ANALYSIS OF RED WINE QUALITY USING PHYSICOCHEMICAL PROPERTIES

Group Information:

Sai Teja Reddy Konala, Sri Harshini Puduri

Section 1: Introduction

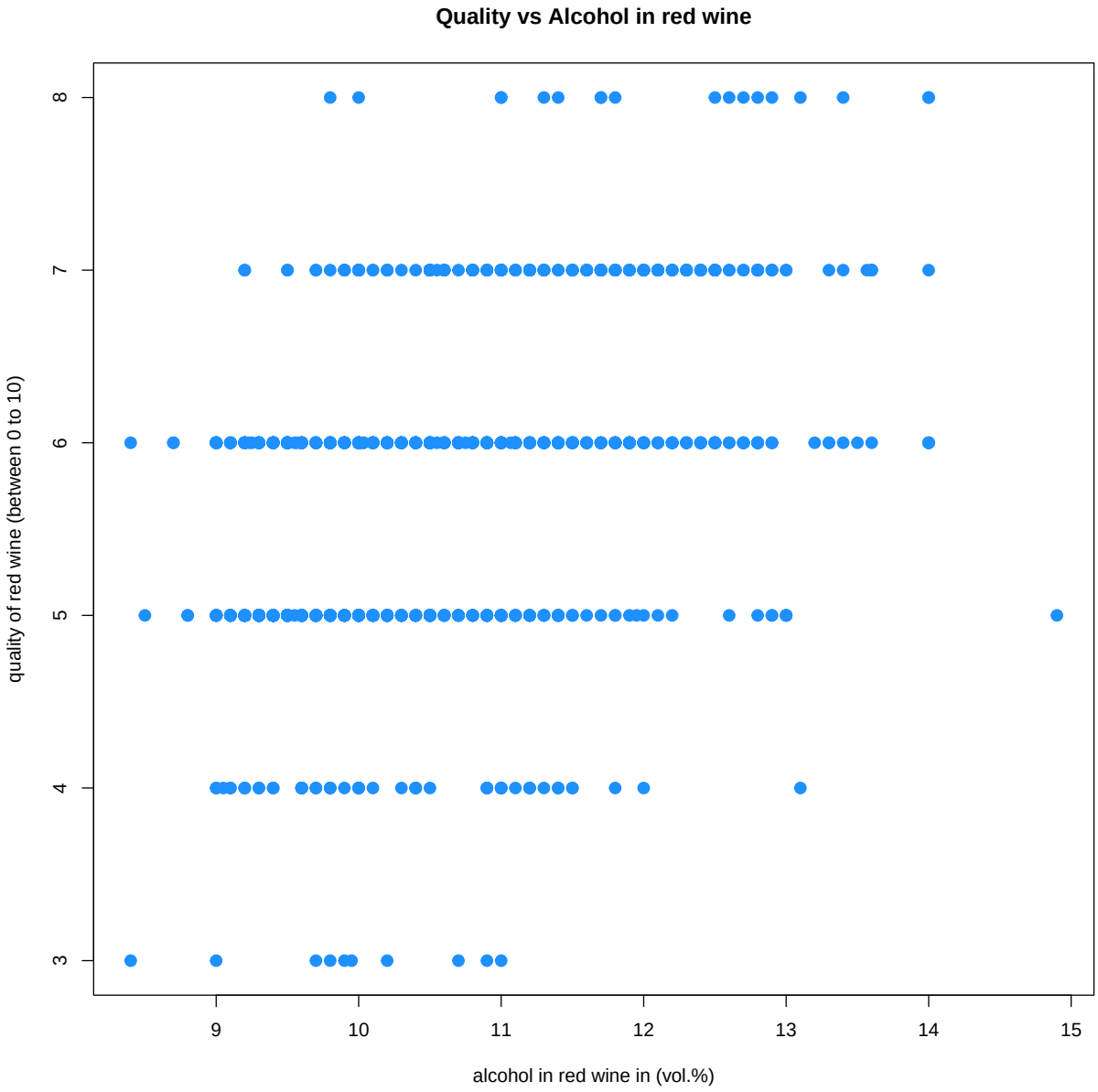
The idea of this project is to examine the correlation between the physicochemical properties and sensory evaluations (quality) of red vinho verde wine from Portugal and to find which physicochemical attributes are important when modeling the 'quality' of wine as the response attribute. The hypotheses is that not every predictor variable has a significant role in predicting the quality of the red wine. So, there might be some predictors that are more significant than the others in determining the quality of the red wine. This is just an assumption which should be tested out.

For this project, we will use the `winequality-red` data set from UC Irvine Machine Learning Repository (Cortez,Paulo, Cerdeira,A., Almeida,F., Matos,T., and Reis,J.. (2009). Wine Quality. UCI Machine Learning Repository. https://doi.org/10.24432/C56S3T.). The dataset in focus involves red variant of the Portuguese "Vinho Verde" wine, containing only physicochemical attributes as inputs and sensory evaluation attribute `quality` as output, excluding specific details such as grape types, wine brands, or prices. This dataset was created by Paulo Cortez and derived when 1599 types of wine were tasted by wine experts and the wine was given a grade quality ranging from 0(worst) to 10(best).

The physicochemical variables in the observations are fixed acidity, volatile acidity, citric acid, residual sugar, chlorides, total sulphur dioxide, free sulphur dioxide, density, pH, sulphates, and alcohol. We will use these physiochemical attributes as the predictor variables and `quality` as the response variable. Some of the variable names from the original dataset have been renamed. The full red wine dataset contains 1599 observations and the following 12 variables (including 1 response variable). Their interpretation and units of measurement are given below:

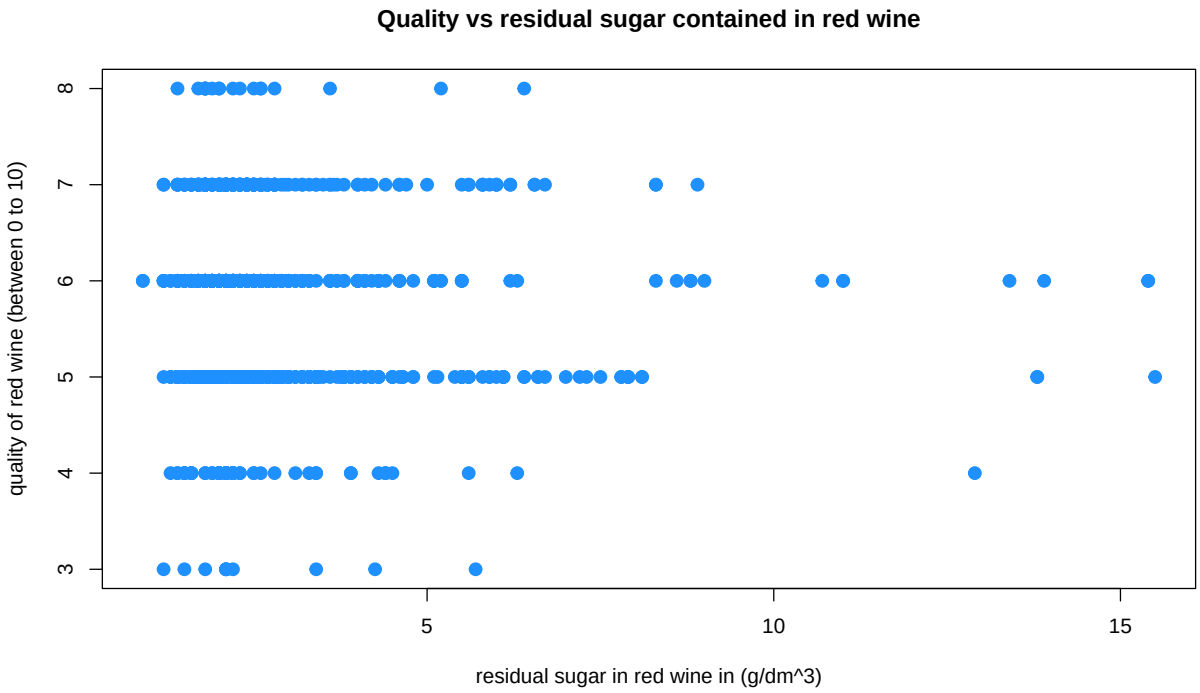
- `FA`: fixed acidity of red wine in $(g(tartaric\ acid)/dm^3)$
- `VA`: volatile acidity of red wine in $(g(acetic\ acid)/dm^3)$
- `CA`: citric acid contained in red wine in (g/dm^3)
- `RS`: residual sugar contained in red wine in (g/dm^3)
- `CL`: chlorides contained in red wine in $(g(sodium\ chloride)/dm^3)$
- `FSO2`: free sulfur dioxide contained in red wine in (mg/dm^3)
- `TSO2`: total sulfur dioxide contained in red wine in (mg/dm^3)
- `DEN`: density of red wine in (g/cm^3)
- `pH`: power of hydrogen (ph level) of red wine on a ph scale of 0 to 14
- `SUL`: sulphates contained in red wine in $(g(pottasium\ sulphate)/dm^3)$
- `ALC`: alcohol contained in red wine in $(vol.\%)$
- `quality`: quality of red wine (between 0 to 10). This is the response variable.

Scatter plot graph between quality and ALC (alcohol in red wine). Because, alcohol is an important metric in alcoholic beverages like red wine.



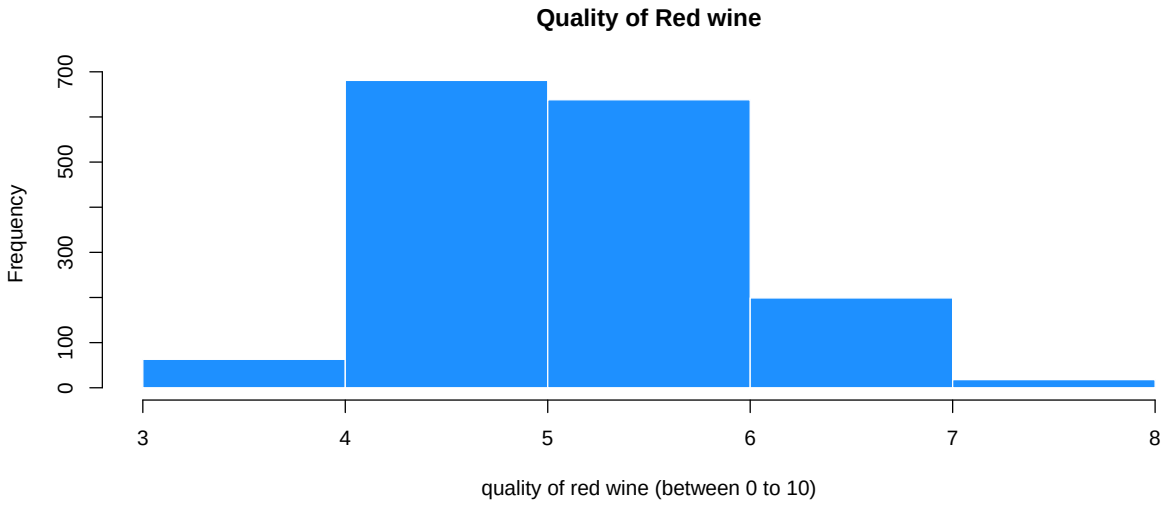
The scatterplot between quality and ALC (alcohol in red wine) seems to have a linear relationship from the scatterplot graph. However, we will test out when we perform regression analysis and finally choosing a relevant model to the given dataset.

Scatter plot graph between quality and RA (residual sugar in red wine).



The scatterplot between quality and RA doesn't seems to have a linear relationship from the scatterplot graph. However, we will test out when we perform regression analysis.

Histogram for the predictor variable which is quality (quality of red wine between 0 to 10.

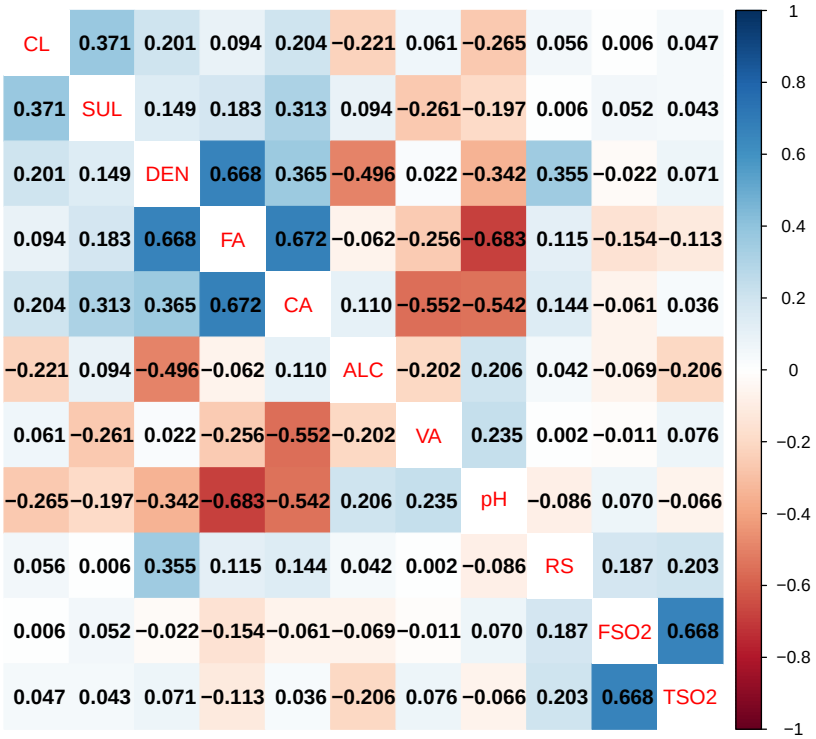


From the histogram, we see that the quality of red wine is mostly between 4-6 and only some observations have a quality of red wine to be less than 4 and more than 7.

Section 2: Regression Analysis

Firstly, collinearity between variables needs to be checked. So let us assume that the variables have high correlation if the pairwise correlation value is greater than 0.6.

The correlation matrix is as follows:



We can see that (DEN, FA), (FA, CA), (FA, pH) and (FSO2, TSO2) appear to be correlated. Let's also check for condition number and VIF's for the model.

```
##    Eigenvalue Condition Index
## 1     10.5269          1.0000
## 2      0.4947          4.6129
## 3      0.3542          5.4515
## 4      0.2153          6.9931
## 5      0.1833          7.5786
## 6      0.1112          9.7317
## 7      0.0579         13.4816
## 8      0.0308         18.5010
## 9      0.0199         23.0260
## 10     0.0054         44.0351
## 11     0.0006        135.3347
## 12     0.0000       6053.4834
```

Here, collinearity exists because the condition number is 6053.4834, which is greater than 30. So, lets also check for VIF's.

```
##        FA        VA        CA        RS        CL      FSO2      TSO2       DEN
## 7.767512 1.789390 3.128022 1.702588 1.481932 1.963019 2.186813 6.343760
##        pH       SUL       ALC
## 3.329732 1.429434 3.031160
```

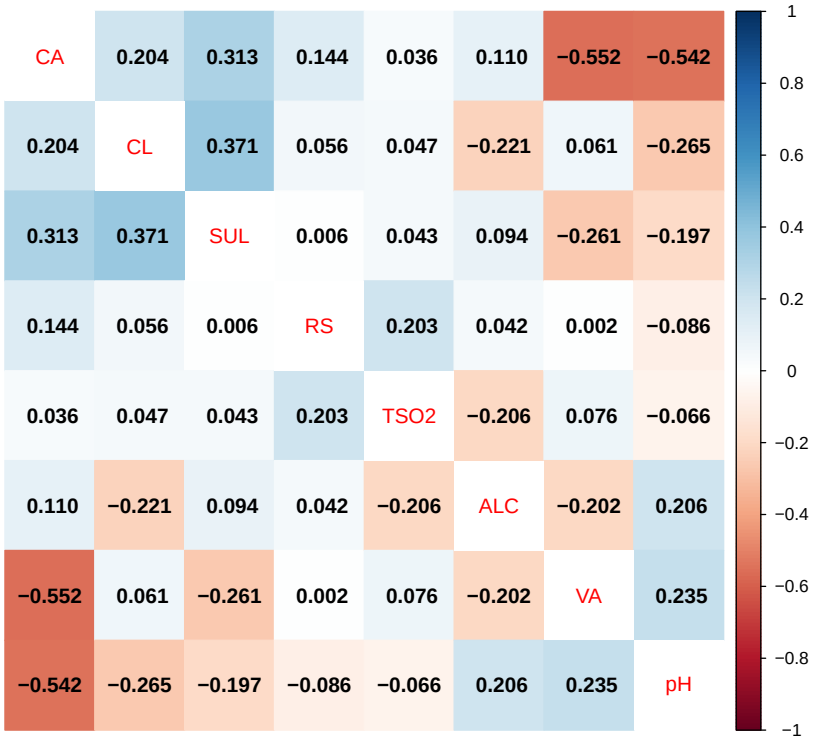
The VIF of FA and DEN is greater than 5. Lets remove FA, DEN and FSO2 from the model to reduce collinearity. Now lets check the VIF's of this new model.

```
##        VA        CA        RS        CL      TSO2        pH       SUL       ALC
## 1.607906 2.134107 1.082538 1.368659 1.109980 1.598680 1.332670 1.276332
```

With FA (fixed acidity), FSO2 (Free sulpher dioxide), DEN (Density) removed, the VIF's of the variables are less than 5. Now let's check for orthagonality for the new model (model_collinearity1):

```
##          VA          CA          RS          CL        TSO2          pH         SUL
## 0.37807326 0.53141998 0.07624511 0.26935808 0.09908306 0.37448413 0.24962668
##         ALC
## 0.21650458
```

RS, CL, SUL, ALC, TSO2 is orthogonal from the new model ($R_k^2 < 0.3$). So, overall removing those predictors seemed to reduce the collinearity of the overall model. Let's check this once again with correlation matrix:



This looks much better than the initial correlation matrix.

Let's check for any outliers in the observations:

```
## named integer(0)
```

There are no outliers in the observations.

Now let's search for any highly influential observations:

```
## [1] 100
```

There are 100 highly influential observations in the dataset.

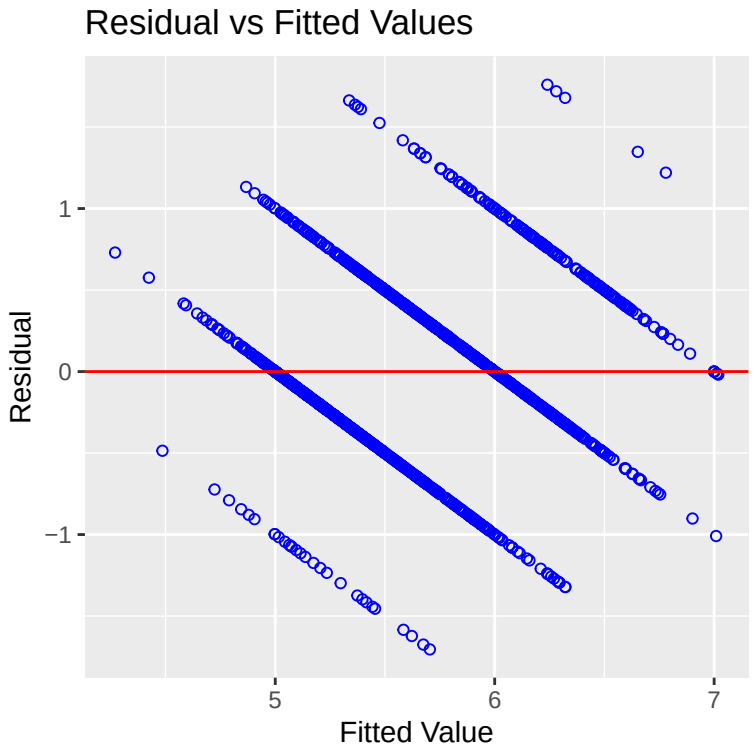
Lets remove these observations from the dataset in the new model.

```
##  (Intercept)           VA           CA           RS           CL         FSO2
## -1.044443232 -0.988011458 -0.308058624  0.017047974 -2.110233537  0.003472691
##         TSO2          DEN           pH          SUL          ALC
## -0.003563788  5.321078137 -0.501995319  1.151032867  0.303129850
```

The coefficients of old model (model_collinearity1):

```
##  (Intercept)           VA           CA           RS           CL         TSO2
##  4.603517188 -1.122819083 -0.181993663  0.011960648 -1.932354242 -0.002419632
##           pH          SUL          ALC
## -0.522499315  0.906094720  0.293511253
```

There seems to be change in the coefficient values with the highly influential observations removed, so lets use the model with the highly influential observations removed for now (model_collinearity2). Now, lets look at the fitted vs residual graph for this model.



From the graph, it seems like both the normality and constant varaince assumptions are being violated. So, to confirm this we use breusch-pagan test for constant variance and shaprio wilk test for normality.

```
##
##  studentized Breusch-Pagan test
##
## data:  model_collinearity2
## BP = 39.379, df = 10, p-value = 2.179e-05
```

The test stastic for breush-pagan test is 39.379 and the p-value is $2.179 * 10^{-5}$. Let's assume significane level (α) to be 0.05. The p-value is lesser than significance level, so constant variance assumption is violated.

```
##
##  Shapiro-Wilk normality test
##
## data:  resid(model_collinearity2)
## W = 0.99592, p-value = 0.0004671
```

The test stastic for shaprio-wilk test is 0.99592 and the p-value is 0.0004671. Let's assume significance level (α) to be 0.05. The p-value is lesser than significance level, so normality assumption is violated.

First, we will use bootstrap for OLS on the model with collinear predictors removed (model_collinearity1). We are using 1500 boostrap samples with a seed of 42 for reproducability.

```
## Bootstrap percent confidence intervals
##
##                 Estimate        2.5 %       97.5 %
## (Intercept)  4.603517188  3.693168963  5.463968506
## VA          -1.122819083 -1.352611313 -0.902065274
## CA          -0.181993663 -0.424034711  0.054135910
## RS           0.011960648 -0.010455568  0.035332692
## CL          -1.932354242 -2.698798067 -1.132000920
## TSO2        -0.002419632 -0.003459375 -0.001369331
## pH          -0.522499315 -0.778472606 -0.254910696
## SUL          0.906094720  0.686219726  1.118608124
## ALC          0.293511253  0.258651825  0.328284874
```

Let's also find the confidence interval for the same model without using bootstrap:

```
##                   2.5 %       97.5 %
## (Intercept)  3.69956483  5.507469548
## VA          -1.34816103 -0.897477135
## CA          -0.42062427  0.056636946
## RS          -0.01152130  0.035442600
## CL          -2.72331763 -1.141390860
## TSO2        -0.00343877 -0.001400494
## pH          -0.78310335 -0.261895281
## SUL          0.68938258  1.122806862
## ALC          0.25977712  0.327245384
```

In this case, the intervals are relatively similar in width and location and there would be no change in our hypothesis tests. So lets perform LAD regression.

```
##
## Call: rq(formula = quality ~ . - FA - DEN - FSO2, data = rwq)
##
## tau: [1] 0.5
##
## Coefficients:
##             Value     Std. Error t value   Pr(>|t|)
## (Intercept)   3.02742   0.32638    9.27572   0.00000
## VA           -0.78762   0.07309  -10.77526   0.00000
## CA           -0.07856   0.08405   -0.93472   0.35008
## RS            0.03569   0.01089    3.27903   0.00106
## CL           -2.11908   0.38077   -5.56518   0.00000
## TSO2         -0.00253   0.00033   -7.60881   0.00000
## pH           -0.33990   0.11189   -3.03788   0.00242
## SUL           1.09174   0.13451    8.11646   0.00000
## ALC           0.34785   0.02011   17.29535   0.00000
```

The confidence intervals for LAD are:

```
##              Coefficient   LowerBound   UpperBound
## (Intercept)  3.027424569  2.387728103  3.667121036
## VA          -0.787615146 -0.930878220 -0.644352072
## CA          -0.078561020 -0.243292168  0.086170127
## RS           0.035693237  0.014358447  0.057028028
## CL          -2.119079860 -2.865384477 -1.372775242
## TSO2        -0.002526401 -0.003177181 -0.001875622
## pH          -0.339896234 -0.559188555 -0.120603914
## SUL          1.091741202  0.828107528  1.355374876
## ALC          0.347847052  0.308427920  0.387266184
```

Let's use huber's method and use bootstrap confidence intervals with R to be 1500.

```
##
## Call: rlm(formula = quality ~ . - FA - DEN - FSO2, data = rwq, maxit = 100)
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2.58048 -0.36870 -0.05986  0.43587  1.97733
##
## Coefficients:
##             Value   Std. Error t value
## (Intercept)  4.2159  0.4533     9.3003
## VA          -1.0308  0.1130    -9.1215
## CA          -0.1607  0.1197    -1.3433
## RS           0.0247  0.0118     2.0962
## CL          -1.7815  0.3966    -4.4913
## TSO2        -0.0027  0.0005    -5.3617
## pH          -0.4480  0.1307    -3.4282
## SUL          0.9287  0.1087     8.5457
## ALC          0.2984  0.0169    17.6386
##
## Residual standard error: 0.59 on 1590 degrees of freedom
```

The confidence intervals for huber's method are:

```
## Bootstrap percent confidence intervals
##
##                 Estimate        2.5 %       97.5 %
## (Intercept)  4.215919219  3.332028913  5.101810626
## VA          -1.030755799 -1.246047327 -0.812591092
## CA          -0.160748212 -0.391441895  0.066252224
## RS           0.024684465  0.002610391  0.046483687
## CL          -1.781483378 -2.560764566 -0.999166031
## TSO2        -0.002740194 -0.003748969 -0.001719665
## pH          -0.448024191 -0.707729776 -0.179000604
## SUL          0.928713635  0.719218880  1.137505757
## ALC          0.298389198  0.263401951  0.332686282
```

Now, lets remove the highly influential observations removed and fit OLS model:

```
##
## Call:
## lm(formula = quality ~ . - FA - FSO2 - DEN, data = rwq, subset = noninfluential_ids)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.71362 -0.36190 -0.05497  0.39191  1.74714
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  4.244118   0.419199  10.124  < 2e-16 ***
## VA          -1.003869   0.110112  -9.117  < 2e-16 ***
## CA          -0.315230   0.111584  -2.825  0.00479 **
## RS           0.022093   0.012534   1.763  0.07816 .
## CL          -2.142128   0.403119  -5.314 1.24e-07 ***
## TSO2        -0.002809   0.000485  -5.793 8.40e-09 ***
## pH          -0.486156   0.120321  -4.041 5.61e-05 ***
## SUL          1.177825   0.112687  10.452  < 2e-16 ***
## ALC          0.299901   0.015851  18.920  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.5542 on 1490 degrees of freedom
## Multiple R-squared:  0.4286, Adjusted R-squared:  0.4255
## F-statistic: 139.7 on 8 and 1490 DF,  p-value: < 2.2e-16
```

Removing highly influential observations does seem to change the estimate of parameter CA is significant, when its not significant in previous OLS model with the highly influential observations.

Now let's fit LAD without the highly influential observations:

```
##
## Call: rq(formula = quality ~ . - FA - DEN - FSO2, data = rwq, subset = noninfluential_ids)
##
## tau: [1] 0.5
##
```

```
## Coefficients:
##             Value    Std. Error t value  Pr(>|t|)
## (Intercept)  2.93094  0.38895    7.53548  0.00000
## VA          -0.76191  0.08090   -9.41750  0.00000
## CA          -0.15416  0.11063   -1.39346  0.16369
## RS           0.03476  0.01106    3.14299  0.00171
## CL          -2.28841  0.36099   -6.33933  0.00000
## TSO2        -0.00253  0.00035   -7.21332  0.00000
## pH          -0.34348  0.11441   -3.00207  0.00273
## SUL          1.22753  0.14049    8.73728  0.00000
## ALC          0.35322  0.01928   18.32240  0.00000
```

The confidence intervals are:

```
##              Coefficient   LowerBound   UpperBound
## (Intercept)  2.930942145  2.168609268  3.693275021
## VA          -0.761907912 -0.920475639 -0.603340185
## CA          -0.154164870 -0.371004832  0.062675093
## RS           0.034755720  0.013082088  0.056429352
## CL          -2.288414131 -2.995935703 -1.580892559
## TSO2        -0.002534077 -0.003222623 -0.001845531
## pH          -0.343476302 -0.567722214 -0.119230391
## SUL          1.227534643  0.952171558  1.502897728
## ALC          0.353216559  0.315432668  0.391000451
```

Here the significance of variables remains the same even with highly influential observations removed, So for LAD, we can use the model with highly influential observations present.

Now, let's remove highly influential observations from the dataset itself and perform Huber's method:

```
##
## Call: rlm(formula = quality ~ . - FA - DEN - FSO2, data = rwq[-c(14,
##     34, 44, 46, 80, 87, 92, 93, 132, 133, 143, 145, 152, 162,
##     170, 199, 200, 235, 240, 279, 282, 292, 354, 365, 367, 379,
##     391, 410, 441, 443, 452, 456, 460, 481, 496, 499, 518, 567,
##     568, 589, 634, 639, 646, 648, 653, 660, 673, 691, 701, 724,
##     724, 755, 777, 814, 829, 833, 862, 877, 900, 904, 905, 938,
##     999, 1044, 1062, 1078, 1079, 1080, 1082, 1091, 1112, 1121,
##     1125, 1177, 1187, 1234, 1236, 1240, 1262, 1270, 1277, 1288,
##     1289, 1290, 1300, 1320, 1375, 1404, 1424, 1430, 1435, 1436,
##     1468, 1470, 1479, 1483, 1485, 1506, 1515, 1516), ], maxit = 100)
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.71542 -0.35485 -0.04585  0.39093  1.74222
##
## Coefficients:
##             Value   Std. Error t value
## (Intercept)  4.1100  0.4342     9.4659
## VA          -0.9857  0.1136    -8.6753
## CA          -0.2826  0.1156    -2.4445
## RS           0.0250  0.0130     1.9194
## CL          -2.1320  0.4182    -5.0986
## TSO2        -0.0030  0.0005    -5.9158
## pH          -0.4585  0.1247    -3.6754
```

```
## SUL          1.1693  0.1169    10.0036
## ALC          0.3023  0.0164    18.3872
##
## Residual standard error: 0.5496 on 1491 degrees of freedom
```

The confidence intervals is as follows for huber's method with no influential observations using bootstrap intervals:

```
## Bootstrap bca confidence intervals
##
##                 Estimate         2.5 %      97.5 %
## (Intercept)  4.110016851  3.2319160970  4.99599757
## VA          -0.985724605 -1.1934079609 -0.74625966
## CA          -0.282637897 -0.5078746541 -0.06311672
## RS           0.024953530  0.0004580381  0.05092594
## CL          -2.131993519 -2.9274233124 -1.28653661
## TSO2        -0.002974049 -0.0039375692 -0.00198312
## pH          -0.458483150 -0.7063467800 -0.21051494
## SUL          1.169288468  0.9162117674  1.39006150
## ALC          0.302294977  0.2709265929  0.33440715
```

Here we can see that CA is significant while it wasn't significant before, which is different from the model with Huber's method with no highly influential observations.

Let's create a table with all these models with the intercept and predictor estimates and bold implies that the predictor or intercept is significant according to the test conducted.

Methods	Intercept	VA	CA	RS	CL	TSO2	pH	SUL	ALC
OLS	**4.603**	**-1.122**	-0.181	0.011	**-1.932**	**-0.002**	**-0.522**	**0.906**	**0.293**
OLS[Refit]	**4.244**	**-1.003**	**-0.315**	0.022	**-2.142**	**-0.002**	**-0.486**	**1.177**	**0.299**
LAD	**3.027**	**-0.787**	-0.078	**0.035**	**-2.119**	**-0.002**	**-0.339**	**1.091**	**0.347**
LAD[Refit]	**2.930**	**-0.761**	-0.154	**0.034**	**-2.288**	**-0.002**	**-0.343**	**1.227**	**0.353**
Huber	**4.215**	**-1.030**	-0.160	**0.024**	**-1.781**	**-0.002**	**-0.448**	**0.928**	**0.298**
Huber[Refit]	**4.110**	**-0.985**	**-0.282**	**0.024**	**-2.131**	**-0.002**	**-0.458**	**1.169**	**0.302**

So we should choose LAD model (model_lad) as the final model because the significance didn't change even after removing the highly influential observations. The regression equation is:

$quality_i$ = 3.027 - 0.787 VA_i - 0.0785 CA_i + 0.035 RS_i - 2.119 CL_i - 0.002 $TSO2_i$ - 0.339 pH_i + 1.091 SUL_i + 0.347 ALC_i.

lets do shaprio wilk test to the final model: (model_lad)

```
##
##  Shapiro-Wilk normality test
##
## data:  resid(model_lad)
## W = 0.98674, p-value = 6.066e-11
```

The test statistic is 0.98674 and the p-value is $6.066 * 10^{-11}$. The normality assumption is viloated as the test statistic is less than the significance level of 0.05.

Lets check the breush pagan test:

```
##
##  studentized Breusch-Pagan test
##
## data:  model_lad
## BP = 75.073, df = 8, p-value = 4.77e-13
```

The test statistic is 75.073 and the p-value is $4.77 * 10^{-13}$. As p-value is less than significance level of 0.05, the constant variance assumption is violated for the final model.

Section 3: Discussion

Let's look at the selected final model (model_lad) and the coefficient interpretation:

- Fixed Acidity (FA): Not included in the final model after addressing collinearity.
- Volatile Acidity (VA): A one-unit decrease in volatile acidity corresponds to an increase of approximately 0.787 units in wine quality.
- Citric Acid (CA): A one-unit decrease in citric acid is associated with a decrease of 0.0785 units in wine quality.
- Residual Sugar (RS): An increase of 1 unit in residual sugar leads to an increase of about 0.035 units in wine quality.
- Chlorides (CL): One unit increase in chlorides results in a substantial decrease of 2.119 units in wine quality.
- Total Sulfur Dioxide (TSO2): A reduction of 1 unit in total sulfur dioxide is associated with a decrease of 0.002 units in wine quality.
- pH level (pH): A one-unit decrease in pH is linked to a decrease of approximately 0.339 units in wine quality.
- Sulphates (SUL): An increase of 1 unit in sulphates corresponds to an increase of 1.091 units in wine quality.
- Alcohol (ALC): A one-unit increase in alcohol content leads to an increase of about 0.347 units in wine quality.
- Density (DEN) : Not included in the final model after addressing collinearity.
- Free SO2 (FSO2) : Not included in the final model after addressing collinearity.

The LAD model says that Citric acid level (CA) is not significant and the other predictors VA, RS, CL, TSO2, pH, SUL, ALC are significant (If the confidence interval didn't have 0 in the particular intercept or predictor, it is significant). We can't directly calculate the values of R^2 for the LAD model. The LAD regression model demonstrates statistical significance, and the chosen predictors provide meaningful insights into the physicochemical attributes influencing red wine quality. So, some predictors are significant than other predictors (CA is insignificant according to LAD model). The LAD model also addressed collinearity as collinear predictor were removed from the model(FA, DEN, FSO2) and also maintains robustness against highly influential observations in the data observations because we used LAD, resulting in a somewhat interpretable model.

At the same time, there assumptions might not be true because the LAD model is still violating the constant variance assumption and also the normality assumption. This is checked by using breush-pagan test for constant variance assumption and the p-value is $4.77 * 10^{-13}$ which is less than significance level of 0.05, so the constant variance assumption is violated. The normality assumption is also violated as the p-value of the shapiro-wilk test is $6.066 * 10-11$ which is again less than significance level of 0.05.

So, this prediction can be potentially used for testing out quality of wine from physicochemical properties if the normality and constant variation assumptions were fixed. Some alternative transformations or modeling techniques can be used to address these issues.

Section 4: Limitations

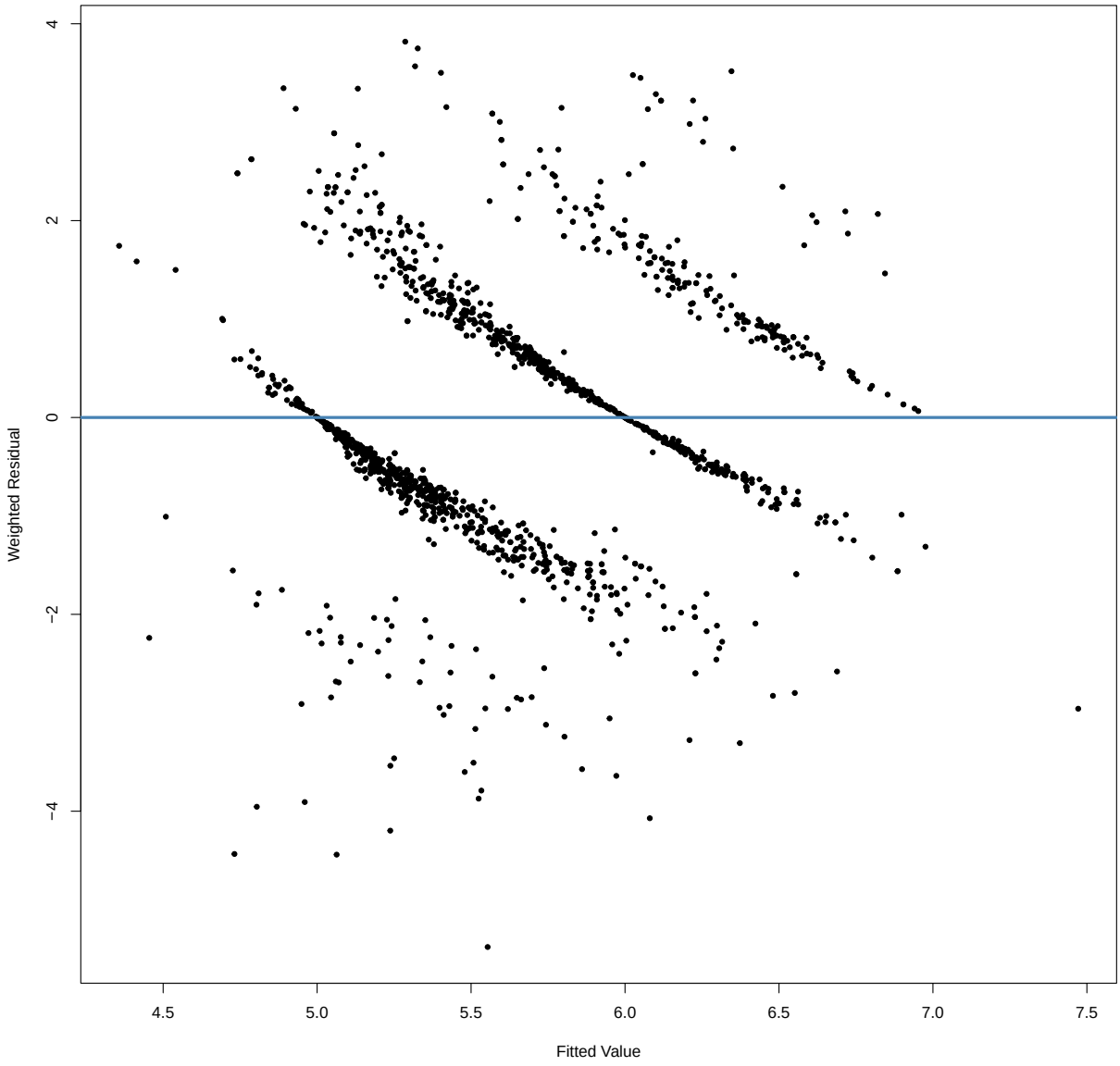
- The reliability and validity of the analysis in our study are subject to several limitations. Even though efforts have been made to address potential non-linearities through variable transformations, such as logarithmic transformations, it is still possible that the underlying functional forms are not fully captured. The 'winequality-red' dataset from the UC Irvine Machine Learning Repository is the only dataset used, which raises questions about how broadly applicable the results will be. Moreover, there may be issues with model assumptions, namely the assumption of normality, due to the categorical nature of the 'quality' variable, which has only 10 discrete values. Although robust regression techniques and diagnostic tests have been used to address these problems, it is important to recognize that residual non-normality may affect the accuracy of parameter estimates.
- Furthermore, the violation of assumptions, such as normality and constant variance, in the residuals of certain regression models raises concerns about the validity of the results. While attempts have been made to address these violations, such as exploring different transformations and using weighted least squares (WLS) in the additional work section and robust regression in section 2, it is crucial to recognize that these methods may not fully remedy the violations. In such cases, it becomes important to acknowledge the limitations of the chosen models and explore alternative statistical techniques that are more resilient to the observed violations. If we were to redo the project or continue working on it, we would conduct a more carry out a more thorough analysis into the nature of influential observations to understand their potential impact on the study. Secondly, in order to provide a more reliable analysis, we would look at non-parametric techniques or other regression models that are less susceptible to assumptions. This might involve taking into account machine learning algorithms like gradient boosting or random forests that can capture complex correlations in the data without strictly adhering to linear assumptions.

Section 5: Conclusions

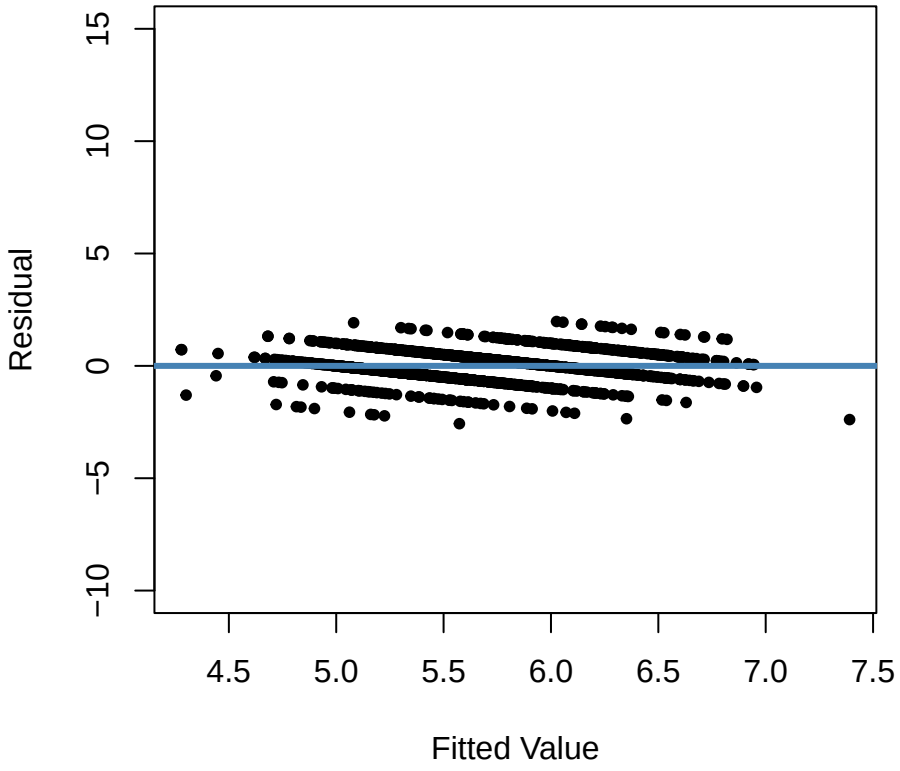
In conclusion, our analysis of the red variant Portuguese "Vinho Verde" wine dataset revealed significant relationships between physicochemical attributes and wine quality. With the quality variable being categorical, the robust LAD regression model turned out to be the most appropriate. There were strong correlations between wine quality and key predictors such alcohol concentration, volatile acidity, and chloride content. However, challenges persisted, including violations of normality and constant variance assumptions. In order to improve the model's applicability, future studies should investigate other statistical methods and take into account additional influencial factors, including grape types or vineyard features. Despite limitations, this study lays the groundwork for future research in the field by providing important insights into the complex relationships between red wine's physicochemical characteristics and sensory evaluations.

Section 6: Additional Work

Using WLS to see if it can fix the constant variance violation and look at the fitted vs weighted residual graph:



Let's look at the normal fitted vs residual graph without weighted residuals:

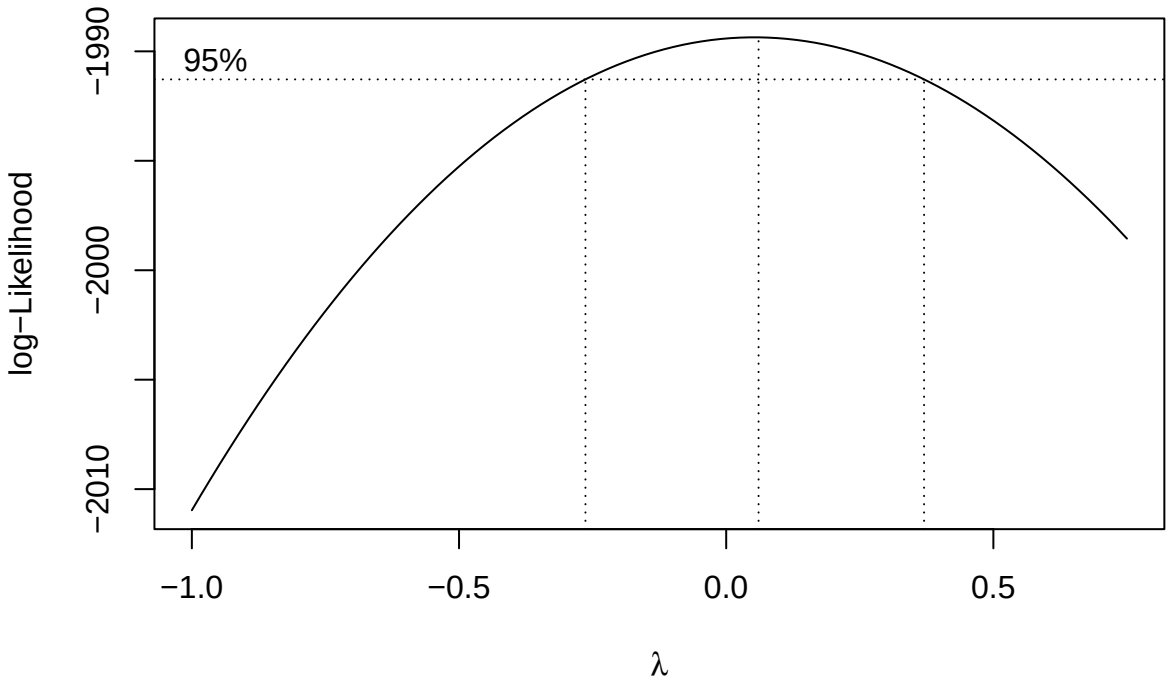


Now let's do breush-pagan test to check homoscedasticity:

```
##
##  studentized Breusch-Pagan test
##
## data:  model_wls
## BP = 6001.6, df = 8, p-value < 2.2e-16
```

The test-statistic is 6001.6 and the p-value is less than $2.2 * 10^{-16}$ which is somehow worse than the model from where we started. This might be because the model (model_collinearity1) we used to get WLS has normality violations along with constant variance violations. So we don't use the wls model (model_wls).

Let's use a box-cox method to find an appropriate transformation of the response.



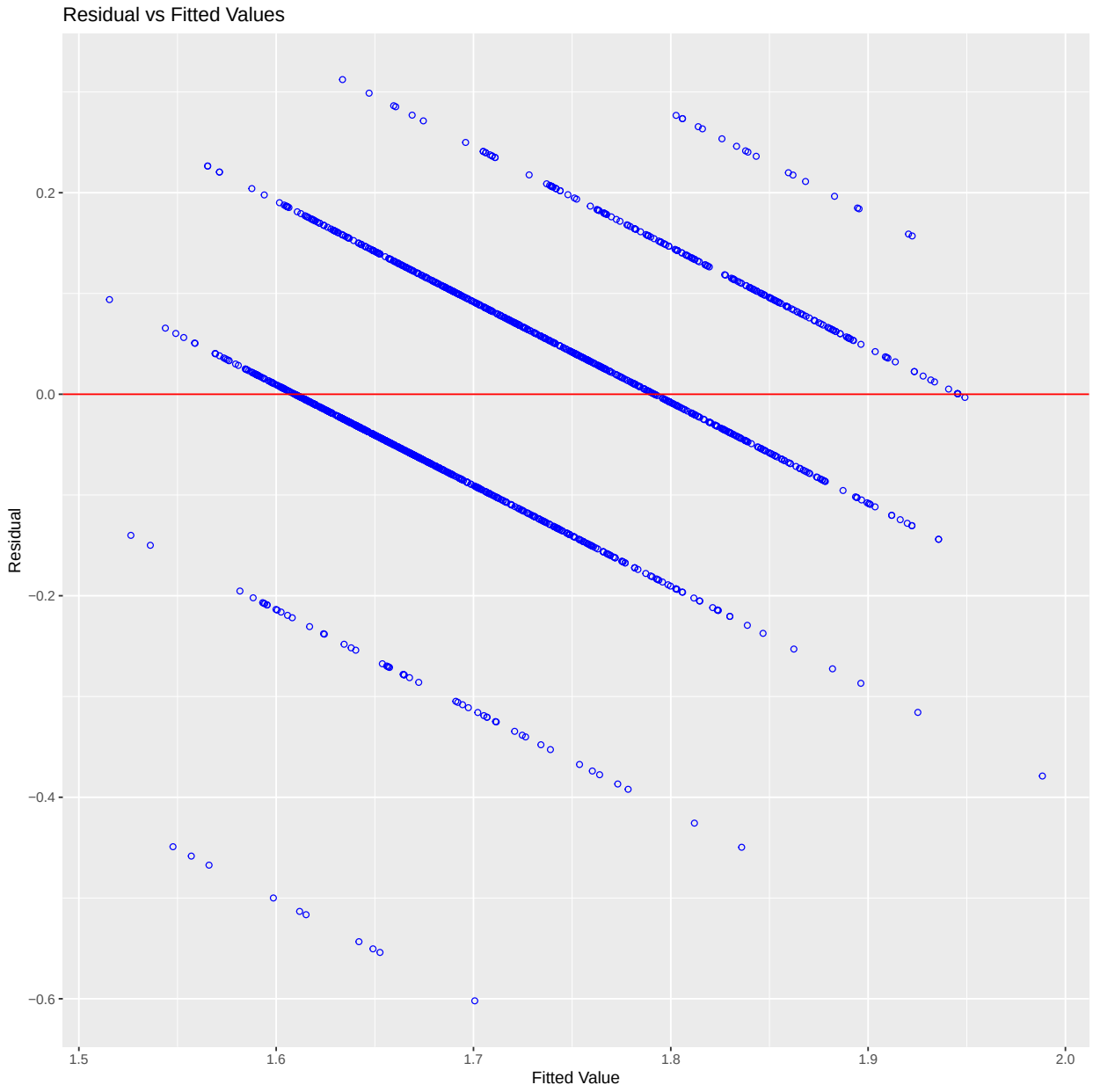
To extract $\hat{\lambda}$, the value that maximizes the log-likelihood (the function being plotted in the previous graph), we use the following code.

```
## [1] 0.06060606
```

let's also get the 95% confidence interval from the box plot:

```
## lower bound upper bound
##  -0.2575758   0.3611111
```

Here, the 95% confidence interval has 0, so we can use a log transformation. So, let's fit the log transformation and check the residual plot.



Now, let's check the breush-pagan test:

```
##
##  studentized Breusch-Pagan test
##
## data:  model_log
## BP = 57.843, df = 8, p-value = 1.233e-09
```

Here the test statistic is 57.843 and p-value is $1.233 * 10^{-9}$ which is less than significance level of 0.05. So still constant variance assumption is violated.

lets do shaprio wilk test:

```
##
##  Shapiro-Wilk normality test
##
## data:  resid(model_log)
## W = 0.96354, p-value < 2.2e-16
```

Here the test statistic is 0.96354 and p-value is less than $2.2 * 10^{-16}$ which is less than significance level of 0.05. So still normality assumption is violated. So we reject this model (model_log) too and try using robust regression methods on the models. (Done in section 2).

Code Appendix

```
#section 1 starts here

library(readr)
library(ggplot2)
library(quantreg)
library(faraway)
library(car)
library(tidyverse)
# loading our red wine dataset into r
rwq = read_csv("D:/UNI/Masters/FSU/Semester 1/Regressions/Project/winequality-red.csv")

# scatterplot between quality and ALC
plot(quality ~ ALC, data = rwq,
     xlab = "alcohol in red wine in (vol.%)",
     ylab = "quality of red wine (between 0 to 10)",
     main = "Quality vs Alcohol in red wine",
     pch  = 20,
     cex  = 2,
     col  = "dodgerblue")

# scatterplot between quality and RS
plot(quality ~ RS, data = rwq,
     xlab = "residual sugar in red wine in (g/dm^3)",
     ylab = "quality of red wine (between 0 to 10)",
     main = "Quality vs residual sugar contained in red wine",
     pch  = 20,
     cex  = 2,
     col  = "dodgerblue")

# histogram for quality
hist(rwq$quality,
     xlab = "quality of red wine (between 0 to 10)",
     main = "Quality of Red wine",
     breaks = 5,
     col    = "dodgerblue",
     border = "white")

#section 1 ends here
```

```r
# section 2 starts here

# collinearity
library(dplyr)
library(corrplot)
library(olsrr)
library(faraway)
library(lmtest)
library(MASS)
# data.frame containing just the predictors
rwq_preds = dplyr::select(rwq, -quality)
round(cor(rwq_preds), 3)

# correlation matrix
corrplot(cor(rwq_preds),
         method = 'color', order = 'hclust',  diag = FALSE,
         number.digits = 3, addCoef.col = 'black', tl.pos= 'd', cl.pos ='r')

# checking the condition number
model_collinearity = lm(quality ~ ., data = rwq)
round(ols_eigen_cindex(model_collinearity)[, 1:2], 4)
# check the vif's for the model_collinearity
vif(model_collinearity)

# check for vif on new model (model_collinearity1)
model_collinearity1 = lm(quality ~ . - FA - FSO2 - DEN, data = rwq)
vif(model_collinearity1)
#orthogonality on new model (model_collinearity1)
1 - 1/vif(model_collinearity1)
rwq_preds1 = dplyr::select(rwq, -FA, - quality, - FSO2, - DEN)
# correlation matrix
corrplot(cor(rwq_preds1),
         method = 'color', order = 'hclust',  diag = FALSE,
         number.digits = 3, addCoef.col = 'black', tl.pos= 'd', cl.pos ='r')
outlier_test_cutoff = function(model, alpha = 0.05) {
    n = length(resid(model))
    qt(alpha/(2 * n), df = df.residual(model) - 1, lower.tail = FALSE)
}

# vector of indices for observations deemed outliers.
cutoff = outlier_test_cutoff(model_collinearity1, alpha = 0.05)

which(abs(rstudent(model_collinearity1)) > cutoff)
# check for highly influential observations on the model : model_collinearity1
highly_inf_obs = which(cooks.distance(model_collinearity1) > 4 / length(cooks.distance(model_collineari
# count the number of highly influential observations
length(highly_inf_obs)

# ids for non-influential observations
noninfluential_ids = which(
    cooks.distance(model_collinearity1) <= 4 / length(cooks.distance(model_collinearity1)))

# fit the model on non-influential subset
```

```r
model_collinearity2 = lm(quality ~ . - FA,
               data = rwq,
               subset = noninfluential_ids)
# return coefficients for new model(model_collinearity2)
coef(model_collinearity2)
# return coefficients for old model(model_collinearity1)
coef(model_collinearity1)

# fitted vs redisual values for the model (model_collinearity2)
ols_plot_resid_fit(model_collinearity2)
# breush-pagan test
bptest(model_collinearity2)
#shapiro-wilk test
shapiro.test(resid(model_collinearity2))

# confidence interval for the model using bootstrap
set.seed(42)
Confint(Boot(model_collinearity1, R = 1500, method = 'residual'))
# confidence interval for regular OLS model without using bootstrap
confint(model_collinearity1)

# LAd regression (model_lad)
model_lad = rq(quality ~ . - FA - DEN - FSO2, data = rwq)
summary(model_lad)
# calculating the lower and upper bounds for LAD (model_lad)
# Extract coefficient estimates and standard errors
coef_estimates = coef(model_lad)
se = summary(model_lad)$coefficients[, "Std. Error"]

# Set the desired confidence level
confidence_level = 0.95

# Calculate confidence intervals
z_value <- qnorm((1 + confidence_level) / 2)  # For a two-sided interval
lower_bound <- coef_estimates - z_value * se
upper_bound <- coef_estimates + z_value * se

# Combine results into a data frame
confidence_intervals <- data.frame(
  Coefficient = coef_estimates,
  LowerBound = lower_bound,
  UpperBound = upper_bound
)

# Print the results
print(confidence_intervals)

# IRWLS with a limit of 100 iterations.
model_hub = rlm(quality ~ . - FA - DEN - FSO2, maxit = 100, data = rwq)
summary(model_hub)
#bootstrap confint for huber's methos (model_hub)
set.seed(42)
Confint(Boot(model_hub, R = 1500, method = 'residual'))
```

```r
# fitting ols model with no highly influential observations
model_ols_no_highinf = lm(quality ~ . - FA - FSO2 - DEN, data = rwq,
                          subset = noninfluential_ids)
summary(model_ols_no_highinf)

# fitting the model with lad and no highly influential observations
model_lad_no_highinf = rq(quality ~ . - FA - DEN - FSO2, data = rwq,
                          subset = noninfluential_ids)
summary(model_lad_no_highinf, alpha = 0.05)

# calculating the lower and upper bounds for LAD with no highly influential observations (model_lad_no_
# Extract coefficient estimates and standard errors
coef_estimates = coef(model_lad_no_highinf)
se = summary(model_lad_no_highinf)$coefficients[, "Std. Error"]

# Set the desired confidence level
confidence_level = 0.95

# Calculate confidence intervals
z_value <- qnorm((1 + confidence_level) / 2)  # For a two-sided interval
lower_bound <- coef_estimates - z_value * se
upper_bound <- coef_estimates + z_value * se

# Combine results into a data frame
confidence_intervals <- data.frame(
  Coefficient = coef_estimates,
  LowerBound = lower_bound,
  UpperBound = upper_bound
)

# Print the results
print(confidence_intervals)

# huber's method with no highly influential observations
model_hub_no_highinf = rlm(quality ~ . - FA - DEN - FSO2, maxit = 100, data = rwq[-c(14,34,44,46,80,87,9
summary(model_hub_no_highinf)
# bootstrap intervals with huber's method and no highly influential observations
set.seed(42)
Confint(Boot(model_hub_no_highinf, R = 1500, method = 'residual'))

# table with all the methods (OLS, OLS refit, LAD, LAD refit, Huber, Huber refit)
Methods = c("**OLS**", "**OLS[Refit]**", "**LAD**", "**LAD[Refit]**", "**Huber**", "**Huber[Refit]**")
Intercept = c("**4.603**", "**4.244**","**3.027**","**2.930**","**4.215**","**4.110**")
VA = c("**-1.122**", "**-1.003**","**-0.787**","**-0.761**","**-1.030**","**-0.985**")
CA = c("-0.181", "**-0.315**","-0.078","-0.154","-0.160","**-0.282**")
RS = c("0.011", "0.022","**0.035**","**0.034**","**0.024**","**0.024**")
CL = c("**-1.932**","**-2.142**","**-2.119**","**-2.288**","**-1.781**","**-2.131**")
TSO2 = c("**-0.002**","**-0.002**","**-0.002**","**-0.002**","**-0.002**","**-0.002**")
pH = c("**-0.522**","**-0.486**","**-0.339**","**-0.343**","**-0.448**","**-0.458**")
SUL = c("**0.906**","**1.177**","**1.091**","**1.227**","**0.928**","**1.169**")
ALC = c("**0.293**","**0.299**","**0.347**","**0.353**","**0.298**","**0.302**")
table40 = data.frame(Methods, Intercept, VA, CA, RS, CL, TSO2, pH, SUL, ALC)
knitr::kable(table40, "pipe", align=c("l" , "c", "c", "c", "c", "c", "c", "c", "c", "c"))
```

```r
# shaprio wilk test for final model
shapiro.test(resid(model_lad))

# breush pagan test for final model
bptest(model_lad)

#section 2 ends here

# additional work starts here

# initializing the weights for the WLS model on model_collinearity1
model_wts = lm(abs(resid(model_collinearity1)) ~ . - FA - quality - FSO2 - DEN, data = rwq)
coef(model_wts)
# calculate the weights as 1 / (fitted values)^2
weights = 1 / fitted(model_wts)^2
# run WLS
model_wls = lm(quality ~ . - FA - FSO2 - DEN, data = rwq, weights = weights)
# fitted vs weighted residual graph
plot(fitted(model_wls), weighted.residuals(model_wls),
     pch = 20, xlab = 'Fitted Value', ylab = 'Weighted Residual')

abline(h=0, lwd=3, col='steelblue')
# OLS fitted-vs-residual plot
plot(fitted(model_collinearity1), resid(model_collinearity1),
     pch = 20, ylim = c(-10, 15),
     xlab = 'Fitted Value', ylab = 'Residual')
abline(h=0, lwd=3, col='steelblue')
#bp-test for wls model
bptest(model_wls)

#box-cox
bc = boxcox(model_collinearity2, lambda = seq(-1, 0.75, by = 0.05), plotit = TRUE)
# getting the value that maximizes the log-likelihood
bc$x[which.max(bc$y)]
get_lambda_ci = function(bc, level = 0.95) {
    # lambda such that
    # L(lambda) > L(hat(lambda)) - 0.5 chisq_{1, alpha}
    CI_values = bc$x[bc$y > max(bc$y) - qchisq(level, 1)/2]

    # 95 % CI
    CI <- range(CI_values)

    # label the columns of the CI
    names(CI) <- c("lower bound","upper bound")

    CI
}

# extract the 95% CI from the box cox object
get_lambda_ci(bc)

# fitting the model_log
model_log = lm(log(quality) ~ log(VA) + CA + log(RS) + log(CL) + log(TSO2) + log(pH) + log(SUL) + log(A
```

```r
# fitted vs residual plot for model_log
ols_plot_resid_fit(model_log)

# breusch-pagan test for model_log
bptest(model_log)
# shaprio test for model_log
shapiro.test(resid(model_log))

# additional work ends here
```