STSCI 4740 - DATA MINING & MACHINE LEARNING

Professor Dr. Yang Ning

Fall 2022

FINAL PROJECT: PREDICTING WINE QUALITY

Eric Huang, David Vilensky, Nick Gembs, Tomer Shamir

Table of Contents:

1. Introduction

The goal of this project is to predict the quality of a given wine on a scale from 0-10, with 10 being the highest quality. Our Wine dataset is a collection of 178 red and white wine samples grown in the same region in Italy but derived from three different cultivars. This includes 1,599 red wine observations and 4,898 white wine observations. Each observation measures 12 different chemical properties of a wine: fixed acidity, volatile acidity, citric acid, residual sugar, chlorides, free sulfur dioxide, total sulfur dioxide, density, pH, sulphates, alcohol %, and color.

In this paper we determine which of these predictors are most useful for predicting wine quality, and which type of Machine Learning model performs best. We try: linear regression (with parameter selection, lasso, and ridge variations), K-Nearest Neighbors, decision trees, and smoothing splines. We discuss the trade-offs of each method with respect to interpretability, bias, and variance and conclude by selecting the single best method of predicting wine quality.

2. Model Assumptions

The wine comes in two colors: red and white, each of which comes in a separate data set. We have the option to create two separate models to predict wine quality for each color. However, we chose to combine the red and white wines into a single dataframe by adding an indicator ‘redYES’ variable, which has value 1 when the wine is red and 0 when the wine is white. We felt that creating a single, general-purpose model that can predict wine quality regardless of color was more appropriate given that all the wines were grown in the same region of Italy.

The wine quality output is an ordered categorical variable, ranked between integers 0 through 10. We choose to utilize mostly regression methods instead of classification methods, in order to better utilize the data assumption that quality of 5 is closer to quality of 6 than quality of 2 is. In other words, we assume that wine quality is ranked on a linear scale. Most classification methods - such as linear discriminant analysis - fail to recognize this assumption of the data.

In this report we fit several models in order to determine which is the most effective at predicting wine quality. Sticking with our wine quality scale assumption, we chose to compare our models on the basis of test Mean Squared error so that the penalty was greatest for estimates that were farthest from the true value. In addition, we used this 10-fold cross-validation (10-fold CV) in order to choose any tuning parameters we have for our models. 10-fold CV is preferred in most models because 10-fold CV trains on the entire data set, while traditional train/test splitting only trains on half of the data set. As a result, 10-fold CV often yields models with lower variance so we chose to use it in our models.

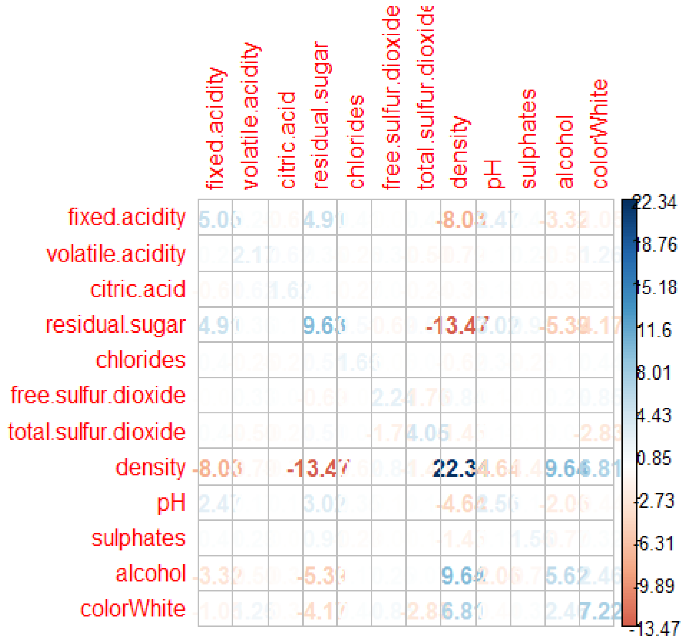
3. Linear Regression

The first models to be fitted to the wine dataset will be under the scope of linear regression. We chose to start with linear regression because it is easiest to implement and interpret.

Linear regression has the following assumptions:

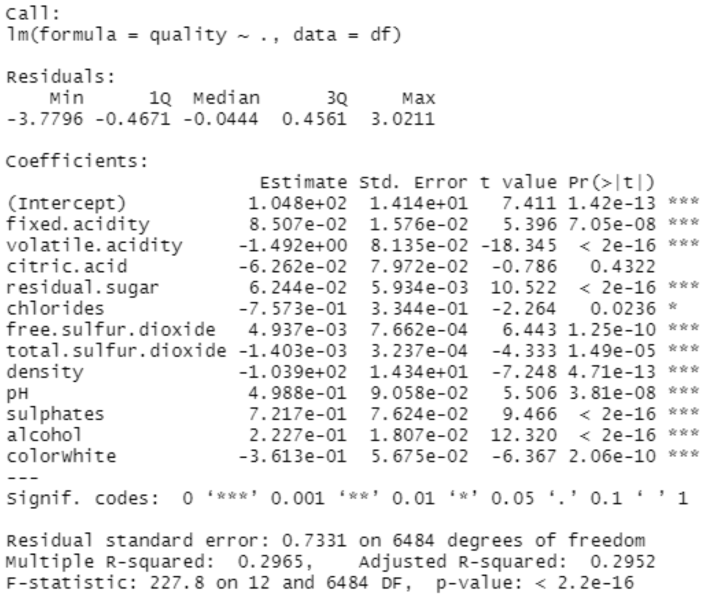
1. A linear relationship between predictors and outcome
2. Multivariate normality
3. Little to no multicollinearity
4. Homoscedasticity

By performing linear regression in R, the first 2 assumptions are assumed to be true, which can be confirmed through the goodness-of-fit of the model. Multicollinearity can be measured by identifying any variables with high correlations. This is visualized in the correlation plot below:



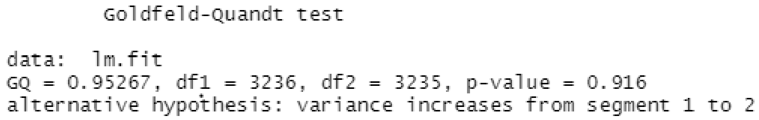
As we can see in the correlation plot, density has an abnormally high correlation with other predictors, especially with fixed.acidity, residual.sugar, and alcohol. Dropping one or more of these variables may be advantageous in fitting a proper linear regression model.

Homoscedasticity can be confirmed with a Goldfeld-Quandt test after fitting our initial regression model. This initial model will be a full, non-sparse model, predicting quality based on all of the predictors with no interactions.



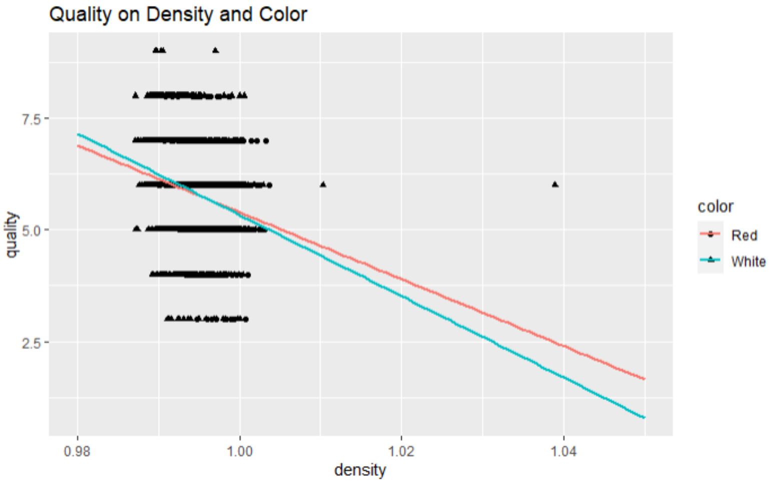
```
Call:
lm(formula = quality ~ ., data = df)

Residuals:
    Min      1Q  Median      3Q     Max 
-3.7796 -0.4671 -0.0444  0.4561  3.0211 

Coefficients:
                       Estimate Std. Error t value Pr(>|t|)    
(Intercept)           1.048e+02  1.414e+01   7.411 1.42e-13 ***
fixed.acidity         8.507e-02  1.576e-02   5.396 7.05e-08 ***
volatile.acidity     -1.492e+00  8.135e-02 -18.345  < 2e-16 ***
citric.acid          -6.262e-02  7.972e-02  -0.786   0.4322    
residual.sugar        6.244e-02  5.934e-03  10.522  < 2e-16 ***
chlorides            -7.573e-01  3.344e-01  -2.264   0.0236 *  
free.sulfur.dioxide   4.937e-03  7.662e-04   6.443 1.25e-10 ***
total.sulfur.dioxide -1.403e-03  3.237e-04  -4.333 1.49e-05 ***
density              -1.039e+02  1.434e+01  -7.248 4.71e-13 ***
pH                    4.988e-01  9.058e-02   5.506 3.81e-08 ***
sulphates             7.217e-01  7.624e-02   9.466  < 2e-16 ***
alcohol               2.227e-01  1.807e-02  12.320  < 2e-16 ***
colorWhite           -3.613e-01  5.675e-02  -6.367 2.06e-10 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.7331 on 6484 degrees of freedom
Multiple R-squared:  0.2965,	Adjusted R-squared:  0.2952 
F-statistic: 227.8 on 12 and 6484 DF,  p-value: < 2.2e-16
```

As seen in the R output, all predictors are significant in predicting quality at a 95% confidence level except for citric.acid. It is likely that this predictor will be dropped in future variable selection models. Chlorides is the only other model not significant to a 99.9% confidence level. The residual standard error for this model is 0.7331. To predict a test mean square error, 10-fold cross-validation will be used to gather a cv error. This mse came out to be 0.539366. To confirm the final assumption of linear regression, a Goldfeld-Quandt test was performed.



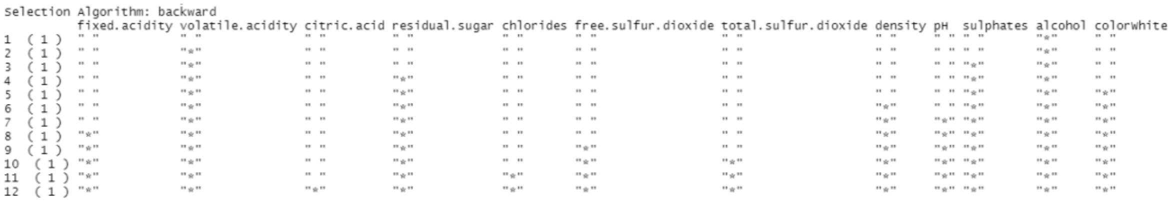
```
	Goldfeld-Quandt test

data:  lm.fit
GQ = 0.95267, df1 = 3236, df2 = 3235, p-value = 0.916
alternative hypothesis: variance increases from segment 1 to 2
```

Since the p-value is not less than 0.05, we fail to reject the null hypothesis. We do not have sufficient evidence to say that heteroscedasticity is present in the regression model. The homoscedasticity assumption has been met. When looking at the coefficients of the full model, it is seen that density, the variable with high correlation, has the largest absolute coefficient. The full model claims that density is a dominant variable, its effect can be seen in an effect plot of density on quality, holding other predictors constant at their means.

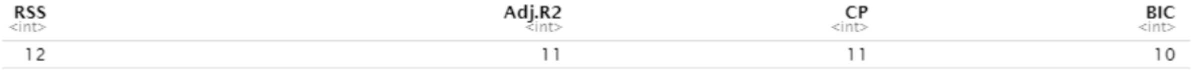


3a. Variable Selection

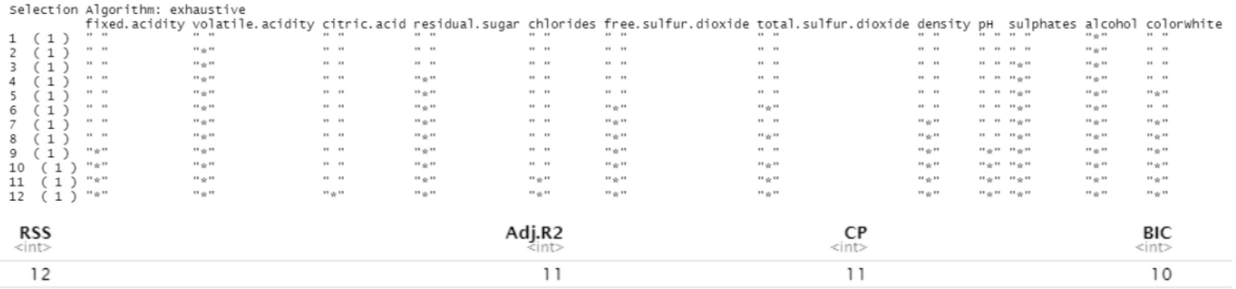
The second regression model to be tested is subset selection. In this section, we will be performing best-subset, forward, and backward selection. It is likely that best-subset selection will yield the best model, unless it overfits due to its higher flexibility. This is because best-subset selection compares all 2^p possible models, which is feasible since we have only 12 predictors. R outputs for each selection model can be seen in the table below. In addition, values for the best model according to each metric are displayed below the model. For example, under the Mallow's Cp metric in exhaustive selection, the model with 11 variables returned the lowest Cp and is therefore deemed the best model. For RSS, it is confirmed that the best model will always be the full model, as least squares regression minimizes training RSS. Finally, since BIC has a heavier penalty, it can be seen that BIC tends to select smaller models. Subset selection has returned two possible models for further testing, an 11 predictor model that drops citric.acid, and a 10 predictor model that drops citric.acid and chlorides. 10-fold CV can be performed to gather test MSE values to compare to other models. The errors are 0.5390804 and 0.5383253, respectively.



```
Selection Algorithm: backward
          fixed.acidity volatile.acidity citric.acid residual.sugar chlorides free.sulfur.dioxide total.sulfur.dioxide density pH  sulphates alcohol colorwhite
1  ( 1 )  " "           " "              " "         " "            " "       " "                 " "                  " "     " " " "       "*"     " "
2  ( 1 )  " "           "*"              " "         " "            " "       " "                 " "                  " "     " " " "       "*"     " "
3  ( 1 )  " "           "*"              " "         " "            " "       " "                 " "                  " "     " " "*"       "*"     " "
4  ( 1 )  " "           "*"              " "         "*"            " "       " "                 " "                  " "     " " "*"       "*"     " "
5  ( 1 )  " "           "*"              " "         "*"            " "       " "                 " "                  " "     " " "*"       "*"     "*"
6  ( 1 )  " "           "*"              " "         "*"            " "       " "                 " "                  "*"     " " "*"       "*"     "*"
7  ( 1 )  " "           "*"              " "         "*"            " "       " "                 " "                  "*"     "*" "*"       "*"     "*"
8  ( 1 )  "*"           "*"              " "         "*"            " "       " "                 " "                  "*"     "*" "*"       "*"     "*"
9  ( 1 )  "*"           "*"              " "         "*"            " "       "*"                 " "                  "*"     "*" "*"       "*"     "*"
10  ( 1 ) "*"           "*"              " "         "*"            " "       "*"                 "*"                  "*"     "*" "*"       "*"     "*"
11  ( 1 ) "*"           "*"              " "         "*"            "*"       "*"                 "*"                  "*"     "*" "*"       "*"     "*"
12  ( 1 ) "*"           "*"              "*"         "*"            "*"       "*"                 "*"                  "*"     "*" "*"       "*"     "*"
```

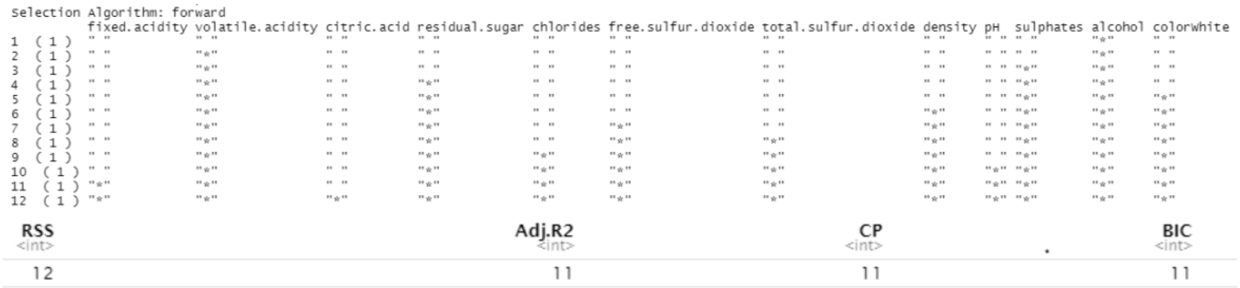


RSS <int>	Adj.R2 <int>	CP <int>	BIC <int>
12	11	11	10



```
Selection Algorithm: exhaustive
          fixed.acidity volatile.acidity citric.acid residual.sugar chlorides free.sulfur.dioxide total.sulfur.dioxide density pH  sulphates alcohol colorWhite
1  ( 1 )  " "           " "              " "         " "            " "       " "                 " "                  " "     " " " "       "*"     " "
2  ( 1 )  " "           "*"              " "         " "            " "       " "                 " "                  " "     " " " "       "*"     " "
3  ( 1 )  " "           "*"              " "         " "            " "       " "                 " "                  " "     " " "*"       "*"     " "
4  ( 1 )  " "           "*"              " "         "*"            " "       " "                 " "                  " "     " " "*"       "*"     " "
5  ( 1 )  " "           "*"              " "         "*"            " "       " "                 " "                  " "     " " "*"       "*"     "*"
6  ( 1 )  " "           "*"              " "         "*"            " "       "*"                 "*"                  " "     " " "*"       "*"     " "
7  ( 1 )  " "           "*"              " "         "*"            " "       "*"                 " "                  "*"     " " "*"       "*"     "*"
8  ( 1 )  " "           "*"              " "         "*"            " "       "*"                 "*"                  "*"     " " "*"       "*"     "*"
9  ( 1 )  "*"           "*"              " "         "*"            " "       "*"                 " "                  "*"     "*" "*"       "*"     "*"
10  ( 1 ) "*"           "*"              " "         "*"            " "       "*"                 "*"                  "*"     "*" "*"       "*"     "*"
11  ( 1 ) "*"           "*"              " "         "*"            "*"       "*"                 "*"                  "*"     "*" "*"       "*"     "*"
12  ( 1 ) "*"           "*"              "*"         "*"            "*"       "*"                 "*"                  "*"     "*" "*"       "*"     "*"
```

RSS	Adj.R2	CP	BIC
<int>	<int>	<int>	<int>
12	11	11	10

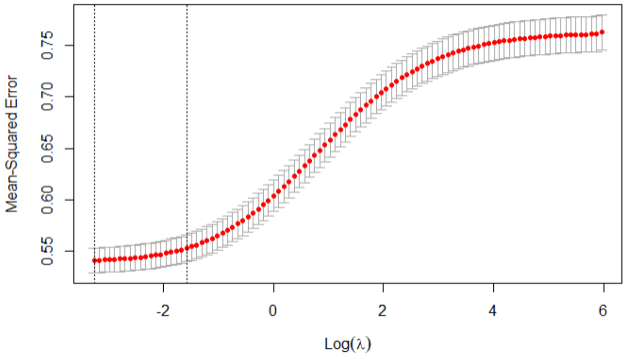


```
Selection Algorithm: forward
          fixed.acidity volatile.acidity citric.acid residual.sugar chlorides free.sulfur.dioxide total.sulfur.dioxide density pH  sulphates alcohol colorWhite
1  ( 1 )  " "           " "              " "         " "            " "       " "                 " "                  " "     " " " "       "*"     " "
2  ( 1 )  " "           "*"              " "         " "            " "       " "                 " "                  " "     " " " "       "*"     " "
3  ( 1 )  " "           "*"              " "         " "            " "       " "                 " "                  " "     " " "*"       "*"     " "
4  ( 1 )  " "           "*"              " "         "*"            " "       " "                 " "                  " "     " " "*"       "*"     " "
5  ( 1 )  " "           "*"              " "         "*"            " "       " "                 " "                  " "     " " "*"       "*"     "*"
6  ( 1 )  " "           "*"              " "         "*"            " "       " "                 " "                  "*"     " " "*"       "*"     "*"
7  ( 1 )  " "           "*"              " "         "*"            " "       "*"                 " "                  "*"     " " "*"       "*"     "*"
8  ( 1 )  " "           "*"              " "         "*"            " "       "*"                 "*"                  "*"     " " "*"       "*"     "*"
9  ( 1 )  " "           "*"              " "         "*"            "*"       "*"                 "*"                  "*"     " " "*"       "*"     "*"
10  ( 1 ) " "           "*"              " "         "*"            "*"       "*"                 "*"                  "*"     "*" "*"       "*"     "*"
11  ( 1 ) "*"           "*"              " "         "*"            "*"       "*"                 "*"                  "*"     "*" "*"       "*"     "*"
12  ( 1 ) "*"           "*"              "*"         "*"            "*"       "*"                 "*"                  "*"     "*" "*"       "*"     "*"
```

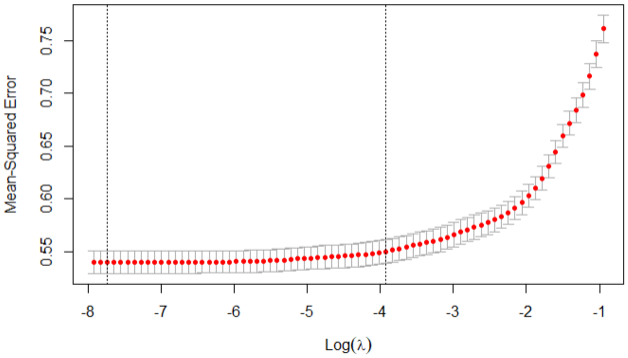
RSS	Adj.R2	CP	BIC
<int>	<int>	<int>	<int>
12	11	11	11

3b. Lasso and Ridge Regression (Shrinkage Methods)

The final regression technique used is ridge and lasso regression. Both are methods that shrink model coefficients in order to achieve a better bias-variance tradeoff by reducing variance. Using the glmnet library in R, ridge and lasso regression can be performed with a tuning parameter λ. The higher the value of λ, the less flexible the model is. The optimal λ was found using 10-fold cross validation. Plots of MSE on log(λ) for ridge (left) and lasso (right) are seen below:



Ridge Regression



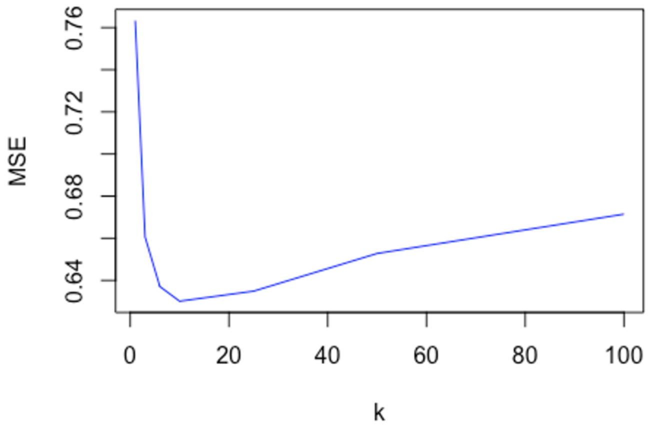
Lasso Regression

With an optimal λ, ridge regression has a cv error of 0.5410264. Since ridge regression does not perform variable selection, no predictors were dropped in this model. Lasso regression has a cv error of 0.5397717. Lasso regression dropped 3 predictors from the model: density, fixed.acidity, and citric.acid. With a lower cv error and a resolution to earlier multicollinearity issues, lasso appears to have a better bias-variance tradeoff so it is preferred to ridge regression.

4. K-Nearest Neighbors

K Nearest Neighbors (KNN) is a supervised machine learning algorithm that can be utilized in both classification and regression problems. KNN is a non-parametric and lazy learning algorithm because it does not assume anything about the underlying distribution. KNN was one of the first methods we looked into because it is fast to train and predict, only relies on distance calculations, and easy to interpret.

KNN uses a user-defined tuning parameter K to select the K-nearest data points on which to train the model. Higher values of K have higher bias in the bias-variance tradeoff, while lower values of K have higher variance in the bias-variance tradeoff. The optimal value for K can be determined using cross-validation, which involves splitting the data into K subsets, training the model on K-1 subsets, and testing the model on the remaining subset. We used 10-fold cross-validation to compare the performance of K={1,3,6,10,25,50,100}, depicted below.

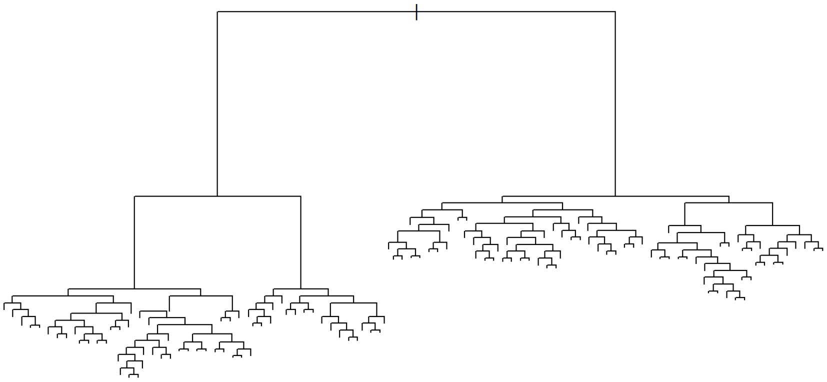


The graph shows that a value of K= 10 yields the lowest test MSE of 0.6282250.

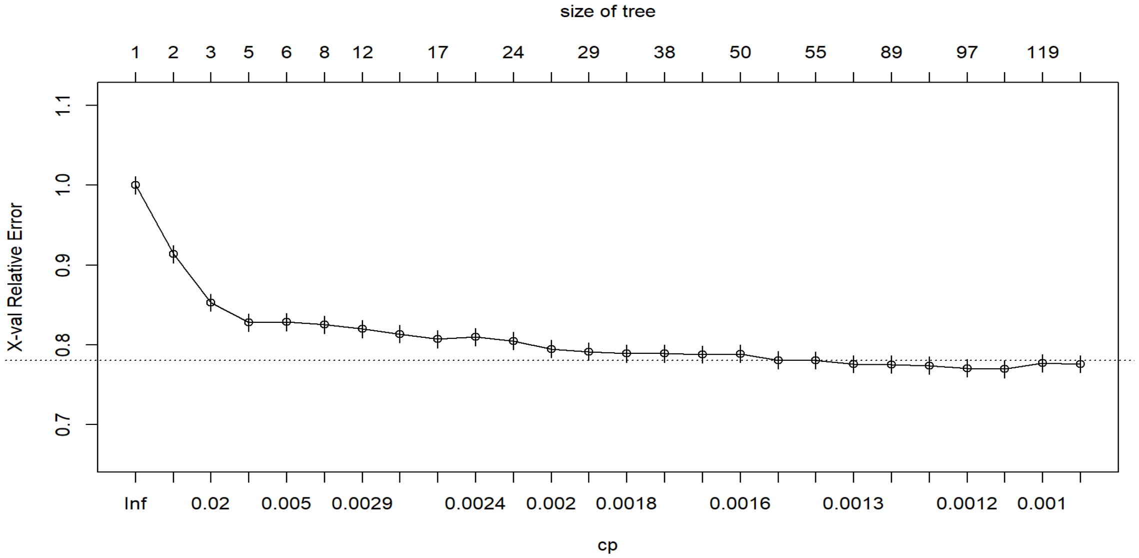
5. Trees

For regression trees, we can build trees based on a complexity parameter, which determines how many splits the tree uses in order to make predictions with the data. In order to tune this parameter, we used a grid search over the complexity parameter values of 0.05, 0.01, 0.009, 0.005, and 0.001. Using 10-fold cross validation, we trained the model using each of these tree complexity parameters in order to estimate which complexity produces the model with the lowest test mean squared error. The complexity parameter that produced the lowest test mean squared error value was 0.001, producing the following tree structure, which had an estimated test MSE of 0.545.

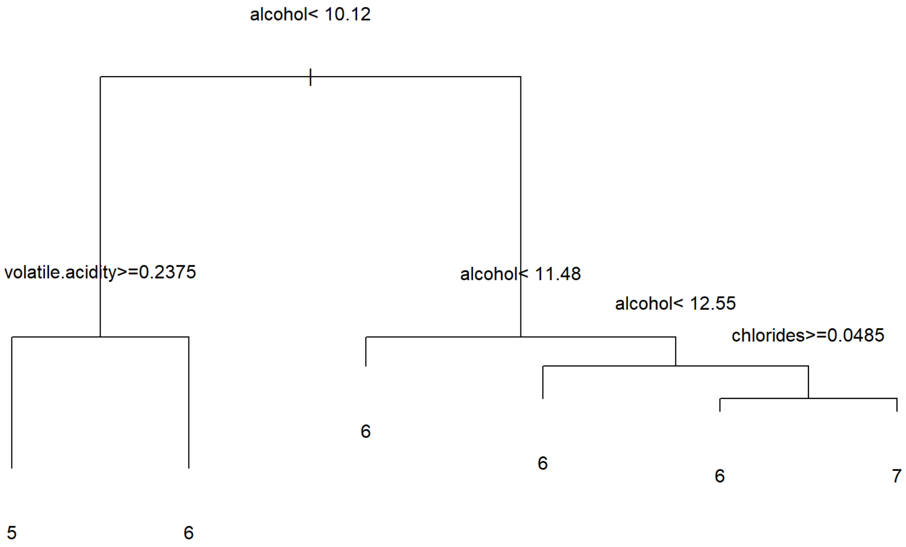
***Note: the split conditions and prediction averages are omitted for readability*



The above tree is a very complex model, as expected by the very low complexity parameter. We decided to plot the relative error as a function of the number of splits in the model to identify a good point to prune the tree while affecting the estimated test MSE as little as possible.



There appears to be an “elbow” in this graph at around a complexity parameter of 0.005 indicating that a decrease of value of the complexity parameter starts to have a reduced effect on the error of the model. Therefore, we chose to prune the tree based on the 0.005 complexity parameter and produced a much simpler tree.



This tree, while much simpler than the other tree we generated, only has an estimated test MSE of 0.554, which is slightly higher than the first tree we generated. A very important observation we can make based on this model is the high importance of alcohol, volatile acidity and chlorides in predicting quality.. The trees model in this sense acts as a predictor selector to some extent, by splitting over predictors where this split is most effective using the recursive binary splitting algorithm.

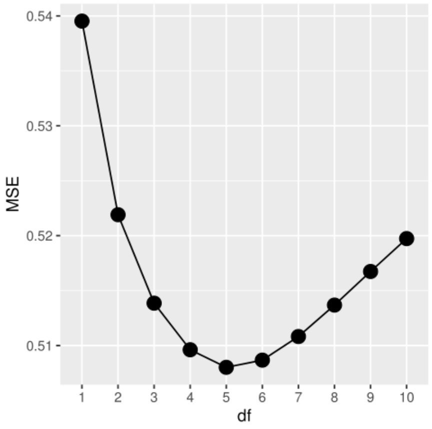
6. Smoothing Splines

We now implement smoothing splines. When implementing smoothing splines, we have the choice of what to set as the smoothing parameter λ. This regulates how closely we follow the training data in the loss function

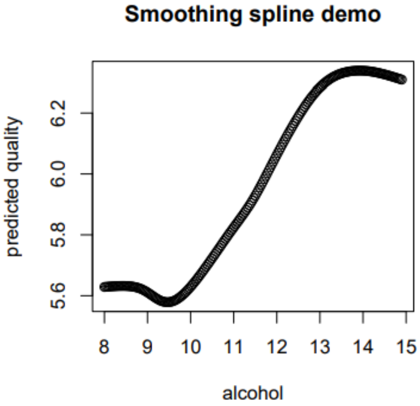
$$\sum_{i=1}^{n}(y_i - g(x_i))^2 + \lambda \int g''(t)dt$$

That is, the lower λ is, the more flexible the model will be, and the higher λ is, the less flexible the model will be. We wish to find the optimal middle-ground value of λ, in which, according to the variance-bias tradeoff, we don't have too flexible of a model with high variance, nor do we have too inflexible of a model with high bias. Setting the correct value of λ allows us to find the "valley" in which test error is minimized.

We implement smoothing splines using the "gam" package. In "gam", instead of setting the exact value of the smoothing parameter, we instead set the effective degrees of freedom. A lower effective degree of freedom corresponds to a higher value of λ, while a higher effective degree of freedom corresponds to a lower value of λ.



We find that the smoothing spline model most effective at predicting wine quality is the model with 5 effective degrees of freedom, with an estimated test MSE of 0.5067132.



We demonstrate an example of what the smoothing spline model that we chose looks like. We choose to show the model as only one variable varies, due to limitations in visualizing the model across 12 variables. We hold all variables except for alcohol constant, and we demonstrate how the model prediction varies as alcohol varies.

7. Conclusion

Throughout this report we compared and contrasted 4 different Machine Learning models based on their bias-variance tradeoff, measured using test MSE, as well as their interpretability. We utilized 10-fold Cross Validation (CV) when applicable to minimize variance. After tuning our parameters for the models and using 10-fold cross validation to estimate the test MSE, we determined that the model that performed the best based on this test MSE estimate was the smoothing spline model. We elaborate on the conclusions of each method below.

We found that K-Nearest Neighbors (KNN) was the worst performing model with a test MSE of 0.62. KNN likely suffers from the curse of dimensionality because we used 12 predictors.

We also tried out linear regression, as one of our assumptions was that wine quality was measured on a linear scale 0-10. Our initial linear regression model trained with 10-fold CV yielded a test MSE of 0.539366. We then utilized best subset selection and found that the “citric acid” and “chloride” predictors had little predictive value and could be dropped from the model. We also tested Ridge Regression and Lasso Regression to see if they gave a more optimal bias-variance tradeoff. The Ridge Regression is a shrinkage method that yielded a test MSE of 0.5410264 but did not remove any predictors. The Lasso Regression is a dimension reduction method that removed density, fixed.acidity and citric.acid, and had a test MSE of 0.5397717.

We found that trees had an estimated test MSE of 0.545. While this is not as strong of a predictor as smoothing splines, trees have other advantages. They have the advantage of being very easy to visualize, even in many dimensions. In contrast, smoothing splines and linear regression struggle to visualize more than 2 predictors at a time. In addition, we found that after pruning the tree to be less complex, we found that the only predictors that were split in the tree were alcohol, volatile acidity, and chlorides, indicating that these three predictors were the most important for predicting wine quality.

We found that smoothing splines had an estimated test MSE of 0.5067 when we used the model with 5 effective degrees of freedom. We may attribute this high performance of smoothing spline models to its ability to adapt very flexibly to the trends in the data, while still keeping variance low.

In conclusion, each model has its own advantages and disadvantages. For this project, in which we aim to predict wine quality, we recommend using a cubic spline model since it has the lowest test MSE. However, for someone trying to understand which variables are most influential in determining wine quality, we would recommend going with either a Lasso regression model or a tree-based method.