Dataset URL: https://archive.ics.uci.edu/dataset/186/wine+quality

Title: Exploratory analysis of wines in order to predict and classify wine quality

```
# I will be conducting exploratory analysis on the wine quality dataset taken fron the UC irvine machin
```

```
str(Data1)
```

```
## 'data.frame':    1599 obs. of  12 variables:
##  $ fixed.acidity       : num  7.4 7.8 7.8 11.2 7.4 7.4 7.9 7.3 7.8 7.5 ...
##  $ volatile.acidity    : num  0.7 0.88 0.76 0.28 0.7 0.66 0.6 0.65 0.58 0.5 ...
##  $ citric.acid         : num  0 0 0.04 0.56 0 0 0.06 0 0.02 0.36 ...
##  $ residual.sugar      : num  1.9 2.6 2.3 1.9 1.9 1.8 1.6 1.2 2 6.1 ...
##  $ chlorides           : num  0.076 0.098 0.092 0.075 0.076 0.075 0.069 0.065 0.073 0.071 ...
##  $ free.sulfur.dioxide : num  11 25 15 17 11 13 15 15 9 17 ...
##  $ total.sulfur.dioxide: num  34 67 54 60 34 40 59 21 18 102 ...
##  $ density             : num  0.998 0.997 0.997 0.998 0.998 ...
##  $ pH                  : num  3.51 3.2 3.26 3.16 3.51 3.51 3.3 3.39 3.36 3.35 ...
##  $ sulphates           : num  0.56 0.68 0.65 0.58 0.56 0.56 0.46 0.47 0.57 0.8 ...
##  $ alcohol             : num  9.4 9.8 9.8 9.8 9.4 9.4 9.4 10 9.5 10.5 ...
##  $ quality             : int  5 5 5 6 5 5 5 7 7 5 ...
```

```
Data1$color <- "red"
Data2$color <- "white"

set.seed(1)
wine <- rbind(Data1, Data2)
wine_df <- wine[sample(1:nrow(wine)), ]
print(nrow(wine_df))
```

```
## [1] 6497
```

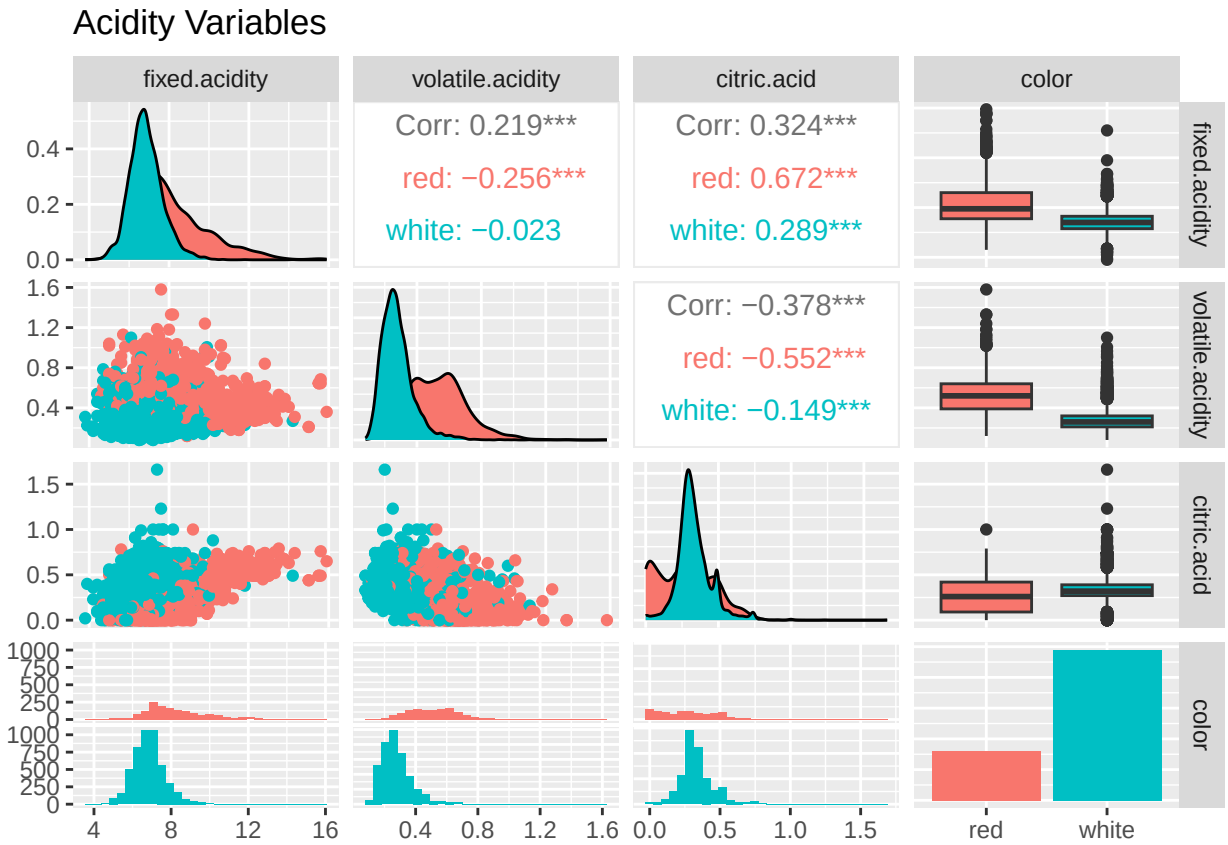
```
summary(wine_df)
```

```
##  fixed.acidity    volatile.acidity  citric.acid     residual.sugar
##  Min.   : 3.800   Min.   :0.0800   Min.   :0.0000   Min.   : 0.600
##  1st Qu.: 6.400   1st Qu.:0.2300   1st Qu.:0.2500   1st Qu.: 1.800
##  Median : 7.000   Median :0.2900   Median :0.3100   Median : 3.000
##  Mean   : 7.215   Mean   :0.3397   Mean   :0.3186   Mean   : 5.443
##  3rd Qu.: 7.700   3rd Qu.:0.4000   3rd Qu.:0.3900   3rd Qu.: 8.100
##  Max.   :15.900   Max.   :1.5800   Max.   :1.6600   Max.   :65.800
##    chlorides       free.sulfur.dioxide total.sulfur.dioxide    density
##  Min.   :0.00900   Min.   :  1.00      Min.   :  6.0        Min.   :0.9871
##  1st Qu.:0.03800   1st Qu.: 17.00      1st Qu.: 77.0        1st Qu.:0.9923
##  Median :0.04700   Median : 29.00      Median :118.0        Median :0.9949
##  Mean   :0.05603   Mean   : 30.53      Mean   :115.7        Mean   :0.9947
##  3rd Qu.:0.06500   3rd Qu.: 41.00      3rd Qu.:156.0        3rd Qu.:0.9970
##  Max.   :0.61100   Max.   :289.00      Max.   :440.0        Max.   :1.0390
##        pH          sulphates         alcohol         quality
##  Min.   :2.720   Min.   :0.2200   Min.   : 8.00   Min.   :3.000
##  1st Qu.:3.110   1st Qu.:0.4300   1st Qu.: 9.50   1st Qu.:5.000
##  Median :3.210   Median :0.5100   Median :10.30   Median :6.000
```

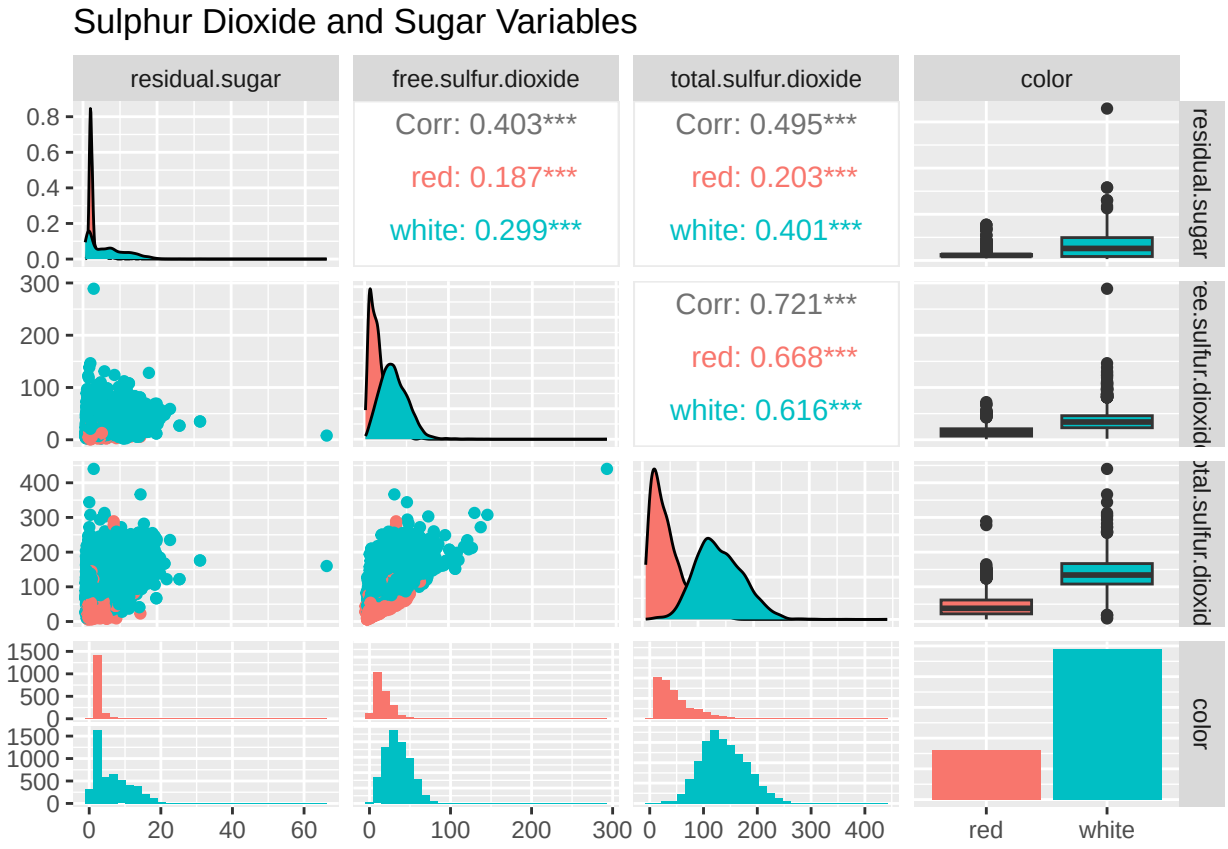
```
##  Mean   :3.219   Mean   :0.5313   Mean   :10.49   Mean   :5.818
##  3rd Qu.:3.320   3rd Qu.:0.6000   3rd Qu.:11.30   3rd Qu.:6.000
##  Max.   :4.010   Max.   :2.0000   Max.   :14.90   Max.   :9.000
##     color
##  Length:6497
##  Class :character
##  Mode  :character
##
##
##
```

```
##   color fixed.acidity volatile.acidity citric.acid residual.sugar  chlorides
## 1   red      8.319637        0.5278205   0.2709756       2.538806 0.08746654
## 2 white      6.854788        0.2782411   0.3341915       6.391415 0.04577236
##   free.sulfur.dioxide total.sulfur.dioxide   density       pH sulphates
## 1            15.87492              46.46779 0.9967467 3.311113 0.6581488
## 2            35.30808             138.36066 0.9940274 3.188267 0.4898469
##    alcohol  quality
## 1 10.42298 5.636023
## 2 10.51427 5.877909
```

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

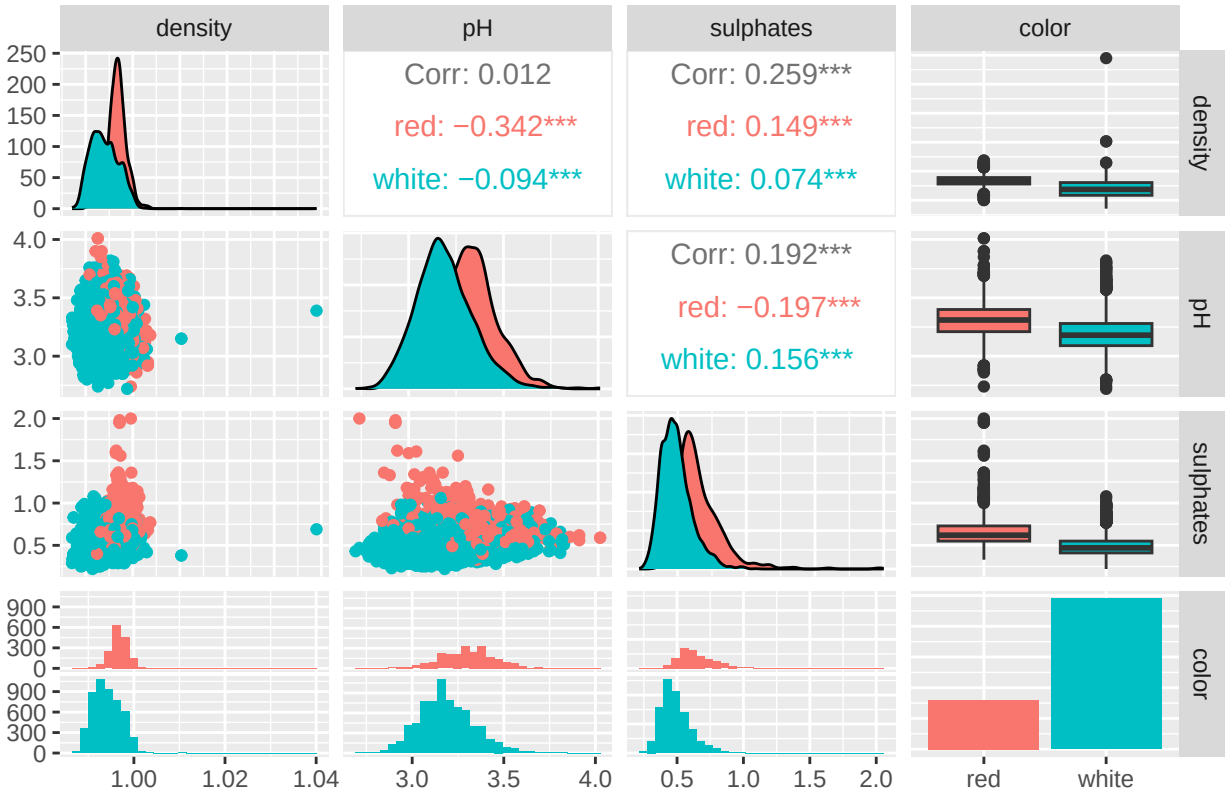


```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

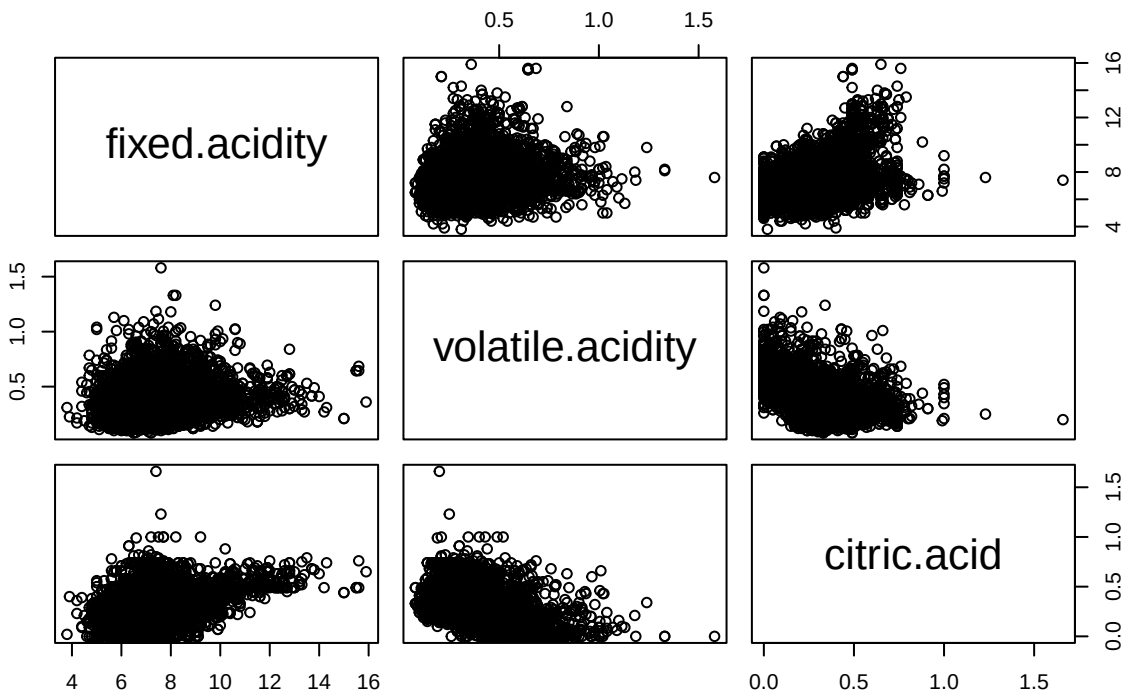



```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

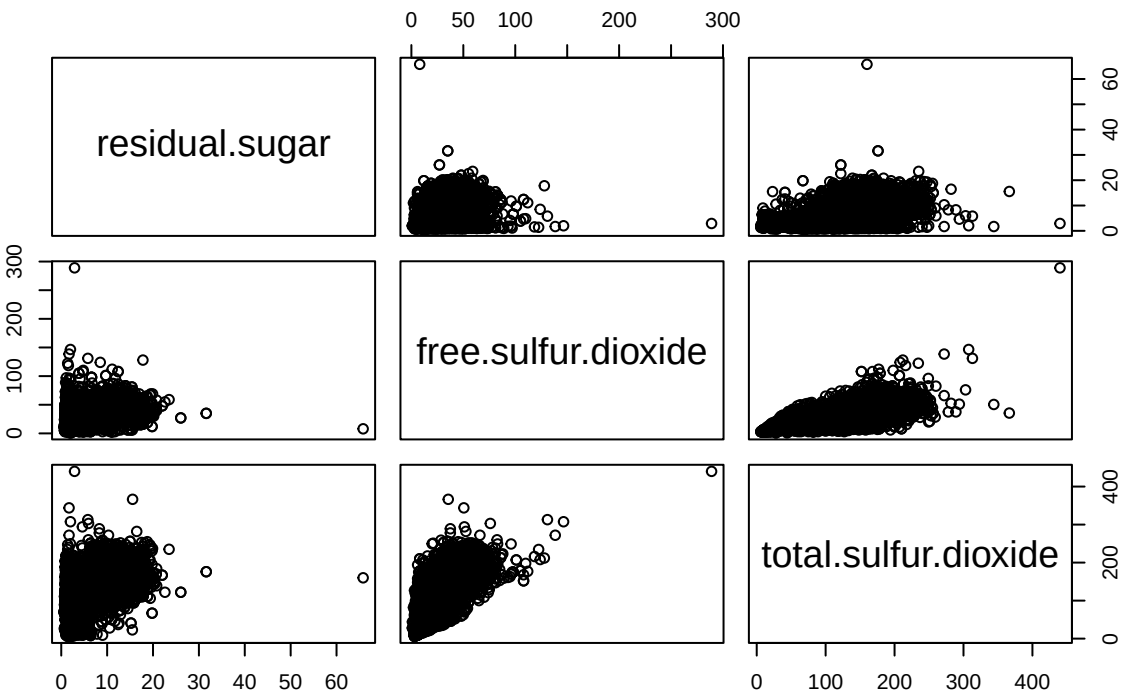
Density, pH, and Sulphates

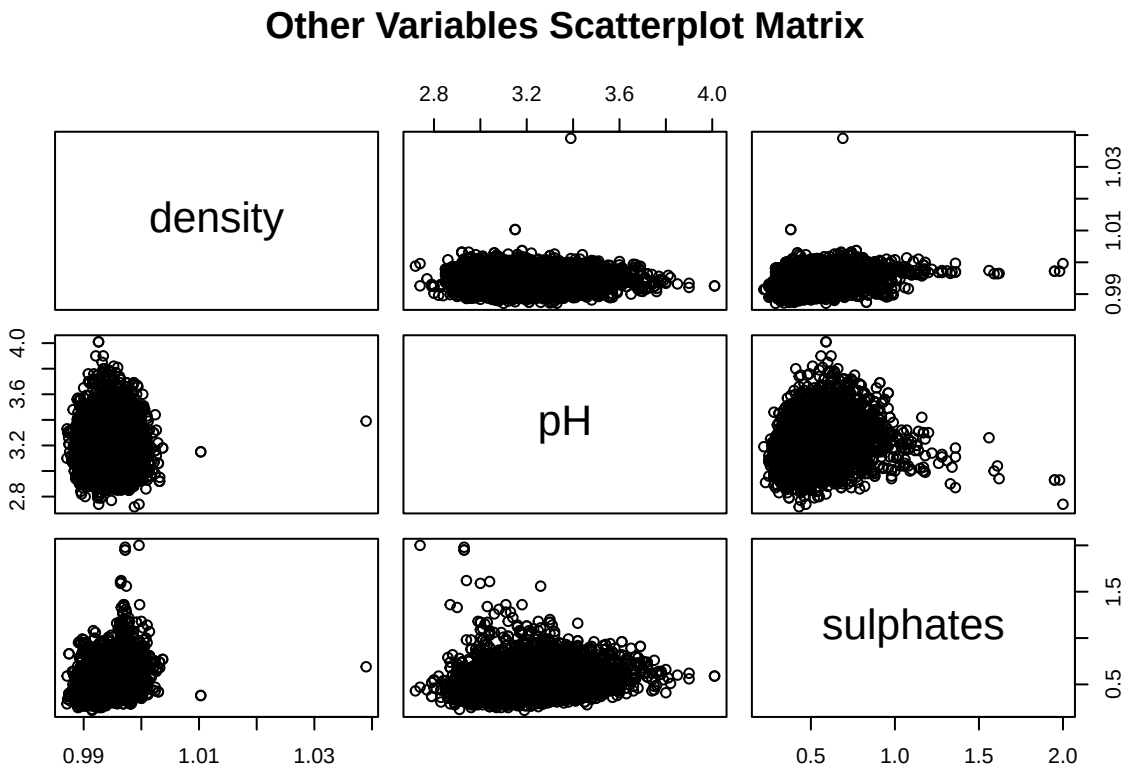


Acidity Variables Scatterplot Matrix



Sulfur Dioxide and Sugar Variables Scatterplot Matrix

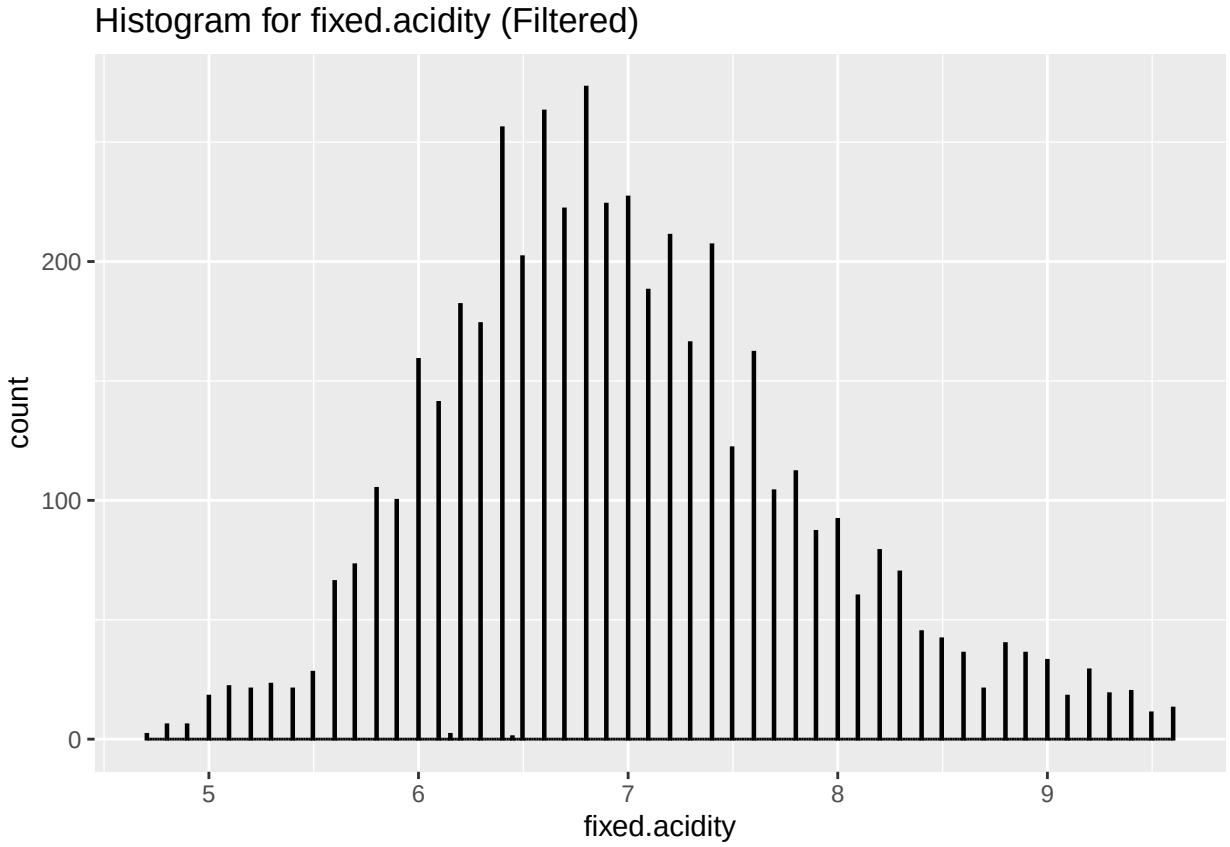




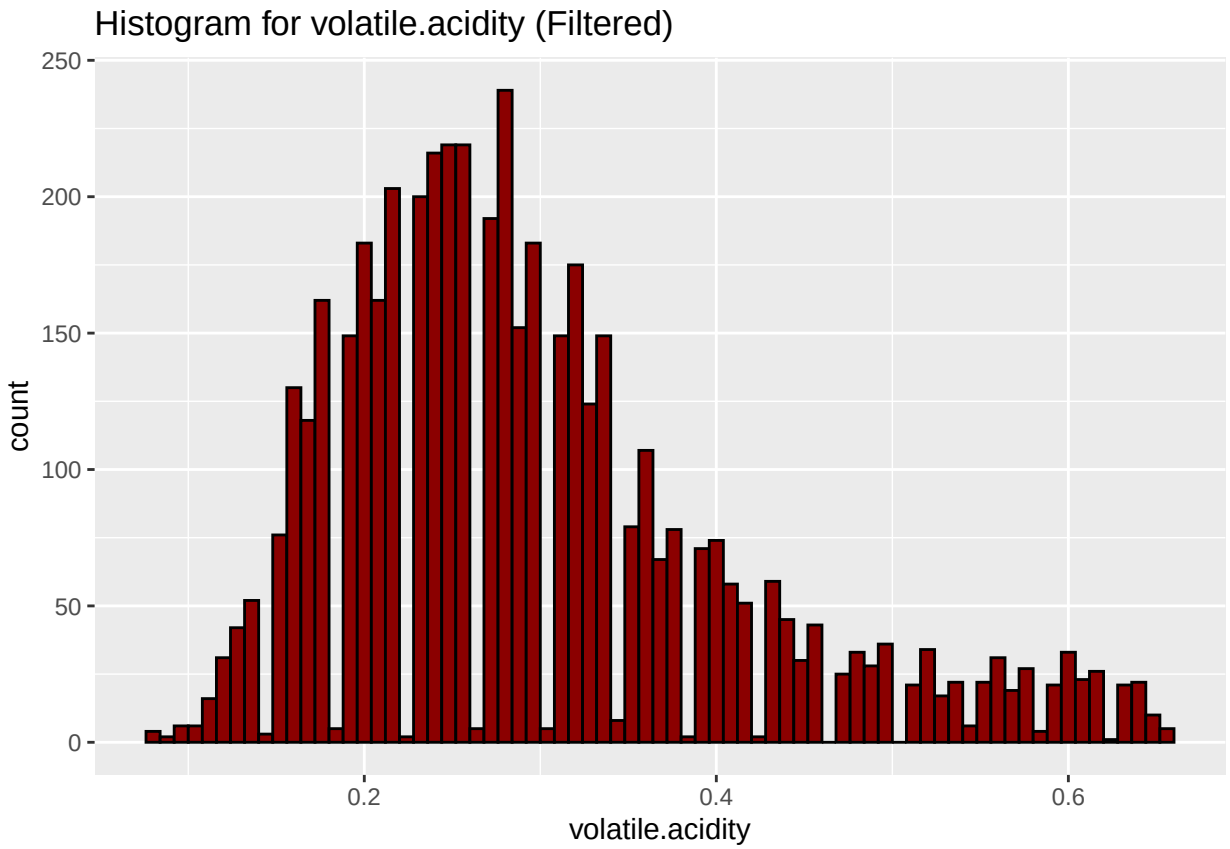

```
#remove outliers
hist_list <- list()

within_iqr <- function(x) {
  lower_bound <- quantile(x, 0.25) - 1.5 * IQR(x)
  upper_bound <- quantile(x, 0.75) + 1.5 * IQR(x)
  x >= lower_bound & x <= upper_bound
}

wine_df_filtered <- wine_df %>%
  filter(if_all(where(is.numeric), within_iqr))

if (nrow(wine_df_filtered) == 0) {
  stop("Filtered dataframe is empty. Adjust filtering criteria.")
}

#Show updated dataframe
for (col in categorical_cols) {
  count_wine_df_filtered <- wine_df_filtered %>% count(!!sym(col))

  p <- ggplot(data = count_wine_df_filtered, aes(x = !!sym(col), y = n)) +
    geom_bar(stat = "identity", fill = "darkred") +
    ggtitle(paste("Bar Plot for", col, "(Filtered)"))

}
```

```
for (col in numeric_cols) {
  p <- ggplot(wine_df_filtered, aes_string(x = col)) +
    geom_histogram(binwidth = 0.008, fill = "darkred", color = "black") +
    ggtitle(paste("Histogram for", col, "(Filtered)"))
  hist_list[[col]] <- p
}

hist_list[["density"]] <- ggplot(wine_df, aes(x = density)) +
  geom_histogram(binwidth = 0.0005, fill = "darkred", color = "black") +
  ggtitle("Histogram for density (Adjusted Binwidth)")

hist_list[["chlorides"]] <- ggplot(wine_df, aes(x = chlorides)) +
  geom_histogram(binwidth = 0.009, fill = "darkred", color = "black") +
  ggtitle("Histogram for chlorides (Adjusted Binwidth)")

hist_list[["residual.sugar"]] <- ggplot(wine_df, aes(x = residual.sugar)) +
  geom_histogram(binwidth = 0.59, bins = 50,  fill = "darkred", color = "black") +
  ggtitle("Histogram for residual.sugar (Adjusted Binwidth)")

print(hist_list[names(hist_list) != "quality"] )
```

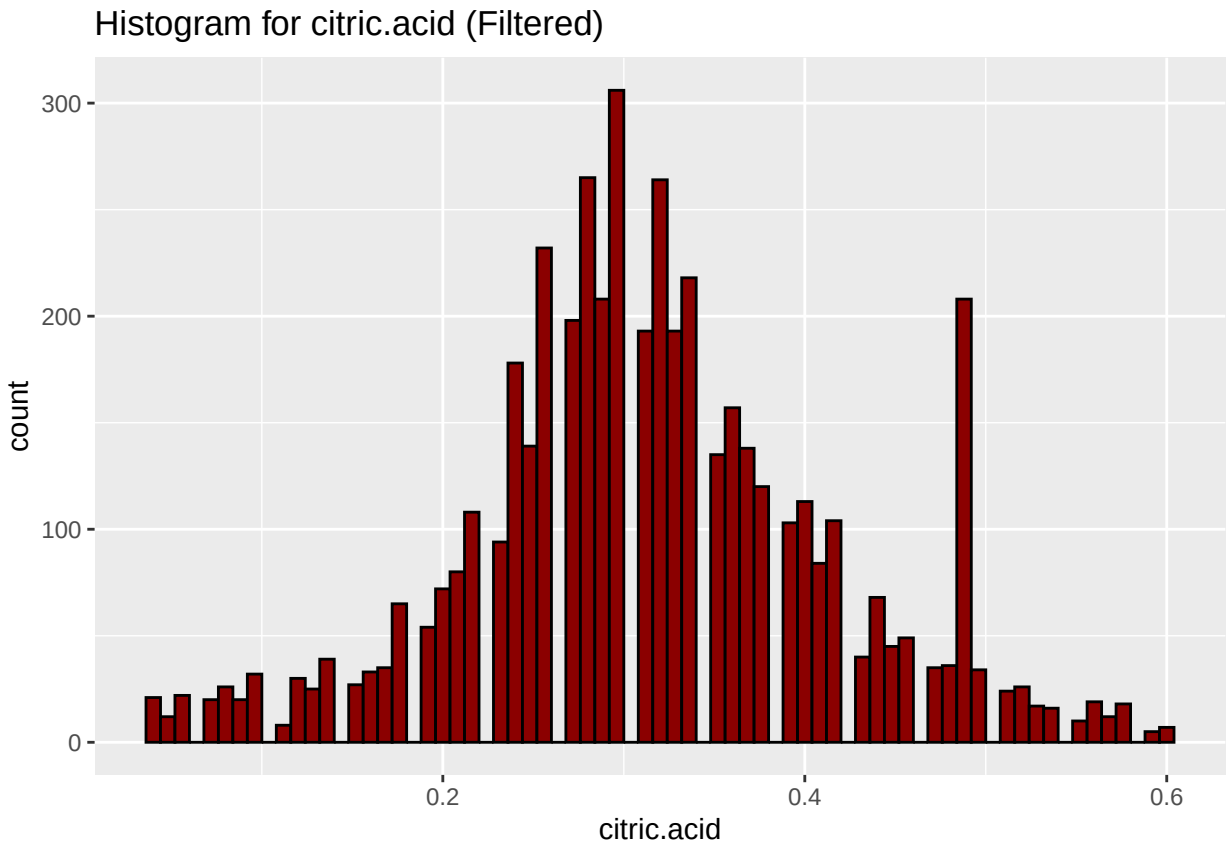
```
## $fixed.acidity
```



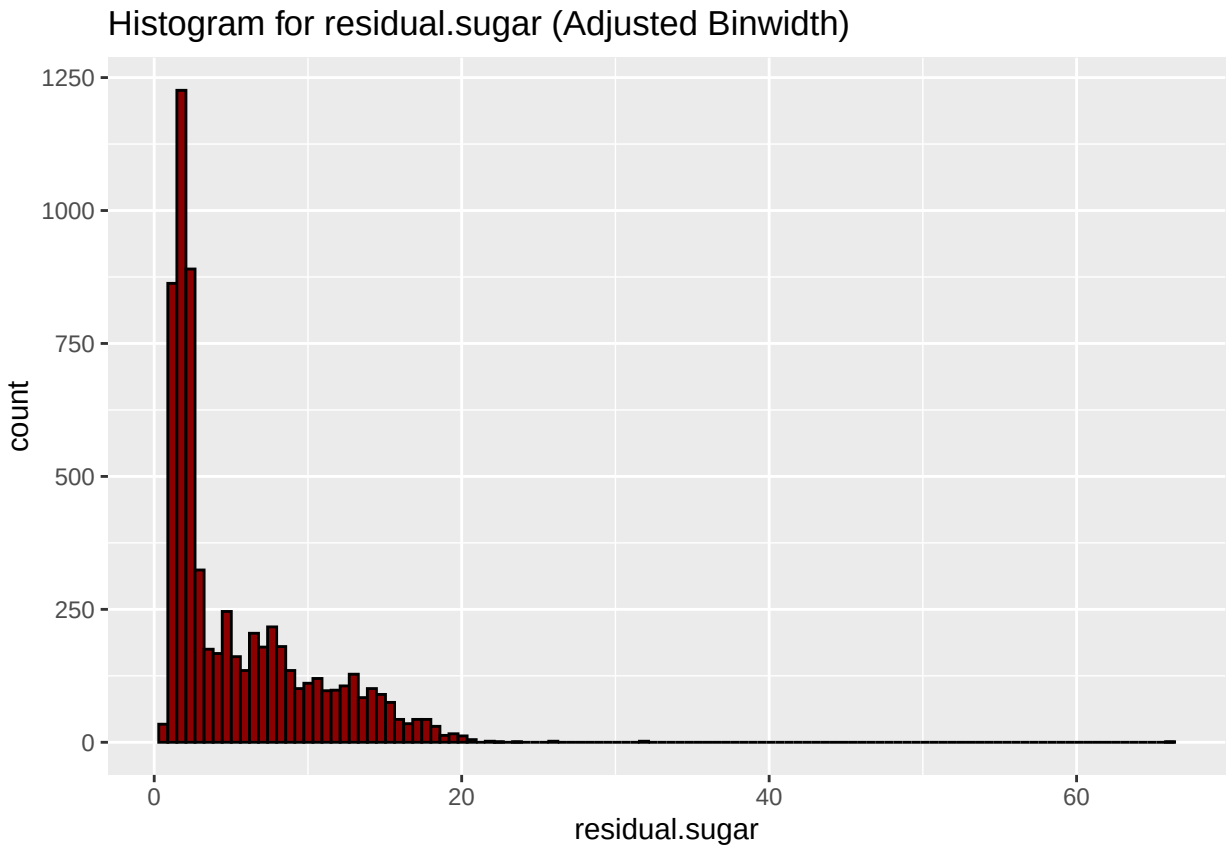

```
##
## $volatile.acidity
```


Histogram for volatile.acidity (Filtered)
250
200
150
100
50
0
count
0.2
0.4
0.6
volatile.acidity

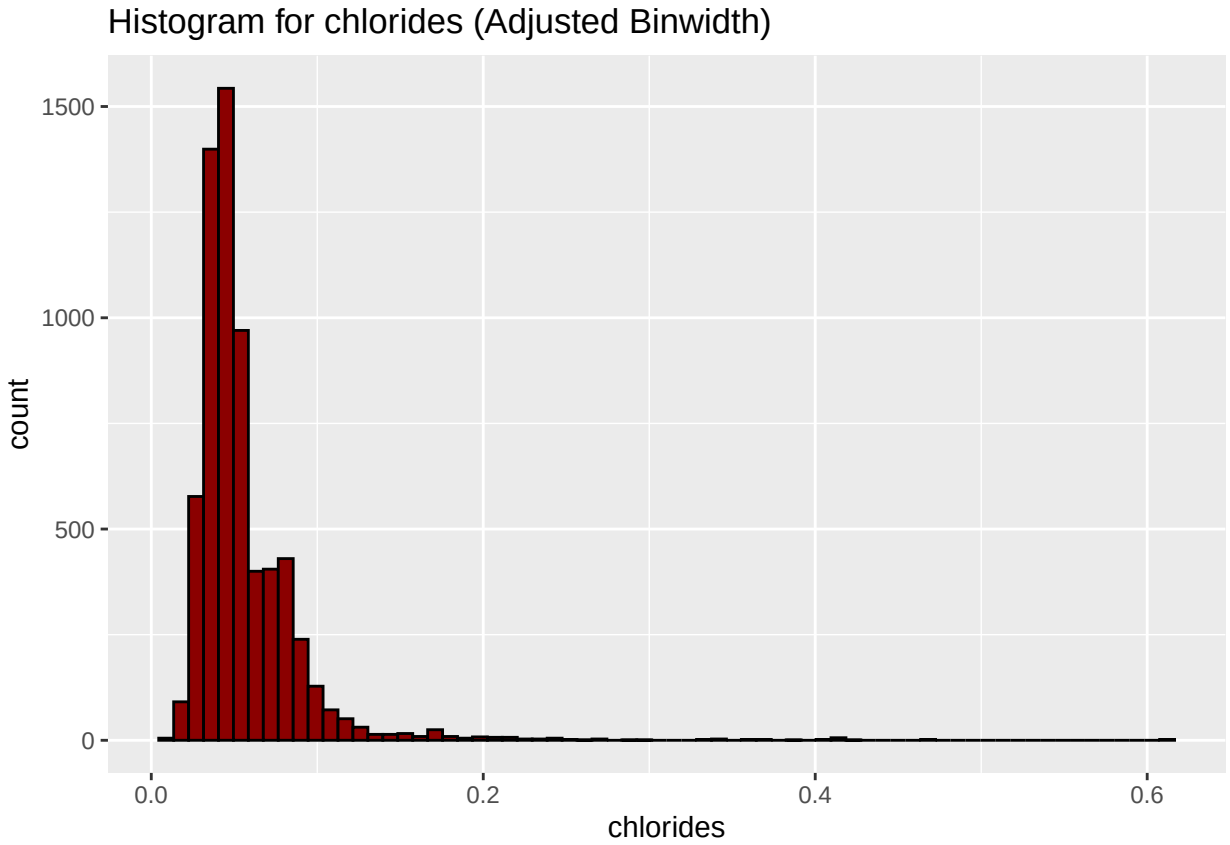
```
##
## $citric.acid
```



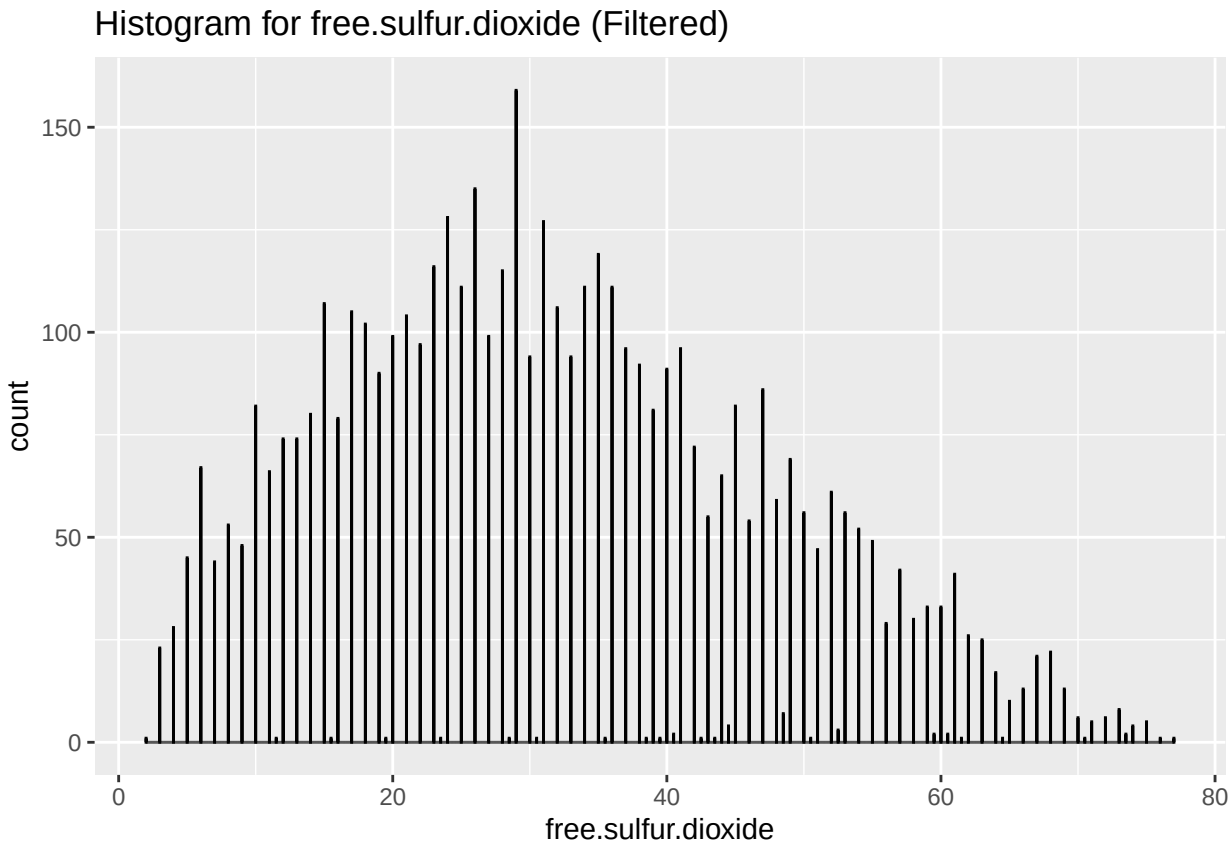

```
##
## $residual.sugar
```


Histogram for residual.sugar (Adjusted Binwidth)
count
1250
1000
750
500
250
0
0
20
40
60
residual.sugar

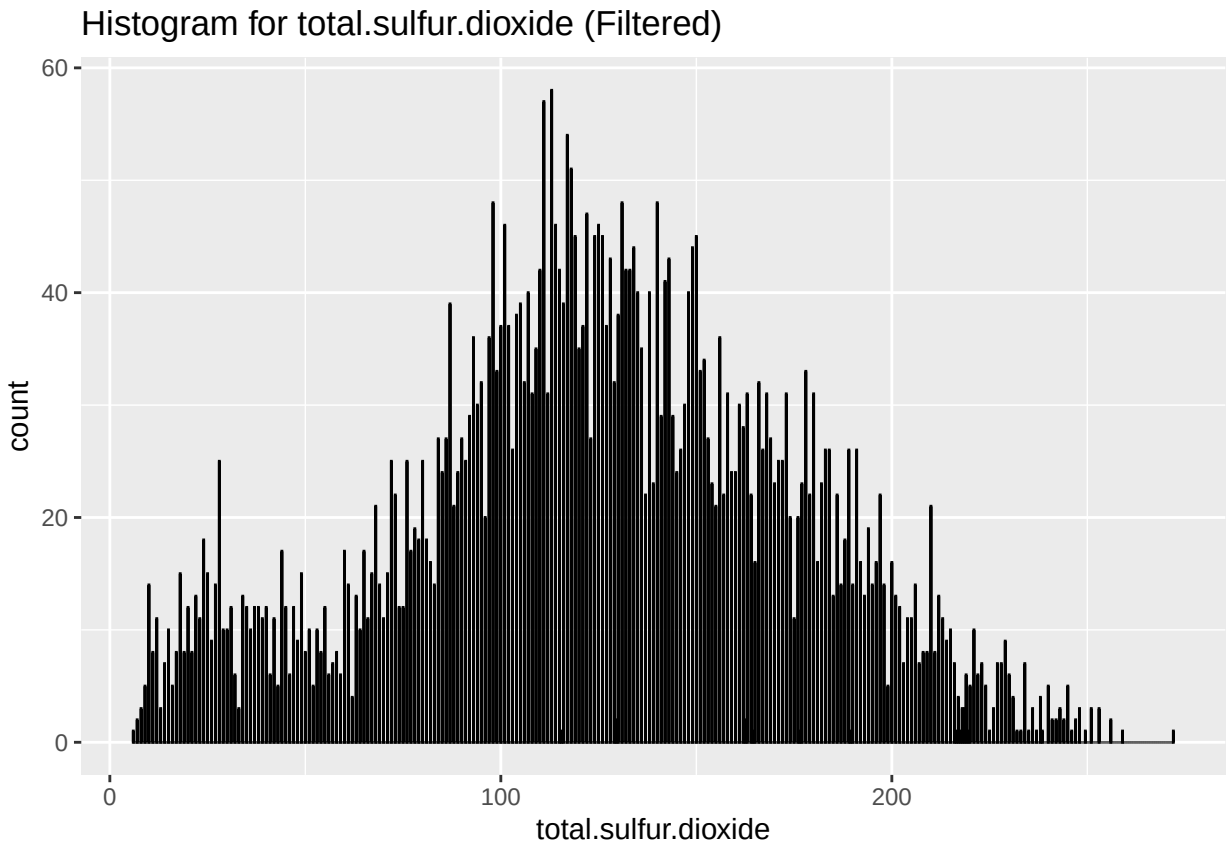
```
##
## $chlorides
```



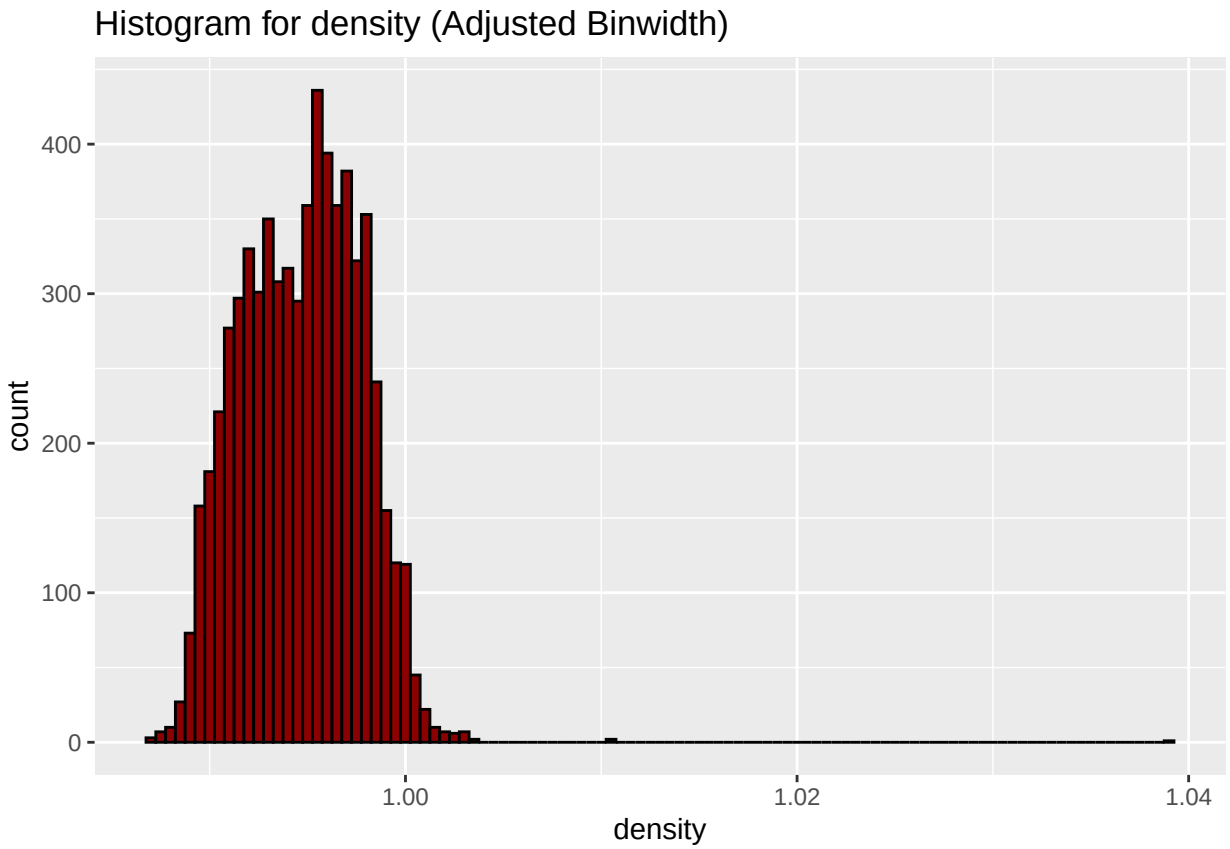

```
##
## $free.sulfur.dioxide
```



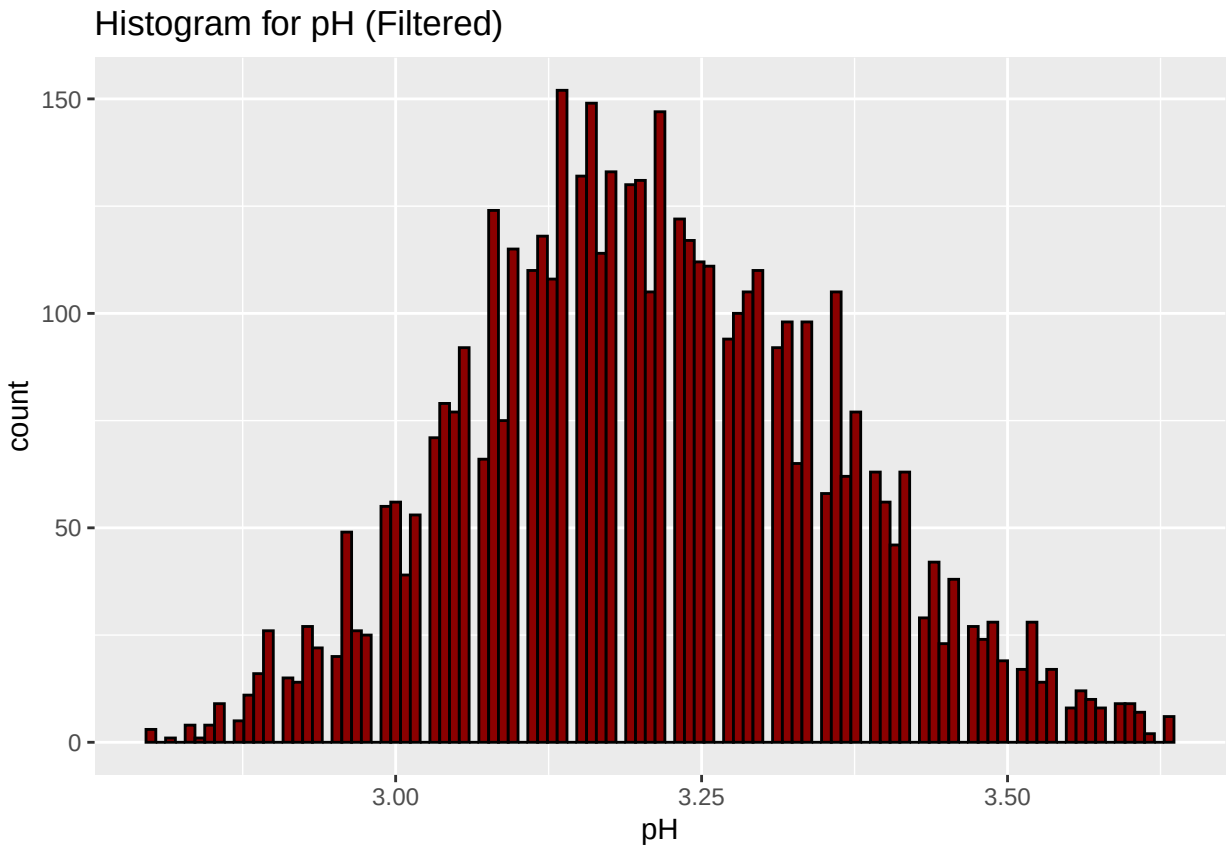

```
##
## $total.sulfur.dioxide
```


Histogram for total.sulfur.dioxide (Filtered)
60
40
20
0
count
0
100
200
total.sulfur.dioxide

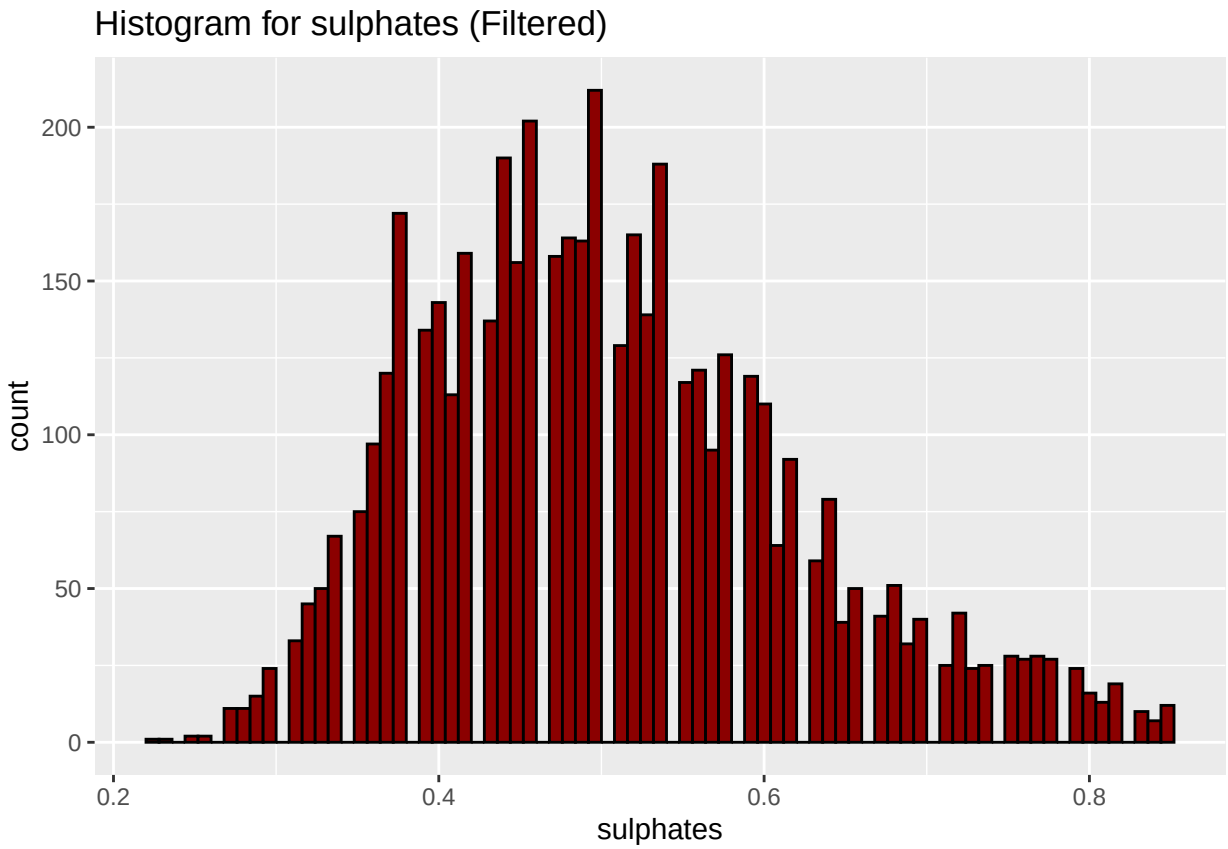
```
##
## $density
```


Histogram for density (Adjusted Binwidth)
count
400
300
200
100
0
1.00
1.02
1.04
density

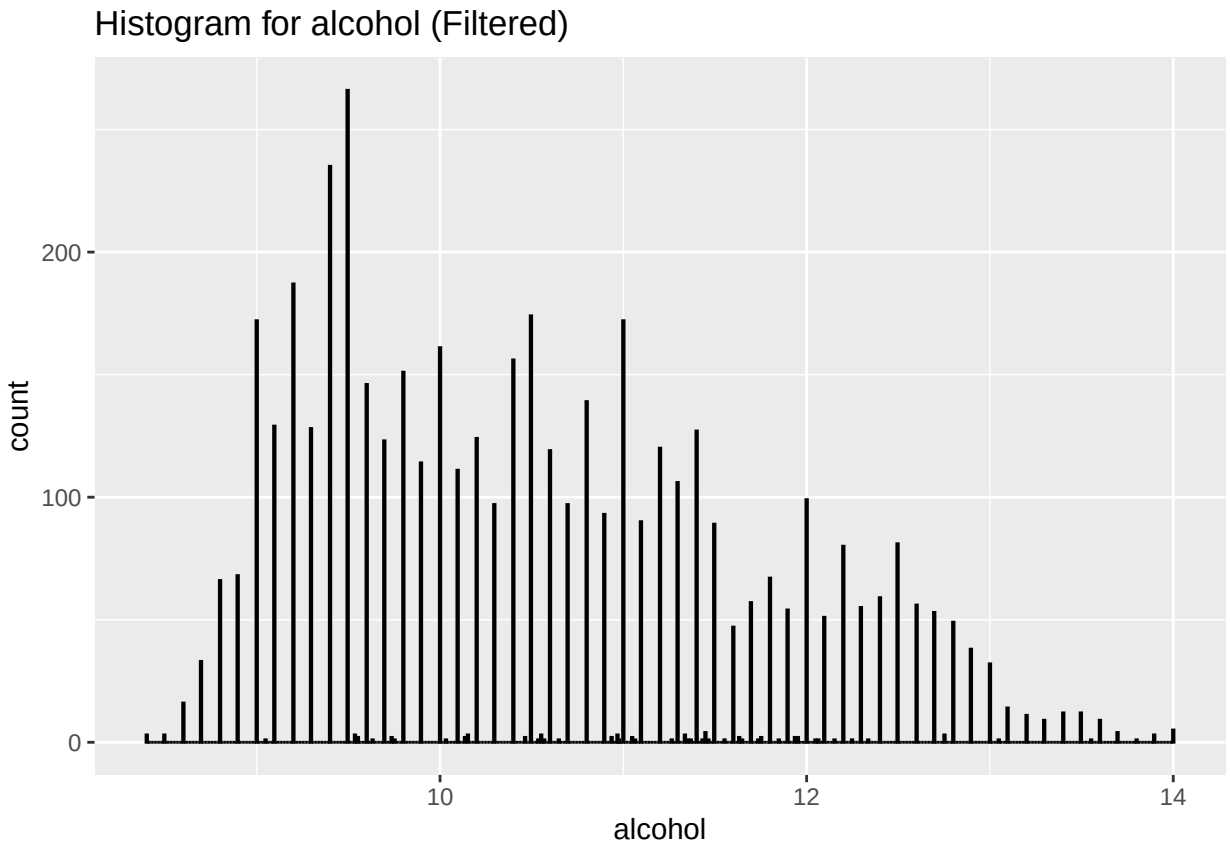
```
##
## $pH
```




```
##
## $sulphates
```


Histogram for sulphates (Filtered)
count
sulphates
200
150
100
50
0
0.2
0.4
0.6
0.8

```
##
## $alcohol
```

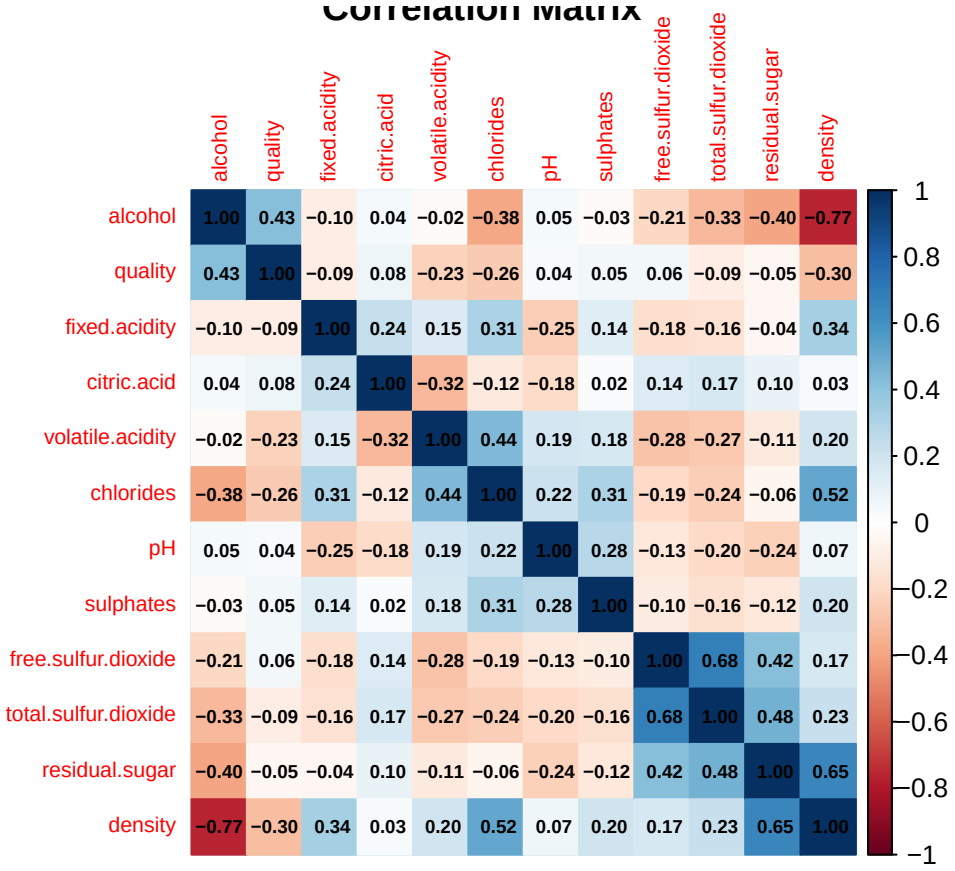



```
cor_matrix <- cor(wine_df_filtered[, !colnames(wine_df_filtered) %in% "color"])
cor_matrix
```

```
##                       fixed.acidity volatile.acidity citric.acid residual.sugar
## fixed.acidity            1.00000000       0.15487361  0.23747614    -0.04137628
## volatile.acidity         0.15487361       1.00000000 -0.32421775    -0.11367437
## citric.acid              0.23747614      -0.32421775  1.00000000     0.09892895
## residual.sugar          -0.04137628      -0.11367437  0.09892895     1.00000000
## chlorides                0.31391759       0.44094791 -0.11720188    -0.05694625
## free.sulfur.dioxide     -0.17749086      -0.28334671  0.14304983     0.42108586
## total.sulfur.dioxide    -0.15969926      -0.26694367  0.17325978     0.47667543
## density                  0.33761864       0.19978065  0.03135473     0.65337688
## pH                      -0.24551943       0.18634457 -0.18361825    -0.23857180
## sulphates                0.13964312       0.17973982  0.02334830    -0.12339320
## alcohol                 -0.09715018      -0.02439932  0.04246182    -0.39790190
## quality                 -0.09118484      -0.22676378  0.07947081    -0.04590283
##                          chlorides free.sulfur.dioxide total.sulfur.dioxide
## fixed.acidity          0.31391759         -0.17749086          -0.15969926
## volatile.acidity       0.44094791         -0.28334671          -0.26694367
## citric.acid           -0.11720188          0.14304983           0.17325978
## residual.sugar        -0.05694625          0.42108586           0.47667543
## chlorides              1.00000000         -0.19212747          -0.24408298
## free.sulfur.dioxide   -0.19212747          1.00000000           0.68043781
## total.sulfur.dioxide  -0.24408298          0.68043781           1.00000000
## density                0.51642939          0.17129521           0.22654339
```

```
## pH                    0.21791267         -0.12816707          -0.19503362
## sulphates             0.31360322         -0.10048468          -0.16042684
## alcohol              -0.38205108         -0.20918241          -0.32985531
## quality              -0.25697645          0.05566345          -0.09068802
##                         density          pH   sulphates     alcohol
## fixed.acidity         0.33761864 -0.24551943  0.13964312 -0.09715018
## volatile.acidity      0.19978065  0.18634457  0.17973982 -0.02439932
## citric.acid           0.03135473 -0.18361825  0.02334830  0.04246182
## residual.sugar        0.65337688 -0.23857180 -0.12339320 -0.39790190
## chlorides             0.51642939  0.21791267  0.31360322 -0.38205108
## free.sulfur.dioxide   0.17129521 -0.12816707 -0.10048468 -0.20918241
## total.sulfur.dioxide  0.22654339 -0.19503362 -0.16042684 -0.32985531
## density               1.00000000  0.06794285  0.20445229 -0.76934221
## pH                    0.06794285  1.00000000  0.27582283  0.04858106
## sulphates             0.20445229  0.27582283  1.00000000 -0.02970806
## alcohol              -0.76934221  0.04858106 -0.02970806  1.00000000
## quality              -0.30431399  0.03956114  0.05435580  0.42878450
##                         quality
## fixed.acidity        -0.09118484
## volatile.acidity     -0.22676378
## citric.acid           0.07947081
## residual.sugar       -0.04590283
## chlorides            -0.25697645
## free.sulfur.dioxide   0.05566345
## total.sulfur.dioxide -0.09068802
## density              -0.30431399
## pH                    0.03956114
## sulphates             0.05435580
## alcohol               0.42878450
## quality               1.00000000
```

```r
corrplot(cor_matrix,
         method = "color",
         order = "hclust",
         addCoef.col = "black",
         title = "Correlation Matrix",
         bg = "lightblue",
         tl.cex = 0.7,
         number.cex = 0.6)
```




```
columns_to_remove <- findCorrelation(
  cor_matrix,
  cutoff = 0.5,
  verbose = FALSE,
  names = TRUE,
)

print(columns_to_remove)
```

```
## [1] "density"              "total.sulfur.dioxide"
```

```
wine_low_cor <- wine_df_filtered[, !(colnames(wine_df_filtered) %in% c("density", "total.sulfur.dioxide
```

```
set.seed(15)
train_indices <- sample(seq_len(nrow(wine_low_cor)), size = 0.66 * nrow(wine_low_cor))

train_data <- wine_low_cor[train_indices, ]
test_data <- wine_low_cor[-train_indices, ]

cat("Train size:", nrow(train_data), "\nTest size:", nrow(test_data))
```

```
## Train size: 3194
## Test size: 1646
```

```r
head(train_data)
```

```
##      fixed.acidity volatile.acidity citric.acid residual.sugar chlorides
## 4591           6.9             0.40        0.37            8.9     0.053
## 1698           7.0             0.17        0.31            4.8     0.034
## 4273           7.2             0.60        0.20            9.9     0.070
## 261            9.3             0.41        0.39            2.2     0.064
## 4079           6.9             0.24        0.37            6.1     0.027
## 1387           6.3             0.51        0.13            2.3     0.076
##      free.sulfur.dioxide   pH sulphates alcohol quality color
## 4591                  36 3.16      0.50     9.3       5 white
## 1698                  34 3.36      0.48     9.6       7 white
## 4273                  21 3.03      0.54     9.1       5 white
## 261                   12 3.26      0.65    10.2       5   red
## 4079                  38 3.19      0.34    12.4       6 white
## 1387                  29 3.42      0.75    11.0       6   red
```

```r
lm.fit <- lm(quality ~ ., data = train_data)
summary(lm.fit)
```

```
##
## Call:
## lm(formula = quality ~ ., data = train_data)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2.64695 -0.43890 -0.00878  0.44468  1.87334
##
## Coefficients:
##                       Estimate Std. Error t value Pr(>|t|)
## (Intercept)          2.1613877  0.4159984   5.196 2.17e-07 ***
## fixed.acidity       -0.0002951  0.0162435  -0.018 0.985506
## volatile.acidity    -1.5984788  0.1273343 -12.553  < 2e-16 ***
## citric.acid         -0.2022371  0.1287417  -1.571 0.116311
## residual.sugar       0.0238260  0.0031326   7.606 3.71e-14 ***
## chlorides           -4.4857454  1.1639314  -3.854 0.000119 ***
## free.sulfur.dioxide  0.0033492  0.0008662   3.867 0.000113 ***
## pH                   0.3145703  0.0924568   3.402 0.000676 ***
## sulphates            0.6131301  0.1124998   5.450 5.42e-08 ***
## alcohol              0.2981358  0.0125023  23.846  < 2e-16 ***
## colorwhite          -0.3218154  0.0602862  -5.338 1.00e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.65 on 3183 degrees of freedom
## Multiple R-squared:  0.2634, Adjusted R-squared:  0.2611
## F-statistic: 113.8 on 10 and 3183 DF,  p-value: < 2.2e-16
```

```r
# Set significance level for elimination
significance_level <- 0.05

# Backward elimination loop
```

```r
while (TRUE) {
  model_summary <- summary(lm.fit)
  p_values <- coef(model_summary)[, "Pr(>|t|)"]
  p_values <- p_values[-1]

  max_p_value <- max(p_values)

  if (max_p_value > significance_level) {
    variable_to_remove <- names(p_values)[which.max(p_values)]

    updated_formula <- update(lm.fit, paste(". ~ . -", variable_to_remove))

    lm.fit <- lm(updated_formula, data = wine_low_cor)

    print(summary(lm.fit))
  } else {

    break
  }
}
```

```
##
## Call:
## lm(formula = updated_formula, data = wine_low_cor)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2.63695 -0.43934  0.00071  0.43899  1.93634
##
## Coefficients:
##                       Estimate Std. Error t value Pr(>|t|)
## (Intercept)           2.412725   0.283895   8.499  < 2e-16 ***
## volatile.acidity     -1.737815   0.105056 -16.542  < 2e-16 ***
## citric.acid          -0.200877   0.100505  -1.999  0.04570 *
## residual.sugar        0.023943   0.002555   9.371  < 2e-16 ***
## chlorides            -4.467642   0.927670  -4.816 1.51e-06 ***
## free.sulfur.dioxide   0.004071   0.000706   5.767 8.55e-09 ***
## pH                    0.227491   0.070158   3.243  0.00119 **
## sulphates             0.607880   0.091273   6.660 3.04e-11 ***
## alcohol               0.305314   0.010080  30.290  < 2e-16 ***
## colorwhite           -0.362162   0.047718  -7.590 3.82e-14 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.6531 on 4830 degrees of freedom
## Multiple R-squared:  0.2732, Adjusted R-squared:  0.2718
## F-statistic: 201.7 on 9 and 4830 DF,  p-value: < 2.2e-16
```

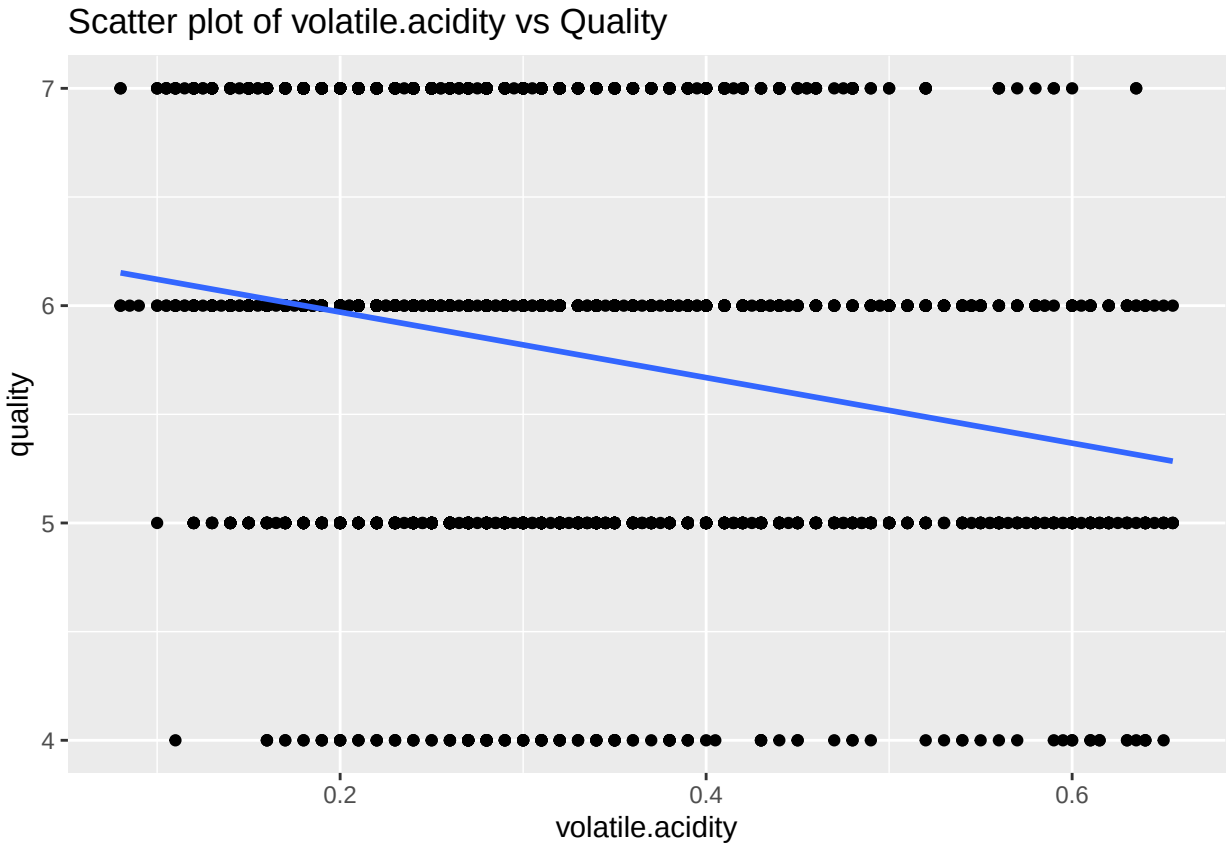
```r
final_model <- lm.fit
```

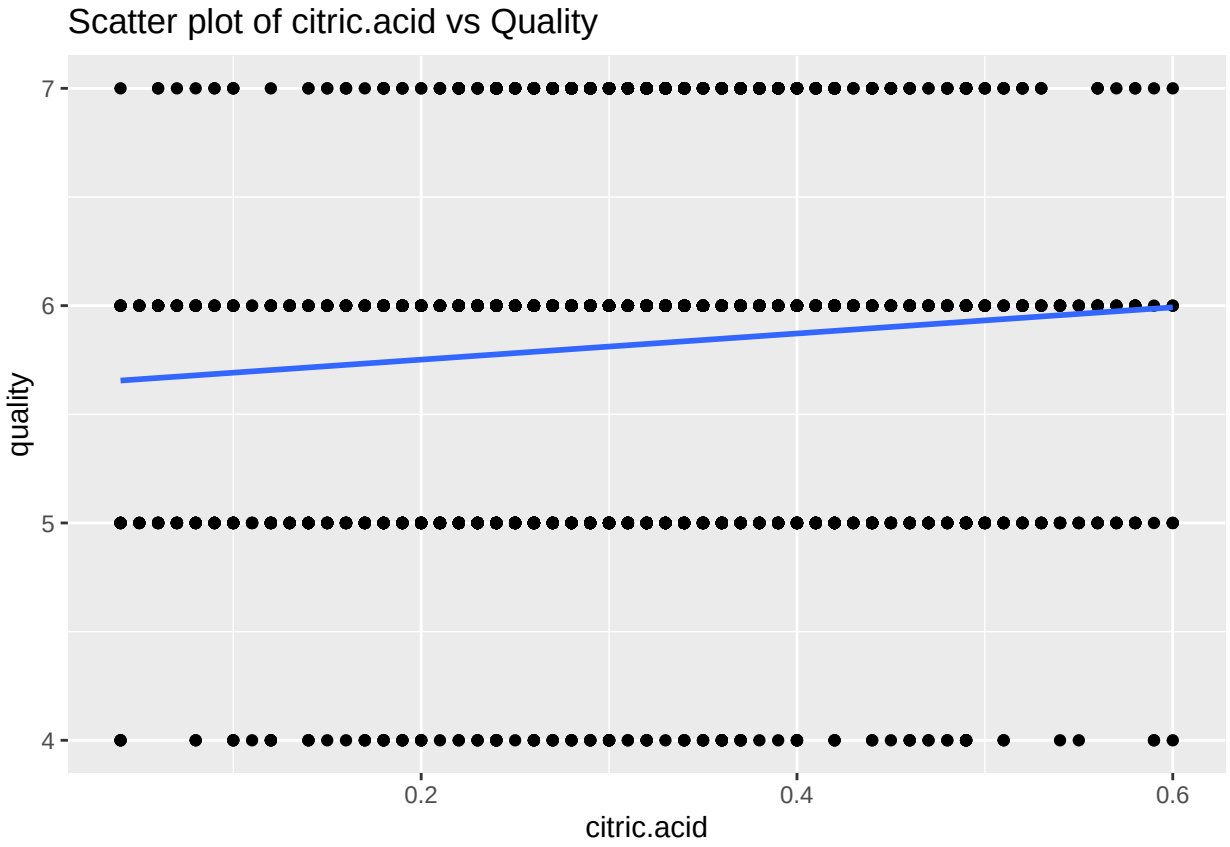
```r
confidence_int <- rbind(
confint(final_model, 'volatile.acidity', level = 0.95),
confint(final_model, 'citric.acid', level = 0.95),
```

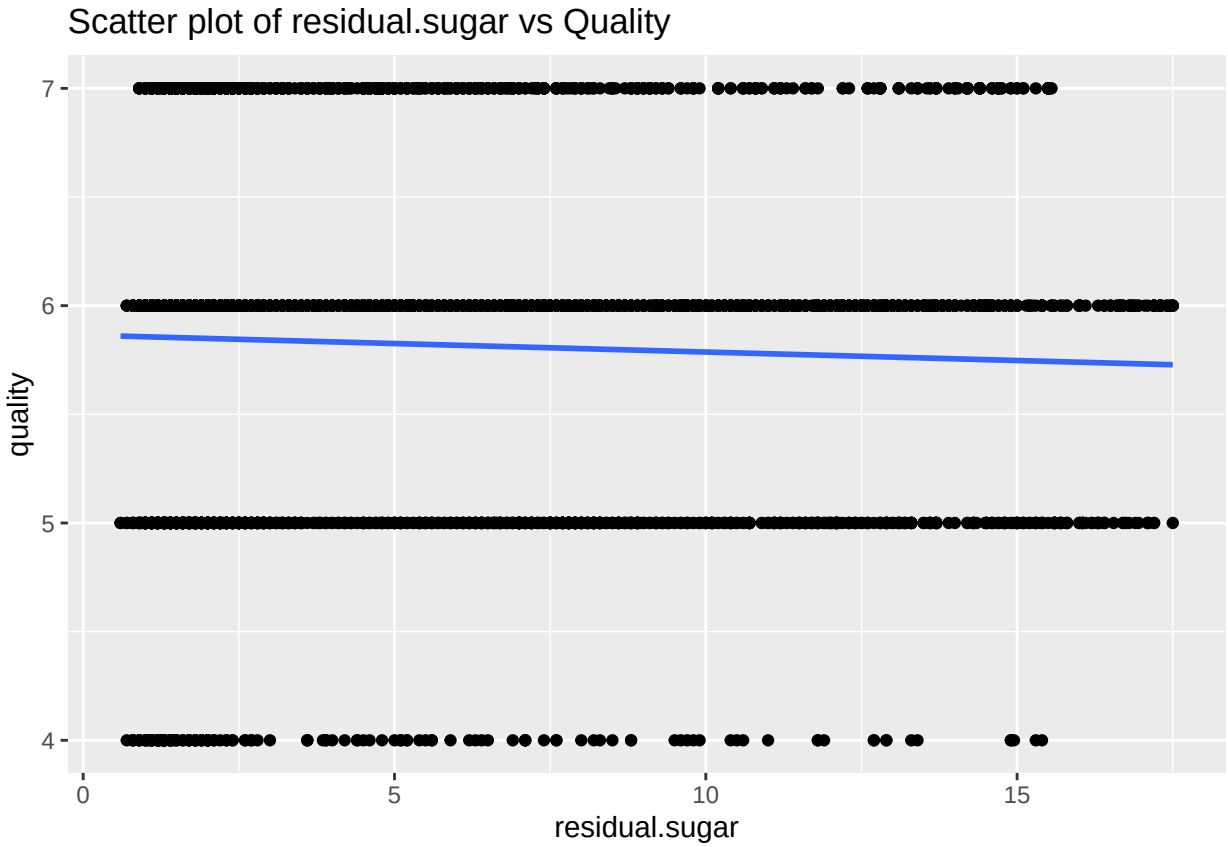
```
confint(final_model, 'residual.sugar', level = 0.95),
confint(final_model, 'chlorides', level = 0.95),
confint(final_model, 'free.sulfur.dioxide', level = 0.95),
confint(final_model, 'sulphates', level = 0.95),
confint(final_model, 'alcohol', level = 0.95),
confint(final_model, 'colorwhite', level = 0.95)
)

print(confidence_int)
```

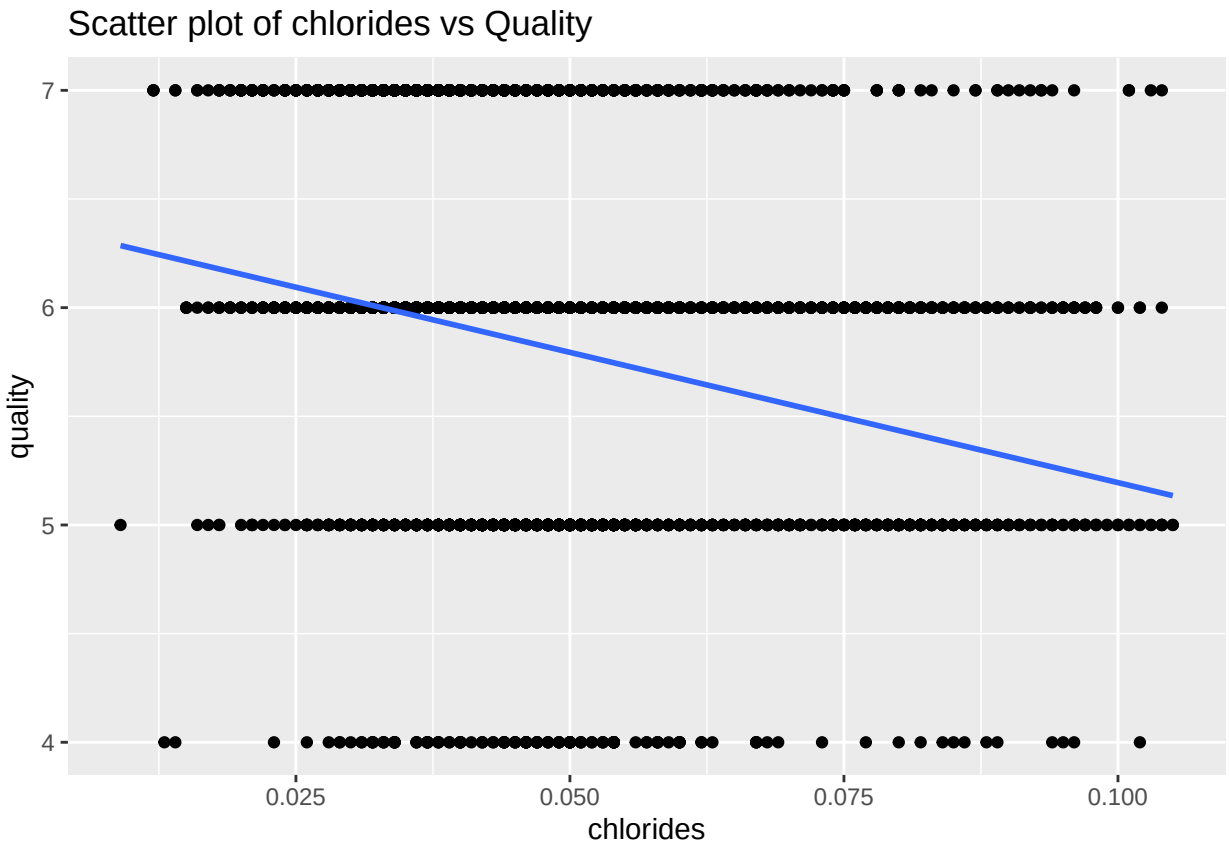
```
##                           2.5 %       97.5 %
## volatile.acidity    -1.943773291 -1.531856727
## citric.acid         -0.397912091 -0.003842550
## residual.sugar       0.018934240  0.028952055
## chlorides           -6.286297314 -2.648987203
## free.sulfur.dioxide  0.002687521  0.005455498
## sulphates            0.428943779  0.786815498
## alcohol              0.285553261  0.325075381
## colorwhite          -0.455710513 -0.268614256
```

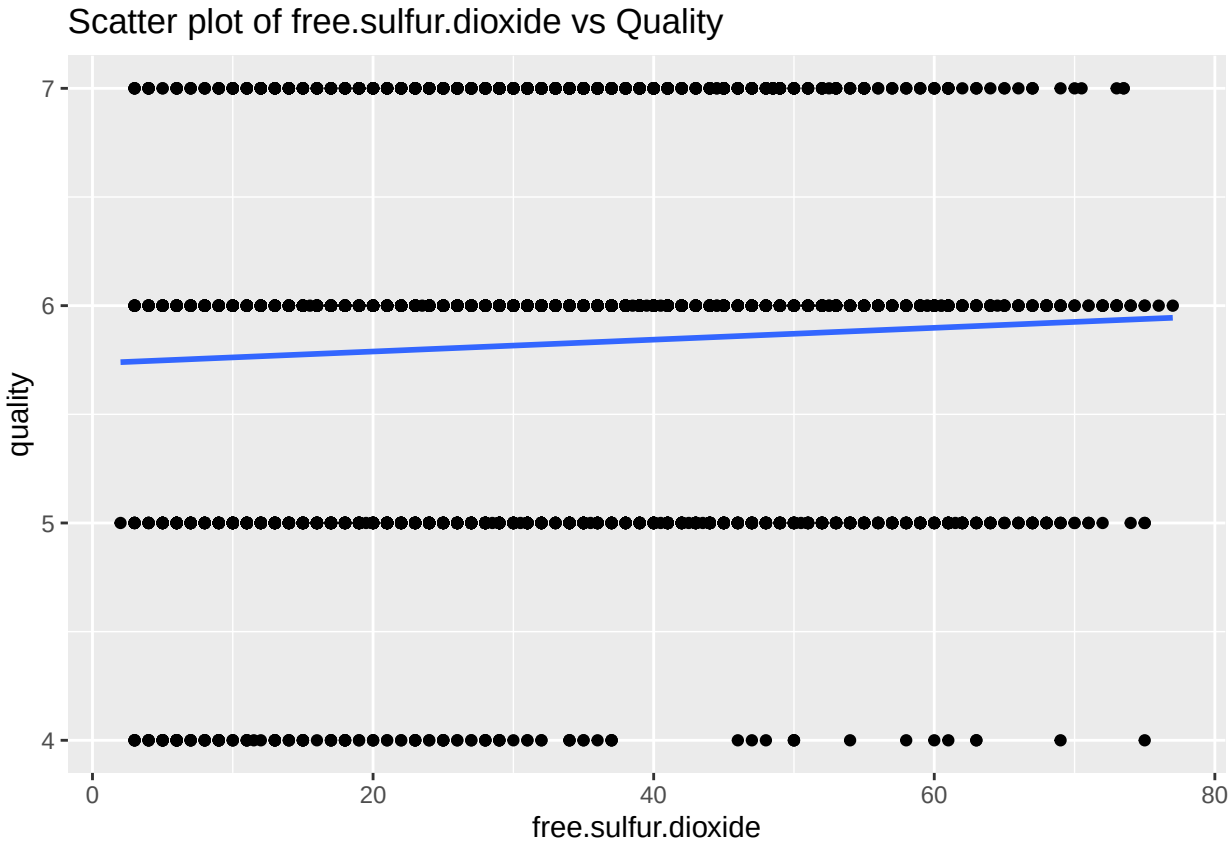
```
final_model_df <- wine_low_cor %>% select(-fixed.acidity)

pred <- setdiff(names(final_model_df), c("quality", "color"))

for (col in pred) {
  p <- ggplot(final_model_df, aes_string(x = col, y = "quality")) +
    geom_point() +
    geom_smooth(method = "lm", se = FALSE) +
    ggtitle(paste("Scatter plot of", col, "vs Quality"))

  print(p)
}
```

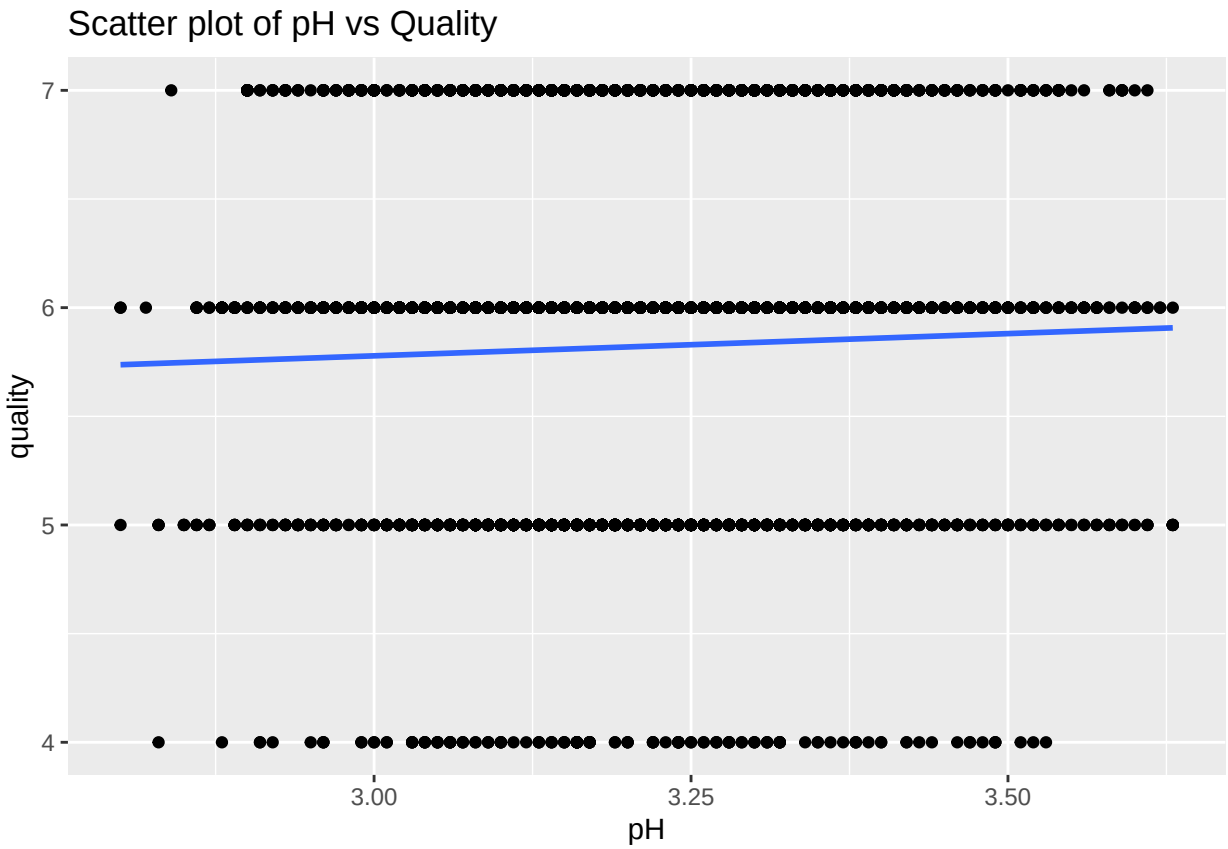

Scatter plot of volatile.acidity vs Quality
quality
7
6
5
4
0.2
0.4
0.6
volatile.acidity

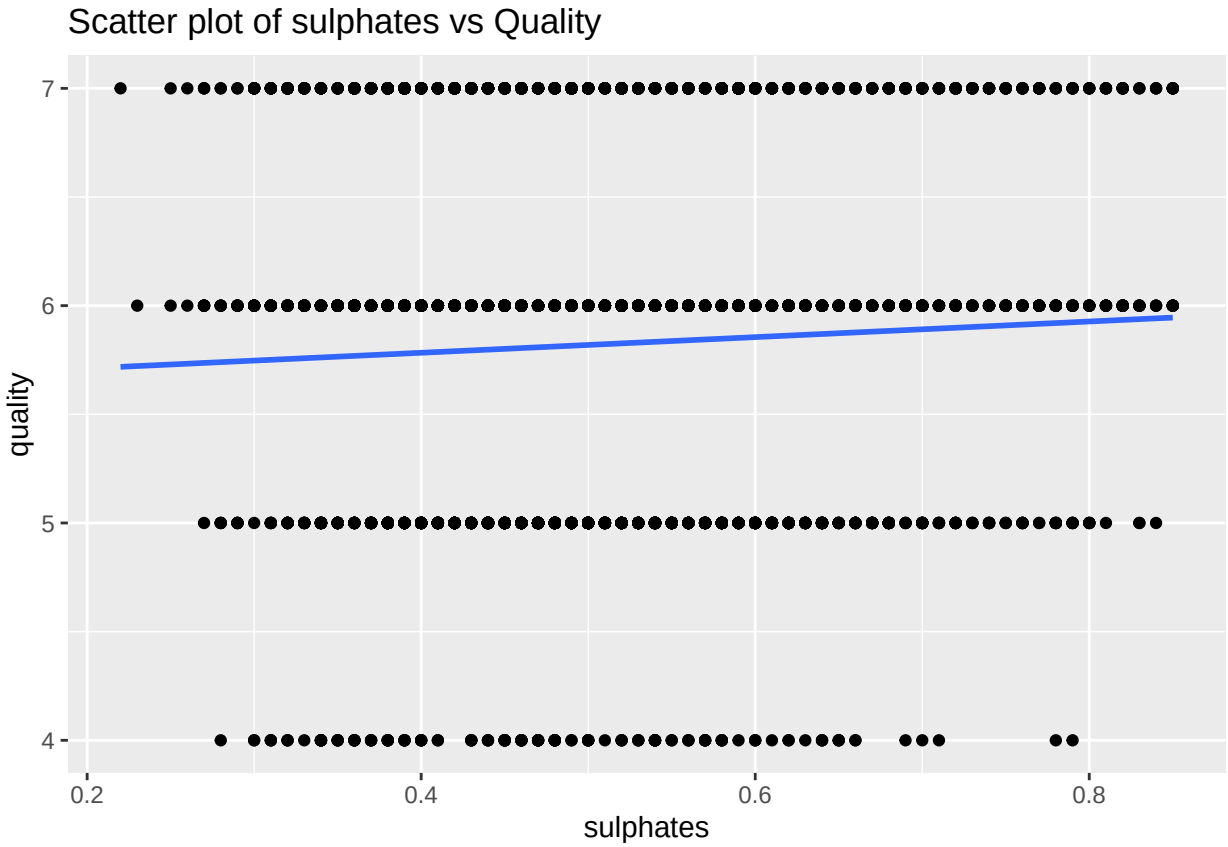

Scatter plot of citric.acid vs Quality
quality
7
6
5
4
0.2
0.4
0.6
citric.acid

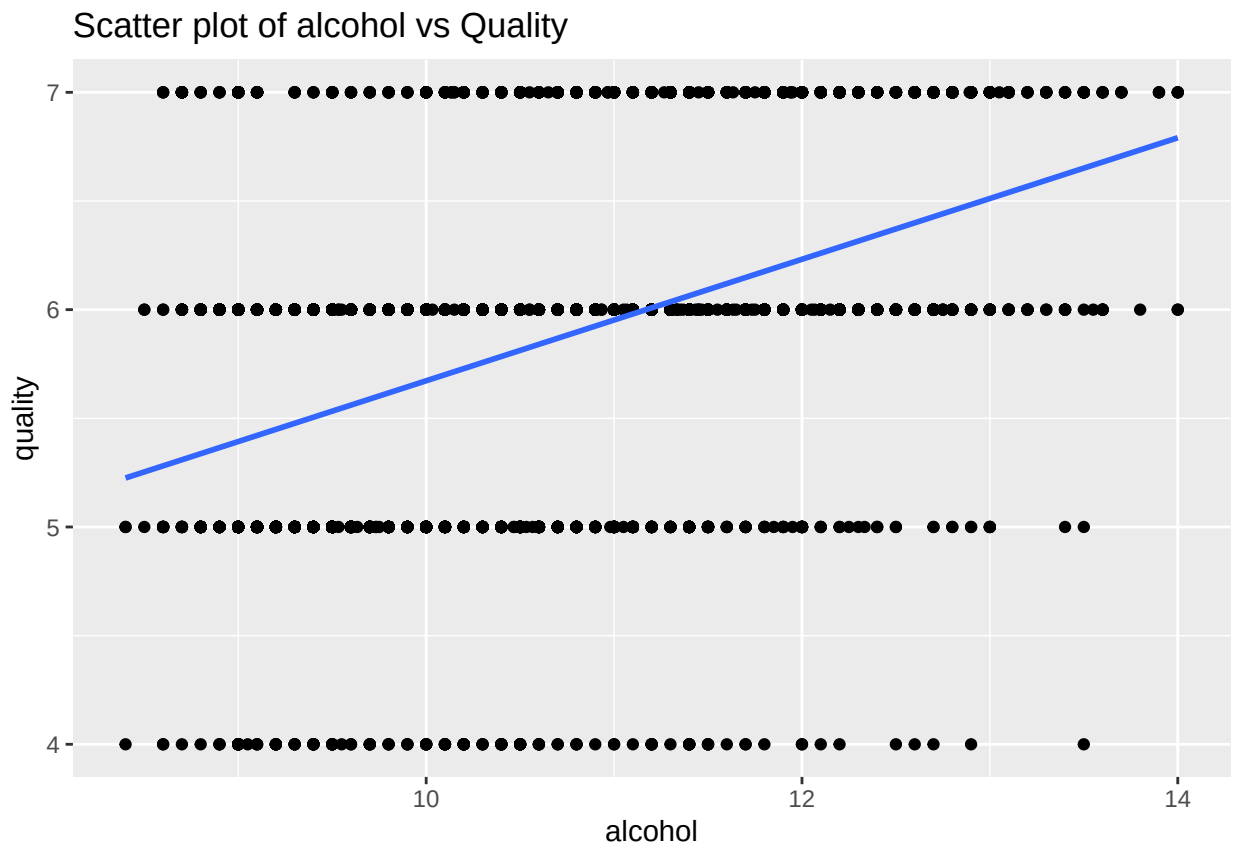

Scatter plot of residual.sugar vs Quality
quality
residual.sugar
7
6
5
4
0
5
10
15


Scatter plot of chlorides vs Quality
quality
7
6
5
4
0.025
0.050
0.075
0.100
chlorides


Scatter plot of free.sulfur.dioxide vs Quality
quality
free.sulfur.dioxide
7
6
5
4
0
20
40
60
80


Scatter plot of pH vs Quality
quality
7
6
5
4
3.00
3.25
3.50
pH


Scatter plot of sulphates vs Quality
quality
sulphates
7
6
5
4
0.2
0.4
0.6
0.8



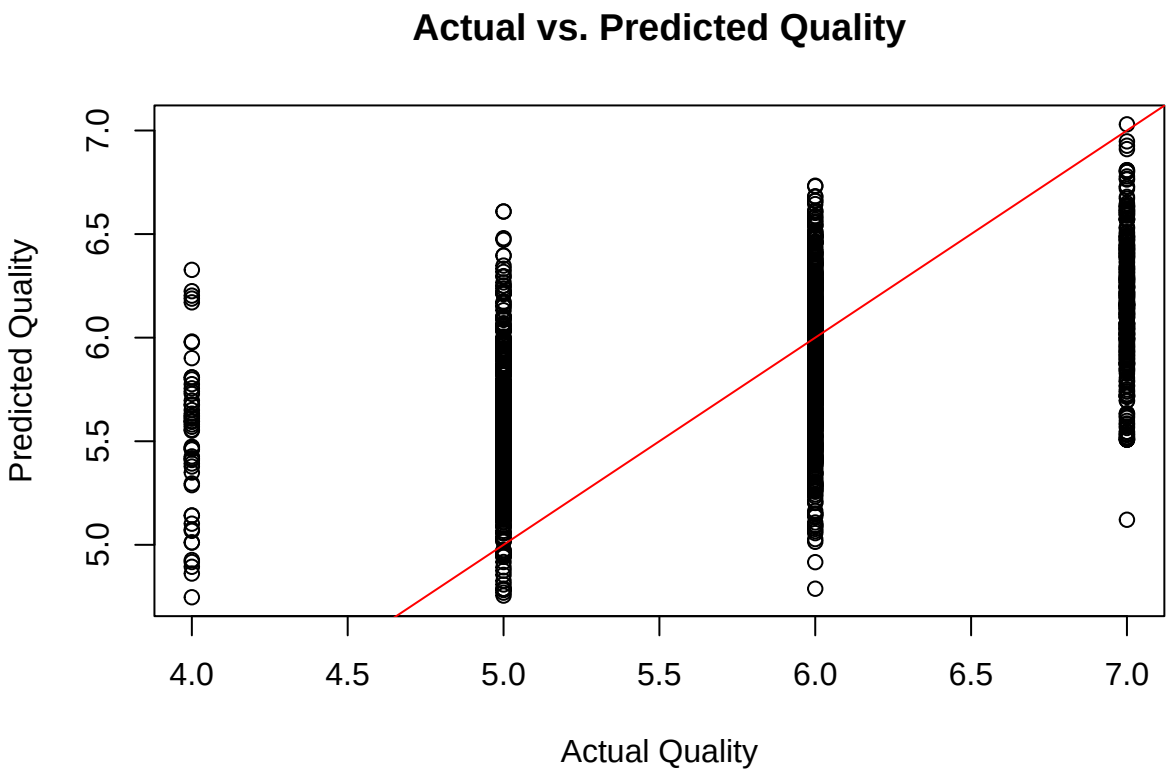

```
predictions <- predict(final_model, newdata = test_data)
cor(predictions, test_data$quality)^2
```

```
## [1] 0.2933634
```

```
rmse <- sqrt(mean((predictions - test_data$quality)^2))
print(rmse)
```

```
## [1] 0.6584743
```

```
plot(test_data$quality, predictions,
     xlab = "Actual Quality", ylab = "Predicted Quality",
     main = "Actual vs. Predicted Quality")
abline(0, 1, col = "red")
```




```
range(test_data$quality)
```

```
## [1] 4 7
```

```
set.seed(123)
wine_low_cor$color <- as.factor(wine_low_cor$color)
trainIndex <- createDataPartition(wine_low_cor$color, p = 0.66, list = FALSE)
train_data <- wine_low_cor[trainIndex, ]
test_data <- wine_low_cor[-trainIndex, ]

rf_model_1 <- randomForest(color ~ fixed.acidity + volatile.acidity + citric.acid +
residual.sugar + chlorides + free.sulfur.dioxide + pH +
sulphates + alcohol, data = train_data, keep.forest = TRUE)

rf_model_2 <- randomForest(quality ~ fixed.acidity + volatile.acidity + citric.acid +
residual.sugar + chlorides + free.sulfur.dioxide + pH +
sulphates + alcohol + color, data = train_data, keep.forest = TRUE)

print(rf_model_1)
```

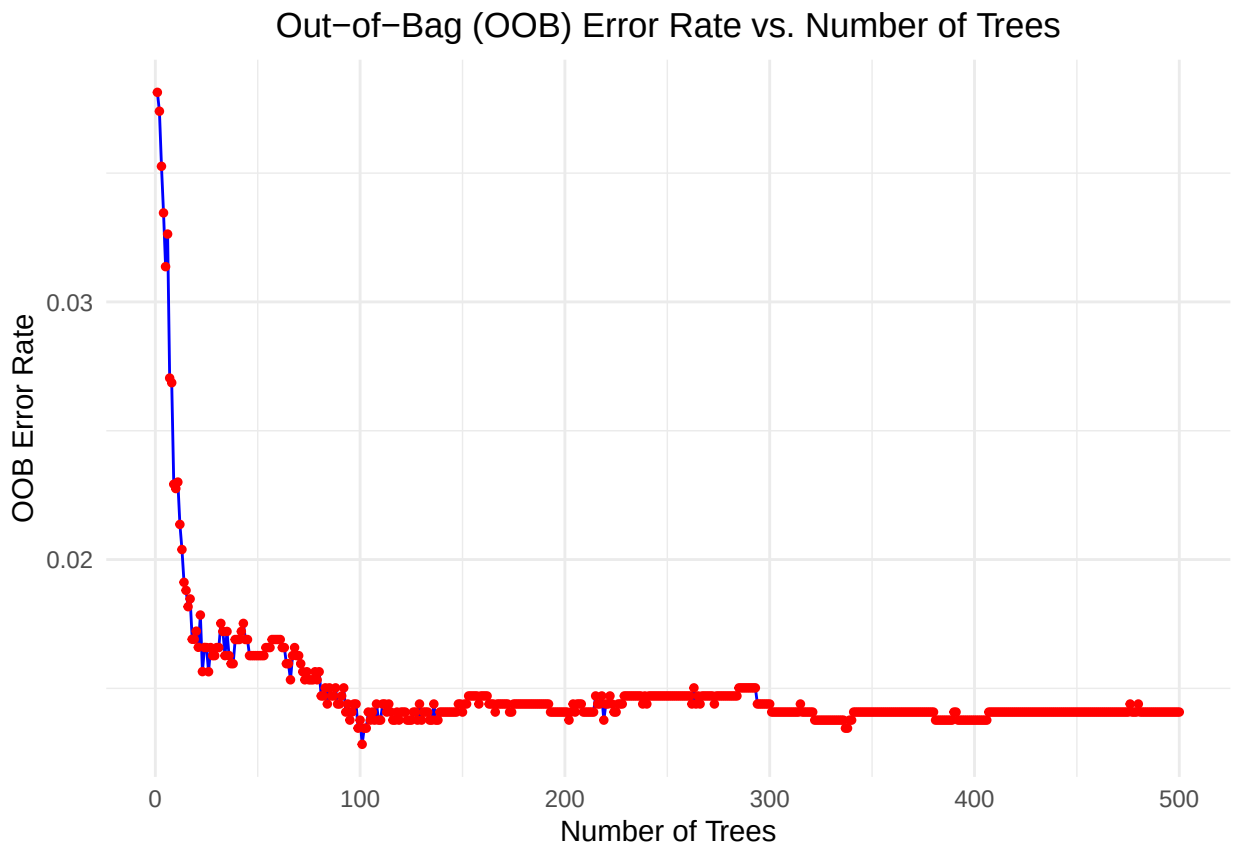
```
##
## Call:
##  randomForest(formula = color ~ fixed.acidity + volatile.acidity +      citric.acid + residual.sugar
##                Type of random forest: classification
```

```
##                       Number of trees: 500
## No. of variables tried at each split: 3
##
##         OOB estimate of  error rate: 1.41%
## Confusion matrix:
##       red white class.error
## red   420    31 0.068736142
## white  14  2731 0.005100182
```

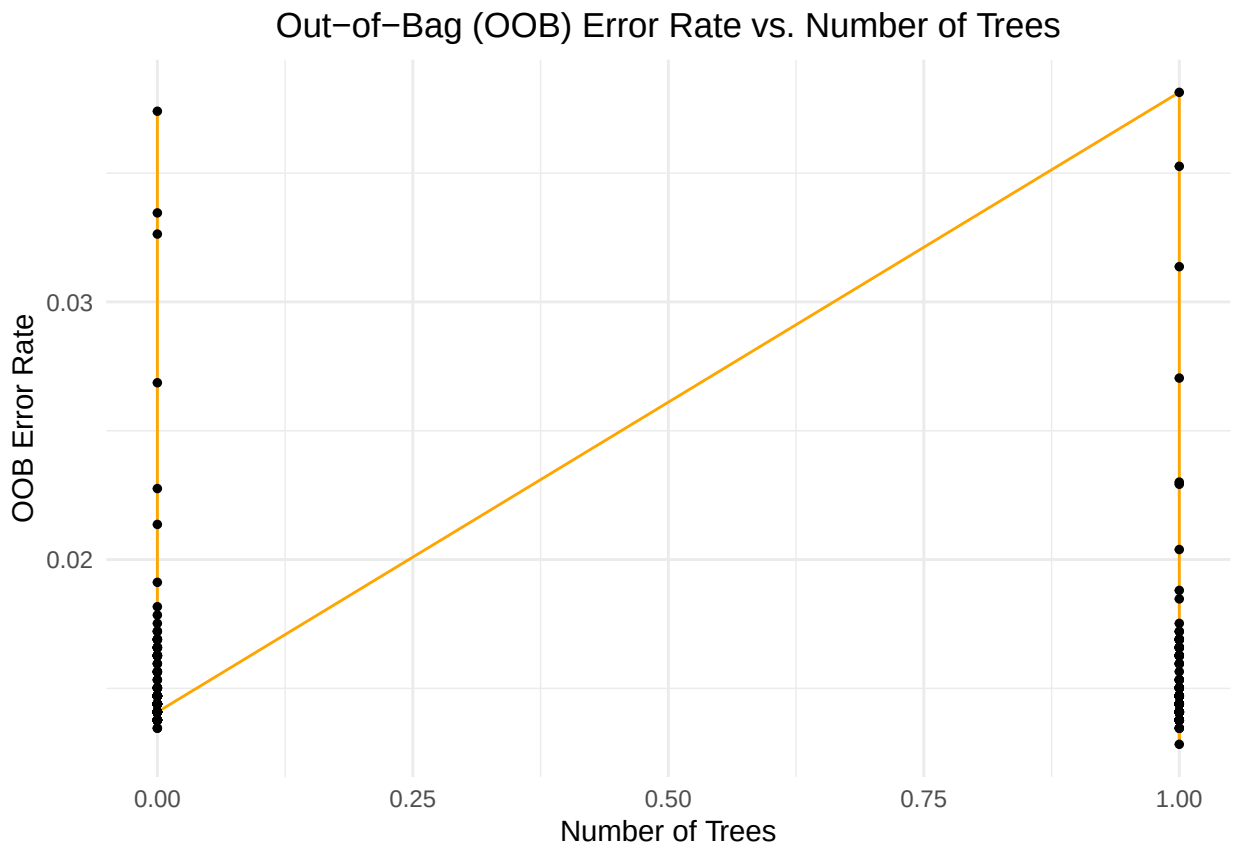
```
print(rf_model_2)
```

```
##
## Call:
##  randomForest(formula = quality ~ fixed.acidity + volatile.acidity +      citric.acid + residual.sug
##                Type of random forest: regression
##                      Number of trees: 500
## No. of variables tried at each split: 3
##
##           Mean of squared residuals: 0.3139498
##                     % Var explained: 46.63
```

```
oob_data_1 <- data.frame(
  Trees = 1:length(rf_model_1$err.rate[, 1]),
  OOB_Error = rf_model_1$err.rate[, 1]
)

oob_data_2 <- data.frame(
  Trees = 1:length(rf_model_2$err.rate[, 1]),
  OOB_Error = rf_model_1$err.rate[, 1]
)

ggplot(oob_data_1, aes(x = Trees, y = OOB_Error)) +
  geom_line(color = "blue") +
  geom_point(color = "red", size = 1) +
  labs(title = "Out-of-Bag (OOB) Error Rate vs. Number of Trees",
       x = "Number of Trees",
       y = "OOB Error Rate") +
  theme_minimal() +
  theme(plot.title = element_text(hjust = 0.5))
```




```
ggplot(oob_data_2, aes(x = Trees, y = OOB_Error)) +
  geom_line(color = "orange") +
  geom_point(color = "black", size = 1) +
  labs(title = "Out-of-Bag (OOB) Error Rate vs. Number of Trees",
       x = "Number of Trees",
       y = "OOB Error Rate") +
  theme_minimal() +
  theme(plot.title = element_text(hjust = 0.5))
```




```
rf_pred_color <- predict(rf_model_1, test_data)
confusionMatrix(factor(rf_pred_color), factor(test_data$color))
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  red white
##      red    219     7
##      white   12  1406
##
##                Accuracy : 0.9884
##                  95% CI : (0.982, 0.993)
##     No Information Rate : 0.8595
##     P-Value [Acc > NIR] : <2e-16
##
##                   Kappa : 0.9517
##
##  Mcnemar's Test P-Value : 0.3588
##
##             Sensitivity : 0.9481
##             Specificity : 0.9950
##          Pos Pred Value : 0.9690
##          Neg Pred Value : 0.9915
##              Prevalence : 0.1405
##          Detection Rate : 0.1332
```

```
##    Detection Prevalence : 0.1375
##       Balanced Accuracy : 0.9715
##
##        'Positive' Class : red
##
```

```
rf_pred_quality <- predict(rf_model_2, test_data)
```

```
#Conclusion
#In the linear regression model the correlation between the predicted and actual values is 0.2933, whic

#The random forest model performs exceptionally well, with an overall accuracy of about 98.85% when pre
```

```
# References
#https://yihui.org/knitr/options/
#https://www.datacamp.com/doc/r/merging
#https://www.datacamp.com/doc/r/category/statistics
#https://www.datacamp.com/doc/r/graphics-with-ggplot2
#https://www.geeksforgeeks.org/how-to-create-and-interpret-pairs-plots-in-r/
#https://www.geeksforgeeks.org/how-to-shuffle-a-dataframe-in-r-by-rows/
#https://www.geeksforgeeks.org/create-interactive-ggplot2-graphs-with-plotly-in-r/
#https://www.sthda.com/english/wiki/visualize-correlation-matrix-using-correlogram
#https://www.rdocumentation.org/packages/corrplot/versions/0.95/topics/corrplot
#P. Cortez, A. Cerdeira, Fernando Almeida, Telmo Matos, J. Reis, 2009, Modeling wine preferences by dat
#Chao Ye, Kevin W. Li, Guozhu Jia, A new red wine prediction framework using machine learning, Journal
#D. Angus, 2019, Modeling Wine Quality from Physicochemical Properties, https://www.semanticscholar.org
#https://www.rdocumentation.org/packages/car/versions/3.1-3/topics/Predict
#https://stackoverflow.com/questions/54977780/how-to-plot-an-oob-error-vs-the-number-of-trees-in-random
#https://statisticsbyjim.com/regression/root-mean-square-error-rmse/
#https://www.geeksforgeeks.org/how-to-calculate-the-oob-of-random-forest-in-r/
#https://www.researchgate.net/figure/Random-Forest-plot-of-classification-error-versus-number-of-trees-
```