the cross-validation method
train_control <- trainControl(method = "cv", number = 10, savePredictions = TRUE)
#fit a regression model and use k-fold CV to evaluate performance
md_lm_data <- train(quality ~ ., data = datatrain_scaled, method = "lm", trControl = train_control)
summary(md_lm_data)
```

```
##  
## Call:  
## lm(formula = .outcome ~ ., data = dat)  
##  
## Residuals:  
##   Min     1Q Median     3Q    Max  
## -3.8246 -0.4507 -0.0397  0.4654  3.0263  
##  
## Coefficients:  
##             Estimate Std. Error t value Pr(>|t|)  
## (Intercept)  6.43912  0.12956 49.700 < 2e-16 ***  
## fixed_acidity 0.15524  0.02751  5.643 1.78e-08 ***  
## volatile_acidity -0.22437  0.01700 -13.202 < 2e-16 ***  
## citric_acid    0.01905  0.01488  1.280  0.200  
## residual_sugar 0.30906  0.03463  8.924 < 2e-16 ***  
## chlorides      -0.02381  0.01485 -1.603  0.109  
## free_sulfur_dioxide 0.09981  0.01740  5.735 1.04e-08 ***  
## total_sulfur_dioxide -0.10024  0.02361 -4.245 2.23e-05 ***  
## density        -0.40283  0.05795 -6.952 4.19e-12 ***  
## p_h            0.12715  0.01897  6.701 2.36e-11 ***  
## sulphates      0.11856  0.01463  8.105 6.98e-16 ***  
## alcohol         0.21578  0.02984  7.231 5.74e-13 ***  
## color          -0.36352  0.07410 -4.906 9.67e-07 ***  
## ---  
## Signif. codes: 0 '****' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1  
##  
## Residual standard error: 0.7279 on 3977 degrees of freedom  
## Multiple R-squared: 0.3117, Adjusted R-squared: 0.3097  
## F-statistic: 150.1 on 12 and 3977 DF, p-value: < 2.2e-16
```

```

# Hàm kiểm định cho hệ số
create_results_dataframe = function(model, boot_model, conf_level = 0.95) {
  # Tính ước lượng cho các hệ số từ mô hình hồi quy
  est = coef(model)

  # Tính độ lệch chuẩn của ước lượng hệ số từ bootstrap
  se = apply(boot_model$t, 2, sd)

  # Tính khoảng tin cậy 95% cho các hệ số từ bootstrap
  ci_95 = sapply(1:ncol(boot_model$t), function(i) {
    CI = boot.ci(boot_model, index = i, type = "perc", conf = conf_level)$percent[1, 4:5]
    paste0("(", round(CI[1], 2), ", ", round(CI[2], 2), ")")
  })

  # Tính p-value cho các hệ số từ bootstrap
  p_values = sapply(1:ncol(boot_model$t), function(x) {
    qt0 = mean(boot_model$t[, x] <= 0)
    if (qt0 < 0.5) {
      return(2*qt0)
    } else {
      return(2*(1 - qt0))
    }
  })

  # Tạo dataframe
  df_results = data.frame(
    Est = est,
    SE = se,
    CI_95 = ci_95,
    p_value = p_values,
    row.names = names(est)
  )

  # Vẽ histogram của các kết quả ước lượng bootstrap của hệ số
  nrow = 2
  ncol <- ceiling(length(est) / nrow)
  par(mfrow = c(nrow, ncol))
  for (i in 1:length(est)) {
    hist(boot_model$t[, i], main = names(est)[i], xlab = names(est)[i])
  }

  return(df_results)
}

boot_func = function(data, ind, formula, ...){
  data_new = data[ind,]
  model = lm(formula = formula, data = data_new, ...)
  return(model$coefficients)
}

```

IV.2.1.2. Kiểm tra tính đa cộng tuyến (Multicollinearity)

Trong multicollinearity (tính đa cộng tuyến), collinearity (tính cộng tuyến) tồn tại giữa ba hoặc nhiều biến ngay cả khi không có cặp biến nào là tương quan cao. Có nghĩa là có sự dư thừa giữa các biến dự đoán.

Khi có tính đa cộng tuyến, kết quả của mô hình hồi quy trở nên không ổn định.

Tính đa cộng tuyến có thể được đánh giá bằng cách tính một chỉ số gọi là hệ số phỏng đại phương sai (variance inflation factor - VIF), nó đo lường mức độ phương sai của một hệ số hồi quy bị phỏng đại do tính đa cộng tuyến trong mô hình.

```

linear_model1 = lm(quality ~., data = datatrain_scaled)
boot_linear_model1 = boot(data = datatrain_scaled, statistic = boot_func, R = 1000, formula = quality ~.)

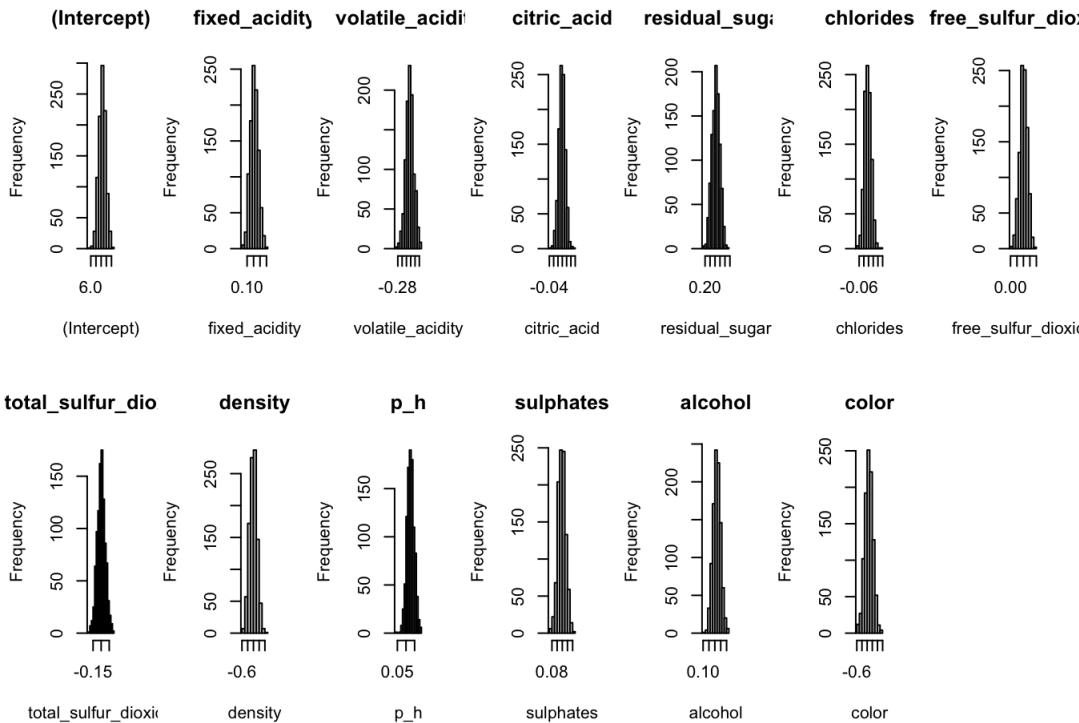
results_df1 = create_results_dataframe(model = linear_model1, boot_model = boot_linear_model1)
print(results_df1)

```

```

##             Est      SE   CI_95 p_value
## (Intercept) 6.43911680 0.13416939 (6.18, 6.72) 0.000
## fixed_acidity 0.15523814 0.03061930 (0.1, 0.22) 0.000
## volatile_acidity -0.22436789 0.01838217 (-0.26, -0.19) 0.000
## citric_acid 0.01904874 0.01475635 (-0.01, 0.05) 0.200
## residual_sugar 0.30905931 0.03927214 (0.23, 0.38) 0.000
## chlorides -0.02380509 0.01395288 (-0.05, 0.01) 0.102
## free_sulfur_dioxide 0.09981077 0.02958338 (0.04, 0.16) 0.000
## total_sulfur_dioxide -0.10023950 0.02474203 (-0.15, -0.05) 0.000
## density -0.40283471 0.06407432 (-0.52, -0.28) 0.000
## p_h 0.12714820 0.02057015 (0.09, 0.17) 0.000
## sulphates 0.11856291 0.01478442 (0.09, 0.15) 0.000
## alcohol 0.21578455 0.03213084 (0.15, 0.28) 0.000
## color -0.36351648 0.07719733 (-0.52, -0.21) 0.000

```



Hệ số VIF của một biến dự đoán là một thước đo mức độ dễ dàng mà nó được dự đoán từ hồi quy tuyến tính sử dụng các biến dự đoán khác.

```
library(car)
```

```
## Loading required package: carData
```

```
## 
## Attaching package: 'car'
```

```
## The following object is masked from 'package:psych':
## 
##   logit
```

```
## The following object is masked from 'package:boot':
## 
##   logit
```

```
## The following object is masked from 'package:dplyr':
## 
##   recode
```

```
## The following object is masked from 'package:purrr':
## 
##   some
```

```
vif(linear_model1)
```

```

## fixed_acidity volatile_acidity citric_acid
## 5.696423 2.174477 1.666137
## residual_sugar chlorides free_sulfur_dioxide
## 9.029706 1.660649 2.279828
## total_sulfur_dioxide density p_h
## 4.197297 25.276416 2.710213
## sulphates alcohol color
## 1.611092 6.704478 7.922001

```

Có thể thấy sự tồn tại của tính đa cộng tuyến ở đây, ta loại bỏ density vì nó gây ra hiện tượng đa cộng tuyến (VIF = 25.2764)

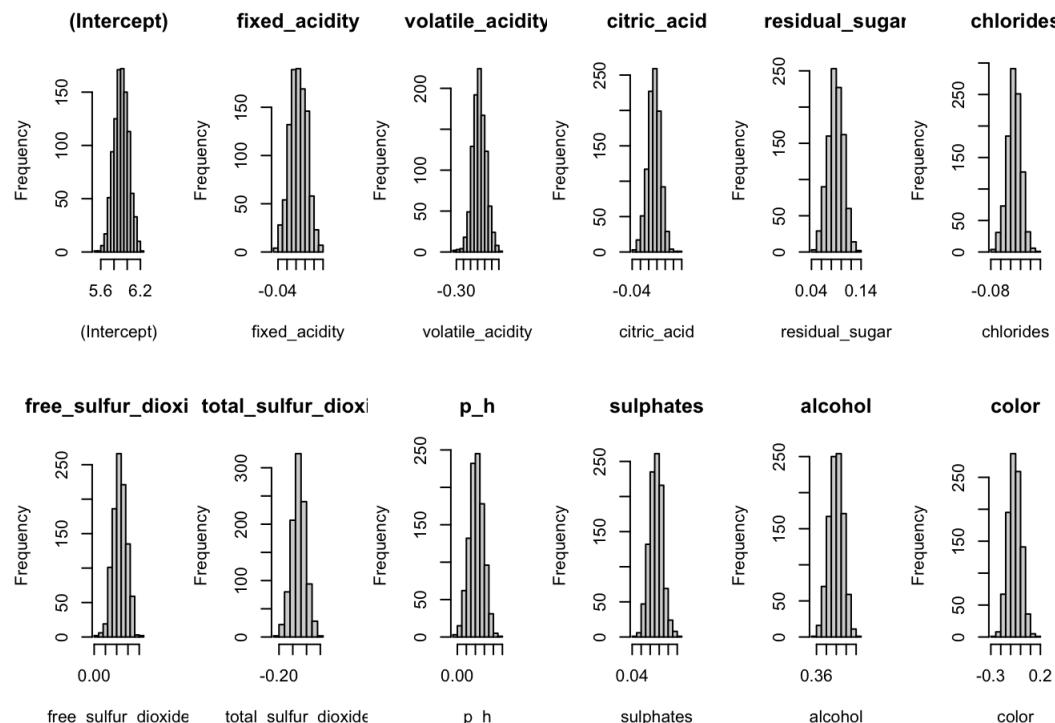
Fitting model mới

```

linear_model2 = lm(quality ~ . - density , data = datatrain_scaled)
boot_linear_model2 = boot(data = datatrain_scaled, statistic = boot_func, R = 1000, formula = quality ~ . - density )

results_df2 = create_results_dataframe(model = linear_model2, boot_model = boot_linear_model2)

```



```
print(results_df2)
```

	Est	SE	CI_95	p_value
## (Intercept)	5.908295740	0.10908915	(5.7, 6.13)	0.000
## fixed_acidity	0.006016967	0.01862397	(-0.03, 0.04)	0.814
## volatile_acidity	-0.234947899	0.01839219	(-0.27, -0.2)	0.000
## citric_acid	0.012764439	0.01524189	(-0.02, 0.04)	0.376
## residual_sugar	0.088556506	0.01514358	(0.06, 0.12)	0.000
## chlorides	-0.032922903	0.01365530	(-0.06, -0.01)	0.014
## free_sulfur_dioxide	0.113531432	0.02956621	(0.06, 0.17)	0.000
## total_sulfur_dioxide	-0.128502062	0.02505602	(-0.18, -0.08)	0.000
## p_h	0.042882089	0.01568089	(0.01, 0.07)	0.006
## sulphates	0.092324685	0.01414038	(0.07, 0.12)	0.000
## alcohol	0.400124614	0.01423662	(0.37, 0.43)	0.000
## color	-0.058727875	0.06324134	(-0.19, 0.06)	0.366

Tương tự, chúng ta xây dựng một model khác sau khi loại bỏ fixed_acidity

```

linear_model3 = lm(quality ~ . -density -fixed_acidity, data = datatrain_scaled)
boot_linear_model3 = boot(data = datatrain_scaled, statistic = boot_func, R = 1000,
                           formula = quality ~ . -density -fixed_acidity)

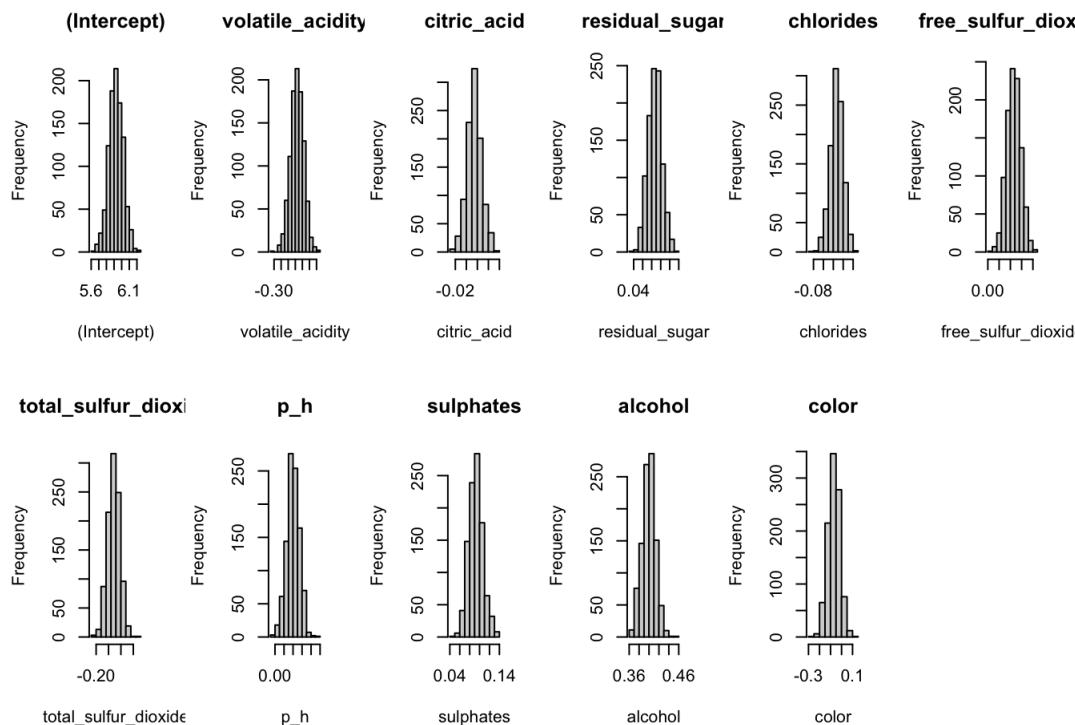
results_df3 = create_results_dataframe(model = linear_model3, boot_model = boot_linear_model3)
print(results_df3)

```

```

##             Est      SE   CI_95 p_value
## (Intercept) 5.92352671 0.09348045 (5.74, 6.11) 0.000
## volatile_acidity -0.23489211 0.01858978 (-0.27, -0.2) 0.000
## citric_acid 0.01480642 0.01321112 (-0.01, 0.04) 0.252
## residual_sugar 0.08837613 0.01524431 (0.06, 0.12) 0.000
## chlorides -0.03353551 0.01292328 (-0.06, -0.01) 0.004
## free_sulfur_dioxide 0.11295248 0.03078563 (0.06, 0.18) 0.000
## total_sulfur_dioxide -0.12837615 0.02479554 (-0.18, -0.08) 0.000
## p_h          0.04061290 0.01430307 (0.01, 0.07) 0.006
## sulphates    0.09267983 0.01429508 (0.07, 0.12) 0.000
## alcohol       0.39960352 0.01344047 (0.37, 0.43) 0.000
## color        -0.06747325 0.05400822 (-0.18, 0.04) 0.178

```



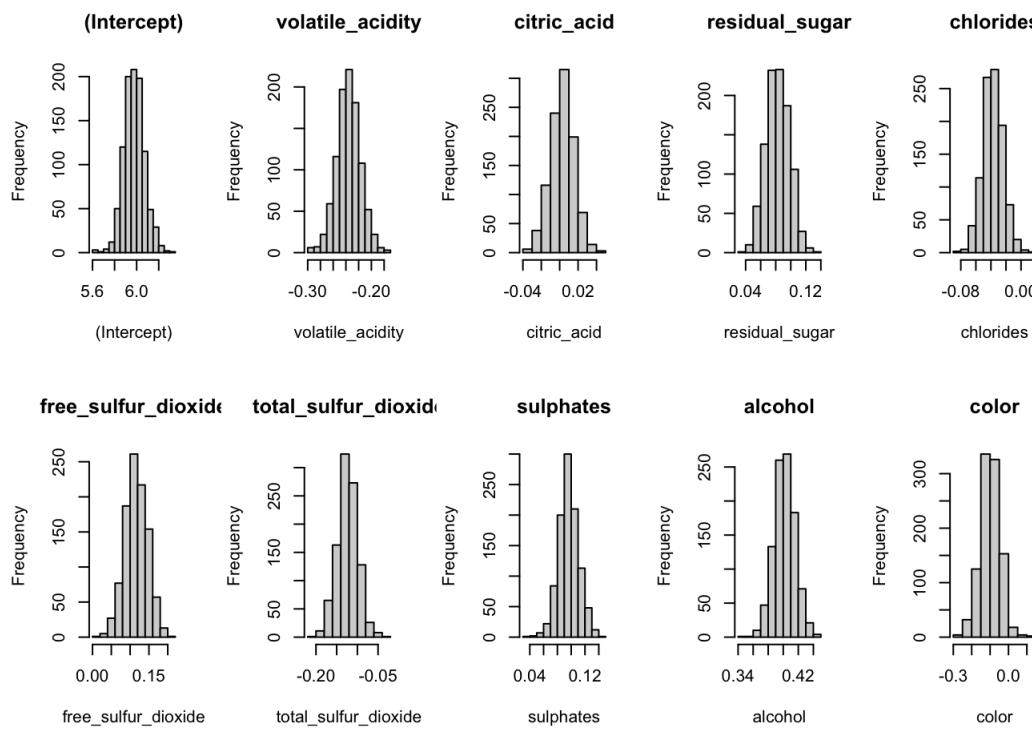
Tương tự, chúng ta xây dựng một model khác sau khi loại bỏ `p_h`

```

linear_model4 = lm(quality ~ . -density -fixed_acidity -p_h, data = datatrain_scaled)
boot_linear_model4 = boot(data = datatrain_scaled, statistic = boot_func, R = 1000,
                           formula = quality ~ . -density -fixed_acidity -p_h)

```

```
results_df4 = create_results_dataframe(model = linear_model4, boot_model = boot_linear_model4)
```



```
print(results_df4)
```

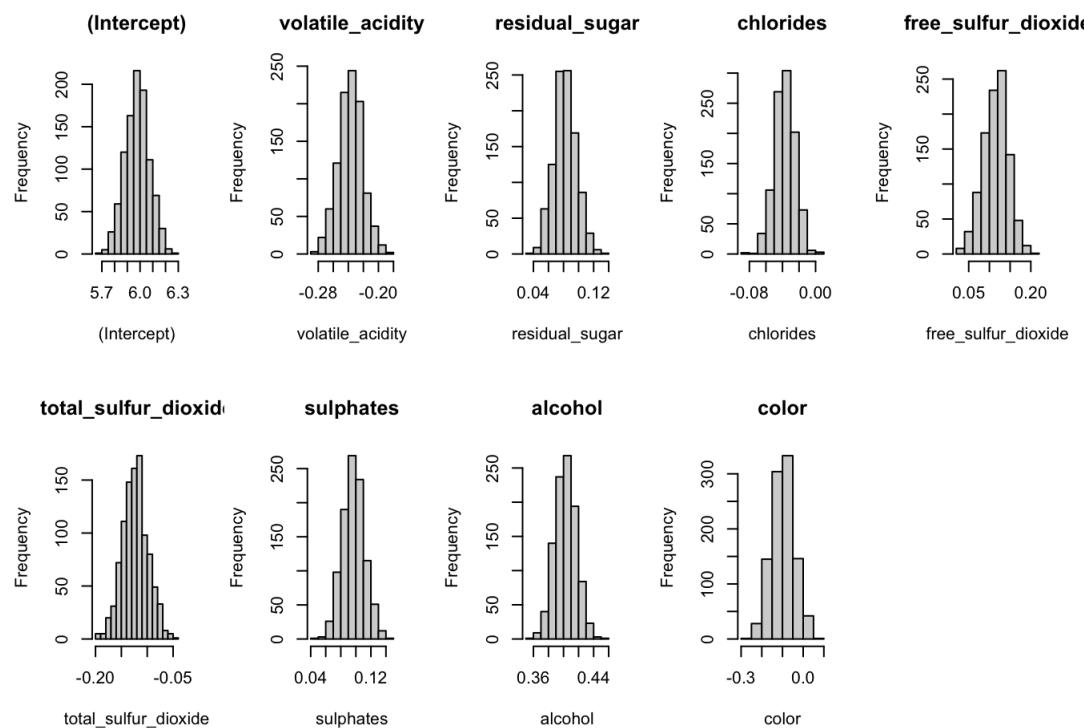
```
##             Est      SE   CI_95 p_value
## (Intercept) 5.977788333 0.09289161 (5.8, 6.17) 0.000
## volatile_acidity -0.236197355 0.01879726 (-0.27, -0.2) 0.000
## citric_acid    0.002487828 0.01286094 (-0.02, 0.03) 0.800
## residual_sugar  0.083079934 0.01541629 (0.05, 0.11) 0.000
## chlorides       -0.037777168 0.01352157 (-0.06, -0.01) 0.010
## free_sulfur_dioxide 0.1114003517 0.03008698 (0.06, 0.17) 0.000
## total_sulfur_dioxide -0.123715055 0.02459891 (-0.17, -0.08) 0.000
## sulphates        0.096450556 0.01454165 (0.07, 0.13) 0.000
## alcohol          0.402766929 0.01406736 (0.37, 0.43) 0.000
## color            -0.098629364 0.05358616 (-0.21, 0) 0.048
```

Tương tự, chúng ta xây dựng một model khác sau khi loại bỏ citric_acid

```
linear_model5 = lm(quality ~ . -density -fixed_acidity -p_h -citric_acid, data = datatrain_scaled)
boot_linear_model5 = boot(data = datatrain_scaled, statistic = boot_func, R = 1000,
                           formula = quality ~ . -density -fixed_acidity -p_h -citric_acid)
```

```
results_df5 = create_results_dataframe(model = linear_model5, boot_model = boot_linear_model5)
print(results_df5)
```

```
##             Est      SE   CI_95 p_value
## (Intercept) 5.97893046 0.09476613 (5.79, 6.16) 0.000
## volatile_acidity -0.23739627 0.01630876 (-0.27, -0.21) 0.000
## residual_sugar  0.08337194 0.01482225 (0.05, 0.11) 0.000
## chlorides       -0.03725380 0.01244252 (-0.06, -0.01) 0.006
## free_sulfur_dioxide 0.11373241 0.03022182 (0.05, 0.17) 0.000
## total_sulfur_dioxide -0.12320999 0.02452046 (-0.17, -0.07) 0.000
## sulphates        0.09666791 0.01457774 (0.07, 0.13) 0.000
## alcohol          0.40302558 0.01413106 (0.38, 0.43) 0.000
## color            -0.09928515 0.05448655 (-0.2, 0.01) 0.086
```



IV.2.1.3. Tại sao lại chọn linear_model5?

- linear_model1 : quality = 4.4 + 0.1 fixed_acidity -0.23 volatile_acidity - 0.009 citric_acid + 0.29 residual_sugar - 0.02 chlorides + 0.06 free_sulfur_dioxide - 0.08 total_sulfur_dioxide - 0.29 density - 0.07 p_h + 0.11 sulphates + 0.27 alcohol -0.33 color
- linear_model5 : quality = 4.04 -0.24 volatile_acidity + 0.1 residual_sugar - 0.03 chlorides + 0.08 free_sulfur_dioxide - 0.1 total_sulfur_dioxide + 0.09 sulphates + 0.29 alcohol -0.1 color

Sự khác biệt chính giữa hai mô hình là linear_model1 bao gồm tất cả các biến, trong khi linear_model5 loại bỏ density, fixed_acidity, p_h và citric_acid.

Kết quả cho thấy rằng linear_model5 giải thích khoảng 28.18% biến quality. Khi các biến bị loại bỏ được thêm lại vào linear_model1, khả năng giải thích tăng lên chỉ 0.58% lên 28.76%. Câu hỏi là liệu việc tăng này 0.58% có ý nghĩa thống kê hay không.

Để trả lời câu hỏi này, thực hiện kiểm định giả thuyết cho từng biến bị loại bỏ và kết quả như sau:

- + density : p = 0.66
- + fixed_acidity : p = 0.71
- + p_h : p = 0.45
- + citric_acid. : p = 0.27

Vì tất cả p-value này đều lớn hơn mức ý nghĩa 0.05, ta kết luận rằng các biến bị loại bỏ không cung cấp thông tin bổ sung hay cải thiện dự đoán về chất lượng hơn những gì đã được giải thích bởi các biến trong linear_model5. Điều này ngụ ý rằng những biến này không có ảnh hưởng đáng kể đối với chất lượng rượu vang đỏ và việc bao gồm chúng vào mô hình không cải thiện khả năng dự đoán.

Tóm lại, sau khi tính toán ảnh hưởng của bảy biến này, các biến bổ sung không có tác động thống kê đáng kể đến dự đoán về chất lượng.

IV.2.1.4 Thực hiện thống kê suy luận

a. F-test

- Thực hiện các thống kê suy luận cho mô hình linear_model5 vừa xác định được. Ta áp dụng phương pháp bootstrap để ước lượng khoảng tin cậy và kiểm định giả thuyết $\beta_j = 0$.
“Có thực sự tồn tại mối liên hệ giữa biến X_j và Y ”.

```
linear_model = lm(quality ~ . -density -fixed_acidity -p_h -citric_acid, data = datatrain_scaled)
boot_linear_model = boot(data = datatrain_scaled, statistic = boot_func, R = 1000,
                         formula = quality ~ . -density -fixed_acidity -p_h -citric_acid)

summary(linear_model)
```

```
##
## Call:
## lm(formula = quality ~ . - density - fixed_acidity - p_h - citric_acid,
##     data = datatrain_scaled)
##
## Residuals:
##   Min     1Q Median     3Q    Max 
## -3.8349 -0.4513 -0.0352  0.4596  3.1244 
## 
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)    
## (Intercept) 5.97893   0.09400  63.605 < 2e-16 ***
## volatile_acidity -0.23740   0.01581 -15.013 < 2e-16 ***
## residual_sugar   0.08337   0.01380  6.041 1.67e-09 ***
## chlorides      -0.03725   0.01446 -2.576  0.0100 *  
## free_sulfur_dioxide 0.11373   0.01726  6.588 5.05e-11 ***
## total_sulfur_dioxide -0.12321   0.02321 -5.308 1.17e-07 ***
## sulphates       0.09667   0.01409  6.859 8.00e-12 ***
## alcohol         0.40303   0.01360 29.641 < 2e-16 ***
## color           -0.09929   0.05356 -1.854  0.0639 .  
## --- 
## Signif. codes:  0 '****' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1 
## 
## Residual standard error: 0.7329 on 3981 degrees of freedom
## Multiple R-squared:  0.3017, Adjusted R-squared:  0.3003 
## F-statistic:  215 on 8 and 3981 DF,  p-value: < 2.2e-16
```

F-Statistic - p value <= 2.2e -16

```
print(boot_linear_model)
```

```

## 
## ORDINARY NONPARAMETRIC BOOTSTRAP
##
## Call:
## boot(data = datatrain_scaled, statistic = boot_func, R = 1000,
##   formula = quality ~ . - density - fixed_acidity - p_h - citric_acid)
##
## 
## Bootstrap Statistics :
##      original    bias   std. error
## t1*  5.97893046 -2.037716e-03  0.09602391
## t2* -0.23739627  1.233660e-03  0.01732058
## t3*  0.08337194 -3.693346e-05  0.01554832
## t4* -0.03725380 -1.872886e-04  0.01243698
## t5*  0.11373241  3.549700e-03  0.02892562
## t6* -0.12320999 -1.919839e-03  0.02428694
## t7*  0.09666791  8.096792e-04  0.01498806
## t8*  0.40302558 -5.620842e-05  0.01405609
## t9* -0.09928515  9.067619e-04  0.05520420

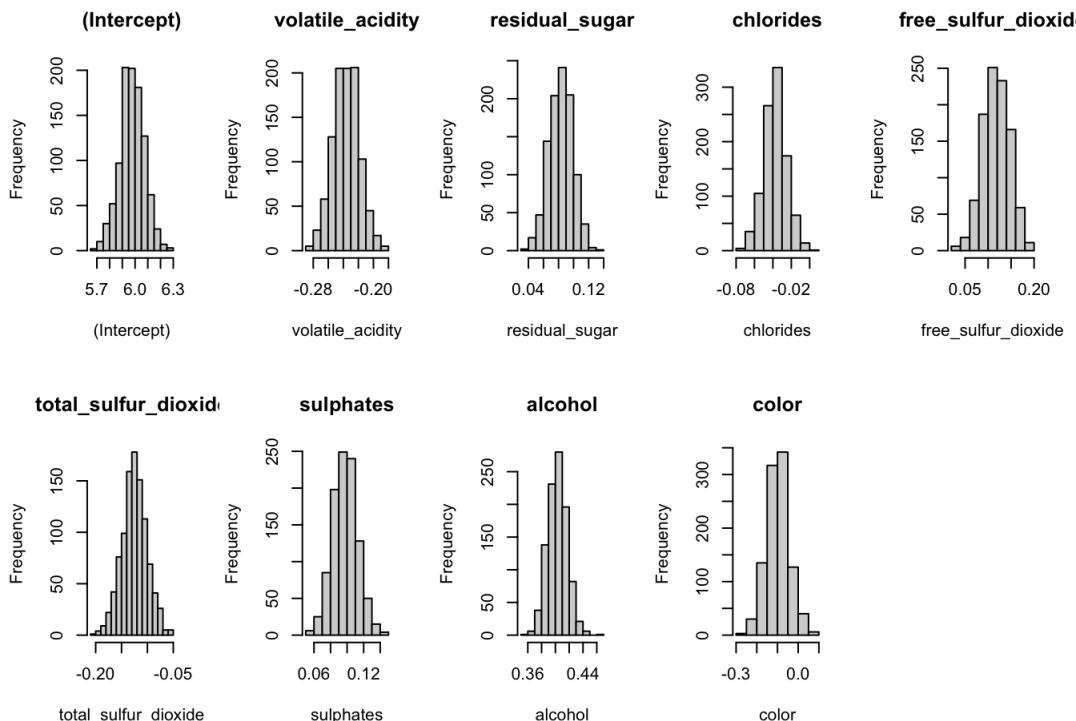
```

```

results_df = create_results_dataframe(model = linear_model, boot_model = boot_linear_model)
print(results_df)

```

	Est	SE	CI_95	p_value
## (Intercept)	5.97893046	0.09602391	(5.78, 6.16)	0.000
## volatile_acidity	-0.23739627	0.01732058	(-0.27, -0.2)	0.000
## residual_sugar	0.08337194	0.01554832	(0.05, 0.11)	0.000
## chlorides	-0.03725380	0.01243698	(-0.06, -0.01)	0.002
## free_sulfur_dioxide	0.11373241	0.02892562	(0.06, 0.17)	0.000
## total_sulfur_dioxide	-0.12320999	0.02428694	(-0.17, -0.08)	0.000
## sulphates	0.09666791	0.01498806	(0.07, 0.13)	0.000
## alcohol	0.40302558	0.01405609	(0.38, 0.43)	0.000
## color	-0.09928515	0.05520420	(-0.2, 0.01)	0.092



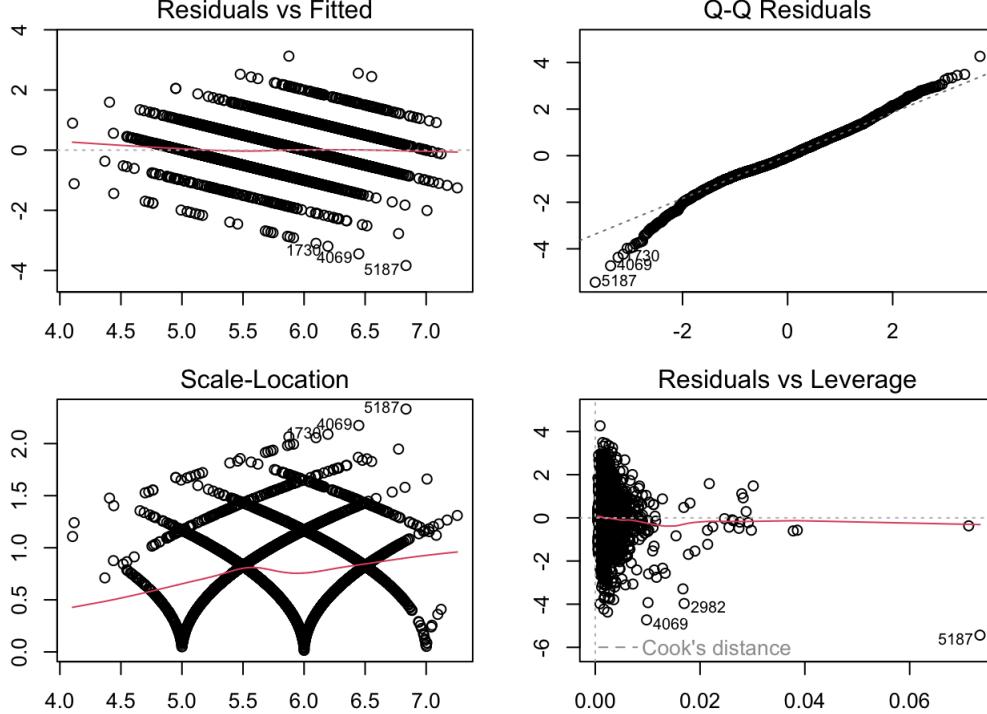
Chúng ta có thể thấy p value nhỏ hơn 0.05. Vì vậy, ta bác bỏ giả thuyết và chấp nhận đối thuyết, hay có thể nói rằng mô hình là có ý nghĩa.

b. ANOVA

```

par(mfrow=c(2,2), oma = c(1,1,0,0) + 0.1, mar = c(3,3,1,1) + 0.1)
plot(linear_model)

```



- Biểu đồ Residuals vs Fitted cho thấy nếu phần dư có mô hình phi tuyến. Phần dư xung quanh một đường ngang mà không có mô hình rõ ràng, đây là dấu hiệu tốt cho thấy không có mối quan hệ phi tuyến.
- Biểu đồ QQ chuẩn cho thấy phần dư fit với đường thẳng. Do đó, có thể gọi là phân phối chuẩn của phần dư.
- Biểu đồ Scale-Location cho thấy nếu phần dư phân bố đều dọc theo các phạm vi của các biến dự đoán. Đây là cách để kiểm tra giả thuyết về đồng nhất phương sai (homoscedasticity). Nó thoa nếu bạn thấy một đường ngang đi qua các điểm phân bố đều (ngẫu nhiên).
- Biểu đồ Residuals vs Leverage có dáng vẻ đặc trưng khi có một trường hợp có ảnh hưởng lớn. Bạn có thể thấy rõ khoảng cách Cook vì tất cả các trường hợp đều nằm trong khoảng cách Cook.

c. Kiểm định khoảng tin cậy 95% cho trung bình

Giả sử nồng độ phần trăm của các chất là:

```
head(datatest_scaled, 1)
```

```
## fixed_acidity volatile_acidity citric_acid residual_sugar chlorides
## 5189 -1.218061 -0.5054646 1.23336 1.472659 -0.7392151
## free_sulfur_dioxide total_sulfur_dioxide density p_h
## 5189 -0.274417 -0.3589284 -0.07272391 0.009440656
## sulphates alcohol color quality
## 5189 -0.2941747 -0.05650814 2 6
```

Khi đó, dựa vào mô hình, ta có thể ước tính được chất lượng rượu như sau:

```
new_data = head(datatest_scaled, 1)
pred_data = predict(linear_model, new_data)
pred_data
```

```
## 5189
## 6.012475
```

Và để tìm khoảng tin cậy cho giá trị trung bình chất lượng rượu, ta sử dụng phương pháp bootstrap. Vì đã chạy bootstrap ở phần trước với 1000 lần lặp để ước tính các hệ số, ta có thể sử dụng kết quả này để tính các giá trị ước đoán trung bình chất lượng rượu trong mỗi lần lặp mẫu:

```
x_wine <- as.numeric(cbind(1,new_data[,1]))
y_wine <- apply(boot_linear_model$t, 1, function(x){x_wine %*% t(x)})
quantile(y_wine, probs = c(0.025, 0.975))
```

```
## 2.5% 97.5%
## -2.180365 7.492234
```

Như vậy, trung bình chất lượng rượu được tiên lượng theo mô hình này, có giá trị thay đổi từ (-2.180365, 7.492234), với độ tin cậy 95%.

d. Ước lượng khoảng tiên đoán 95% độ tin cậy

Ngoài ra, ta còn có thể sử dụng phương pháp bootstrap để ước lượng khoảng tiên đoán cho chất lượng rượu, theo nồng độ các chất tan có trong

rượu, dựa trên mô hình:

```
resid_data = residuals(linear_model)
y_wine_pd_pci = y_wine + sample(resid_data, size = 1000, replace = TRUE)
quantile(y_wine_pd_pci, probs = c(0.025, 0.975))
```

```
## 2.5% 97.5%
## -2.742206 7.918966
```

Như vậy, khoảng tiên đoán cho chất lượng rượu được tiên lượng theo mô hình này, có giá trị thay đổi từ (-2.742206, 7.918966), với độ tin cậy 95%.

IV.2.1.5. Đánh giá mô hình

Tập dữ liệu test ban đầu:

```
head(datatest_scaled$quality)
```

```
## [1] 6 6 5 6 7 7
```

Giá trị chất lượng dự đoán:

```
pred <- predict(linear_model, datatest_scaled)
head(pred)
```

```
## 5189 2218 179 153 2511 2314
## 6.012475 5.702439 5.629716 5.344769 5.995686 6.462089
```

Làm tròn các giá trị dự đoán này, ta được:

```
pred = round(pred)
head(pred)
```

```
## 5189 2218 179 153 2511 2314
## 6 6 6 5 6 6
```

Từ đó, ta có thể xây dựng một confusion matrix đơn giản như sau:

```
tst_tab <- table(predicted = pred, actual = datatest_scaled$quality)
tst_tab
```

```
##      actual
## predicted 3 4 5 6 7 8 9
##        4 0 1 2 0 0 0 0
##        5 1 36 236 107 6 2 0
##        6 4 27 215 410 155 18 1
##        7 0 0 5 33 56 13 1
##        8 0 0 1 0 0 0 0
```

```
RMSE_value <- RMSE(pred, datatest_scaled$quality)
MAE_value <- MAE(pred, datatest_scaled$quality)
R_squared <- R2(pred, datatest_scaled$quality)
accuracy <- sum(diag(tst_tab))/length(datatest_scaled$quality)
# Hiển thị kết quả
cat("RMSE:", RMSE_value, "\n")
```

```
## RMSE: 0.8064589
```

```
cat("MAE:", MAE_value, "\n")
```

```
## MAE: 0.5270677
```

```
cat("R-squared:", R_squared, "\n")
```

```
## R-squared: 0.2156041
```

```
cat("Accuracy: ", accuracy, "\n")
```

IV.2.1.6. Nhận xét và rút ra kết luận cho mô hình

linear_model5

```
##
## Call:
## lm(formula = quality ~ . - density - fixed_acidity - p_h - citric_acid,
##     data = datatrain_scaled)
##
## Coefficients:
## (Intercept) volatile_acidity residual_sugar
## 5.97893      -0.23740       0.08337
## chlorides free_sulfur_dioxide total_sulfur_dioxide
## -0.03725      0.11373      -0.12321
## sulphates alcohol color
## 0.09667      0.40303      -0.09929
```

Cụ thể, thông tin từ tổng hợp bootstrap cho thấy các hệ số cho, residual_sugar free_sulfur_dioxide , sulphates và alcohol đều dương trên tất cả các mẫu bootstrap, cho thấy các giá trị cao hơn của các biến này liên quan đến chất lượng rượu vang tốt hơn. Ngược lại, các hệ số cho total_sulfur_dioxide , chlorides , volatile_acidity và color lại âm, cho thấy các giá trị cao hơn của các biến này liên quan đến chất lượng rượu vang kém hơn. If the winemaker wants to increase the quality of the data wine, it would be beneficial to: Nếu nhà sản xuất rượu muốn tăng chất lượng của rượu vang, họ có thể làm bằng cách:

- Tăng nồng độ residual_sugar : Đường dư có thể góp phần vào vị ngọt và cấu trúc của rượu, có thể cải thiện hương vị và chất lượng tổng thể của sản phẩm.
- Tăng nồng độ free_sulfur_dioxide : Lưu huỳnh dioxit tự do là chất chống oxy hóa giúp ngăn ngừa sự oxy hóa và hư hỏng trong rượu. Tăng nồng độ này có thể giúp cải thiện sự tươi mới và chất lượng tổng thể của rượu.
- Tăng nồng độ sulphates : Sunfua là loại chất bảo quản giúp ngăn ngừa sự hư hỏng và oxy hóa trong rượu. Tăng nồng độ sunfua có thể giúp cải thiện sự ổn định và chất lượng của rượu.
- Tăng nồng độ alcohol : Cồn là yếu tố quan trọng ảnh hưởng đến chất lượng rượu vang, ảnh hưởng đến cấu trúc, hương vị và khả năng lão hóa của rượu. Rượu vang với mức cồn lý tưởng (thường từ 12% đến 15% ABV) thường cân bằng và hài hòa hơn, trong khi rượu vang với mức cồn quá thấp hoặc quá cao có thể thiếu cân bằng và thiếu tính cá nhân. Tăng nồng độ cồn có thể giúp cải thiện hương vị và chất lượng tổng thể của rượu vang.

Hoặc:

- Giảm nồng độ total_sulfur_dioxide : Mặc dù SO2 là chất bảo quản quan trọng trong rượu, mức độ cao có thể ảnh hưởng tiêu cực đến hương vị và mùi của rượu. Giảm nồng độ SO2 tổng có thể giúp cải thiện tính cách và chất lượng tổng thể của sản phẩm.
- Giảm nồng độ color : Mặc dù màu sắc là một yếu tố quan trọng trong rượu, màu sắc quá mạnh có thể liên quan đến chất lượng thấp hơn của rượu. Giảm độ mạnh màu sắc có thể cải thiện chất lượng tổng thể của sản phẩm.
- Giảm nồng độ chlorides : Clo là một loại muối góp phần vào vị đắng và cảm giác khô. Giảm nồng độ clo có thể cải thiện hương vị và chất lượng tổng thể của rượu.
- Giảm nồng độ volatile_acidity : Độ axit bay hơi đề cập đến nồng độ axit như axit axetic, góp phần vào hương vị và mùi của rượu. Mặc dù một số axit bay hơi là cần thiết, mức độ cao có thể gây ra hương vị và mùi không mong muốn, ảnh hưởng tiêu cực đến chất lượng sản phẩm. Giảm nồng độ axit bay hơi có thể cải thiện hương vị và chất lượng tổng thể của rượu.

Bằng cách điều chỉnh các biến này, nhà sản xuất rượu có thể cải thiện chất lượng tổng thể của rượu vang đỏ. Tuy nhiên, cần lưu ý rằng quá trình sản xuất rượu vang là phức tạp, và thay đổi các biến này có thể có tác động không mong muốn đến hương vị, mùi và tính chất tổng thể của sản phẩm. Ngoài ra, cũng có những yếu tố khác ảnh hưởng đến chất lượng của rượu vang như loại nho, khí hậu và kỹ thuật chế biến rượu.

IV.2.2. Stepwise Regression

IV.2.2.1 Lựa chọn mô hình

a. Forward Stepwise Selection

```
#define intercept-only model
intercept_only <- lm(quality ~ 1, data=datatrain_scaled)

#define model with all predictors
all <- lm(quality ~ ., data=datatrain_scaled)

#perform forward stepwise regression
forward <- stepAIC(intercept_only, direction="forward", scope=formula(all), trace=0, k=20)

#view results of forward stepwise regression
forward$anova
```

```

##      Step Df Deviance Resid. Df Resid. Dev     AIC
## 1        NA    NA  3989  3061.856 -1036.433
## 2 + alcohol -1 664.40143   3988  2397.454 -1992.441
## 3 + volatile_acidity -1 166.90171   3987  2230.553 -2260.352
## 4 + sulphates -1 35.72579   3986  2194.827 -2304.776
## 5 + residual_sugar -1 16.67200   3985  2178.155 -2315.200
## 6 + color -1 11.63292   3984  2166.522 -2316.566

```

Các hệ số (coefficients):

```
forward$coefficients
```

```

## (Intercept)    alcohol volatile_acidity    sulphates
## 6.14039628  0.42585497 -0.26003133  0.07999948
## residual_sugar    color
## 0.08712331 -0.19199614

```

b. Backward Stepwise Selection

```

#perform backward stepwise regression
backward <- step(all, direction="backward", scope=formula(all), trace=0, k=20)

#view results of backward stepwise regression
backward$anova

```

```

##      Step Df Deviance Resid. Df Resid. Dev     AIC
## 1        NA    NA  3977  2107.359 -2287.039
## 2 - citric_acid  1 0.868732   3978  2108.228 -2305.394
## 3 - chlorides  1 1.032977   3979  2109.261 -2323.440
## 4 - total_sulfur_dioxide 1 8.945979   3980  2118.207 -2326.553
## 5 - free_sulfur_dioxide 1 8.584578   3981  2126.792 -2330.415

```

Các hệ số (coefficients):

```
backward$coefficients
```

```

## (Intercept) fixed_acidity volatile_acidity residual_sugar
## 6.6314268  0.1726331 -0.2496573  0.3423449
## density      p_h      sulphates    alcohol
## -0.4505972  0.1361123  0.1134914  0.2096685
## color
## -0.4739377

```

c. Both-Direction Stepwise Selection

```

#perform backward stepwise regression
both <- step(intercept_only, direction="both", scope=formula(all), trace=0, k=20)

#view results of backward stepwise regression
both$anova

```

```

##      Step Df Deviance Resid. Df Resid. Dev     AIC
## 1        NA    NA  3989  3061.856 -1036.433
## 2 + alcohol -1 664.40143   3988  2397.454 -1992.441
## 3 + volatile_acidity -1 166.90171   3987  2230.553 -2260.352
## 4 + sulphates -1 35.72579   3986  2194.827 -2304.776
## 5 + residual_sugar -1 16.67200   3985  2178.155 -2315.200
## 6 + color -1 11.63292   3984  2166.522 -2316.566

```

Các hệ số (coefficients):

```
both$coefficients
```

```

## (Intercept)    alcohol volatile_acidity    sulphates
## 6.14039628  0.42585497 -0.26003133  0.07999948
## residual_sugar    color
## 0.08712331 -0.19199614

```

IV.2.2.2 Thực hiện thống kê suy luận

```
summary(both)
```

```

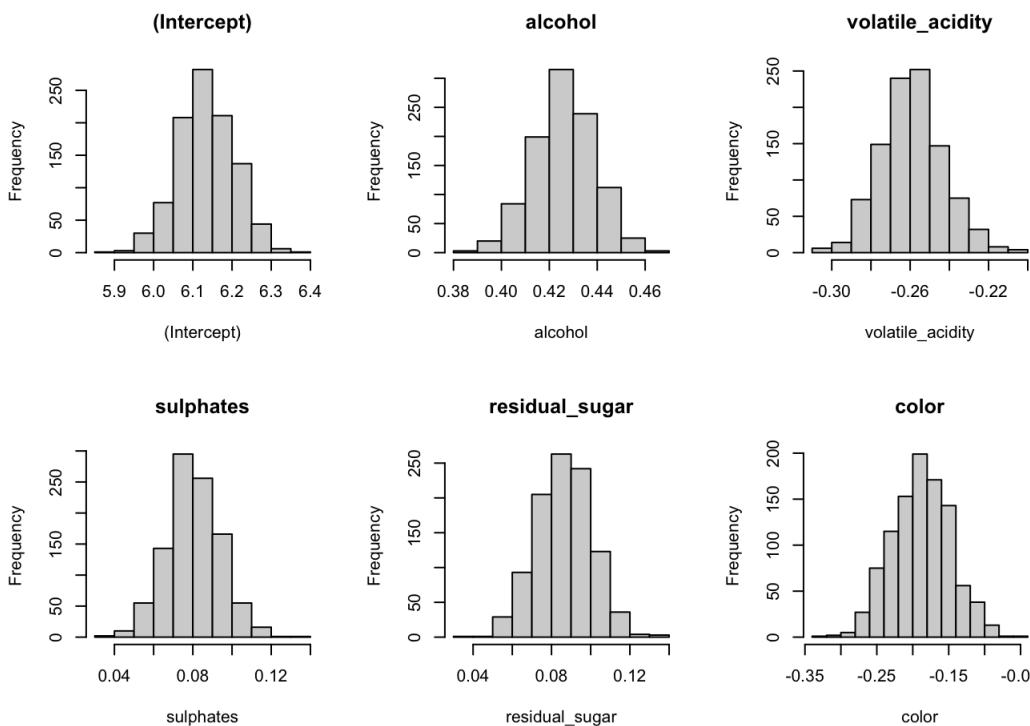
## 
## Call:
## lm(formula = quality ~ alcohol + volatile_acidity + sulphates +
##     residual_sugar + color, data = datatrain_scaled)
##
## Residuals:
##   Min     1Q Median     3Q    Max 
## -3.2398 -0.4517 -0.0245  0.4688  3.1146 
## 
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)    
## (Intercept) 6.14040   0.07323 83.847 < 2e-16 ***
## alcohol      0.42585   0.01249 34.107 < 2e-16 ***
## volatile_acidity -0.26003  0.01558 -16.688 < 2e-16 ***
## sulphates     0.08000   0.01362  5.874 4.59e-09 ***
## residual_sugar 0.08712  0.01323  6.586 5.11e-11 *** 
## color        -0.19200  0.04151 -4.625 3.86e-06 *** 
## --- 
## Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1 
## 
## Residual standard error: 0.7374 on 3984 degrees of freedom 
## Multiple R-squared: 0.2924, Adjusted R-squared: 0.2915 
## F-statistic: 329.3 on 5 and 3984 DF, p-value: < 2.2e-16

```

```
both_model = lm(formula = quality ~ alcohol + volatile_acidity + sulphates + residual_sugar +
color, data = datatrain_scaled)
```

```
boot_stepwise_model = boot(data = datatrain_scaled, statistic = boot_func, R = 1000,
formula = quality ~ alcohol + volatile_acidity + sulphates
+ residual_sugar + color)
```

```
results_df_stepwise = create_results_dataframe(model = both_model, boot_model = boot_stepwise_model)
```



```
results_df_stepwise
```

	Est	SE	CI_95	p_value
## (Intercept)	6.14039628	0.07080794	(5.99, 6.27)	0
## alcohol	0.42585497	0.01266124	(0.4, 0.45)	0
## volatile_acidity	-0.26003133	0.01596568	(-0.29, -0.23)	0
## sulphates	0.07999948	0.01369507	(0.05, 0.11)	0
## residual_sugar	0.08712331	0.01390161	(0.06, 0.11)	0
## color	-0.19199614	0.04075517	(-0.26, -0.11)	0

- Mô hình Stepwise Regression có dạng: $\text{quality} = 6.14 + 0.43\text{alcohol} - 0.26\text{volatile_acidity} + 0.08\text{sulphates} + 0.09\text{residual_sugar} - 0.19\text{color}$

Kết quả cho thấy hồi quy từng bước giải thích được khoảng 28,10% sự thay đổi về chất lượng và khi các biến bị loại trừ được thêm lại vào thì khả năng giải thích chỉ tăng từ 0,66% lên 28,76%. Câu hỏi đặt ra là liệu mức tăng 0,66% này có ý nghĩa thống kê hay không.

Để trả lời câu hỏi này, ta thực hiện kiểm tra giả thuyết cho từng biến bị loại trừ và thu được các p_value sau: + fixed_acidity: p = 0.71 + free_sulfur_dioxide: p = 0.09 + citric_acid: p = 0.49 + chlorides: p = 0.12 + p_h: p = 0.45 + ...

Vì tất cả các giá trị p này đều lớn hơn mức ý nghĩa alpha là 0,05, nên ta kết luận rằng các biến bị loại trừ không cung cấp thông tin bổ sung hoặc cải thiện dự đoán về chất lượng ngoài những gì đã được giải thích bởi các biến trong mô hình tuyến tính linear_model1, cụ thể là volatile_acidity, citric_acid, chlorides, total_sulfur_dioxide, p_h, sulphates, and alcohol.

IV.2.2.3. Nhận xét

Dựa trên 3 mô hình hồi quy từng bước, ta xác định được các biến quan trọng nhất ảnh hưởng đến chất lượng rượu vang, đó là: alcohol, residual_sugar, sulphates, color, volatile_acidity.

Cụ thể hơn, ta nhận thấy rằng:

- alcohol, residual_sugar và sulphates có tác động tích cực đến chất lượng rượu, nghĩa là tăng nồng độ của chúng có thể cải thiện chất lượng rượu.
- color và volatile_acidity có tác động tiêu cực đến chất lượng rượu, nghĩa là tăng nồng độ của chúng có thể giảm chất lượng rượu.

Điều này cho thấy rằng các nhà sản xuất rượu nên tập trung vào:

- Tối ưu hóa mức độ của:
 - sulphates : để cải thiện độ ổn định và chất lượng tổng thể của rượu
 - alcohol : để nâng cao hương vị và chất lượng tổng thể của rượu
 - residual_sugar : để đóng góp vào độ ngọt và thể tích của rượu, từ đó nâng cao hương vị và chất lượng tổng thể của rượu
- Giảm thiểu mức độ của:
 - color : để tránh tác động tiêu cực đến chất lượng rượu
 - volatile_acidity : để ngăn chặn hương vị khó chịu, giống như giấm

Bằng cách hiểu mối quan hệ giữa các biến này và chất lượng rượu, các nhà sản xuất rượu có thể đưa ra quyết định sáng suốt về cách tối ưu hóa quá trình sản xuất rượu để sản xuất rượu chất lượng cao.

Đáng chú ý là những phát hiện này cũng có thể được sử dụng để xác định các khu vực cần cải thiện trong quá trình sản xuất rượu, chẳng hạn như điều chỉnh quá trình lên men để tối ưu hóa hàm lượng của alcohol, residual_sugar và sulphates, và thực hiện các kỹ thuật để giảm thiểu mức độ ảnh hưởng của color và volatile_acidity. Bằng cách làm vậy, các nhà sản xuất rượu có thể tinh chỉnh kỹ thành phần rượu của mình và sản xuất rượu vang chất lượng cao đáp ứng các tiêu chuẩn mong đợi.

IV.2.3. Hồi quy đa thức

IV.2.3.1. Xây dựng mô hình

Xét mô hình hồi quy tối ưu, được lựa chọn trước đó, linear_model5 có dạng như sau:

quality = $\beta_0 + \beta_1 \text{alcohol} + \beta_2 \text{sulphates} + \beta_3 \text{volatile_acidity} + \beta_4 \text{chlorides} + \beta_5 \text{total_sulfur_dioxide} + \beta_6 \text{citric_acid} + \beta_7 \text{p_h} + \epsilon$

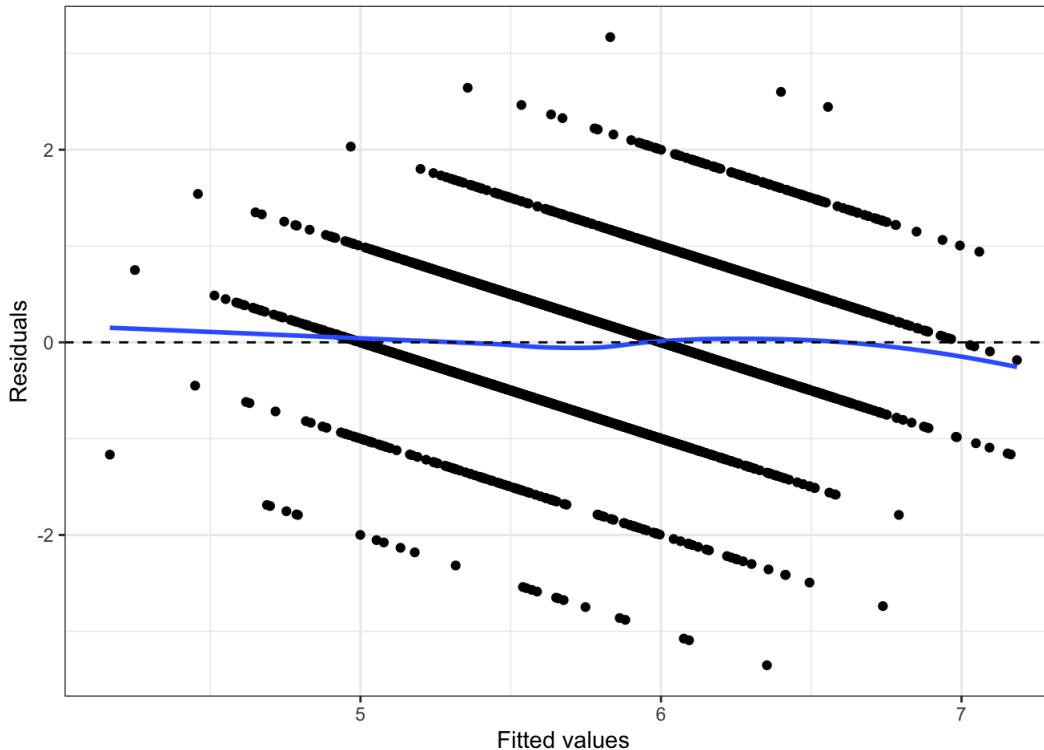
```
linear_model5 <- lm(quality ~ alcohol + sulphates + volatile_acidity + chlorides
+ total_sulfur_dioxide + citric_acid + p_h, data = datatrain_scaled)
summary(linear_model5)
```

```
##
## Call:
## lm(formula = quality ~ alcohol + sulphates + volatile_acidity +
##     chlorides + total_sulfur_dioxide + citric_acid + p_h, data = datatrain_scaled)
##
## Residuals:
##   Min     1Q Median     3Q    Max 
## -3.3525 -0.4454 -0.0417  0.4868  3.1689 
## 
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)    
## (Intercept) 5.80602   0.01173 495.020 < 2e-16 ***
## alcohol      0.37743   0.01307 28.872 < 2e-16 ***
## sulphates    0.09557   0.01345  7.108 1.39e-12 ***
## volatile_acidity -0.22908   0.01470 -15.581 < 2e-16 ***
## chlorides    -0.03273   0.01466 -2.232  0.02567 *  
## total_sulfur_dioxide -0.02688   0.01421 -1.892  0.05861 .  
## citric_acid     0.01764   0.01376  1.282  0.19988    
## p_h            0.03705   0.01296  2.859  0.00427 ** 
## --- 
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## 
## Residual standard error: 0.7409 on 3982 degrees of freedom
## Multiple R-squared:  0.2862, Adjusted R-squared:  0.2849 
## F-statistic:  228 on 7 and 3982 DF, p-value: < 2.2e-16
```

a. Kiểm tra tính tuyến tính của mô hình với biểu đồ Residuals vs Fitted.

```
ggplot(data = linear_model5, mapping = aes(x = .fitted, y = .resid)) +
  geom_point() +
  geom_smooth(method = "loess", se = FALSE) +
  geom_hline(yintercept = 0, linetype = "dashed") +
  labs(x = "Fitted values", y = "Residuals") +
  theme_bw()
```

```
## `geom_smooth()` using formula = 'y ~ x'
```



Hình vẽ không cho thấy xu hướng đường cong nào đáng kể. => Giả định về tính tuyến tính của mô hình là phù hợp.

b. Kiểm tra tính tuyến tính từng phần của các biến có trong mô hình bằng biểu đồ thăng dư từng phần.

```
# Hàm vẽ biểu đồ thăng dư từng phần
plotPartialResidual <- function(dataset, model, feature) {
  # kết quả cho từng thành phần tuyến tính
  terms_md <- predict(model, type = "terms")
  head(terms_md)

  # các giá trị thăng dư từng phần
  partial_resid_md <- residuals(model, type = "partial")
  head(partial_resid_md)
  data_part_resid_md <- tibble(
    features = dataset[[feature]],
    terms = terms_md[, feature],
    partial_resid = partial_resid_md[, feature]
  )

  ggplot(data_part_resid_md, mapping = aes(features, partial_resid)) +
    geom_point(colour = 'cyan4', alpha = 0.4) +
    geom_smooth(method = "loess", se = FALSE, linetype = "dashed", color = "red") +
    geom_line(aes(x = features, y = terms), color = "blue") +
    labs(x = sprintf("%s", feature), y = "Partial Residuals") +
    theme_bw()
}
```

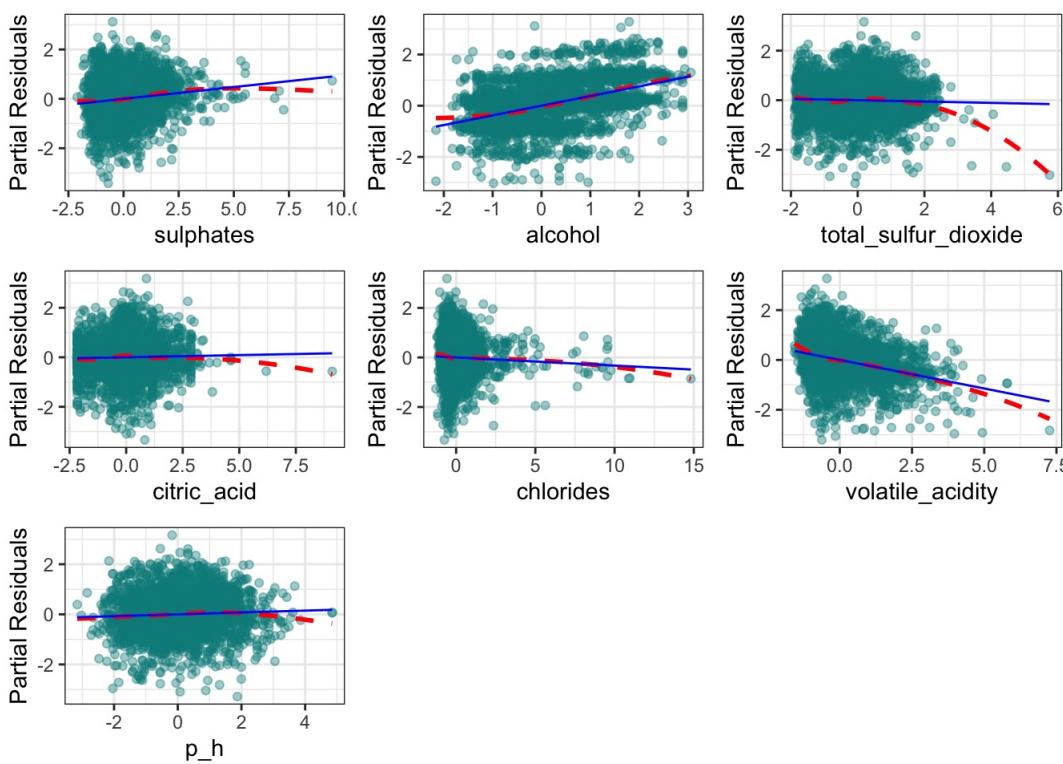
```
# Create a list of plots
plots <- list()

# Create a list of data frames
feature_list <- list('sulphates', 'alcohol', 'total_sulfur_dioxide', 'citric_acid', 'chlorides', 'volatile_acidity', 'pH')

# Create a list of plot titles
titles <- c('Sulphates', 'Alcohol', 'Total Sulfur Dioxide', 'Citric Acid', 'Chlorides', 'Volatile Acidity', 'pH')

# Create individual plots and add to the list
for(i in seq_along(feature_list)) {
  plots[[i]] <- plotPartialResidual(datatrain_scaled, linear_model5, feature_list[[i]])
}

# Arrange the plots in a grid
grid.arrange(grobs = plots, nrow = 3)
```



Thông qua biểu đồ thăng du từng phần, nhận thấy có một số feature mà mối quan hệ giữa nó và biến mục tiêu quality không hẳn là tuyến tính. Do đó, để cải thiện mô hình, ta có thể sử dụng mô hình hồi quy mở rộng như sau:

```
poly_model <- lm(quality ~ alcohol + poly(sulphates,2) + volatile_acidity +
    poly(chlorides,2) + poly(total_sulfur_dioxide,2) + citric_acid +
    p_h, data = datatrain_scaled)

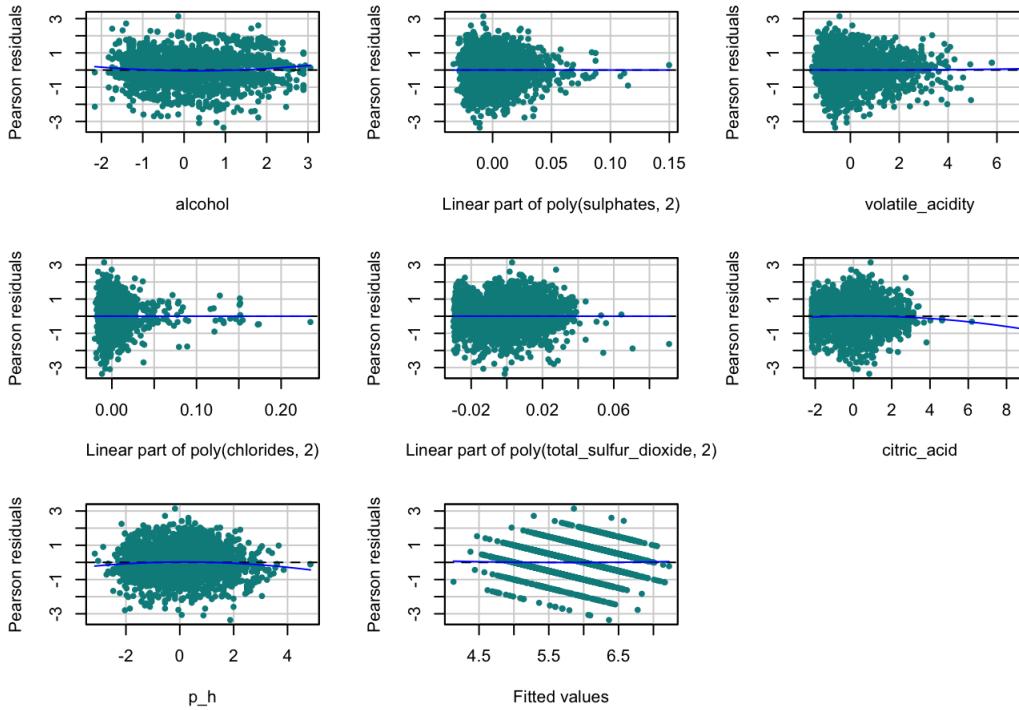
summary(poly_model)
```

```

## 
## Call:
## lm(formula = quality ~ alcohol + poly(sulphates, 2) + volatile_acidity +
##     poly(chlorides, 2) + poly(total_sulfur_dioxide, 2) + citric_acid +
##     p_h, data = datatrain_scaled)
## 
## Residuals:
##   Min     1Q Median     3Q    Max 
## -3.3629 -0.4426 -0.0416  0.4888  3.1403 
## 
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)    
## (Intercept) 5.80602   0.01170 496.062 < 2e-16 ***
## alcohol      0.37463   0.01370 27.338 < 2e-16 ***
## poly(sulphates, 2)1 6.56335   0.85718  7.657 2.38e-14 ***
## poly(sulphates, 2)2 -1.25317   0.80254 -1.561  0.11849  
## volatile_acidity -0.21609   0.01545 -13.989 < 2e-16 *** 
## poly(chlorides, 2)1 -1.52983   0.96445 -1.586  0.11277  
## poly(chlorides, 2)2 -0.11566   0.85671 -0.135  0.89261  
## poly(total_sulfur_dioxide, 2)1 -1.07244   0.92520 -1.159  0.24646  
## poly(total_sulfur_dioxide, 2)2 -3.48641   0.81185 -4.294 1.79e-05 *** 
## citric_acid       0.01965   0.01379  1.425  0.15420  
## p_h                 0.03602   0.01318  2.732  0.00631 ** 
## ---                
## Signif. codes:  0 '****' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## 
## Residual standard error: 0.7393 on 3979 degrees of freedom
## Multiple R-squared:  0.2897, Adjusted R-squared:  0.2879 
## F-statistic: 162.3 on 10 and 3979 DF,  p-value: < 2.2e-16

```

```
residualPlots(poly_model, pch=19, col="cyan4", cex=0.6)
```

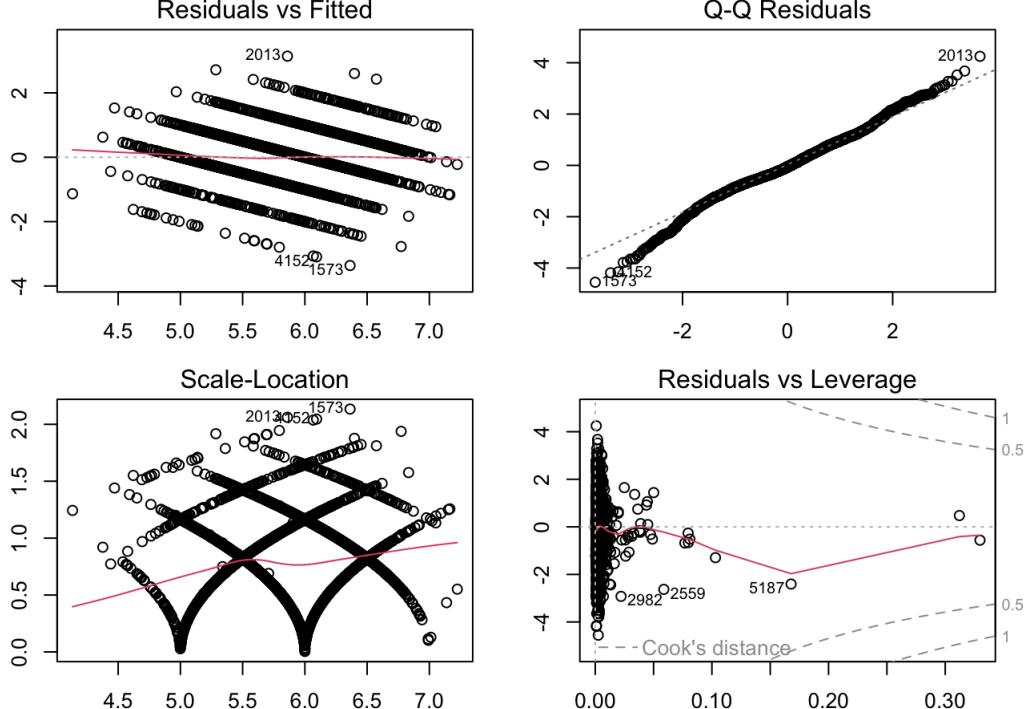


```

##             Test stat Pr(>|Test stat|)    
## alcohol          3.9418    8.227e-05 ***
## poly(sulphates, 2) 0.2870    0.774127    
## volatile_acidity  0.2870    0.774127    
## poly(chlorides, 2) 0.2870    0.774127    
## poly(total_sulfur_dioxide, 2) 0.2870    0.774127    
## citric_acid       -2.0656   0.038932 *   
## p_h               -2.7249   0.006461 **  
## Tukey test        0.7051    0.480737    
## ---                
## Signif. codes:  0 '****' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

```

```
par(mfrow=c(2,2), oma = c(1,1,0,0) + 0.1, mar = c(3,3,1,1) + 0.1)
plot(poly_model)
```



IV.2.3.2. Đánh giá mô hình

```
preds_poly <- predict(poly_model, datatest_scaled)

# round the numeric values to the nearest integer values
preds_poly = round(preds_poly)
head(preds_poly)
```

```
## 5189 2218 179 153 2511 2314
## 6 6 6 5 6 6
```

```
tst_tab <- table(predicted = preds_poly, actual = datatest_scaled$quality)
tst_tab
```

```
##      actual
## predicted 3 4 5 6 7 8 9
##       4 0 1 1 0 0 0 0
##       5 2 33 254 125 6 2 0
##       6 3 29 198 393 165 21 0
##       7 0 1 6 32 46 10 2
```

```
RMSE_value <- RMSE(preds_poly, datatest_scaled$quality)
MAE_value <- MAE(preds_poly, datatest_scaled$quality)
R_squared <- R2(preds_poly, datatest_scaled$quality)
accuracy <- sum(diag(tst_tab))/length(datatest_scaled$quality)

# Hiển thị kết quả
cat("RMSE:", RMSE_value, "\n")
```

```
## RMSE: 0.8143451
```

```
cat("MAE:", MAE_value, "\n")
```

```
## MAE: 0.5368421
```

```
cat("R-squared:", R_squared, "\n")
```

```
## R-squared: 0.2028565
```

```
cat("Accuracy: ", accuracy, "\n")
```

```
## Accuracy: 0.1977444
```

IV.2.4. Ridge Regression

IV.2.4.1. Xây dựng mô hình

```
X_train <- as.matrix(datatrain_scaled[, -which(names(datatrain_scaled)
  %in% c("quality"))])
y_train <- datatrain$quality

X_test <- as.matrix(datatest_scaled[, -which(names(datatest_scaled) == "quality")])
```

```
#### block này lỗi --> check lại
ridge_model <- glmnet(X_train, y_train, alpha = 0)

# Chọn hyperparameter lambda tối ưu bằng cross-validation
cv_ridge_model <- cv.glmnet(X_train, y_train, alpha = 0)
best_lambda <- cv_ridge_model$lambda.min

# Huấn luyện mô hình với lambda tối ưu
ridge_model <- glmnet(X_train, y_train, alpha = 0, lambda = best_lambda)

# Dự đoán trên tập kiểm tra
preds_ridge <- predict(ridge_model, newx = X_test, s = best_lambda)
range(preds_ridge)
```

```
## [1] 4.246041 7.474774
```

Giá trị hồi quy dao động trong khoảng từ 4 đến 8.

IV.2.4.2. Đánh giá mô hình

```
# round the numeric values to the nearest integer values
preds_ridge = round(preds_ridge)
head(preds_ridge)
```

```
##      s1
## 5189 6
## 2218 6
## 179  6
## 153  5
## 2511 6
## 2314 6
```

```
tst_tab <- table(predicted = preds_ridge, actual = datatest_scaled$quality)
tst_tab
```

```
##      actual
## predicted 3 4 5 6 7 8 9
##       4 0 1 1 0 0 0 0
##       5 1 35 238 108 5 2 0
##       6 4 28 218 415 169 21 0
##       7 0 0 2 27 43 10 2
```

```
RMSE_value <- RMSE(preds_ridge, datatest_scaled$quality)
MAE_value <- MAE(preds_ridge, datatest_scaled$quality)
R_squared <- R2(preds_ridge, datatest_scaled$quality)
accuracy <- sum(diag(tst_tab))/length(datatest_scaled$quality)
```

```
# Hiển thị kết quả
cat("RMSE:", RMSE_value, "\n")
```

```
## RMSE: 0.8031891
```

```
cat("MAE:", MAE_value, "\n")
```

```
## MAE: 0.5293233
```

```
cat("R-squared:", R_squared, "\n")
```

```
## R-squared: 0.2119854
```

```
cat("Accuracy: ", accuracy, "\n")
```

```
## Accuracy: 0.2105263
```

IV.3. Classification models

IV.3.1. Naive Bayes

IV.3.1.1. Chuẩn bị dữ liệu

Tạo một feature mới tên là ‘segmentation’ nhận ba giá trị tương ứng với 3 phân khúc, được chia dựa vào điểm đánh giá chất lượng (quality): - poor (phân khúc tầm thấp): có điểm đánh giá từ 3 - 5. - medium (phân khúc tầm trung): có điểm đánh giá bằng 6. - excellent (phân khúc cao cấp): có điểm đánh giá từ 7 trở lên.

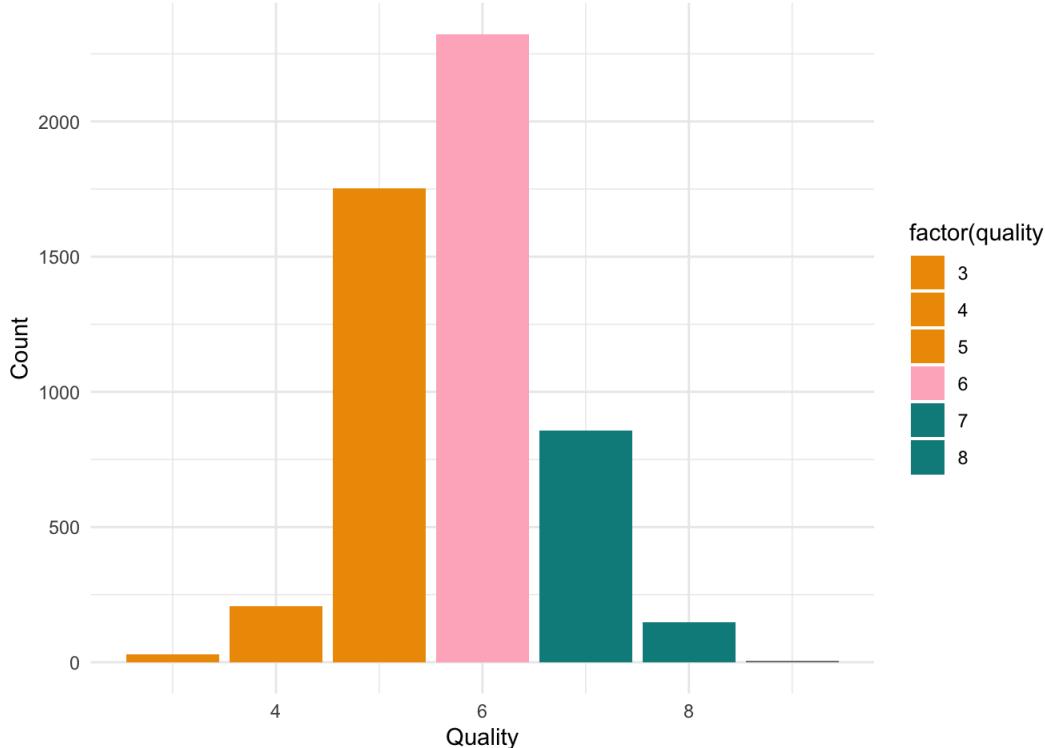
```
# scale on data
data_scaled = scale_data(datatrain, datatest)
datatrain_scaled = data_scaled$train_scaled
datatest_scaled = data_scaled$test_scaled
```

```
#wine quality is turned into nominal data
datatrain_scaled$segmentation <- ifelse(datatrain_scaled$quality < 6, 'poor', 'excellent')
datatrain_scaled$segmentation[datatrain_scaled$quality==6] <- 'medium'
datatrain_scaled$segmentation <- as.factor(datatrain_scaled$segmentation)

datatest_scaled$segmentation <- ifelse(datatest_scaled$quality < 6, 'poor', 'excellent')
datatest_scaled$segmentation[datatest_scaled$quality==6] <- 'medium'
datatest_scaled$segmentation <- as.factor(datatest_scaled$segmentation)
```

```
data$segmentation <- ifelse(data$quality < 6, 'poor', 'excellent')
data$segmentation[data$quality==6] <- 'medium'
data$segmentation <- as.factor(data$segmentation)

ggplot(data=data) +
  geom_bar(mapping = aes(x=quality, fill = factor(quality)), stat = "count") +
  scale_fill_manual(values = c("3" = "orange2", "4" = "orange2", "5" = "orange2", "6" = "pink1", "7" = "cyan4", "8" = "cyan4")) +
  labs(x = "Quality", y = "Count") +
  theme_minimal()
```



```
datatrain_scaled <- subset(datatrain_scaled, select = -c(quality))
datatest_scaled <- subset(datatest_scaled, select = -c(quality))
```

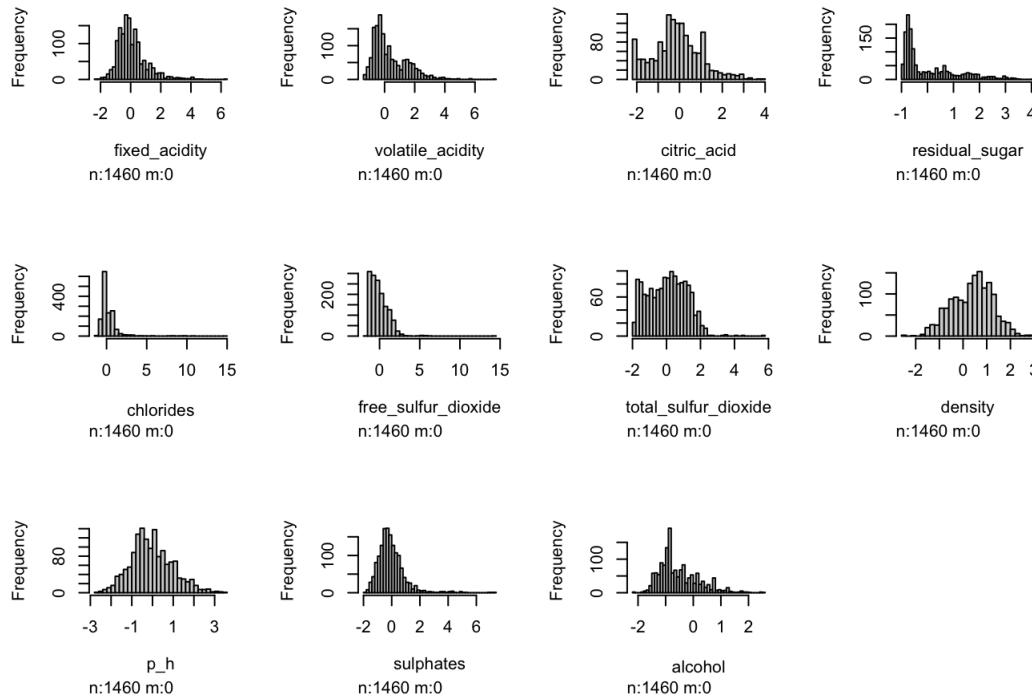
IV.3.1.2. Kiểm tra tính phân phối chuẩn của từng biến giải thích trong từng nhóm phân khúc chất lượng.

```
#split dataframe into groups
seg <- split(datatrain_scaled, f = datatrain_scaled$segmentation)

poor = seg$poor
medium = seg$medium
excellent = seg$excellent
```

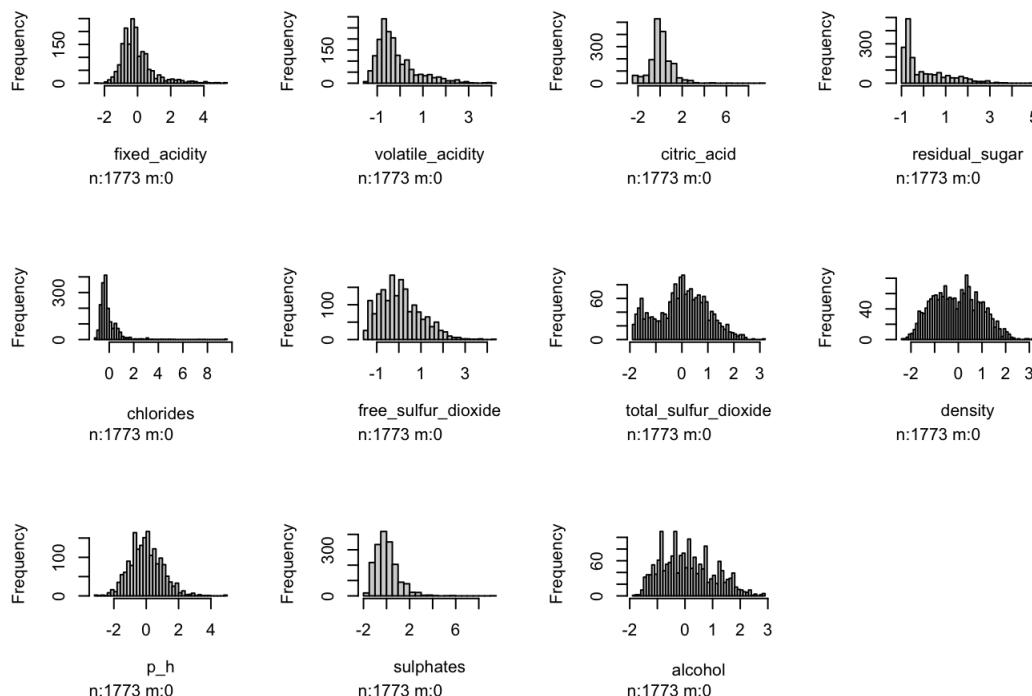
a. Nhóm phân khúc thấp (poor):

```
poor <- subset(poor, select = -segmentation)
hist.data.frame(poor)
```



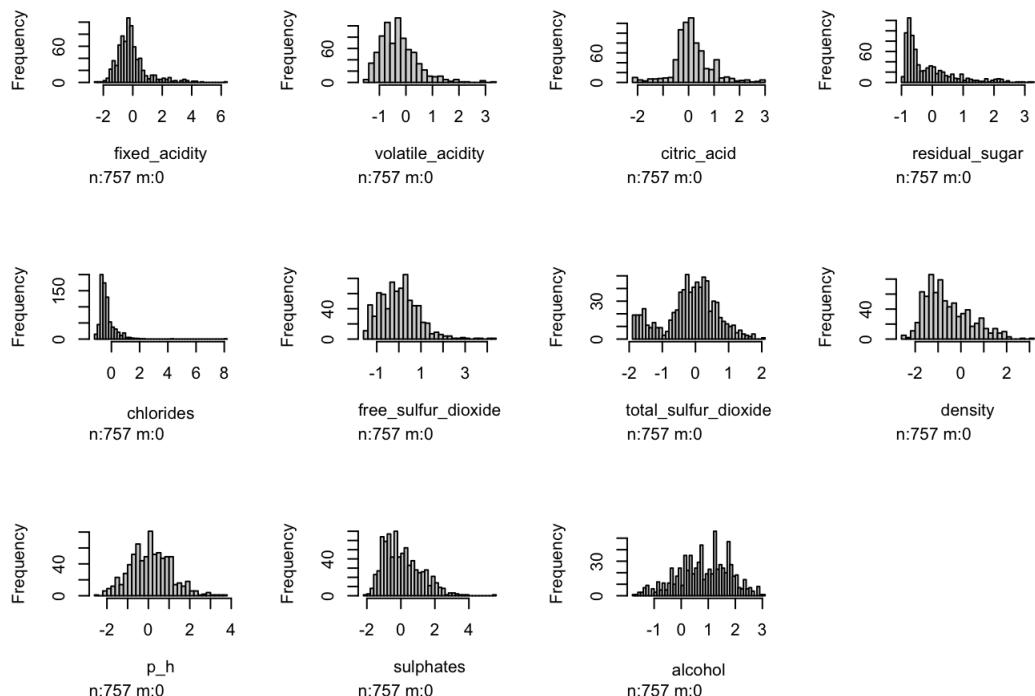
b. Nhóm phân khúc tâm trung (medium):

```
medium <- subset(medium, select = -segmentation)
hist.data.frame(medium)
```



c. Nhóm phân khúc cao cấp (excellent):

```
excellent <- subset(excellent, select = -segmentation)
hist.data.frame(excellent)
```



IV.3.1.3. Huấn luyện và đánh giá mô hình.

Naive Bayes sử dụng ước lượng mật độ của phân phối chuẩn được xây dựng sử dụng các biến có phân phối chuẩn và sự độc lập giữa các biến như sau:

```
nb_model <- NaiveBayes(formula = segmentation ~ density + volatile_acidity +
  p_h + fixed_acidity + sulphates,
  data = datatrain_scaled)

preds_nb <- predict(nb_model, newdata = datatest_scaled)
```

```
## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 13
```

```
## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 52
```

```
## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 81
```

```
## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 331
```

```
## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 397
```

```
## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 555
```

```
## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 714
```

```
## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 1026
```

```
## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 1178
```

```
## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 1197
```

```
## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 1309
```

```
nb_conf_matrix <- confusionMatrix(preds_nb$class, datatest_scaled$segmentation)
print(nb_conf_matrix)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction excellent medium poor
##   excellent      39    26     8
##   medium        194   417   308
##   poor          19    107   212
##
## Overall Statistics
##
##           Accuracy : 0.5023
##             95% CI : (0.475, 0.5295)
##   No Information Rate : 0.4135
##   P-Value [Acc > NIR] : 4.2e-11
##
##           Kappa : 0.1745
##
## McNemar's Test P-Value : < 2e-16
##
## Statistics by Class:
##
##           Class: excellent Class: medium Class: poor
## Sensitivity       0.15476    0.7582    0.4015
## Specificity       0.96846    0.3564    0.8429
## Pos Pred Value    0.53425    0.4538    0.6272
## Neg Pred Value    0.83055    0.6764    0.6815
## Prevalence        0.18947    0.4135    0.3970
## Detection Rate    0.02932    0.3135    0.1594
## Detection Prevalence 0.05489    0.6910    0.2541
## Balanced Accuracy  0.56161    0.5573    0.6222
```

Naive Bayes sử dụng ước lượng mật độ Kernel:

```
nb_model_k <- NaiveBayes(formula = segmentation ~.,
                           data = datatrain_scaled[, !(names(datatrain_scaled) %in% c('quality'))],
                           usekernel = TRUE)

preds_nb_k <- predict(nb_model_k, newdata = datatest_scaled)
```

```
## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 10
```

```
## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 13
```

```
## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 23
```

```
## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 26
```

```
## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 29
```

```
## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 52
```

```
## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 68
```

```
## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 71
```

```
## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with  
## observation 81
```

```
## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with  
## observation 87
```

```
## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with  
## observation 98
```

```
## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with  
## observation 113
```

```
## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with  
## observation 119
```

```
## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with  
## observation 121
```

```
## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with  
## observation 123
```

```
## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with  
## observation 124
```

```
## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with  
## observation 142
```

```
## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with  
## observation 152
```

```
## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with  
## observation 219
```

```
## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with  
## observation 258
```

```
## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with  
## observation 267
```

```
## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with  
## observation 288
```

```
## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with  
## observation 299
```

```
## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with  
## observation 306
```

```
## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with  
## observation 307
```

```
## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with  
## observation 331
```

```
## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with  
## observation 342
```

```
## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with  
## observation 365
```

```
## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with  
## observation 397
```

```
## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with  
## observation 402
```

```
## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with  
## observation 492
```

```
## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with  
## observation 498
```

```
## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with  
## observation 515
```

```
## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with  
## observation 547
```

```
## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with  
## observation 551
```

```
## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with  
## observation 555
```

```
## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with  
## observation 560
```

```
## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with  
## observation 585
```

```
## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with  
## observation 589
```

```
## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with  
## observation 611
```

```
## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with  
## observation 612
```

```
## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with  
## observation 629
```

```
## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with  
## observation 655
```

```
## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with  
## observation 659
```

```
## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with  
## observation 664
```

```
## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with  
## observation 683
```

```
## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with  
## observation 690
```

```
## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with  
## observation 698
```

```
## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with  
## observation 714
```

```
## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with  
## observation 742
```

```
## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with  
## observation 789
```

```
## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with  
## observation 794
```

```
## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with  
## observation 822
```

```
## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with  
## observation 825
```

```
## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with  
## observation 869
```

```
## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with  
## observation 875
```

```
## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with  
## observation 945
```

```
## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with  
## observation 1026
```

```
## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with  
## observation 1032
```

```
## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with  
## observation 1062
```

```
## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with  
## observation 1086
```

```
## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with  
## observation 1087
```

```
## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with  
## observation 1099
```

```
## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with  
## observation 1107
```

```
## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with  
## observation 1112
```

```
## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with  
## observation 1130
```

```
## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with  
## observation 1142
```

```
## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with  
## observation 1156
```

```
## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with  
## observation 1158
```

```
## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with  
## observation 1164
```

```
## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with  
## observation 1177
```

```
## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with  
## observation 1178
```

```
## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with  
## observation 1180
```

```
## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with  
## observation 1182
```

```
## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with  
## observation 1183
```

```
## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with  
## observation 1197
```

```
## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with  
## observation 1228
```

```
## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with  
## observation 1236
```

```
## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with  
## observation 1268
```

```
## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with  
## observation 1280
```

```
## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with  
## observation 1282
```

```
## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with  
## observation 1296
```

```
## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with  
## observation 1309
```

```
## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with  
## observation 1310
```

```
## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with  
## observation 1319
```

```
#Evaluate the model's performance  
nb_conf_matrix_k <- confusionMatrix(preds_nb_k$class, datatest_scaled$segmentation)  
print(nb_conf_matrix_k)
```

```

## Confusion Matrix and Statistics
##
##      Reference
## Prediction excellent medium poor
##   excellent     133    102    23
##   medium        98    279   177
##   poor         21    169   328
##
## Overall Statistics
##
##           Accuracy : 0.5564
##             95% CI : (0.5292, 0.5833)
##   No Information Rate : 0.4135
##   P-Value [Acc > NIR] : <2e-16
##
##           Kappa : 0.3029
##
## McNemar's Test P-Value : 0.9492
##
## Statistics by Class:
##
##           Class: excellent Class: medium Class: poor
## Sensitivity          0.5278      0.5073      0.6212
## Specificity          0.8840      0.6474      0.7631
## Pos Pred Value       0.5155      0.5036      0.6332
## Neg Pred Value       0.8890      0.6508      0.7537
## Prevalence           0.1895      0.4135      0.3970
## Detection Rate       0.1000      0.2098      0.2466
## Detection Prevalence 0.1940      0.4165      0.3895
## Balanced Accuracy    0.7059      0.5774      0.6922

```

IV.3.2. LDA

```

lda_model <- lda(segmentation ~ ., data = datatrain_scaled[, !(names(datatrain_scaled)
                                                       %in% c('quality'))])
preds_lda <- predict(lda_model, newdata = datatest_scaled[, !(names(datatrain_scaled)
                                                       %in% c('quality'))])
summary(preds_lda$class)

```

```

## excellent   medium   poor
##      154      691     485

```

```

lda_conf_matrix <- confusionMatrix(preds_lda$class, datatest_scaled$segmentation)
print(lda_conf_matrix)

```

```

## Confusion Matrix and Statistics
##
##      Reference
## Prediction excellent medium poor
##   excellent     90    54   10
##   medium       153   351  187
##   poor        9    145  331
##
## Overall Statistics
##
##      Accuracy : 0.5805
##      95% CI : (0.5534, 0.6071)
##      No Information Rate : 0.4135
##      P-Value [Acc > NIR] : < 2.2e-16
##
##      Kappa : 0.3216
##
## McNemar's Test P-Value : 2.11e-11
##
## Statistics by Class:
##
##      Class: excellent Class: medium Class: poor
## Sensitivity          0.35714    0.6382    0.6269
## Specificity          0.94063    0.5641    0.8080
## Pos Pred Value       0.58442    0.5080    0.6825
## Neg Pred Value       0.86224    0.6886    0.7669
## Prevalence           0.18947    0.4135    0.3970
## Detection Rate       0.06767    0.2639    0.2489
## Detection Prevalence 0.11579    0.5195    0.3647
## Balanced Accuracy    0.64889    0.6011    0.7174

```

IV.3.3. Multinomial Logistic`

IV.3.3.1 Lựa chọn mô hình

```
head(datatrain_scaled)
```

```

## fixed_acidity volatile_acidity citric_acid residual_sugar chlorides
## 1 0.1449982 2.0835759 -2.161256 -0.7016920 0.5171176
## 2 0.4479001 3.1427289 -2.161256 -0.5414767 1.1051883
## 3 0.4479001 2.4366269 -1.889687 -0.6101404 0.9448054
## 4 3.0225667 -0.3877809 1.640714 -0.7016920 0.4903871
## 5 0.1449982 1.8482086 -2.161256 -0.7245799 0.4903871
## 6 0.5236256 1.4951576 -1.753902 -0.7703557 0.3300042
## free_sulfur_dioxide total_sulfur_dioxide density p_h sulphates
## 1 -1.0476099 -1.3974329 1.1432778 1.7470191 0.1735907
## 2 -0.2744170 -0.8165744 0.7978228 -0.1767285 0.9754743
## 3 -0.8266976 -1.0453975 0.8669138 0.1956098 0.7750034
## 4 -0.7162415 -0.9397868 1.2123688 -0.4249539 0.3072380
## 5 -0.9371538 -1.2918223 1.1432778 1.7470191 0.1735907
## 6 -0.8266976 -0.9573886 0.6596408 0.4438353 -0.4946456
## alcohol color segmentation
## 1 -0.9826600 1 poor
## 2 -0.6458775 1 poor
## 3 -0.6458775 1 poor
## 4 -0.6458775 1 medium
## 5 -0.9826600 1 poor
## 6 -0.9826600 1 poor

```

```
md_mlogit1 = nnet::multinom(segmentation~, data = datatrain_scaled, maxit = 500)
```

```

## # weights: 42 (26 variable)
## initial value 4383.463032
## iter 10 value 3519.710380
## iter 20 value 3465.543810
## iter 30 value 3349.835695
## final value 3349.814554
## converged

```

Thuật toán hội tụ sau 30 lần lặp. Tổng hợp kết quả ước lượng mô hình như sau:

```
md_mlogit1
```

```

## Call:
## nnet::multinom(formula = segmentation ~ ., data = datatrain_scaled,
##   maxit = 500)
##
## Coefficients:
##   (Intercept) fixed_acidity volatile_acidity citric_acid residual_sugar
## medium 0.9171971 -0.6543810 0.2764179 -0.06031218 -0.7769028
## poor -0.5473110 -0.7794531 0.9585154 -0.03831687 -1.2163011
##   chlorides free_sulfur_dioxide total_sulfur_dioxide density p_h
## medium 0.2619832 -0.2075470 0.3214175 1.160668 -0.4873156
## poor 0.3012093 -0.5219812 0.6422195 1.540384 -0.6530659
##   sulphates alcohol color
## medium -0.3030049 -0.3151711 0.3185663
## poor -0.5864910 -1.1156337 0.8730507
##
## Residual Deviance: 6699.629
## AIC: 6751.629

```

Calculate coefficient and standard error

```
summary(md_mlogit1)
```

```

## Call:
## nnet::multinom(formula = segmentation ~ ., data = datatrain_scaled,
##   maxit = 500)
##
## Coefficients:
##   (Intercept) fixed_acidity volatile_acidity citric_acid residual_sugar
## medium 0.9171971 -0.6543810 0.2764179 -0.06031218 -0.7769028
## poor -0.5473110 -0.7794531 0.9585154 -0.03831687 -1.2163011
##   chlorides free_sulfur_dioxide total_sulfur_dioxide density p_h
## medium 0.2619832 -0.2075470 0.3214175 1.160668 -0.4873156
## poor 0.3012093 -0.5219812 0.6422195 1.540384 -0.6530659
##   sulphates alcohol color
## medium -0.3030049 -0.3151711 0.3185663
## poor -0.5864910 -1.1156337 0.8730507
##
## Std. Errors:
##   (Intercept) fixed_acidity volatile_acidity citric_acid residual_sugar
## medium 0.5646334 0.1166140 0.08600457 0.06688286 0.1534922
## poor 0.6499983 0.1338992 0.09620991 0.07509900 0.1737158
##   chlorides free_sulfur_dioxide total_sulfur_dioxide density p_h
## medium 0.1149981 0.07080241 0.1076514 0.2585045 0.07702497
## poor 0.1191752 0.08497441 0.1205166 0.2923244 0.09078271
##   sulphates alcohol color
## medium 0.05672158 0.1291377 0.3232280
## poor 0.07048475 0.1496702 0.3712664
##
## Residual Deviance: 6699.629
## AIC: 6751.629

```

Giải thích dữ liệu huấn luyện mô hình:

Sau khi chạy mô hình, nnet báo cáo tóm tắt các vòng lặp và thông báo về sự hội tụ của mô hình. Trong trường hợp này, mô hình hội tụ khá nhanh sau 30 vòng lặp. Đầu ra thực thi mô hình hiển thị một số lịch sử vòng lặp và bao gồm log-likelihood âm cuối cùng là 4194. Giá trị này được nhân đôi như được hiển thị trong tóm tắt mô hình dưới dạng Residual Deviance.

Akaike Information Criterion (AIC) là 8441.024 và nó cung cấp một phương pháp để đánh giá chất lượng của mô hình của bạn thông qua việc so sánh các mô hình liên quan. Nếu chúng ta có nhiều hơn một mô hình ứng viên tương tự (nơi tất cả các biến của mô hình đơn giản hơn xuất hiện trong các mô hình phức tạp hơn), thì chúng ta nên chọn mô hình có AIC nhỏ nhất. Nó hữu ích để so sánh các mô hình. Tuy nhiên, không giống như R-squared, con số đó không có ý nghĩa.

IV.3.3.2. Thực hiện thống kê suy luận

a. Tính giá trị Z score và p-Value cho các biến

```

# Z score
mlogit_output = summary(md_mlogit1)
z <- mlogit_output$coefficients/mlogit_output$standard.errors
p <- (1-norm(abs(z),0,1))*2
print(z, digits=2)

```

```

## (Intercept) fixed_acidity volatile_acidity citric_acid residual_sugar
## medium      1.62      -5.6       3.2     -0.90      -5.1
## poor       -0.84      -5.8      10.0     -0.51      -7.0
## chlorides free_sulfur_dioxide total_sulfur_dioxide density p_h
## medium      2.3       -2.9       3.0      4.5 -6.3
## poor        2.5       -6.1       5.3      5.3 -7.2
## sulphates alcohol color
## medium     -5.3     -2.4  0.99
## poor      -8.3     -7.5  2.35

```

```
print(p, digits=2)
```

```

## (Intercept) fixed_acidity volatile_acidity citric_acid residual_sugar
## medium      0.1    2.0e-08    0.0013    0.37    4.2e-07
## poor       0.4    5.8e-09    0.0000    0.61    2.5e-12
## chlorides free_sulfur_dioxide total_sulfur_dioxide density p_h
## medium     0.023    3.4e-03    2.8e-03 7.1e-06 2.5e-10
## poor      0.011    8.1e-10    9.9e-08 1.4e-07 6.3e-13
## sulphates alcohol color
## medium    9.2e-08 1.5e-02 0.324
## poor     0.0e+00 9.1e-14 0.019

```

p-Value cho chất lượng cho thấy axit citric và residual sugar (cho ba phân khúc chất lượng) không có ý nghĩa. Bây giờ ta sẽ khám phá toàn bộ tập dữ liệu và phân tích xem liệu chúng ta có thể loại bỏ bất kỳ biến nào không góp phần vào hiệu suất của mô hình hay không.

```

Pquality5 <- rbind(mlogit_output$coefficients[2, ],mlogit_output$standard.errors[2, ],z[2, ],p[2, ])
rownames(Pquality5) <- c("Coefficient", "Std. Errors", "z stat", "p value")
Pquality5 = as.data.frame(Pquality5)
Pquality5

```

```

## (Intercept) fixed_acidity volatile_acidity citric_acid
## Coefficient -0.5473110 -7.794531e-01    0.95851544 -0.03831687
## Std. Errors  0.6499983 1.338992e-01    0.09620991  0.07509900
## z stat      -0.8420191 -5.821193e+00    9.96275162 -0.51021808
## p value     0.3997772 5.842900e-09    0.00000000  0.60989869
## residual_sugar chlorides free_sulfur_dioxide total_sulfur_dioxide
## Coefficient -1.216301e+00 0.30120928   -5.219812e-01    6.422195e-01
## Std. Errors  1.737158e-01 0.11917519   8.497441e-02    1.205166e-01
## z stat      -7.001674e+00 2.52744952   -6.142805e+00    5.328886e+00
## p value     2.529310e-12 0.01148943   8.107655e-10    9.881687e-08
## density     p_h sulphates alcohol color
## Coefficient 1.540384e+00 -6.530659e-01 -0.58649101 -1.115634e+00 0.87305070
## Std. Errors 2.923244e-01 9.078271e-02  0.07048475 1.496702e-01 0.37126638
## z stat      5.269435e+00 -7.193725e+00 -8.32082103 -7.453945e+00 2.35154801
## p value     1.368447e-07 6.303846e-13  0.00000000  9.059420e-14 0.01869548

```

b. Tính toán tỷ lệ chênh lệch (trích xuất các hệ số từ mô hình và lấy lũy thừa)

```

oddsML <- exp(coef(mlogit_output))
print(oddsML, digits = 2)

```

```

## (Intercept) fixed_acidity volatile_acidity citric_acid residual_sugar
## medium      2.50      0.52       1.3     0.94      0.46
## poor       0.58      0.46       2.6     0.96      0.30
## chlorides free_sulfur_dioxide total_sulfur_dioxide density p_h
## medium      1.3       0.81       1.4     3.2 0.61
## poor        1.4       0.59       1.9     4.7 0.52
## sulphates alcohol color
## medium     0.74     0.73  1.4
## poor       0.56     0.33  2.4

```

Tỷ lệ risk/odds cho việc tăng một đơn vị biến residual_sugar là 0.46 cho việc thuộc nhóm chất lượng kém (poor) so với bình thường (normal). Nghĩa là việc tăng một đơn vị residual_sugar làm giảm khả năng thuộc nhóm chất lượng kém đi 54%.

Tỷ lệ risk/odds cho việc tăng một đơn vị biến alcohol là 0.73 cho việc thuộc nhóm chất lượng kém (poor) so với bình thường (normal). Nghĩa là việc tăng một đơn vị alcohol làm giảm khả năng thuộc nhóm chất lượng kém đi 27%.

Kết luận: Các biến residual_sugar và alcohol có ảnh hưởng tiêu cực đến chất lượng rượu vang. Việc tăng residual_sugar có ảnh hưởng tiêu cực mạnh hơn việc tăng alcohol đến chất lượng rượu vang.

IV.3.3.3. Tính toán các biến quan trọng

Thực hiện tính toán trên các biến quan trọng sử dụng hàm Caret:

```

mostImportantVariables <- varImp(md_mlogit1)
mostImportantVariables$Variables <- row.names(mostImportantVariables)
mostImportantVariables <- mostImportantVariables[order(-mostImportantVariables$Overall),]
print(head(mostImportantVariables))

```

```

##          Overall      Variables
## density    2.701052    density
## residual_sugar 1.993204 residual_sugar
## fixed_acidity 1.433834 fixed_acidity
## alcohol     1.430805   alcohol
## volatile_acidity 1.234933 volatile_acidity
## color       1.191617   color

```

Tầm quan trọng của những đặc điểm này trong việc phân loại chất lượng rượu vang có thể được quy cho các thuộc tính hóa học và sinh học của chúng, ảnh hưởng đến tính chất tổng thể và chất lượng được cảm nhận của rượu vang. Dưới đây là một lời giải thích ngắn gọn cho mỗi đặc điểm:

Cấp độ 1: Các đặc điểm có tác động cao

- density (Mật độ): Mật độ, hay trọng lượng riêng, là thước đo khối lượng của một chất trên một đơn vị thể tích. Trong rượu vang, mật độ bị ảnh hưởng bởi các yếu tố như hàm lượng đường, hàm lượng cồn và độ chua. Rượu vang có mật độ cao hơn có thể cho thấy hàm lượng đường hoặc cồn cao hơn, điều này có thể góp phần tạo nên rượu vang đậm đà, đầy đặn hơn. Rượu vang có mật độ thấp hơn có thể cho thấy hàm lượng đường hoặc cồn thấp hơn, điều này có thể dẫn đến rượu vang nhẹ nhàng, thanh tao hơn. Mật độ là một yếu tố quan trọng trong phân loại rượu vang, vì nó có thể cung cấp thông tin chi tiết về độ ngọt, độ đậm đà và chất lượng tổng thể của rượu vang.
- residual_sugar (Đường dư): Đường dư là lượng đường còn lại trong rượu vang sau khi quá trình lên men hoàn tất. Đây là một yếu tố chính trong việc xác định độ ngọt của rượu vang, điều này có thể có tác động đáng kể đến chất lượng được cảm nhận của rượu vang. Rượu vang có hàm lượng đường dư cao hơn có thể ngọt hơn và đậm đà hơn, trong khi rượu vang có hàm lượng đường dư thấp hơn có thể khô hơn và thanh tao hơn. Đường dư là một yếu tố dự đoán quan trọng về chất lượng rượu vang, vì nó có thể cung cấp thông tin chi tiết về sự cân bằng, độ phức tạp và tính chất tổng thể của rượu vang.
- color (Màu sắc): Màu sắc của rượu vang là một khía cạnh quan trọng của tính chất tổng thể và chất lượng được cảm nhận của rượu vang. Màu sắc của rượu vang bị ảnh hưởng bởi giống nho, tiếp xúc với vỏ và quá trình lão hóa. Các màu sắc khác nhau có thể cho thấy các cấu trúc hương vị, mức độ tanin và tiềm năng lão hóa khác nhau. Trong phân loại rượu vang, màu sắc là một yếu tố chính trong việc phân biệt giữa các phong cách rượu vang và mức độ chất lượng khác nhau.

Cấp độ 2: Các đặc điểm có tác động trung bình

- volatile_acidity (Độ chua dễ bay hơi): Độ chua dễ bay hơi để cập đến lượng axit axetic và các hợp chất khác góp phần tạo nên hương thơm và hương vị của rượu vang. Nồng độ axit dễ bay hơi cao có thể khiến rượu vang có vị khó chịu, chua như giấm. Mặt khác, nồng độ vừa phải có thể thêm độ phức tạp và tính chất cho rượu vang. Trong phân loại rượu vang, độ chua dễ bay hơi là một yếu tố quan trọng trong việc xác định chất lượng và phong cách của rượu vang.
- alcohol (Cồn): Hàm lượng cồn là một yếu tố quan trọng trong việc xác định chất lượng rượu vang, vì nó có thể ảnh hưởng đến độ đậm đà, hương vị và hương thơm của rượu vang. Hàm lượng cồn cao hơn có thể góp phần tạo nên rượu vang đậm đà, mạnh mẽ hơn, trong khi hàm lượng cồn thấp hơn có thể dẫn đến rượu vang nhẹ nhàng, thanh tao hơn. Hàm lượng cồn cũng là một yếu tố dự đoán quan trọng về phong cách rượu vang, vì các vùng sản xuất rượu vang và giống nho khác nhau được biết đến là sản xuất rượu vang với mức độ cồn riêng biệt.

Cấp độ 3: Các đặc điểm có tác động thấp

- fixed_acidity (Độ chua cố định): Độ chua cố định để cập đến các axit không dễ bay hơi có trong rượu vang, chẳng hạn như axit tartaric và axit malic. Những axit này đóng vai trò quan trọng trong việc duy trì sự cân bằng, độ tươi mát và tiềm năng lão hóa của rượu vang. Trong phân loại rượu vang, độ chua cố định là một yếu tố quan trọng trong việc xác định chất lượng và phong cách của rượu vang.

Những đặc điểm này rất quan trọng bởi vì chúng ảnh hưởng đến các thuộc tính hóa học và sinh học của rượu vang, điều này lại ảnh hưởng đến hương vị, hương thơm và tính chất tổng thể của rượu vang. Hiểu rõ tầm quan trọng của những đặc điểm này trong việc dự đoán chất lượng rượu vang có thể giúp các nhà sản xuất rượu vang và chuyên gia rượu vang tối ưu hóa quy trình sản xuất của họ và đưa ra quyết định sáng suốt hơn về phân loại rượu vang. Bằng cách tập trung vào các yếu tố như mật độ, đường dư và màu sắc, các nhà sản xuất rượu vang có thể tạo ra rượu vang có nhiều khả năng được coi là có chất lượng cao và hấp dẫn người tiêu dùng. Ngoài ra, việc hiểu rõ tác động của cồn và độ chua đến chất lượng rượu vang có thể giúp các nhà sản xuất rượu vang tạo ra rượu vang cân bằng, ổn định và có khả năng lão hóa tốt hơn. Hãy nhớ rằng chất lượng rượu vang là một khái niệm phức tạp và đa chiều, và nhiều yếu tố khác có thể ảnh hưởng đến nó, bao gồm giống nho, thổ nhưỡng, kỹ thuật sản xuất rượu vang và nhiều hơn nữa. Tuy nhiên, sáu đặc điểm này cung cấp một nền tảng vững chắc để hiểu rõ các khía cạnh hóa học và sinh học của chất lượng rượu vang.

IV.3.3.4. Lean Multinomial Model

Sử dụng 4 biến quan trọng nhất để làm các biến chính xây dựng một mô hình tinh gọn với số lượng biến ít hơn:

```

md_mlogit2 = nnet::multinom(segmentation ~ density + residual_sugar + color + alcohol,
                             data = datatrain_scaled, maxit = 1000)

```

```
## # weights: 18 (10 variable)
## initial value 4383.463032
## iter 10 value 3694.689844
## final value 3586.318867
## converged
```

```
summary(md_mlogit2)
```

```
## Call:
## nnet::multinom(formula = segmentation ~ density + residual_sugar +
##   color + alcohol, data = datatrain_scaled, maxit = 1000)
##
## Coefficients:
## (Intercept) density residual_sugar color alcohol
## medium 1.728342 -0.07695876 0.03837985 -0.2903843 -0.8234473
## poor 1.779757 -0.06121696 -0.18162521 -0.5803544 -1.8295888
##
## Std. Errors:
## (Intercept) density residual_sugar color alcohol
## medium 0.3952343 0.1292338 0.08898406 0.2214409 0.08431454
## poor 0.4283464 0.1431318 0.09871566 0.2404099 0.09762952
##
## Residual Deviance: 7172.638
## AIC: 7192.638
```

IV.3.3.5. Đánh giá các Multinomial Logistics Models

```
mlogit_ModelFit<- rbind(pscl::pR2(md_mlogit1)[1:6],pscl::pR2(md_mlogit2)[1:6])
```

```
## fitting null model for pseudo-r2
## # weights: 6 (2 variable)
## initial value 4383.463032
## final value 4164.203301
## converged
## fitting null model for pseudo-r2
## # weights: 6 (2 variable)
## initial value 4383.463032
## final value 4164.203301
## converged
```

```
rownames(mlogit_ModelFit) <- c("Model-1", "Model-2")
print(mlogit_ModelFit, digits = 2)
```

```
##      llh llhNull G2 McFadden r2ML r2CU
## Model-1 -3350 -4164 1629  0.20 0.34 0.38
## Model-2 -3586 -4164 1156  0.14 0.25 0.29
```

IV.3.3.6. Tính toán sai số của Multinomial Model

```
mlogit1_output = summary(md_mlogit1)
mlogit2_output = summary(md_mlogit2)
mlogitModelError <- as.data.frame(rbind(cbind(mlogit1_output$deviance,mlogit1_output$AIC),cbind(mlogit2_output$deviance,mlogit2_output$AIC)))

names(mlogitModelError) <- c("Deviance", "AIC")
print(mlogitModelError, digits = 3)
```

```
##  Deviance AIC
## 1  6700 6752
```

Ta quan sát thấy mô hình md_mlogit1 với tất cả các biến có điểm McFadden và r2 scores cao nhất (0,20), đồng thời độ lệch chuẩn và AIC thấp nhất.

IV.3.3.7. Tiên đoán xác suất phân loại và dự đoán kết quả phân loại

Giả sử ta có nồng độ phần trăm các chất tan có trong rượu

```
new_data_wine = head(datatest_scaled, 5)
new_data_wine
```

```

## fixed_acidity volatile_acidity citric_acid residual_sugar chlorides
## 5189 -1.2180606 -0.5054646 1.23336024 1.4726586 -0.7392151
## 2218 0.8265276 -1.2115666 -0.39605558 -0.7932436 -0.2046054
## 179 1.2051550 0.1417955 1.09757559 -0.6559162 1.0784578
## 153 0.5993511 0.4360047 -1.00708652 -0.6788041 0.4369262
## 2511 -0.5365312 -0.9173574 0.01129837 -0.8161315 -0.4451798
## free_sulfur_dioxide total_sulfur_dioxide density p_h
## 5189 -0.27441702 -0.3589284 -0.07272391 0.009440656
## 2218 1.43765287 1.3308417 0.03782170 -0.114672087
## 179 0.49877583 0.5563638 1.03964132 -1.169630407
## 153 -1.32375018 -1.6790612 0.93600480 0.381778887
## 2511 0.05695134 0.3275407 -0.79127040 0.816173489
## sulphates alcohol color segmentation
## 5189 -0.29417473 -0.05650814 2 medium
## 2218 0.57453253 -0.89846437 2 medium
## 179 3.31430159 -1.06685561 1 poor
## 153 0.50770890 -1.15105124 1 medium
## 2511 0.03994345 0.19607873 2 excellent

```

Với actual segmentation là:

```
new_data_wine$segmentation
```

```

## [1] medium medium poor medium excellent
## Levels: excellent medium poor

```

Khi đó, ta tiên đoán xác suất phân loại như sau:

```

new_data_wine = head(datatest_scaled, 5)
out_prob_mlogit_wine = predict(md_mlogit1, new_data_wine, type = "prob")
out_prob_mlogit_wine

```

```

## excellent medium poor
## 5189 0.26117426 0.5579328 0.1808930
## 2218 0.10708836 0.5321990 0.3607126
## 179 0.04530773 0.4975444 0.4571479
## 153 0.02879643 0.3546908 0.6165127
## 2511 0.23831252 0.5699780 0.1917094

```

Và nhóm dự đoán là:

```

out_class_mlogit_wine <- predict(md_mlogit1, newdata = new_data_wine, type = "class")
out_class_mlogit_wine

```

```

## [1] medium medium medium poor medium
## Levels: excellent medium poor

```

IV.3.3.8. Đánh giá mô hình

Confusion matrix

```

predictedML <- predict(md_mlogit1, datatest_scaled, na.action = na.pass, type = "probs")
predicted_classML <- predict(md_mlogit1, datatest_scaled)

caret::confusionMatrix(as.factor(predicted_classML), as.factor(datatest_scaled$segmentation))

```

```

## Confusion Matrix and Statistics
##
##      Reference
## Prediction excellent medium poor
##   excellent     74    41    8
##   medium       166   360   188
##   poor         12    149   332
##
## Overall Statistics
##
##           Accuracy : 0.5759
## 95% CI : (0.5489, 0.6027)
## No Information Rate : 0.4135
## P-Value [Acc > NIR] : < 2.2e-16
##
##           Kappa : 0.3086
##
## McNemar's Test P-Value : < 2.2e-16
##
## Statistics by Class:
##
##           Class: excellent Class: medium Class: poor
## Sensitivity          0.29365    0.6545    0.6288
## Specificity          0.95455    0.5462    0.7993
## Pos Pred Value       0.60163    0.5042    0.6734
## Neg Pred Value       0.85253    0.6916    0.7658
## Prevalence            0.18947    0.4135    0.3970
## Detection Rate       0.05564    0.2707    0.2496
## Detection Prevalence 0.09248    0.5368    0.3707
## Balanced Accuracy    0.62410    0.6003    0.7140

```

Tỷ lệ chính xác:

```

mean(as.character(predicted_classML) != as.character(data$segmentation))
## [1] 0.4240602

```

Confusion Matrix cho thấy mô hình Multinomial Logist với tất cả các biến có accuracy là 57.59% và tỷ lệ accuracy khoảng 42,41%.

IV.4. Nhận xét và kết luận về kết quả thu được sau quá trình phân tích

Một số nhận xét có thể rút ra sau quá trình phân tích:

- Nhận thấy có sự khác biệt giữa thành phần rượu vang đỏ và rượu vang trắng. Rượu vang đỏ chứa nồng độ clorua và sunfat cao, trong khi, rượu vang trắng có đặc điểm là hàm lượng axit dễ bay hơi thấp. Tuy nhiên, không có sự kết hợp nào giữa các thành phần này ảnh hưởng đến chất lượng rượu một cách đáng kể. Hay nói cách khác là, các thành phần này ảnh hưởng đến chất lượng rượu vang một cách độc lập với màu rượu (do đó, có thể sử dụng chung một model dự đoán chất lượng cho cả hai loại rượu vang đỏ và rượu vang trắng).
- Quá trình phân tích và xây dựng mô hình còn cho thấy các feature quan trọng có ảnh hưởng nhiều nhất đến chất lượng rượu vang được xác định là:
 - + Mật độ (density): Rượu vang có mật độ cao hơn có thể cho thấy hàm lượng đường hoặc cồn cao hơn, điều này có thể góp phần tạo nên hương vị rượu vang đậm đà, đầy đặn hơn. Do đó, mật độ là một yếu tố quan trọng trong phân loại rượu vang, vì nó có thể cung cấp thông tin chi tiết về độ ngọt, độ đậm đà và chất lượng tổng thể của rượu vang.
 - + Độ cồn (alcohol): Hàm lượng cồn là một yếu tố dự đoán quan trọng về phong cách rượu vang, vì các vùng sản xuất rượu vang và giống nho khác nhau được biết đến là sản xuất rượu vang với mức độ cồn riêng biệt. Hàm lượng cồn cao hơn cũng góp phần tạo nên hương vị rượu vang đậm đà, mạnh mẽ hơn.
 - + Hàm lượng đường dư (residual_sugar): Đường dư là lượng đường còn lại trong rượu vang sau khi quá trình lên men hoàn tất. Đây là yếu tố chính trong việc xác định độ ngọt của rượu vang, điều này có thể có tác động đáng kể đến chất lượng được cảm nhận của rượu vang.
 - + Độ axit dễ bay hơi (volatile_acidity): Độ chua dễ bay hơi đề cập đến lượng axit axetic và các hợp chất khác góp phần tạo nên hương thơm và hương vị của rượu vang. Nồng độ axit dễ bay hơi cao có thể khiến rượu vang có vị khó chịu, chua như giấm. Mặt khác, nồng độ vừa phải có thể thêm độ phức tạp và chiều sâu cho hương vị của rượu vang.
 - + Màu rượu vang (color): Màu sắc của rượu vang bị ảnh hưởng bởi giống nho, do nó tiếp xúc trực tiếp với vỏ và quá trình lên men. Các màu sắc khác nhau có thể cho thấy các cấu trúc hương vị, mức độ tannin và tiêm năng lên men khác nhau.
- Từ đó, ta xây dựng thành công các mô hình hồi quy và phân loại với độ chính xác tương đối. Trong đó:
 - + VỚI MÔ HÌNH HỒI QUY:

Mô hình	RMSE	MAE	R-squared	Accuracy
---------	------	-----	-----------	----------

Mô hình	RMSE	MAE	R-squared	Accuracy
Linear Regression	0.8064	0.5271	0.2156	0.2135
Hồi quy đa thức	0.7625	0.4812	0.2463	0.2256
Ridge Regression	0.8032	0.5293	0.2120	0.2105

Hồi quy đa thức là mô hình đạt hiệu suất tốt nhất với điểm RMSE và MAE thấp nhất (0.7625 và 0.4812) chỉ ra khả năng dự đoán chất lượng rượu vang với sai số ít hơn so với các mô hình khác, và độ chính xác (0.2256) cao hơn so với các mô hình khác. Điều này đúng như dự đoán, vì hồi quy đa thức có khả năng mô hình hóa mối quan hệ phi tuyến, đây có thể là yếu tố giúp mô hình dự đoán chính xác hơn với thực tế.

+) Với mô hình phân loại:

Mô hình	Accuracy
Gaussian Naive Bayes	0.5023
Kernel Naive Bayes	0.5564
LDA	0.5805
Multinomial Logistic	0.5759

LDA là mô hình phân loại đạt độ chính xác cao nhất (0.5805), chỉ ra khả năng phân loại chất lượng rượu vang chính xác hơn so với các mô hình khác. LDA là mô hình phân loại tuyến tính, cho phép giải thích dễ dàng hơn về mối quan hệ giữa các thành phần lý hóa sinh và chất lượng rượu vang. Ngoài ra, LDA có khả năng tính toán hiệu quả, phù hợp cho các ứng dụng thực tế cần xử lý lượng dữ liệu lớn.

- Tuy nhiên, các mô hình dự đoán chất lượng rượu vang hiện tại có tiềm năng nhưng vẫn còn một số hạn chế. Để cải thiện độ chính xác và khả năng tổng quát của mô hình, cần bổ sung thêm các yếu tố quan trọng khác, đồng thời nghiên cứu kỹ lưỡng hơn về mối liên hệ giữa các yếu tố này và chất lượng rượu vang. Việc thu thập thêm dữ liệu và áp dụng các kỹ thuật học máy tiên tiến hơn cũng có thể góp phần nâng cao hiệu quả của mô hình.
- Không có thông số có thể được xem là có ảnh hưởng đáng kể đến chất lượng rượu vang (corr của tất cả các thông số với quality đều < 0.5). Vì vậy, có thể kết luận rằng chất lượng rượu chủ yếu phụ thuộc vào sở thích cá nhân của các chuyên gia đánh giá. Do đó, có thể cần xem xét lại việc lựa chọn các feature, và thay vào đó, ta có thể ghi nhận thêm những feature mới khác, chẳng hạn như loại nho, năm sản xuất, thời gian ủ, hay thậm chí là các thông số liên quan đến điều kiện uống rượu như nhiệt độ hoặc độ ẩm để phân tích và xây dựng được các mô hình tốt hơn.
- Ngoài ra, có thể thấy chất lượng rượu vang là một khái niệm phức tạp và mang tính chủ quan, phụ thuộc vào nhiều yếu tố khác nhau. Do đó, việc sử dụng mô hình dự đoán chỉ nên được xem như một công cụ hỗ trợ, cần kết hợp thêm với đánh giá của chuyên gia và sở thích cá nhân để có thể đưa ra các đánh giá chính xác nhất về chất lượng rượu vang.

Một số đề xuất cải thiện mô hình:

- Bổ sung các yếu tố quan trọng: Loại nho, năm sản xuất, thời gian ủ, điều kiện bảo quản, nhiệt độ uống, độ ẩm,...
- Nghiên cứu mối liên hệ giữa các yếu tố: Phân tích kỹ lưỡng hơn về mối liên hệ giữa các yếu tố được xác định và chất lượng rượu vang.
- Thu thập thêm dữ liệu: Thu thập thêm dữ liệu về các loại rượu vang khác nhau và đánh giá của chuyên gia để cải thiện độ chính xác của mô hình.

V. Đề xuất các phương pháp cải tiến sản phẩm và chiến lược kinh doanh

1. Cải tiến sản phẩm

Tinh chỉnh thành phần: Việc điều chỉnh mức độ, tỉ lệ của các thành phần có ảnh hưởng chính như nồng độ cồn, độ axit và độ pH có thể giúp đạt được các mục tiêu cải thiện chất lượng cụ thể.

Tối ưu hóa quá trình lên men: Giám sát và tối ưu hóa các thành phần chính trong quá trình lên men để đạt được hương vị và mùi thơm mong muốn.

Giảm thiểu các thành phần không mong muốn: Việc giải quyết các yếu tố góp phần tạo ra các thuộc tính không mong muốn, chẳng hạn như độ axit dễ bay hơi, có thể giúp tạo ra rượu vang có chất lượng tốt hơn.

2. Chiến lược tiếp thị

Phân khúc khách hàng: Sử dụng xếp hạng chất lượng được mô hình dự đoán để phân loại rượu thành các phân khúc chất lượng, cho phép tiếp thị có mục tiêu đến từng khách hàng phù hợp.

Lấy các thuộc tính chính làm USP (Unique Selling Point) của sản phẩm: Tận dụng các thuộc tính có ảnh hưởng nhất được mô hình xác định trong các kế hoạch tiếp thị để thu hút người tiêu dùng đang tìm kiếm những tính chất sản phẩm cụ thể.

3. Chiến lược thu hút khách hàng

Đề xuất rượu vang: Cung cấp các đề xuất rượu được cá nhân hóa cho khách hàng dựa trên sở thích và các giao dịch mua trước đây của họ.

Tạo nội dung có giá trị giáo dục: Giáo dục người tiêu dùng về tầm quan trọng của các thuộc tính khác nhau trong chất lượng rượu vang, thúc đẩy sự đánh giá và hiểu biết sâu sắc hơn về rượu.

4. Tối ưu hóa quy trình sản xuất

Giám sát thời gian thực: Triển khai giám sát thời gian thực các thuộc tính chính trong quá trình sản xuất để phát hiện sai lệch và điều chỉnh kịp thời.

Phân tích dự đoán: Sử dụng mô hình để dự đoán các vấn đề chất lượng tiềm ẩn một cách chủ động trước khi chúng xảy ra, từ đó kịp thời can thiệp.

5. Đảm bảo quy trình kiểm tra chất lượng

Kiểm tra tính nhất quán: Sử dụng mô hình để theo dõi tính nhất quán của chất lượng rượu theo thời gian giữa các lô.

Kiểm soát chất lượng: Thực hiện các quy trình kiểm soát chất lượng dựa trên xếp hạng chất lượng dự đoán để đảm bảo rằng chỉ những loại rượu đáp ứng các tiêu chuẩn nhất định mới được xuất xưởng.