Alisha C. Souza

Merrimack College

Predictive Modeling-Final Project

Executive Summary:

In this project I worked with two different data sets seeking to answer a quantitative response problem, qualitative response problem and problem specifically using principal component regression. The following models were used throughout this project and a final model was choose based on statistical results while keeping in mind trade-offs such as Prediction Accuracy VS Interpretability, whether the model is a Good fit VS Over-fit, Under-fit right, whether the model shows Parsimony VS Black-box and lastly keeping in mind the interpretability vs flexibility of the model.

The first dataset, called the wine quality dataset, focuses on solving a numerical response problem. Specifically, I am using the red wine subset from the dataset (accessible at https://archive.ics.uci.edu/dataset/186/wine+quality). This dataset contains information about wine samples from northern Portugal, including attributes like fixed acidity, volatile acidity, citric acid, residual sugar, chlorides, free sulfur dioxide, total sulfur dioxide, density, pH, sulphates, alcohol, and quality. In my analysis, I performed algorithms and techniques to explore the relationship between these attributes (predictors) and the target variable, which is the quality of the wine. I have created a model to comprehend this relationship, aiming to make accurate predictions about the future quality of wine based on the most influential predictors and the best-performing model. An exploratory data analysis identified key factors like alcohol that strongly influenced wine quality. I then conducted a linear regression where significant variables like acidity, chlorides, and alcohol were pinpointed. The model was statistically significant, explaining about 36.06% of quality variation. I then refined the model, considering all variables and only significant ones. Ridge and lasso regression models were introduced to handle complex relationships. Lasso, emphasizing variable selection, stood out. Confidence intervals added a layer of uncertainty awareness, enhancing interpretability. Splitting the data into training and test sets, we evaluated model performance, using a simple linear regression as a benchmark (RMSE of 0.6744579). Ridge regression, addressing data complexity, exhibited a higher RMSE (0.8244), indicating a less accurate fit. Lasso regression showed a similar trend, with a final RMSE of 0.6766 after testing different parameters. Further evaluation involved Principal Component Regression (PCR) and Partial Least Squares (PLS) models. PCR improved with more components, stabilizing after three. PLS yielded an RMSE of 0.6750. Regression tree models were then explored, with the initial tree performing well (RMSE of 0.5274). Pruning for simplicity slightly increased mean squared error (0.6175), highlighting the trade-off between complexity and accuracy. Overall, the bagging model excelled with an impressively low error of

0.439, outperforming other models, including boosting. Bagging's robustness, achieved through aggregating predictions from various models, made it the optimal choice for predicting red wine quality. Its simplicity, combined with successful handling of selected predictor variables, solidified its position as the best model for this dataset.

The second dataset, called the with diabetes dataset, focuses on solving a qualitative response problem. The goal is to be able to predict whether a patient has diabetes based on the predictors. The overall decision to choose the first decision tree model, incorporating High Blood Pressure (HighBP), Body Mass Index (BMI), and Age as predictors with 5 nodes and a mean squared error (MSE) of 0.2011, is justified by its favorable balance between model simplicity and accuracy. The decision tree exhibits a reasonably low residual mean deviance, indicating a strong fit to the training data. The inclusion of HighBP, BMI, and Age as predictor variables aligns with established associations between these factors and diabetes risk, enhancing the interpretability of the model. With only 5 nodes, the model avoids unnecessary complexity, making it more easily understandable and interpretable for stakeholders. The MSE of 0.2011 signifies a relatively accurate prediction performance in comparison to the other models studied, and the decision to keep this level of error is justified by the desire to maintain a parsimonious model that balances accuracy with simplicity. This decision is particularly relevant when considering the trade-off between model complexity and the risk of overfitting, and the selected decision tree configuration offers a compelling compromise for practical and comprehensible diabetes prediction.

Data & Approach:

In this project I will be using two datasets to address and analyze different problems. The first dataset I will be using is the wine quality dataset to apply algorithms and techniques to address a quantitate response problem. Two datasets were included and the red wine dataset is used for this analysis. (https://archive.ics.uci.edu/dataset/186/wine+quality) This data set includes wine samples from the north of Portugal and has the following variables: fixed acidity, volatile acidity, citric acid, residual sugar, chlorides, free sulfur dioxide, total sulfur dioxide, density, pH, sulphates, alcohol, quality. In this approach I use techniques to understand the relationship between the predictors in relation to the target variable quality. I worked on creating a model to understand this relationship and be able to make predictions on the future quality of wine based on the most significant predictors and most accurate model.

In this quantitative analysis, the approach follows a step-by-step process to understand and predict the quality of red wine. Initially, during Data Exploration, the goal is to get a grasp of the dataset's characteristics. I began by conducting a thorough exploration of the data sets characteristics. Checking the dataset's dimensions and identifying any missing values, set the foundation for data integrity. Summary statistics and visualizations, including scatter plots and box plots, facilitated a comprehensive understanding of variable distributions and relationships with the predictors. Moving on, the focus shifts to picking out important predictors. Methods like Best Subset Selection or Stepwise Selection systematically help in choosing predictors that have the most impact on explaining the variation in the response. The initial linear regression model (red.lm.model1) aimed to predict wine quality based on all available variables. Acknowledging the significance of certain predictors, the model was refined by creating linear regression models with only the significant variables (red.lm.model_significant) and with significant positively correlated variables (red.lm.model_significant_pos_cor) offering a more focused perspective on the critical factors influencing wine quality.

To assess model performance, the data was split into training and testing sets. The training set was used to build the models, and the test set served as an independent dataset for evaluation. Mean Squared Error (MSE) and R-squared values were employed to quantify the predictive accuracy of each model, providing a robust assessment of their performance. Regularization techniques such as Ridge and Lasso Regression were then used utilizing cross-validated error and examining the impact of regularization on coefficients, gauging the models' ability to prevent overfitting and improve generalization to unseen data. Principal Component Regression (PCR) and Partial Least Squares (PLS) regression were then explored to capture non-linear relationships and potentially improve model flexibility. Cross-validation and validation plots were employed to fine-tune model parameters and assess their predictive accuracy. A regression tree model was constructed, visualized, and pruned based on cross-validation. This process aimed to strike a balance between capturing intricate patterns in the data and preventing overfitting, ultimately optimizing the tree's predictive performance. Bagging (Random Forest) and Boosting techniques were introduced to enhance predictive accuracy through aggregation of multiple models. Mean Squared Error (MSE) then served as a quantitative measure for

evaluating the performance of these ensemble methods. Overall, this method combined in a comprehensive comparison of various models, utilizing metrics like MSE and R-squared. Bar plots provided a visual representation of the relative performance of models, offering insights into their strengths and weaknesses in predicting red wine quality. This meticulous approach, combining exploratory data analysis, variable selection, regularization, diverse regression techniques, and ensemble methods, allowed for a significant understanding of the predictive models for red wine quality based on chemical attributes.

The second dataset I will be using is the CDC Diabetic Health Indicators data set to apply the algorithms and techniques to address a quantitative response problem. The data set I will be using contains healthcare statistics and lifestyle survey information about people in general along with their diagnosis of diabetes. The features consist of some demographics, lab test results, and answers to survey questions for each patient. The target variable for classification is whether a patient has diabetes/is pre-diabetic, or healthy. The purpose of this data set is to better understand the relationship between lifestyle and diabetes in the US. (Data set link: https://www.cdc.gov/brfss/annual_data/annual_2014.html) Variables in this data set include: Has Diabetes, High BP, High Chol, Chol Check, BMI, Smoker, Stroke, Heart Disease or Attack, Physical Activity, Fruits, Veggie, Heavy Alcohol Consumption, Any Healthcare, Cost, Gen Health, Mental Health, Physical Health, Diff Walk, Sex, Age, Education and Income.

In this qualitative analysis, the approach also follows a step-by-step process however aims to predict based on classification. Data Exploration was also conducted providing an understanding of the dataset's characteristics. Again, checking the dataset's dimensions and identifying any missing values set the foundation for data integrity. Summary statistics, class distribution and relationship exploration was explored providing a comprehensive understanding of variable distributions and relationships with the predictors. The dataset was then divided into training and testing sets, a standard practice to evaluate how well the model generalizes to unseen data. Various classification algorithms were applied, including logistic regression, k-Nearest Neighbors (KNN), decision trees, random forests, and boosting. The models are trained on the training set, and their performance is evaluated on the testing set using metrics such as accuracy, error, and confusion matrices. Hyperparameters of the chosen algorithms were fine-tuned to optimize performance. For example, different values of "k" are experimented with in KNN, and parameters like and boosting rounds are adjusted in boosting techniques. The impact of these hyperparameter tuning on the model's performance metrics is assessed, ensuring improved performance without overfitting. The goal is to build a classification model that accurately predicts whether or not someone is likely to have diabetes based on significant predictors with the choice of algorithm, data characteristics, and considerations like interpretability and computational efficiency.

Detailed Findings:

Working with the first dataset (quantitative) exploratory data analysis provided us with a comprehensive general understanding of the red wine data. The dataset does not have any missing values and is composed of 1598 rows and 12 columns. The dataset comprises various features related to red wine, such as fixed acidity, volatile acidity, citric acid, residual sugar, chlorides, free sulfur dioxide, total sulfur dioxide, density, pH, sulphates, alcohol, and quality. The summary statistics reveal the ranges and averages for each variable. For instance, the average fixed acidity is 8.321, volatile acidity is 0.528, and so on. The quality of the wine which is our target variable, rated on a scale from 3 to 8, has an average of 5.636. An output of histograms highlighting the quantitative variables that are nonbinary ("density", "pH", "sulphates", "alcohol") were assessed for distribution skewness. Both density and pH were evenly distributed around the mean whereas alcohol and sulphates were left skewed. Looking at computed matrix of correlation of variables, quality has a positive correlation of approximately 0.12 with fixed acidity, 0.22 citric acid and .25 with sulphates. A negative correlation of about -0.39 with volatile acidity. This suggests that higher fixed acidity and citric acid are weakly associated with higher wine quality, while higher volatile acidity is weakly associated with lower quality. Notably, alcohol has a relatively strong positive correlation of about 0.48 with quality. This suggests that wines with higher alcohol content are moderately associated with higher quality in this dataset.

The linear regression model attempts to predict the quality of red wine based on various chemical attributes. The coefficients provide insights into the direction and strength of the relationships between predictor variables and the response variable. The p-values help assess the statistical significance of these relationships. The significant variables (based on a commonly used significance level of 0.05) are Volatile Acidity, Chlorides, Free Sulfur Dioxide, Total Sulfur Dioxide, pH, Sulphates, and Alcohol. These are the variables that have a statistically significant impact on the predicted quality of red wine in this model. The coefficients for these variables indicate their impact on wine quality. For instance, higher volatile acidity and chlorides are associated with lower quality, while higher sulphates and alcohol content are linked to higher quality. The model overall is statistically significant (p-value < 2.2e-16), explaining approximately 36.06% of the variability in wine quality. The residuals, representing the differences between observed and predicted values, have a standard deviation of 0.648. The results of the linear regression analysis on the red wine quality data set provide compelling evidence to reject the null hypothesis for the examined variables. The null hypothesis posits that the coefficients of the predictor variables are equal to zero, implying no significant impact on the response variable (quality). However, the calculated p-values for each variable are well below the commonly accepted threshold of 0.05, indicating that these predictors are indeed statistically significant in influencing wine quality. Therefore, we can confidently reject the null hypothesis and assert that there is a meaningful relationship between volatile acidity, chlorides, free sulfur dioxide, total sulfur dioxide, pH, sulphates, alcohol, and the perceived quality of red wines in this dataset.

These results offer insights into the chemical factors influencing red wine quality, providing a valuable tool for understanding and potentially improving wine production. The overall model is statistically significant, but the individual predictors vary in their impact on the quality of red wine. Looking at a diagnostic plot of this model we can see that the Residuals vs Fitted Values show linearity and homoscedasticity (constant variance) of residuals based on each level of quality and the Normal Q-Q Plot suggests that the residuals are normally distributed and the residuals vs Leverage indicated a high leverage point (152).

The linear regression model provided critical information for choosing significant predictors moving forward. Additional regression models were tested and analyzed. Specifically, I evaluated results from linear regression model with all variables to the model with only significant variables and model with significant and positively correlated variables.

	All Variables	Significant Variables	Significant and Positive Correlated Variables
Residual standard error	0.648 on 1587 degrees of freedom	0.6477 on 1591 degrees of freedom	0.7275 on 1593 degrees of freedom
Multiple R-squared	0.3606	0.3595	0.1909
Adjusted R-squared	0.3561	0.3567	0.1884
F-statistic	81.35 on 11 and 1587	127.6 on 7 and 1591	75.18 on 5 and 1593
p-value	< 2.2e-16	< 2.2e-16	< 2.2e-16

First looking at the models with all variables and significant variables. Both models have similar residual standard errors, indicating that they perform similarly in terms of predicting the response variable (quality). Multiple R-squared and Adjusted R-squared are very close between the two models, suggesting that the model with only significant variables explains a similar proportion of the variability in the response variable as the model with all variables. The F-statistic in the model with only significant variables is higher, indicating a better fit to the data. This is expected since the model is more parsimonious, focusing on the most important variables. The model with all variables has more degrees of freedom due to the additional variables included. Overall, the model with only significant variables performs comparably in terms of explaining the variability in the quality of red wine, while being more concise and interpretable. This suggests that the selected significant variables capture the essential information for predicting wine quality in this dataset. Now comparing the model with significant positively correlated values to the model with all variables and just significant values which has the same p-value as the additional models. However, this model was not beneficial as it has a significantly lower F-Statistic and a lower r squared indicating a worse fit. Considering the specific goals of this analysis, the nature of the data, and the trade-off between model complexity and performance when deciding on the "best" model the best model chosen is the model with just significant values.

The 95% confidence intervals for the coefficients of the significant variables in the linear regression model helped quantify the uncertainty associated with the estimated coefficients, providing a range of plausible values for each variable's effect on wine quality. For example, the 95% confidence interval for the coefficient of alcohol is approximately (0.26,0.32). This suggests that we are 95% confident that the true effect of alcohol content on wine quality falls within this range. The following ranges were analyzed for the remainder of coefficients: 95% confidence interval for the intercept is approximately (3.64,5.22), 95% confidence interval for the coefficient of volatile acidity is approximately (−1.21,−0.81), 95% confidence interval for the coefficient of chlorides is approximately (−2.80,−1.24), 95% confidence interval for the coefficient of free sulfur dioxide is approximately (0.0009,0.0092), 95% confidence interval for the coefficient of total sulfur dioxide is approximately (−0.0048,−0.0021), 95% confidence interval for the coefficient of pH is approximately (−0.71,−0.25), 95% confidence interval for the coefficient of sulphates is approximately (0.67,1.10), 95% confidence interval for the coefficient of alcohol is approximately (0.26,0.32).

Once the data was split into training and test data a simple linear regression was performed on a randomly split dataset, making predictions on the test set, and calculating the RMSE as a measure of the model's performance on the unseen data. RMSE is a metric that quantifies the average difference between the predicted and actual values. A lower RMSE indicates that the model's predictions are closer to the actual values. The specific value (0.6744579) provided a numerical measure of how well the linear regression model performs on the test set in terms of predicting wine quality and will be compared against future models.

Next, I performed ridge regression which is a regularization method that adds a penalty to the linear regression objective function, improving the generalization performance of the model and addressing issues associated with multicollinearity and overfitting. The ridge regression technique introduces a regularization parameter (lambda) that controls the strength of the penalty term. The choice of this parameter is crucial, and it can be selected through techniques like cross-validation. The ridge regression performed provided a RMSE of .6745. The cross-validation plot shows how the coefficients are affected by the regularization. As lambda increases, the penalty on large coefficients becomes more substantial, leading to a greater shrinkage of the coefficients. The variables with steeper lines are more sensitive to changes in lambda. The final ridge regression provided a higher RMSE of .8244 indicating a poorer model.

In the Lasso regression model results provided, the coefficients offer valuable insights into the relationships between predictor variables and the dependent variable, considering the regularization effect of the Lasso algorithm. The intercept term, representing the expected value of the dependent variable when all predictors are zero, remains present but may be influenced by Lasso's tendency to shrink some coefficients to zero for variable selection. The second intercept with a value of "." suggests that the intercept might not be as relevant in this specific Lasso model. Notably, variables like "volatile acidity," "chlorides," "pH," "sulphates," and "alcohol" have non-zero coefficients, indicating that Lasso has retained them in the model as relevant predictors. The negative coefficients for "volatile acidity," "chlorides," and "pH" suggest negative impacts on the dependent variable, while the positive coefficients for "sulphates" and

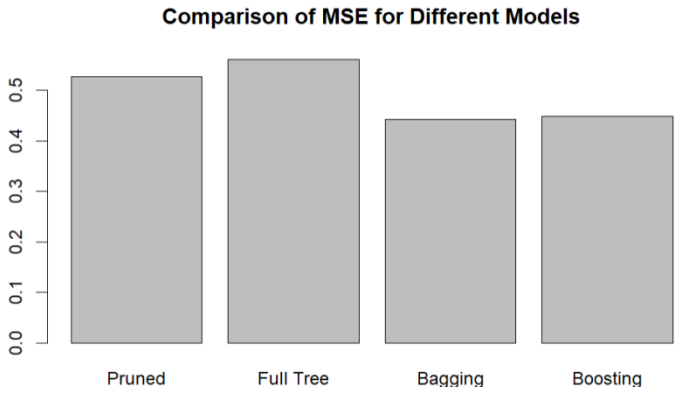
"alcohol" indicate positive impacts. The smaller magnitudes of some coefficients, such as for "free sulfur dioxide" and "total sulfur dioxide," imply reduced impacts, potentially even to the point of insignificance. In summary, these results showcase the Lasso model's capacity for variable selection, effectively shrinking some coefficients while retaining others based on their relevance and impact on the dependent variable. The overall RMSE for the lasso model was .6766. like ridge regression the lambda value was obtained (2) and tested which provided a final RMSE of .8606. For the continuation of the comparison of models I performed PCR and PLS to assess the model performance through validation plots, and evaluate the prediction accuracy using RMSE on the test set. Summary statistics show that the PCR model achieves better predictive performance with an increasing number of components and that adding components beyond 3 does not lead to a significant improvement in predictive performance and the RMSE for PCR is .6808. Result from PLS model indicated a RMSE of .6750. The RMSE values of the models were compared along with r squared.

	RMSE
OLS	.6744
Ridge	.6745
Lasso	.677
PCR	.6808
PLS	.675

Next, a regression tree fit on the training data. The tree regression model is trying to predict quality explained by all the other variables in the data set. Results of the regression tree, showed 14 terminal nodes, effectively captures the patterns in the red wine quality dataset. The residuals, with a mean close to zero, suggest that the model is making predictions that are, on average, unbiased. The distribution of residuals shows that the model tends to have errors within a reasonable range, providing a nuanced understanding of its predictive performance across different quality levels. The regression tree uses 7 predictor variables to make predictions about the "quality" outcome. The tree has 14 terminal nodes, suggesting that it divides the dataset into nine distinct groups based on the predictor variables. The low residual mean deviance .3317 indicates a good fit of the model to the training data and has had the lowers RMSE so far of .5274. The initial tree, (red.tree) demonstrated good predictive performance on the test set, as indicated by the mean squared error.. This suggests that the model captures the relationships between the selected features and wine quality reasonably well. For cross-validation the plot of deviance against the number of trees helped identify an optimal tree size, facilitating a balance between model complexity and predictive accuracy. However, the pruned tree, (prune.red) represents a more generalized version that may avoid overfitting to the training data. The mean squared error increased slightly after pruning (0.6175), reflecting a trade-off between model complexity and performance. This comparison suggests that pruning the decision tree to 4 nodes has led to an increase in the Test MSE compared to the unpruned tree. In other words, the

unpruned tree is performing better on the test dataset in terms of Mean Squared Error. While the pruned tree is simpler and less prone to overfitting, it may sacrifice a small amount of predictive accuracy compared to the original, more complex tree. The choice between the original and pruned tree depends on the specific questions of the analysis, balancing model simplicity and predictive performance and comparison to additional models.

The same method with training and test datasets were performed with bagging. In comparison, bagging has decreased the error to .439 providing a more improved mode (from:.527 of unpruned and .561 pruned). This indicates that the bagging model with 10 randomly selected predictors (best mtry) at each split performs well on the test set, as evidenced by the relatively low mean squared error. Bagging, by aggregating predictions from multiple trees trained on bootstrapped samples, tends to enhance model robustness, and mitigate overfitting, contributing to improved predictive accuracy. The same method was also performed for boosting which provided a MSE of .4476. All of the models provided additional insight on the effects of the predictors on the quality target variable with similar MSE. Bagging only slightly outperformed the other models with specific parameters. The overall efficiency and simplicity of bagging along with the selected predictor variable have shown it to be the best fit when predicting the quality of red wine.



Working with the second dataset (qualitative) exploratory data analysis provided us with a comprehensive understanding of the diabetes data. The dataset does not have any missing values and is composed of 70692 rows and 22 columns. The dataset comprises various features related to diabetes, such as has Diabetes, High BP, High Chol, Chol Check, BMI, Smoker, Stroke. The summary statistics reveal the ranges and averages for each variable we also find that some of the data is binary. Our target variable diabetes is binary with values 0 or 1, indicating the absence or presence of diabetes. Histograms for the selected qualitative non binary values were assessed showing the distribution and skewness of variables.

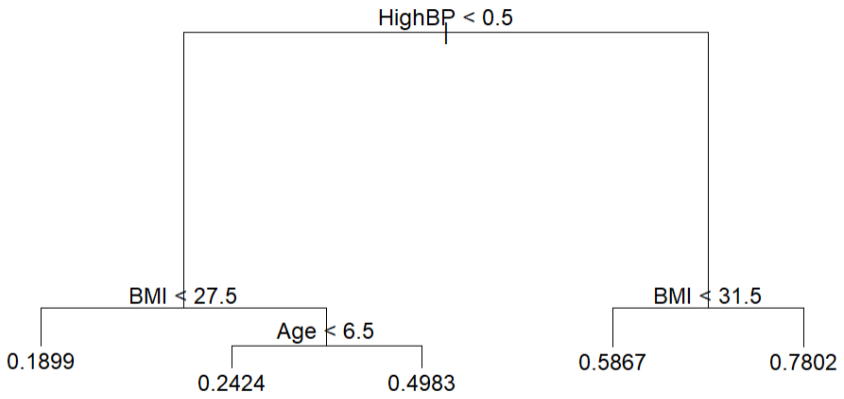
A Correlation with Diabetes binary shows the highest positive correlation with Diabetes binary is GenHlth with a correlation coefficient of approximately 0.41. This suggests that as the general

health perception increases, the likelihood of having diabetes also increases. Other variables with notable positive correlations include Age (0.28) and HighBP (0.38). The variable with the highest negative correlation is PhysActivity with a coefficient of approximately -0.16. This implies that higher levels of physical activity are associated with a lower likelihood of having diabetes. There are also correlations among other pairs of variables. For example, GenHlth has positive correlations with variables such as PhysHlth (0.55), DiffWalk (0.48), and Age (0.16). Variables like Income and Education have a relatively strong positive correlation of 0.46. Variables such as Fruits, Veggies, and HvyAlcoholConsump have relatively low correlations with Diabetes binary. This suggests that these lifestyle factors may not be strongly correlated with the presence of diabetes in the dataset. Sex has a very low positive correlation with Diabetes binary (0.04), indicating a weak association between gender and diabetes in this dataset.

A logistic regression was performed with diabetes as the response and all variables as predictors. Results concluded passed of p values that the following predictors are shown to be significant: High Blood, Pressure (HighBP), High Cholesterol (HighChol), Cholesterol Check (CholCheck), Body Mass Index (BMI), Stroke, Heart Disease or Heart Attack, General Health (GenHlth), Difficulty Walking (DiffWalk), Sex, Age, Education, and Income. Additional logistic regressions were performed by removing predictors and assessing the significance of the models in comparison to one another. As additional models where created less significant predictors were removed when choosing a final set of predictors. Deviance residuals measure how well the model fits the data. Results showed the values are extremely close to zero, indicating a very good fit overall. The residuals are at the scale of 10-6, which also suggests that the model fits the data very well. The estimated coefficients for the estimates are very close to zero, and the standard errors are very large. This might however indicate a problem with the model fitting process, leading to unreliable coefficient estimates. For each iteration of the logistic regression model a contingency table was created evaluating the performance of the logistic regression model for predicting the accuracy whether individuals have diabetes. The first logistic regression model shoed an error rate of .2519 and the final model with 10 significant predictors had only a slightly increased error rate of .2659. The data was then split into a train and test data and the logistic regression was ran under these conditions. Results indicated an increase in error of the model of .3040. and a lower AIC score indicating a better fit (52898 vs 75568).

An LDA model was then used to predict diabetes status based on various predictors using training data and make predictions on test data. A confusion matrix was then used to evaluate the model's performance, and calculated the accuracy of the model. Accuracy results where .2664 showing a lower accuracy in comparison to the logistic regression. The same environment was set up and a naive bayes model was ran showing an accuracy of .2746. QDA was also compared in the same training and test environment providing an accuracy of .2861. KNN (k-Nearest Neighbors) was then used to make predictions by considering the majority class or average value of the nearest data points to a given instance in the feature space. This model relies on memorizing the training data to classify new instances based on their proximity to known data points. Results from this model were looked at where k=5 and k =10 indicating the number of neighbors. The accuracy of the KNN model where k=5 was .7156 (error=.2843) and the accuracy where k=10 was .7219 (error=.2780)

The decision tree model was constructed using the significant variables from the previous models after splitting the data into training and test. The residual mean deviance of this model is 0.2007 reflecting the average squared difference between the predicted and observed values, normalized by the degrees of freedom (35340) and incorporated the predictors HighBP, BMI and age. A lower deviance suggests a better fit of the model to the training data. In this case, the residual mean deviance is reasonably low, indicating a relatively good fit. In addition to this, the mean squared error is .2011 with a total of 5 nodes for this model.



Next, cross-validation was performed on the decision tree model and a plot was used to visualize how the model's performance changes with different numbers of trees. Results from the plot indicated the best number of trees is 4. The mean squared error from the new model is .2038. This error rate is comparative to the original model but provides a simpler model. While looking to reduce error the importance of balancing parsimony verses black box comes to play. The first tree model has one of the overall lowest errors compared to all models in this analysis. This model also provides a sufficient amount of predictors without over or under doing it and providing a good fit.

Validity & Reliability Assessment:

The accuracy of the recommendations plays an important role in many ways and should be tested in future implementation. Analysis are always balancing the tradeoff between bias and variance where this trade off looks to see how close the prediction is to the actual. In this case we would be trying to balance accuracy and uncertainty and testing for these parameters can continue to build on an effective model.

One way we can test this is to look at the validity of the data overall. Do all of the predictors reflect the relationships we are testing in the model. In the diabetes data set I came across this with the variable no deductible cost which happened to work out as it was a non-significant predictor. Another way that the accuracy can be tested in the future is by comparing against additional sample dataset to show reliability. Comparing and contrasting this consistent analysis will help us to learn more about our recommendations. One example of this can be done with the wine dataset. As mentioned previously the red wine data set was used for the previous associated models however a white wine dataset was also available in which the results from using the same model techniques would help for future implementation and knowledge of the datasets and model performance.

In addition to this, assessment techniques can be used to further investigate the models. In this analysis I often used various types of cross validation to assess the performance and generalizability of predictive models. The primary goal of cross-validation is to provide an unbiased evaluation of a model's ability to predict new, unseen data. For example, in this analysis I techniques like k-fold cross-validation to assess the model's performance on multiple subsets of the data, ensuring validity and reliability. Specifications of the choice of k (e.g., 5-fold, 10-fold) provided more variety to the performance technique. In the future, other common assessment techniques should be looked at while working with the range or parameters to get a full scope of the model's performance. All of these examples emphasize the importance of ongoing evaluation and testing in both regression and classification problems to ensure that models are both valid (meaningful and accurate) and reliable (consistent and stable).

References:

Cortez, P., Cerdeira, A.L., Almeida, F., Matos, T., & Reis, J. (2009). Modeling wine preferences by data mining from physicochemical properties. *Decis. Support Syst., 47*, 547-553.

James G., Witten D., Hastie T., &Tibshirani R. (2021). An Introduction to Statistical Learning with Applications in R, Second Edition.

Predictive Modeling Final

Alisha Souza

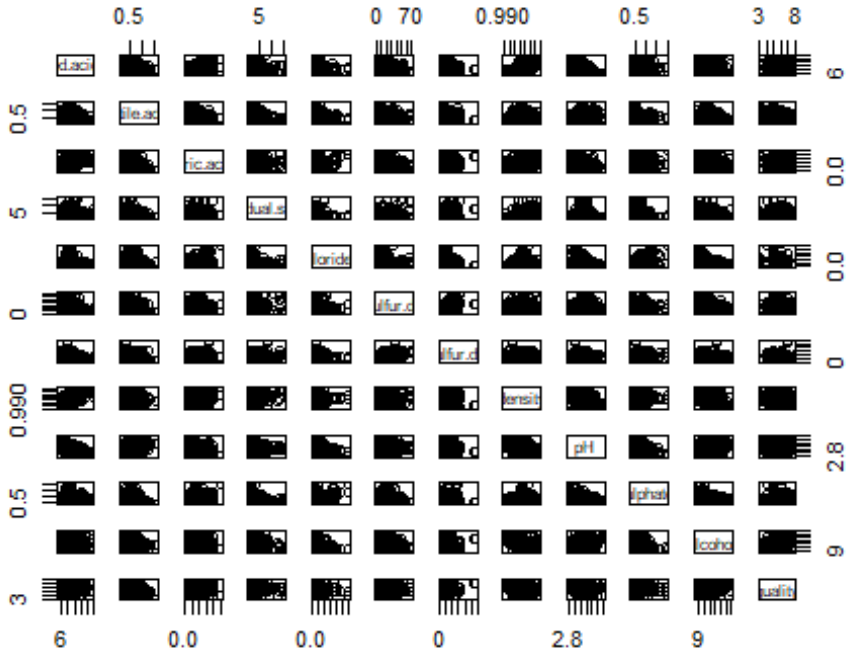
2023-12-10

```
library(tidyverse)

## ── Attaching core tidyverse packages ──────────────────────── tidyverse
2.0.0 ──
## ✔ dplyr     1.1.1     ✔ readr     2.1.4
## ✔ forcats   1.0.0     ✔ stringr   1.5.0
## ✔ ggplot2   3.4.4     ✔ tibble    3.2.1
## ✔ lubridate 1.9.2     ✔ tidyr     1.3.0
## ✔ purrr     1.0.1
## ── Conflicts ────────────────────────────────────────── tidyverse_conflicts() ──
## ✖ dplyr::filter() masks stats::filter()
## ✖ dplyr::lag()    masks stats::lag()
## ℹ Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to
become errors

library(dplyr)
library(stringr)
library(tidyverse)
library(ggplot2)
library(GGally)

## Registered S3 method overwritten by 'GGally':
##   method from
##   +.gg   ggplot2
```

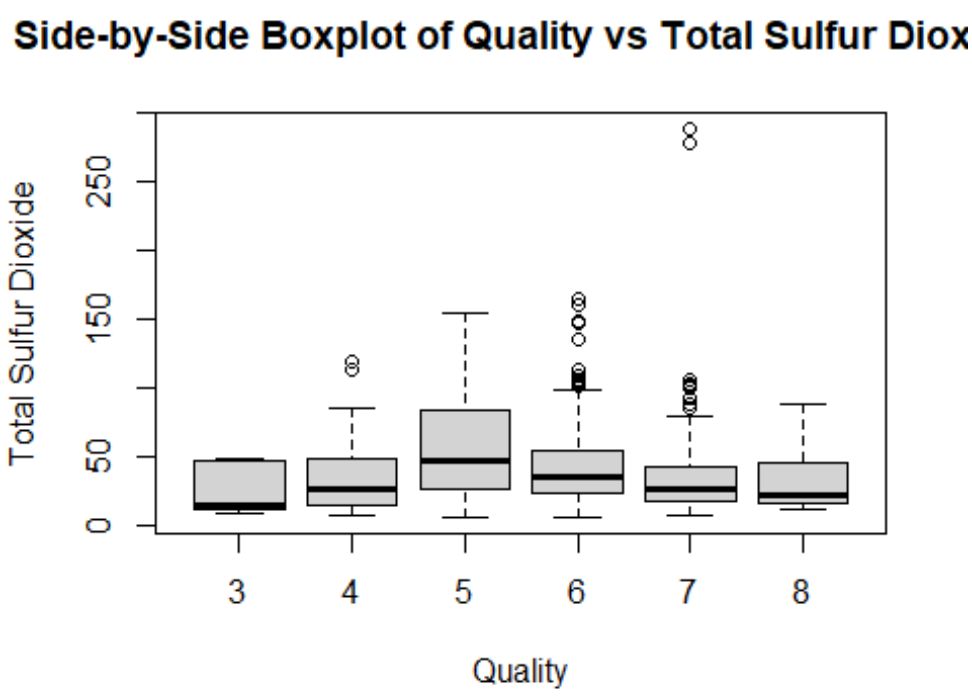
```
library(readr)
library(corrplot)

## corrplot 0.92 loaded

library(ggcorrplot)
library(glmnet)

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

library(tree)
require(randomForest)

## Loading required package: randomForest
## randomForest 4.7-1.1
## Type rfNews() to see new features/changes/bug fixes.
##
## Attaching package: 'randomForest'
##
## The following object is masked from 'package:dplyr':
##
##     combine
##
## The following object is masked from 'package:ggplot2':
```

```
##
##     margin

library(gbm)

## Loaded gbm 2.1.8.1

library(pls)

##
## Attaching package: 'pls'
##
## The following object is masked from 'package:corrplot':
##
##     corrplot
##
## The following object is masked from 'package:stats':
##
##     loadings

red_wine_quality<-read.csv("C:\\Users\\alish\\Downloads\\M.S. Data
Science\\winequality-red.csv")

#check number of rows and columns
dim(red_wine_quality)

## [1] 1598   12

# Check for any missing values in the entire data frame
has_na <- any(is.na(red_wine_quality))


print(has_na)

## [1] FALSE

summary(red_wine_quality)
```

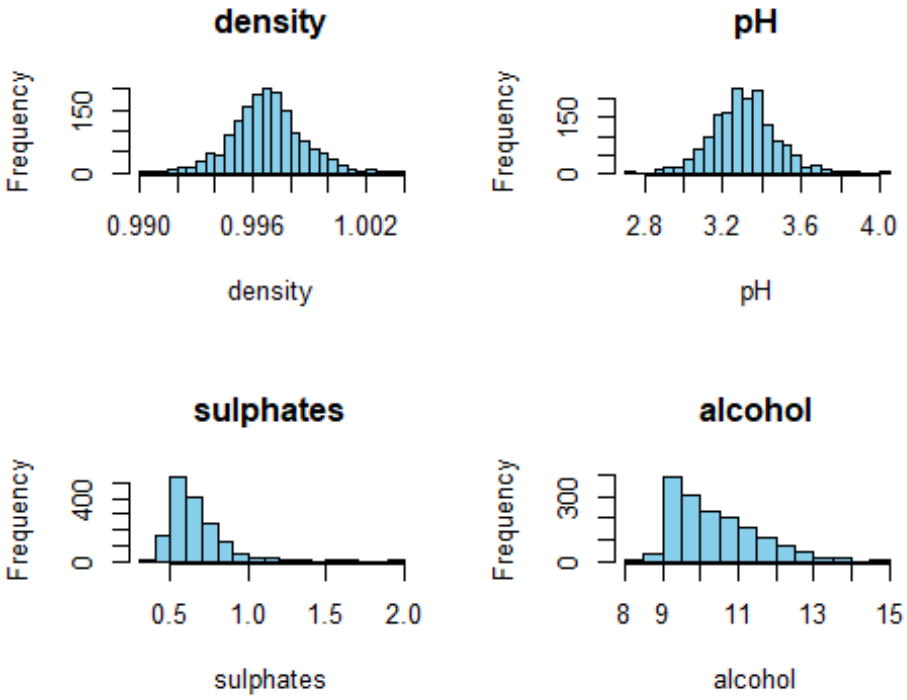
```
##  fixed.acidity    volatile.acidity  citric.acid     residual.sugar
##  Min.   : 4.600   Min.   :0.120    Min.   :0.0000   Min.   : 0.900
##  1st Qu.: 7.100   1st Qu.:0.390    1st Qu.:0.0900   1st Qu.: 1.900
##  Median : 7.900   Median :0.520    Median :0.2600   Median : 2.200
##  Mean   : 8.321   Mean   :0.528    Mean   :0.2709   Mean   : 2.538
##  3rd Qu.: 9.200   3rd Qu.:0.640    3rd Qu.:0.4200   3rd Qu.: 2.600
##  Max.   :15.900   Max.   :1.580    Max.   :1.0000   Max.   :15.500
##    chlorides       free.sulfur.dioxide total.sulfur.dioxide    density
##  Min.   :0.01200   Min.   : 1.00       Min.   :  6.00       Min.   :0.9901
##  1st Qu.:0.07000   1st Qu.: 7.00       1st Qu.: 22.00       1st Qu.:0.9956
##  Median :0.07900   Median :14.00       Median : 38.00       Median :0.9968
##  Mean   :0.08748   Mean   :15.87       Mean   : 46.47       Mean   :0.9967
##  3rd Qu.:0.09000   3rd Qu.:21.00       3rd Qu.: 62.00       3rd Qu.:0.9978
##  Max.   :0.61100   Max.   :72.00       Max.   :289.00       Max.   :1.0037
##        pH          sulphates         alcohol         quality
##  Min.   :2.740   Min.   :0.3300   Min.   : 8.40   Min.   :3.000
##  1st Qu.:3.210   1st Qu.:0.5500   1st Qu.: 9.50   1st Qu.:5.000
##  Median :3.310   Median :0.6200   Median :10.20   Median :6.000
##  Mean   :3.311   Mean   :0.6581   Mean   :10.42   Mean   :5.636
##  3rd Qu.:3.400   3rd Qu.:0.7300   3rd Qu.:11.10   3rd Qu.:6.000
##  Max.   :4.010   Max.   :2.0000   Max.   :14.90   Max.   :8.000

#scatter plot matrix of variables in red wine quality data set
pairs(red_wine_quality)
```



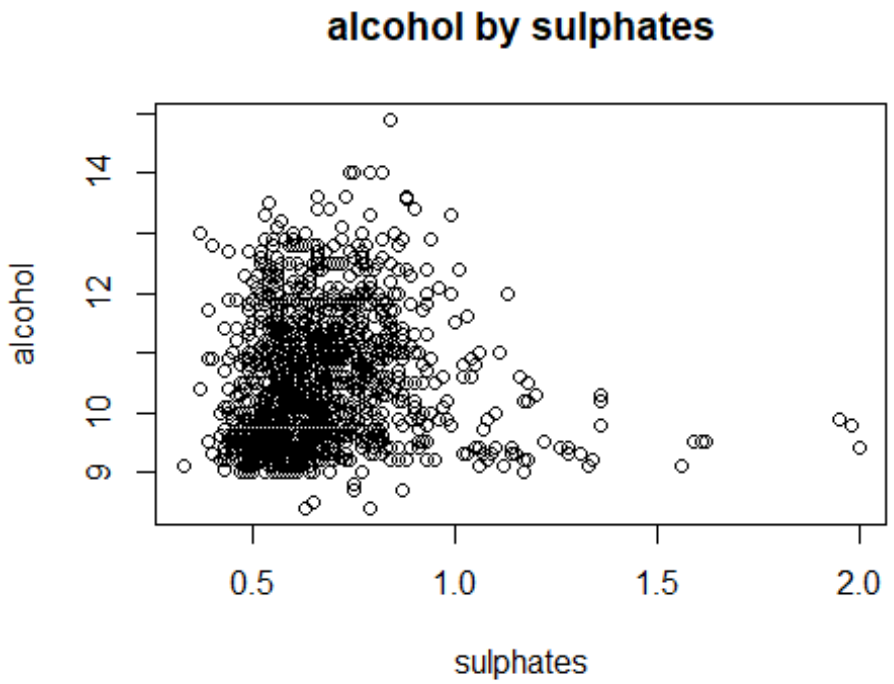
```
boxplot(total.sulfur.dioxide ~ quality, data = red_wine_quality, xlab = "Quality", ylab =
"Total Sulfur Dioxide",
        main = "Side-by-Side Boxplot of Quality vs Total Sulfur Dioxide")
```




```r
quantitative_vars <- c("density", "pH", "sulphates", "alcohol")
par(mfrow = c(2, 2))

for (var in quantitative_vars) {
  hist(red_wine_quality[[var]], main = var, xlab = var, col = "skyblue", breaks = 20)
}
```




```
par(mfrow = c(1, 1))

plot(alcohol ~ sulphates, data = red_wine_quality, xlab = "sulphates", ylab = "alcohol",
     main = "alcohol by sulphates")
```



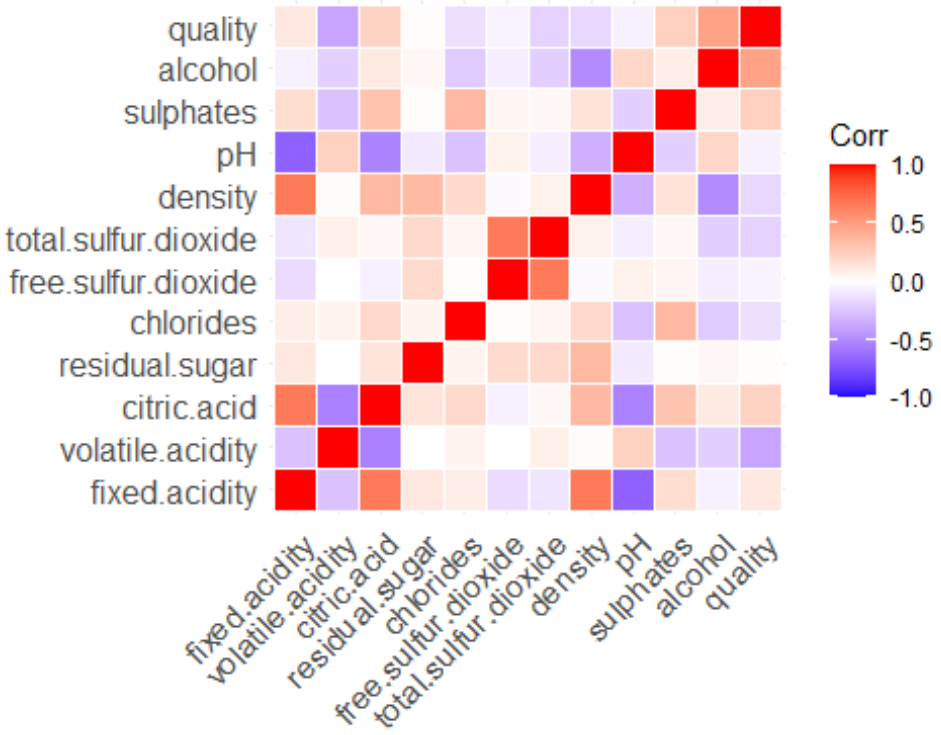

```
#computed matrix of correlation of variables
cor(red_wine_quality)

##                       fixed.acidity volatile.acidity citric.acid residual.sugar
## fixed.acidity          1.00000000     -0.257408032  0.67314986    0.115489296
## volatile.acidity      -0.25740803      1.000000000 -0.55215372    0.002492795
## citric.acid            0.67314986     -0.552153719  1.00000000    0.143167832
## residual.sugar         0.11548930      0.002492795  0.14316783    1.000000000
## chlorides              0.09339986      0.060998412  0.20417993    0.055827687
## free.sulfur.dioxide   -0.15371214     -0.010354008 -0.06112887    0.186988831
## total.sulfur.dioxide  -0.11335842      0.076402396  0.03563177    0.203129096
## density                0.66795579      0.021532007  0.36554336    0.355709648
## pH                    -0.68298738      0.235454862 -0.54245267   -0.085915501
## sulphates              0.18311657     -0.261099373  0.31286533    0.005522954
## alcohol               -0.06125623     -0.201987722  0.10960273    0.041831509
## quality                0.12450477     -0.390420212  0.22617248    0.013522480
##                      chlorides free.sulfur.dioxide total.sulfur.dioxide
```

```
## fixed.acidity         0.093399858       -0.153712139        -0.11335842
## volatile.acidity      0.060998412       -0.010354008         0.07640240
## citric.acid           0.204179935       -0.061128871         0.03563177
## residual.sugar        0.055827687        0.186988831         0.20312910
## chlorides             1.000000000        0.005617871         0.04736656
## free.sulfur.dioxide   0.005617871        1.000000000         0.66769621
## total.sulfur.dioxide  0.047366564        0.667696213         1.00000000
## density               0.200490720       -0.021864444         0.07122315
## pH                   -0.264924338        0.070319154        -0.06645692
## sulphates             0.371285451        0.051656852         0.04294801
## alcohol              -0.221026482       -0.069484511        -0.20562796
## quality              -0.128799652       -0.050717274        -0.18507480
##                       density         pH    sulphates     alcohol
## fixed.acidity         0.66795579 -0.68298738  0.183116572 -0.06125623
## volatile.acidity      0.02153201  0.23545486 -0.261099373 -0.20198772
## citric.acid           0.36554336 -0.54245267  0.312865333  0.10960273
## residual.sugar        0.35570965 -0.08591550  0.005522954  0.04183151
## chlorides             0.20049072 -0.26492434  0.371285451 -0.22102648
## free.sulfur.dioxide  -0.02186444  0.07031915  0.051656852 -0.06948451
## total.sulfur.dioxide  0.07122315 -0.06645692  0.042948015 -0.20562796
## density               1.00000000 -0.34156163  0.148531589 -0.49606841
## pH                   -0.34156163  1.00000000 -0.196667181  0.20549493
## sulphates             0.14853159 -0.19666718  1.000000000  0.09359964
## alcohol              -0.49606841  0.20549493  0.093599643  1.00000000
## quality              -0.17476669 -0.05788401  0.251409996  0.47608749
##                       quality
## fixed.acidity         0.12450477
## volatile.acidity     -0.39042021
## citric.acid           0.22617248
## residual.sugar        0.01352248
## chlorides            -0.12879965
```

```
## free.sulfur.dioxide  -0.05071727
## total.sulfur.dioxide -0.18507480
## density              -0.17476669
## pH                   -0.05788401
## sulphates             0.25141000
## alcohol               0.47608749
## quality               1.00000000
```

#quality and alcohol have the highest correlation (in relation to quality)

```
cor_matrix <- cor(red_wine_quality)



# Create a square correlation plot
ggcorrplot(cor_matrix, type="full", outline.col = "white", lab_size = 3)
```



#This plot is a visual representation of the correlation matrix. Each square in the plot represents the correlation between two variables, with color intensity indicating the strength and direction of the correlation. Positive correlations arerepresented in shades of blue, while negative correlations

are represented in shades of red.The size and intensity of the squares provide a visual summary of the relationships between variables in the dataset.

```
#linear regression model to predict the quality of red wine based on all available variables in the "red_wine_quality" data set

red.lm.model1 <- lm(quality ~ ., data = red_wine_quality)
summary(red.lm.model1)

##
## Call:
## lm(formula = quality ~ ., data = red_wine_quality)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2.6891 -0.3665 -0.0476  0.4521  2.0250
##
## Coefficients:
##                       Estimate Std. Error t value Pr(>|t|)
## (Intercept)          2.196e+01  2.121e+01    1.035   0.3007
## fixed.acidity         2.497e-02  2.601e-02    0.960   0.3372
## volatile.acidity     -1.084e+00  1.211e-01  -8.945  < 2e-16 ***
## citric.acid          -1.825e-01  1.475e-01  -1.237   0.2161
## residual.sugar        1.633e-02  1.501e-02    1.088   0.2766
## chlorides            -1.874e+00  4.196e-01  -4.468 8.47e-06 ***
## free.sulfur.dioxide   4.362e-03  2.172e-03    2.008   0.0448 *
## total.sulfur.dioxide -3.265e-03  7.292e-04  -4.477 8.11e-06 ***
## density              -1.787e+01  2.165e+01  -0.826   0.4091
## pH                   -4.137e-01  1.917e-01  -2.158   0.0311 *
## sulphates             9.163e-01  1.144e-01   8.011 2.18e-15 ***
## alcohol               2.762e-01  2.649e-02  10.425  < 2e-16 ***
## ---
```

```
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.6482 on 1586 degrees of freedom
## Multiple R-squared:  0.3605, Adjusted R-squared:  0.356
## F-statistic: 81.27 on 11 and 1586 DF,  p-value: < 2.2e-16
```

#linear regression model: the algorithm used the least squares method to find the coefficients that minimize the sum of squared differences between the actual and predicted values of the dependent variable (wine quality). This method is widely used for its simplicity and mathematical tractability. The results obtained, including coefficient estimates, standard errors, t-values, and p-values, are based on the least squares approach.

#This linear regression model attempts to predict the quality of red wine based on various chemical attributes. The coefficients provide insights into the direction and strength of the relationships between predictor variables and the response variable. The p-values help assess the statistical significance of these relationships.

#he significant variables (based on a commonly used significance level of 0.05) are Volatile Acidity, Chlorides, Free Sulfur Dioxide, Total Sulfur Dioxide, pH, Sulphates, and Alcohol. These are the variables that have a statistically significant impact on the predicted quality of red wine in this model.

#The model considers variables such as volatile acidity, chlorides, sulphates, and alcohol. The coefficients for these variables indicate their impact on wine quality. For instance, higher volatile acidity and chlorides are associated with lower quality, while higher sulphates and alcohol content are linked to higher quality. The model overall is statistically significant (p-value < 2.2e-16), explaining approximately 36.06% of the variability in wine quality. The residuals, representing the differences between observed and predicted values, have a standard deviation of 0.648. These results offer insights into the chemical factors influencing red wine quality, providing a valuable tool for understanding and potentially improving wine production.The overall model is statistically significant, but the individual predictors vary in their impact on the quality of red wine.

#Interpertation of coefficients showing how each of these chemical attributes effects the quality of red wine. #Volatile Acidity (Coefficient: -1.084) #For each one-unit increase in volatile acidity, the model predicts a decrease of approximately 1.084 units in the quality of red wine. #Higher levels of volatile acidity are associated with lower quality wine.

#Chlorides (Coefficient: -1.874) #Higher levels of chlorides are associated with lower quality wine.

#Free Sulfur Dioxide (Coefficient: 0.004361) #Higher levels of free sulfur dioxide are associated with higher quality wine.

#Total Sulfur Dioxide (Coefficient: -0.003265) #Higher levels of total sulfur dioxide are associated with lower quality wine.

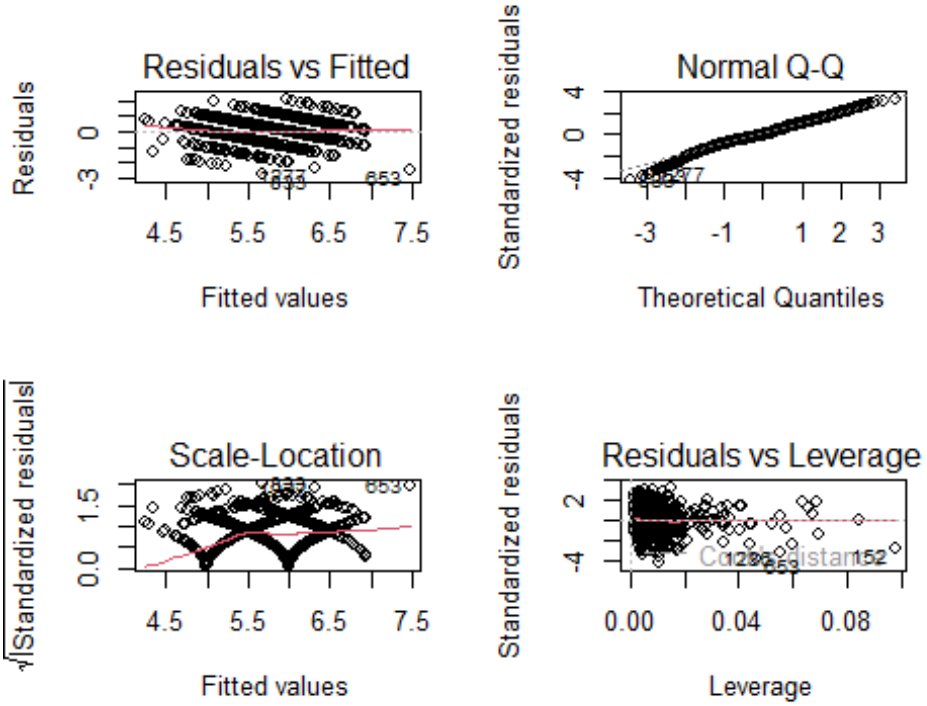
#pH (Coefficient: -0.4137) #Higher pH levels are associated with lower quality wine.

#Sulphates (Coefficient: 0.9163) #Higher levels of sulphates are associated with higher quality wine.

#Alcohol (Coefficient: 0.2762) #Higher alcohol content is associated with higher quality wine.

#The results of the linear regression analysis on the red wine quality data set provide compelling evidence to reject the null hypothesis for the examined variables. The null hypothesis posits that the coefficients of the predictor variables are equal to zero, implying no significant impact on the response variable (quality). However, the calculated p-values for each variable are well below the commonly accepted threshold of 0.05, indicating that these predictors are indeed statistically significant in influencing wine quality. Therefore, we can confidently reject the null hypothesis and assert that there is a meaningful relationship between volatile acidity, chlorides, free sulfur dioxide, total sulfur dioxide, pH, sulphates, alcohol, and the perceived quality of red wines in this dataset.

```
par(mfrow = c(2, 2))
plot(red.lm.model1)
```



```
#linear regression model to predict the quality of red wine based on only significant
variables in the "red_wine_quality" data set

red.lm.model_significant <- lm(quality ~ volatile.acidity + chlorides + free.sulfur.dioxide +
                          total.sulfur.dioxide + pH + sulphates + alcohol,
                          data = red_wine_quality)

summary(red.lm.model_significant)

##
## Call:
## lm(formula = quality ~ volatile.acidity + chlorides + free.sulfur.dioxide +
##     total.sulfur.dioxide + pH + sulphates + alcohol, data = red_wine_quality)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2.68925 -0.36781 -0.04617  0.46081  2.02955
##
## Coefficients:
##                       Estimate Std. Error t value Pr(>|t|)
## (Intercept)           4.429714   0.403068  10.990  < 2e-16 ***
## volatile.acidity     -1.013024   0.100925 -10.037  < 2e-16 ***
## chlorides            -2.017846   0.397666  -5.074 4.35e-07 ***
## free.sulfur.dioxide   0.005078   0.002126   2.388    0.017 *
## total.sulfur.dioxide -0.003482   0.000687  -5.069 4.47e-07 ***
## pH                   -0.482495   0.117611  -4.102 4.29e-05 ***
## sulphates             0.882622   0.109944   8.028 1.91e-15 ***
## alcohol               0.289306   0.016801  17.220  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
```

```
## Residual standard error: 0.6479 on 1590 degrees of freedom
## Multiple R-squared:  0.3594, Adjusted R-squared:  0.3566
## F-statistic: 127.4 on 7 and 1590 DF,  p-value: < 2.2e-16

#linear regression model to predict the quality of red wine based on significant variables
that are positivly correlated variables in the "red_wine_quality" data set

red.lm.model_significant_pos_cor <- lm(quality ~ volatile.acidity + chlorides +
free.sulfur.dioxide +
                        total.sulfur.dioxide + pH,
                        data = red_wine_quality)

summary(red.lm.model_significant_pos_cor)

##
## Call:
## lm(formula = quality ~ volatile.acidity + chlorides + free.sulfur.dioxide +
##     total.sulfur.dioxide + pH, data = red_wine_quality)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2.78996 -0.51312 -0.01359  0.46133  3.08802
##
## Coefficients:
##                       Estimate Std. Error t value Pr(>|t|)
## (Intercept)          7.1393962  0.4272133  16.712  < 2e-16 ***
## volatile.acidity    -1.6334104  0.1068935 -15.281  < 2e-16 ***
## chlorides           -1.7583639  0.4048644  -4.343 1.49e-05 ***
## free.sulfur.dioxide  0.0070096  0.0023857   2.938  0.00335 **
## total.sulfur.dioxide -0.0052667  0.0007614  -6.917 6.64e-12 ***
## pH                  -0.1068938  0.1291878  -0.827  0.40812
## ---
```

```
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.7278 on 1592 degrees of freedom
## Multiple R-squared:  0.1908, Adjusted R-squared:  0.1883
## F-statistic: 75.09 on 5 and 1592 DF,  p-value: < 2.2e-16
```

#comparing results from linear regression model with all variables to the model with only significant variables and model with significant and positivly correlated variables.

#All variables #Residual standard error: 0.648 on 1587 degrees of freedom #Multiple R-squared: 0.3606, Adjusted R-squared: 0.3561 #F-statistic: 81.35 on 11 and 1587 DF, p-value: < 2.2e-16

#Significant variables #Residual standard error: 0.6477 on 1591 degrees of freedom #Multiple R-squared: 0.3595, Adjusted R-squared: 0.3567 #F-statistic: 127.6 on 7 and 1591 DF, p-value: < 2.2e-16

#Significant positive correlated #Residual standard error: 0.7275 on 1593 degrees of freedom #Multiple R-squared: 0.1909, Adjusted R-squared: 0.1884 #F-statistic: 75.18 on 5 and 1593 DF, p-value: < 2.2e-16

#Residual Standard Error: Both models have similar residual standard errors, indicating that they perform similarly in terms of predicting the response variable (quality).

#Multiple R-squared and Adjusted R-squared:The values are very close between the two models, suggesting that the model with only significant variables explains a similar proportion of the variability in the response variable as the model with all variables.

#F-statistic:The F-statistic in the model with only significant variables is higher, indicating a better fit to the data. This is expected since the model is more parsimonious, focusing on the most important variables.

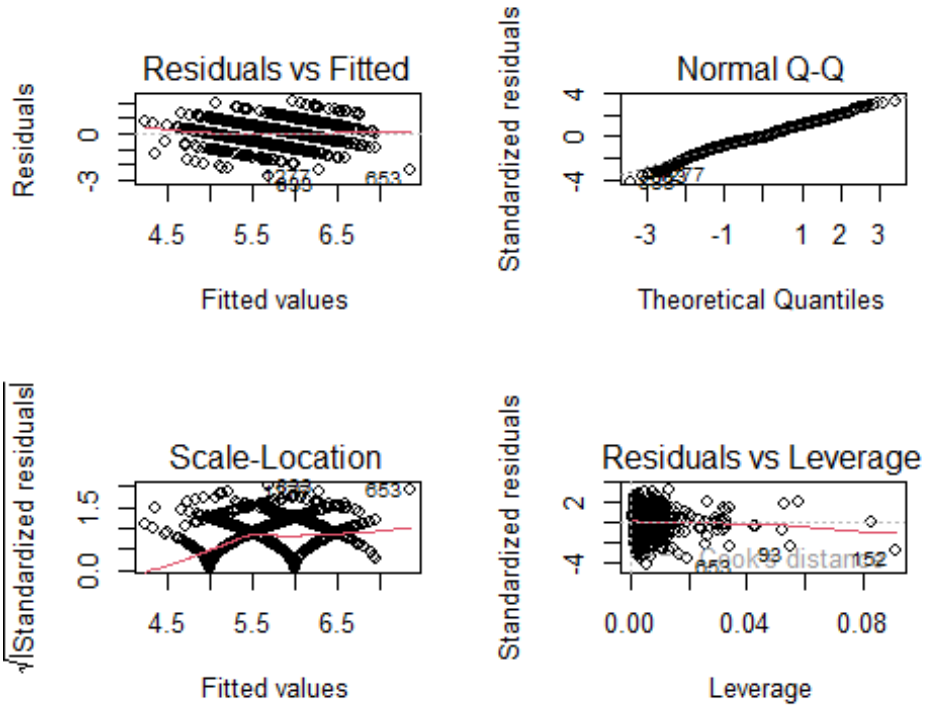
#Degrees of Freedom:The model with all variables has more degrees of freedom due to the additional variables included.

#Overall, the model with only significant variables performs comparably in terms of explaining the variability in the quality of red wine, while being more concise and interpretable. This suggests that the selected significant variables capture the essential information for predicting wine quality in this dataset.

#Comparing the model with significant positively correlated values to the model with all variables and just significant values which has the same p-value as the additional models. However this model was not beneficial as it has a significantly lower F-Statistic and a lower r squared indicating a worse fit.

#Considering the specific goals of your analysis, the nature of your data, and the trade-off between model complexity and performance when deciding on the “best” model the best model is the model with just significant values. Additional validation or cross-validation can provide insights into the model’s generalization performance.

```
par(mfrow = c(2, 2))
plot(red.lm.model_significant)
```




```
confint(red.lm.model_significant)
```

```
##                          2.5 %       97.5 %
## (Intercept)          3.6391129405  5.220315075
## volatile.acidity    -1.2109844217 -0.815063116
## chlorides           -2.7978511928 -1.237841520
## free.sulfur.dioxide  0.0009075961  0.009248534
## total.sulfur.dioxide -0.0048298515 -0.002134818
## pH                  -0.7131845969 -0.251806247
## sulphates            0.6669718642  1.098272467
## alcohol              0.2563516626  0.322260869
```

#The 95% confidence intervals for the coefficients of the significant variables in the linear regression model help quantify the uncertainty associated with the estimated coefficients, providing a range of plausible values for each variable's effect on wine quality.

#Intercept:The 95% confidence interval for the intercept is approximately (3.64,5.22). This means we are 95% confident that the true average quality of wine, when all predictor variables are zero, falls within this range.

#volatile.acidity:The 95% confidence interval for the coefficient of volatile acidity is approximately (−1.21,−0.81) This implies that we are 95% confident that the true effect of volatile acidity on wine quality falls within this range.

#chlorides:The 95% confidence interval for the coefficient of chlorides is approximately (−2.80,−1.24). We are 95% confident that the true effect of chlorides on wine quality falls within this range.

#free.sulfur.dioxide:The 95% confidence interval for the coefficient of free sulfur dioxide is approximately (0.0009,0.0092). This suggests that we are 95% confident that the true effect of free sulfur dioxide on wine quality falls within this range.

#total.sulfur.dioxide:The 95% confidence interval for the coefficient of total sulfur dioxide is approximately (−0.0048,−0.0021). We are 95% confident that the true effect of total sulfur dioxide on wine quality falls within this range.

#pH:The 95% confidence interval for the coefficient of pH is approximately (−0.71,−0.25). This indicates that we are 95% confident that the true effect of pH on wine quality falls within this range.

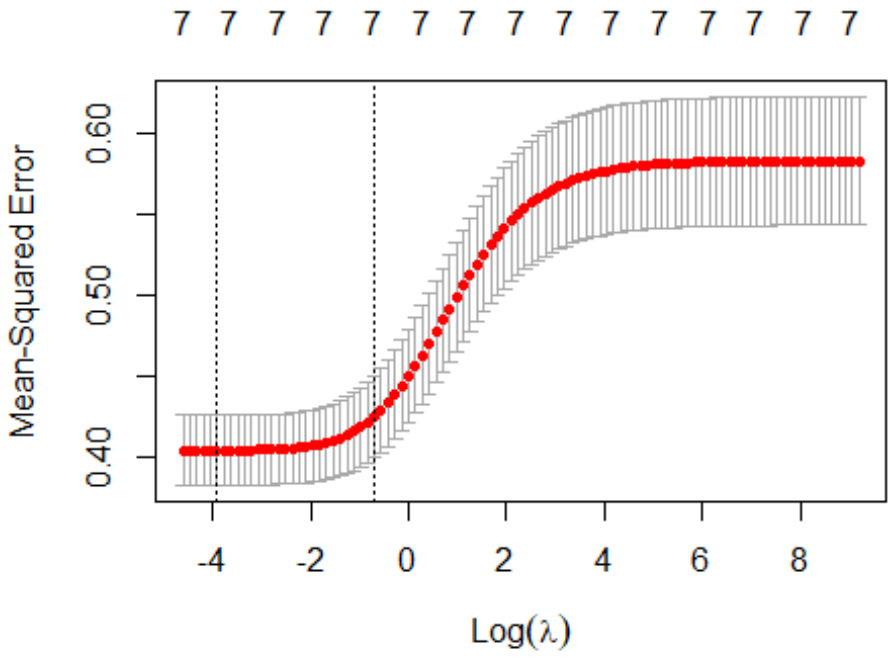
#sulphates:The 95% confidence interval for the coefficient of sulphates is approximately (0.67,1.10). We are 95% confident that the true effect of sulphates on wine quality falls within this range.

#alcohol:The 95% confidence interval for the coefficient of alcohol is approximately (0.26,0.32). This suggests that we are 95% confident that the true effect of alcohol content on wine quality falls within this range.

```
#Split data into test and train
sample_split <- function(dataset, split_ratio, seed = NULL) {
  set.seed(seed)

  train_subset <- sample(nrow(dataset) * split_ratio)

  return(list(
    train = dataset[train_subset, ],
```

```
    test = dataset[-train_subset, ]
  ))
}

red_wine_quality_split <- sample_split(red_wine_quality, 0.5, seed = 123)

#linear model using least squares on the training set


ls_model<-lm(quality ~ volatile.acidity + chlorides + free.sulfur.dioxide +
                         total.sulfur.dioxide + pH + sulphates + alcohol,
                         data = red_wine_quality_split$train)

ls_predictions <- predict(ls_model, red_wine_quality_split$test)
head(ls_predictions)

##      800      801      802      803      804      805
## 5.793962 4.955455 5.539863 6.251154 5.378521 5.570045

calc_rmse <- function(y, y_hat) {
  return(sqrt(mean((y - y_hat)^2)))
}

ls_rmse <- calc_rmse(red_wine_quality_split$test$quality, ls_predictions)
print(ls_rmse)

## [1] 0.6744579

train_matrix <- model.matrix(quality ~ volatile.acidity + chlorides + free.sulfur.dioxide +
                         total.sulfur.dioxide + pH + sulphates + alcohol, data =
red_wine_quality_split$train)

test_matrix <- model.matrix(quality ~ volatile.acidity + chlorides + free.sulfur.dioxide +
                         total.sulfur.dioxide + pH + sulphates + alcohol, data =
```

```
red_wine_quality_split$test)

grid <- 10 ^ seq(4, -2, length = 100)




ridge_model <- cv.glmnet(train_matrix, red_wine_quality_split$train$quality, alpha = 0, lambda = grid, thresh = 1e-12)

ridge_predictions <- predict(ridge_model, test_matrix, s = ridge_model$lambda.min)

ridge_rmse <- calc_rmse(red_wine_quality_split$test$quality, ridge_predictions)
print(ridge_rmse)

## [1] 0.6745557

#cross validated error to choose the optimal lambda
plot(ridge_model)
```



#This plot shows how the coefficients are affected by the regularization. As lambda increases, the penalty on large coefficients becomes more substantial, leading to a greater shrinkage of the coefficients. The variables with steeper lines are more sensitive to changes in lambda.

#Method Ridge regression-Use the entire dataset to split into training and testing sets. This ensures that both subsets have instances from the entire range of your data, including the significant predictors. Then, created model matrices for training and testing data using the significant predictors.This ensures that the model is trained and tested on the same set of features.

#We wre using the RRMSE to evaluate-The RMSE is a measure of the model's accuracy, indicating how well the model's predictions align with the actual values in the test set. The lower the RMSE, the better the model's predictive performance.

```
lasso_model <- cv.glmnet(train_matrix, red_wine_quality_split$train$quality, alpha = 1,
lambda = grid, thresh = 1e-12)

lasso_predictions <- predict(lasso_model, test_matrix, s = lasso_model$lambda.min)

lasso_rmse <- calc_rmse(red_wine_quality_split$test$quality, lasso_predictions)
print(lasso_rmse)
```

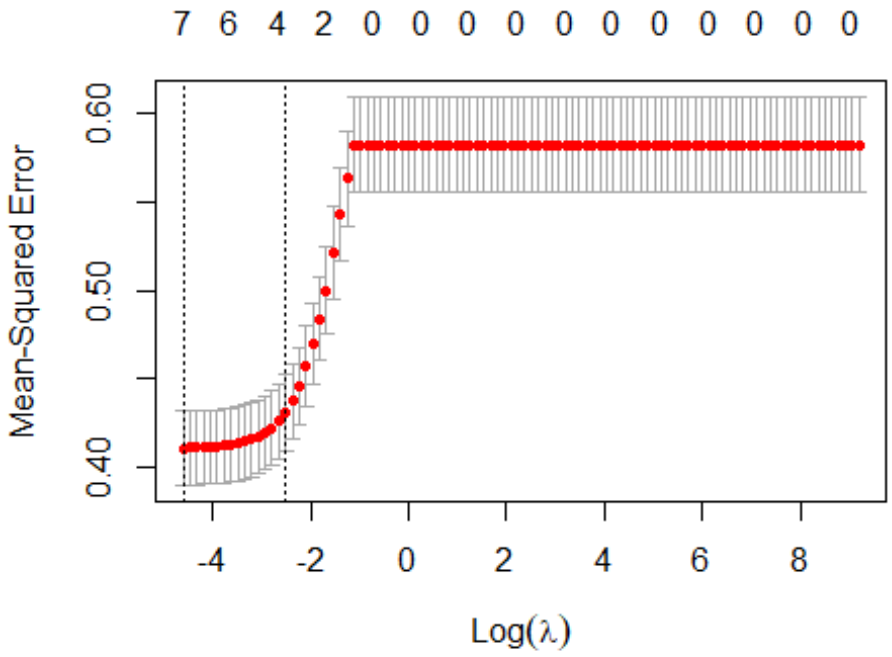
```
## [1] 0.6766518
```

```
predict(lasso_model, s = lasso_model$lambda.min, type = "coefficients")
```

```
## 9 x 1 sparse Matrix of class "dgCMatrix"
##                              s1
## (Intercept)          3.685603489
## (Intercept)          .
## volatile.acidity    -0.923757174
## chlorides           -1.173370836
## free.sulfur.dioxide  0.001822863
## total.sulfur.dioxide -0.004322404
## pH                  -0.130719469
## sulphates            0.612077209
## alcohol              0.264722116
```

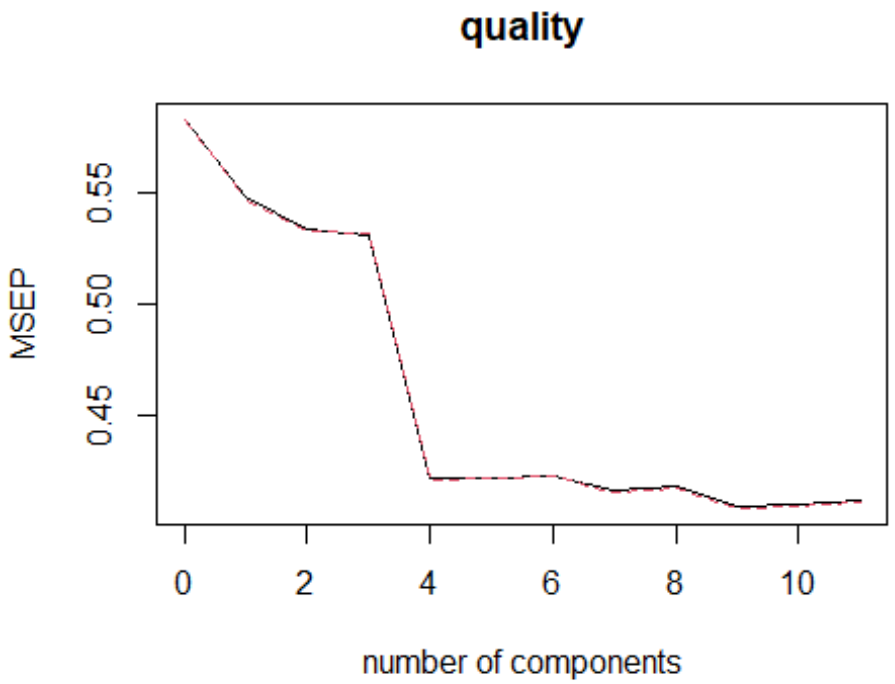
#In the Lasso regression model results provided, the coefficients offer valuable insights into the relationships between predictor variables and the dependent variable, taking into account the regularization effect of the Lasso algorithm. The intercept term, representing the expected value of the dependent variable when all predictors are zero, remains present but may be influenced by Lasso's tendency to shrink some coefficients to zero for variable selection. The second intercept with a value of "." suggests that the intercept might not be as relevant in this specific Lasso model. Notably, variables like "volatile.acidity," "chlorides," "pH," "sulphates," and "alcohol" have non-zero coefficients, indicating that Lasso has retained them in the model as relevant predictors. The negative coefficients for "volatile.acidity," "chlorides," and "pH" suggest negative impacts on the dependent variable, while the positive coefficients for "sulphates" and "alcohol" indicate positive impacts. The smaller magnitudes of some coefficients, such as for "free.sulfur.dioxide" and "total.sulfur.dioxide," imply reduced impacts, potentially even to the point of insignificance. In summary, these results showcase the Lasso model's capacity for variable selection, effectively shrinking some coefficients while retaining others based on their relevance and impact on the dependent variable.

```
plot(lasso_model)
```



#lasso optimal number of variables is 2

```
pcr_model <- pcr(quality ~ ., data=red_wine_quality_split$train, scale = TRUE,
validation = "CV")

validationplot(pcr_model, val.type = "MSEP")
```



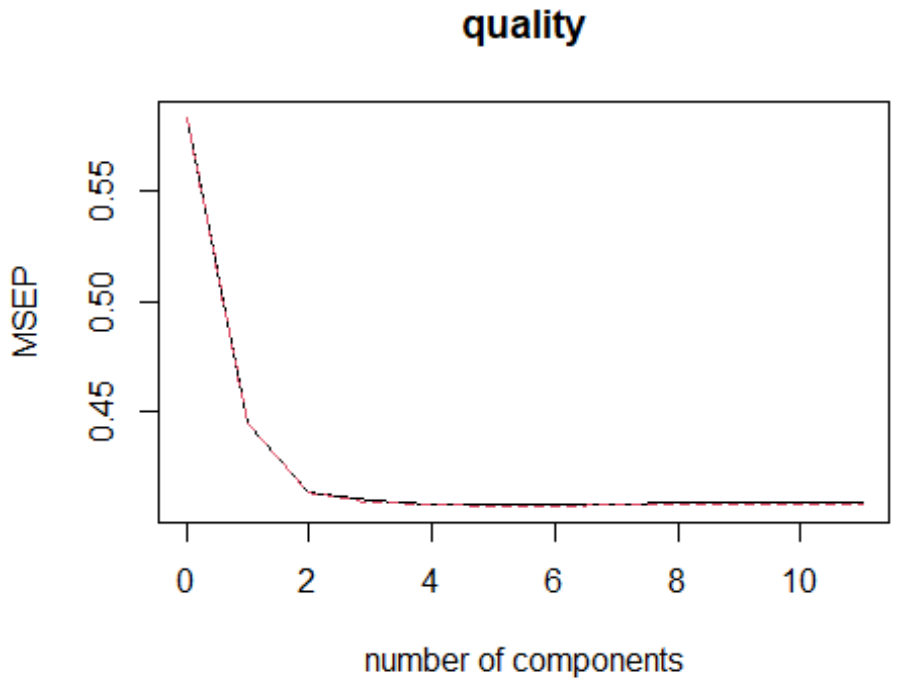

```
summary(pcr_model)

## Data:    X dimension: 799 11
##  Y dimension: 799 1
## Fit method: svdpc
## Number of components considered: 11
##
## VALIDATION: RMSEP
## Cross-validated using 10 random segments.
##        (Intercept)  1 comps  2 comps  3 comps  4 comps  5 comps  6 comps
## CV           0.7633   0.7400   0.7303   0.7288   0.6497   0.6499   0.6505
## adjCV        0.7633   0.7399   0.7301   0.7290   0.6491   0.6495   0.6502
##        7 comps  8 comps  9 comps  10 comps  11 comps
## CV      0.6454   0.6469   0.6397    0.6402    0.6419
## adjCV   0.6449   0.6464   0.6391    0.6396    0.6412
##
## TRAINING: % variance explained
```

```
##          1 comps  2 comps  3 comps  4 comps  5 comps  6 comps  7 comps  8 comps
## X          29.28    45.90   58.943    70.48    79.52    86.04    91.00    95.19
## quality     6.39     9.15    9.833    28.54    28.65    28.71    29.96    29.96
##          9 comps  10 comps  11 comps
## X          97.80     99.43    100.00
## quality    31.84     31.91     31.91
```

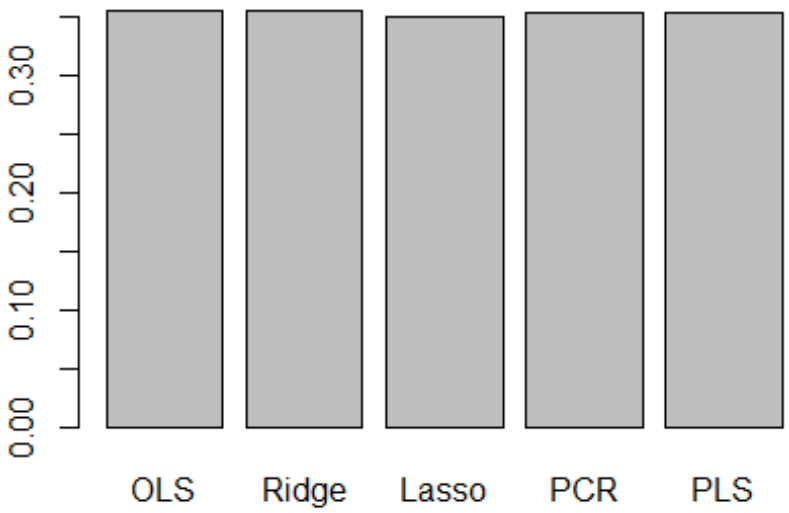
```
pcr_predictions <- predict(pcr_model, red_wine_quality_split$test, ncomp = 10)

pcr_rmse <- calc_rmse(red_wine_quality_split$test$quality, pcr_predictions)
print(pcr_rmse)
```

```
## [1] 0.6751337
```

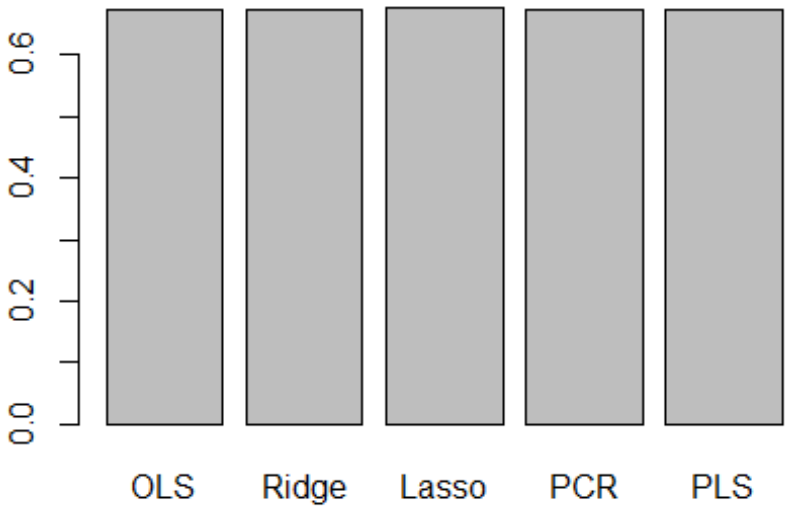
```
pls_model <- plsr(quality ~ ., data = red_wine_quality_split$train, scale = TRUE,
validation = "CV")

validationplot(pls_model, val.type = "MSEP")
```



```r
pls_predictions <- predict(pls_model, red_wine_quality_split$test, ncomp = 10)

pls_rmse <- calc_rmse(red_wine_quality_split$test$quality, pls_predictions)
print(pls_rmse)

## [1] 0.6750484

calc_r2 <- function(y, y_hat) {
  y_bar <- mean(y)
  rss <- sum((y - y_hat)^2)
  tss <- sum((y - y_bar)^2)
  return(1 - (rss / tss))
}

ols_r2 <- calc_r2(red_wine_quality_split$test$quality, ls_predictions)
ridge_r2 <- calc_r2(red_wine_quality_split$test$quality, ridge_predictions)
lasso_r2 <- calc_r2(red_wine_quality_split$test$quality, lasso_predictions)
pcr_r2 <- calc_r2(red_wine_quality_split$test$quality, pcr_predictions)
pls_r2 <- calc_r2(red_wine_quality_split$test$quality, pls_predictions)

barplot(c(ols_r2, ridge_r2, lasso_r2, pcr_r2, pls_r2),
        names.arg = c("OLS", "Ridge", "Lasso", "PCR", "PLS"))
```




```
barplot(c(ls_rmse, ridge_rmse, lasso_rmse, pcr_rmse, pls_rmse),
        names.arg = c("OLS", "Ridge", "Lasso", "PCR", "PLS"))
```

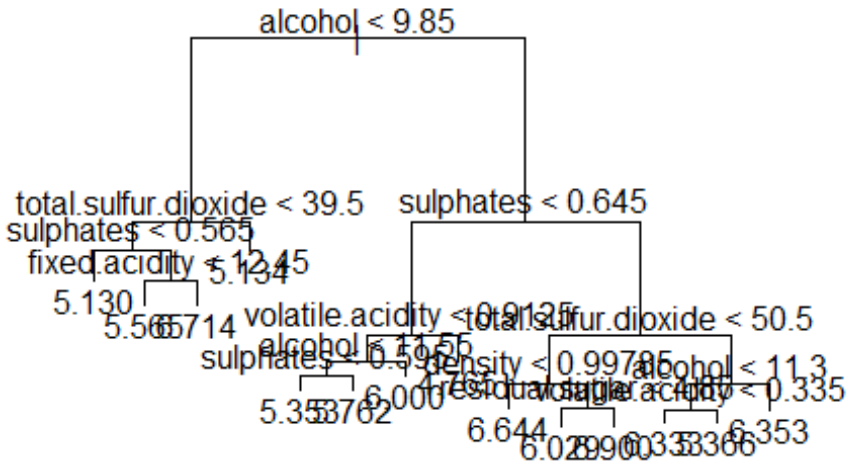


```
#
#A regression tree fit on the training data. I will also ouput the tree plot and the test MSE. The tree regression model is trying to predict sales explained by all the other variables in the data set.

red.tree <- tree(quality ~ . , data = red_wine_quality_split$train)
summary(red.tree)

##
## Regression tree:
## tree(formula = quality ~ ., data = red_wine_quality_split$train)
## Variables actually used in tree construction:
## [1] "alcohol"              "total.sulfur.dioxide" "sulphates"
## [4] "fixed.acidity"        "volatile.acidity"     "density"
## [7] "residual.sugar"
## Number of terminal nodes:  14
## Residual mean deviance:  0.3317 = 260.4 / 785
## Distribution of residuals:
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
## -2.5650 -0.3529 -0.1345  0.0000  0.4348  1.6470
```

#The regression tree, with its 14 terminal nodes, effectively captures the patterns in the red wine quality dataset. The residuals, with a mean close to zero, suggest that the model is making predictions that are, on average, unbiased. The distribution of residuals shows that the model tends to have errors within a reasonable range, providing a nuanced understanding of its predictive performance across different quality levels.

#The regression tree usees 7 predictor variables to make predictions about the “quality” outcome.The tree has 14 terminal nodes, suggesting that it divides the dataset into nine distinct groups based on the predictor variables.The low residual mean deviance .3317 indicates a good fit of the model to the training data.

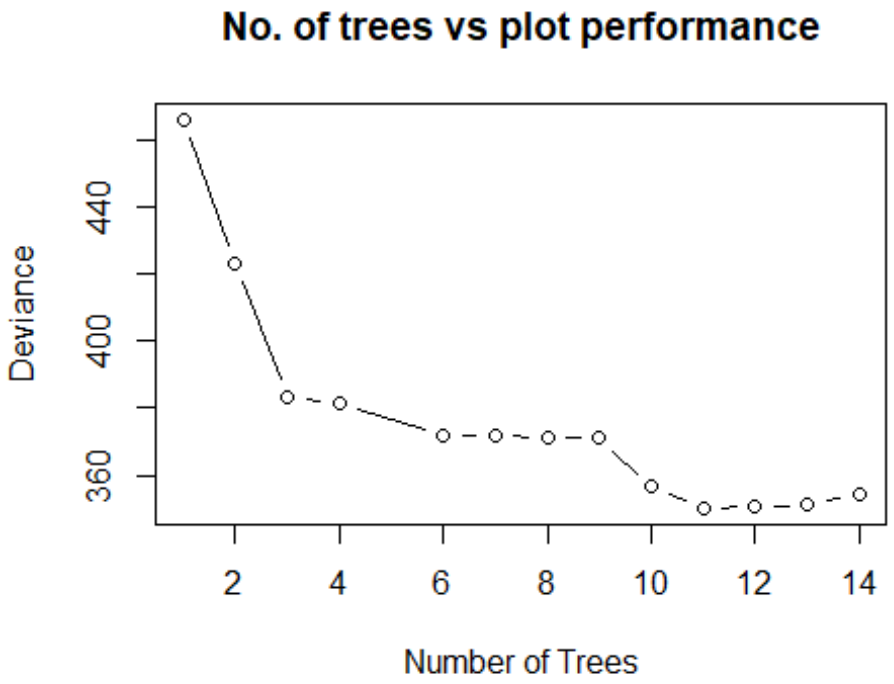
```
plot(red.tree)
text(red.tree, pretty=0)
```




```
yhat <- predict(red.tree, newdata = red_wine_quality_split$test)
full_mse<-mean((yhat - red_wine_quality_split$test$quality)^2)
print(full_mse)

## [1] 0.5274672
```

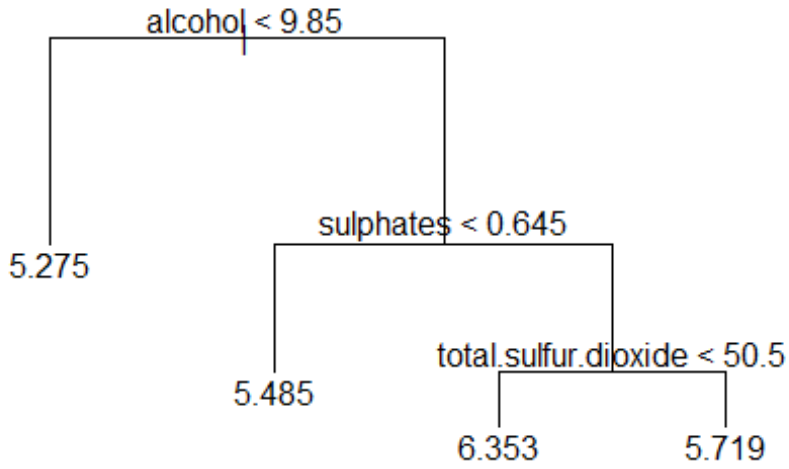
#Now I will do a tree model using cross validation. This is done in order to determine if pruning the tree helps.

```
cv.red <- cv.tree(red.tree)


plot(cv.red$size, cv.red$dev, type='b', main='No. of trees vs plot performance',
xlab='Number of Trees', ylab='Deviance')
```



#tough to determine exactly the best number of tree nodes. Levels out around 4.

```
prune.red <- prune.tree(red.tree, best=4)
plot(prune.red)
text(prune.red, pretty=0)
```




```
yhat <- predict(prune.red, red_wine_quality_split$test)
prune_mse<-mean((yhat-red_wine_quality_split$test$quality)^2)
print(prune_mse)

## [1] 0.5619024
```

#regression tree

#method/results #In this analysis, a regression tree model was constructed using the red wine quality dataset, where the quality of red wine is predicted based on various chemical characteristics. The initial tree, red.tree, was built using variables such as alcohol content, total sulfur dioxide, sulphates, fixed acidity, volatile acidity, density, and residual sugar. The tree had 14 terminal nodes, and its structure was summarized, revealing a residual mean deviance of 0.3317. Subsequent predictions were made on a separate test set, and the mean squared error was calculated, resulting in a value of 0.5275, indicating the average squared difference between predicted and actual values.To further optimize the model, cross-validation was employed to explore the relationship between the number of trees and deviance. The plot of deviance against the number of trees provided insights into the trade-off between model complexity and performance. Pruning, a technique to simplify the tree and prevent overfitting, was then applied based on the optimal tree size identified during cross-validation. The pruned tree, prune.red, was visualized, and terminal nodes were labeled for better interpretability. Predictions were made using the pruned tree on the same test set, yielding a mean squared error of 0.561. The process of cross-validation and pruning aims to enhance the model's

generalization to new data by finding an optimal level of complexity. The choice between the original and pruned tree depends on the specific requirements of the analysis, balancing model simplicity and predictive performance. This comprehensive approach ensures a robust evaluation of the regression tree model in predicting red wine quality.

#interpertation #Alcohol content, total sulfur dioxide, sulphates, fixed acidity, volatile acidity, density, and residual sugar play crucial roles in determining the quality of red wine.Model Performance:The initial tree, (red.tree) demonstrated good predictive performance on the test set, as indicated by a mean squared error of 0.5275. This suggests that the model captures the relationships between the selected features and wine quality reasonably well.For cross-validation the plot of deviance against the number of trees helped identify an optimal tree size, facilitating a balance between model complexity and predictive accuracy.However, the pruned tree, (prune.red) represents a more generalized version that may avoid overfitting to the training data. The mean squared error increased slightly after pruning (0.6175), reflecting a trade-off between model complexity and performance.#This comparison suggests that pruning the decision tree to 4 nodes has led to an increase in the Test MSE compared to the unpruned tree. In other words, the unpruned tree is performing better on the test dataset in terms of Mean Squared Error. While the pruned tree is simpler and less prone to overfitting, it may sacrifice a small amount of predictive accuracy compared to the original, more complex tree.

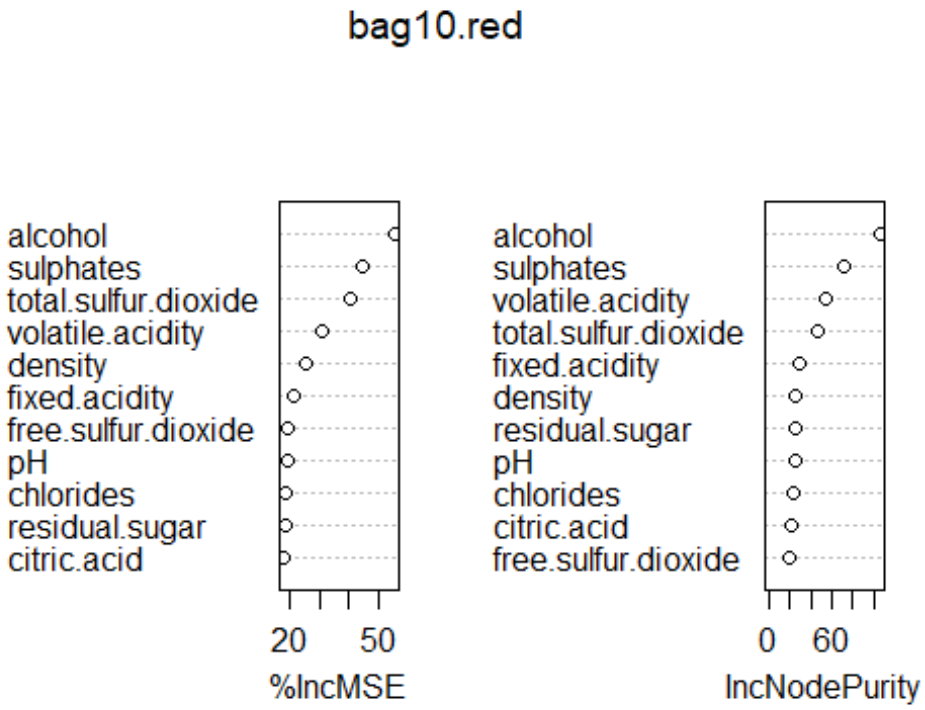
```
#bagging


bag10.red <- randomForest(quality ~., data = red_wine_quality_split$train, mtry=10, importance=TRUE)

yhat.bag <- predict(bag10.red, red_wine_quality_split$test)
bag_mse<-mean((yhat.bag-red_wine_quality_split$test$quality)^2)
print(bag_mse)

## [1] 0.4424187
```

##Bagging has decresed the error to .439 providing a more improved model(from:.527 of unpruned and .561 pruned)

#This analysis indicates that the bagging model with 10 randomly selected predictors at each split performs well on the test set, as evidenced by the relatively low mean squared error. Bagging, by aggregating predictions from multiple trees trained on bootstrapped samples, tends to enhance model robustness and mitigate overfitting, contributing to improved predictive accuracy.

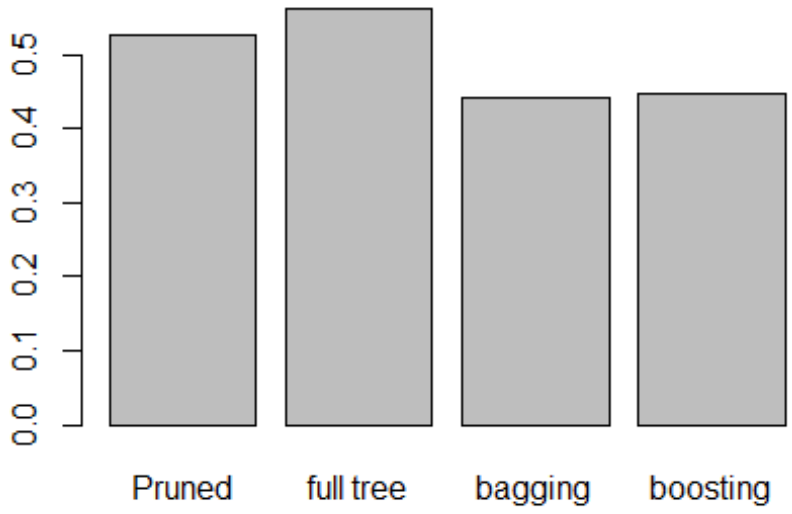
```
importance(bag10.red)
```

```
##                      %IncMSE IncNodePurity
## fixed.acidity        21.30427     28.74575
## volatile.acidity     30.72691     54.55113
## citric.acid          17.84925     21.17532
## residual.sugar       18.31977     25.14243
## chlorides            18.43473     24.12385
## free.sulfur.dioxide  19.44837     19.21571
## total.sulfur.dioxide 40.76417     47.59279
## density              25.67348     26.23152
## pH                   19.14823     24.53102
## sulphates            44.39440     71.53274
## alcohol              55.56326    106.89672

varImpPlot(bag10.red)
```



##After looking at the variables we can see that the most important variable is alcohol followed by sulpfates, total.sulfur.dioxide and volatile.acidity.

```r
#boosting

boost.red <- gbm(quality ~., data = red_wine_quality_split$train, distribution =
"gaussian", n.trees = 1000, shrinkage = 0.01)

predictions <- predict(boost.red, newdata = red_wine_quality_split$test, n.trees = 1000)

# Calculate Mean Squared Error
boost_mse <- mean((predictions - red_wine_quality_split$test$quality)^2)
print(boost_mse)

## [1] 0.447697

barplot(c(full_mse, prune_mse, bag_mse, boost_mse),
        names.arg = c("Pruned", "full tree", "bagging", "boosting"))
```



Predictive Modeling Final-Qualitative

Alisha Souza

2023-12-11

```
library(tidyverse)

## ── Attaching core tidyverse packages ──────────────────────── tidyverse 2.0.0 ──
## ✔ dplyr     1.1.1     ✔ readr     2.1.4
## ✔ forcats   1.0.0     ✔ stringr   1.5.0
## ✔ ggplot2   3.4.4     ✔ tibble    3.2.1
## ✔ lubridate 1.9.2     ✔ tidyr     1.3.0
## ✔ purrr     1.0.1
## ── Conflicts ────────────────────────────────────────── tidyverse_conflicts() ──
## ✖ dplyr::filter() masks stats::filter()
## ✖ dplyr::lag()    masks stats::lag()
## ℹ Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

```
library(dplyr)
library(stringr)
library(tidyverse)
library(ggplot2)
library(GGally)

## Registered S3 method overwritten by 'GGally':
##   method from
##   +.gg   ggplot2

library(readr)
library(corrplot)

## corrplot 0.92 loaded

library(ggcorrplot)
library(glmnet)

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

library(tree)
library(randomForest)

## randomForest 4.7-1.1
## Type rfNews() to see new features/changes/bug fixes.
##
```

```
## Attaching package: 'randomForest'
##
## The following object is masked from 'package:dplyr':
##
##     combine
##
## The following object is masked from 'package:ggplot2':
##
##     margin
```

```
library(gbm)
```

```
## Loaded gbm 2.1.8.1
```

```
library(pls)
```

```
##
## Attaching package: 'pls'
##
## The following object is masked from 'package:corrplot':
##
##     corrplot
##
## The following object is masked from 'package:stats':
##
##     loadings
```

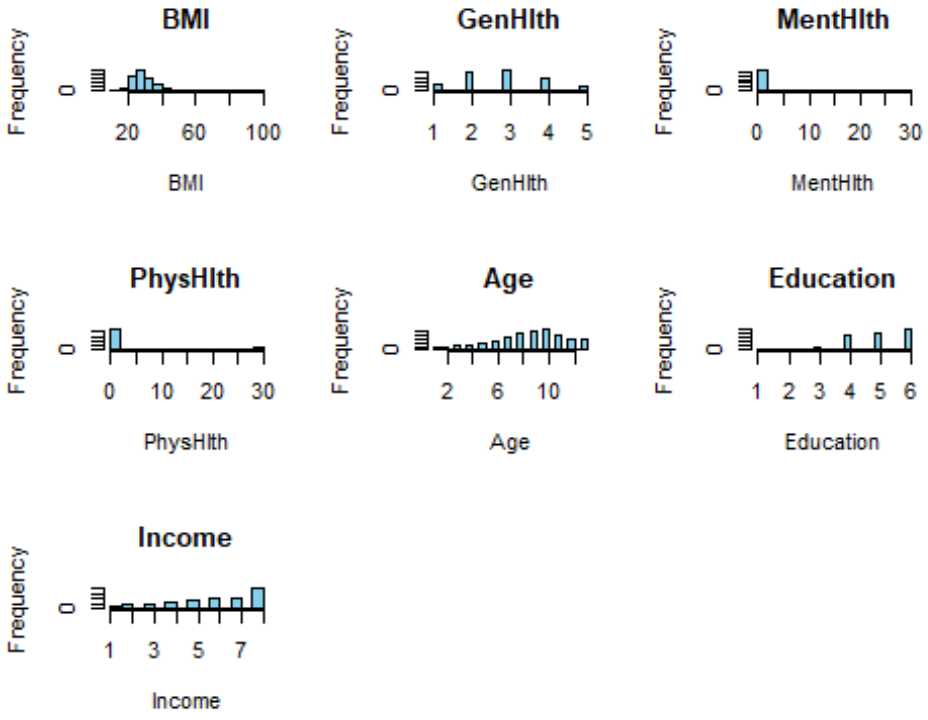
```
library(MASS)
```

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

```
##
##     select

library(e1071)

diabetes_data<-read.csv("C:||Users||alish||Downloads||M.S. Data
Science||diabetes_data.csv")

diabetes<-diabetes_data


#check number of rows and columns
dim(diabetes)

## [1] 70692    22

# Check for any missing values in the entire data frame
has_na <- any(is.na(diabetes))


print(has_na)

## [1] FALSE

summary(diabetes)

##  Diabetes_binary     HighBP          HighChol        CholCheck
##  Min.   :0.0     Min.   :0.0000   Min.   :0.0000   Min.   :0.0000
##  1st Qu.:0.0     1st Qu.:0.0000   1st Qu.:0.0000   1st Qu.:1.0000
##  Median :0.5     Median :1.0000   Median :1.0000   Median :1.0000
##  Mean   :0.5     Mean   :0.5635   Mean   :0.5257   Mean   :0.9753
##  3rd Qu.:1.0     3rd Qu.:1.0000   3rd Qu.:1.0000   3rd Qu.:1.0000
##  Max.   :1.0     Max.   :1.0000   Max.   :1.0000   Max.   :1.0000
##       BMI            Smoker           Stroke       HeartDiseaseorAttack
##  Min.   :12.00   Min.   :0.0000   Min.   :0.00000   Min.   :0.0000
##  1st Qu.:25.00   1st Qu.:0.0000   1st Qu.:0.00000   1st Qu.:0.0000
##  Median :29.00   Median :0.0000   Median :0.00000   Median :0.0000
```

```
##  Mean   :29.86   Mean   :0.4753   Mean   :0.06217   Mean   :0.1478
##  3rd Qu.:33.00   3rd Qu.:1.0000   3rd Qu.:0.00000   3rd Qu.:0.0000
##  Max.   :98.00   Max.   :1.0000   Max.   :1.00000   Max.   :1.0000
##   PhysActivity       Fruits          Veggies       HvyAlcoholConsump
##  Min.   :0.000   Min.   :0.0000   Min.   :0.0000   Min.   :0.00000
##  1st Qu.:0.000   1st Qu.:0.0000   1st Qu.:1.0000   1st Qu.:0.00000
##  Median :1.000   Median :1.0000   Median :1.0000   Median :0.00000
##  Mean   :0.703   Mean   :0.6118   Mean   :0.7888   Mean   :0.04272
##  3rd Qu.:1.000   3rd Qu.:1.0000   3rd Qu.:1.0000   3rd Qu.:0.00000
##  Max.   :1.000   Max.   :1.0000   Max.   :1.0000   Max.   :1.00000
##  AnyHealthcare    NoDocbcCost         GenHlth         MentHlth
##  Min.   :0.000   Min.   :0.00000   Min.   :1.000   Min.   : 0.000
##  1st Qu.:1.000   1st Qu.:0.00000   1st Qu.:2.000   1st Qu.: 0.000
##  Median :1.000   Median :0.00000   Median :3.000   Median : 0.000
##  Mean   :0.955   Mean   :0.09391   Mean   :2.837   Mean   : 3.752
##  3rd Qu.:1.000   3rd Qu.:0.00000   3rd Qu.:4.000   3rd Qu.: 2.000
##  Max.   :1.000   Max.   :1.00000   Max.   :5.000   Max.   :30.000
##     PhysHlth        DiffWalk           Sex            Age
##  Min.   : 0.00   Min.   :0.0000   Min.   :0.000   Min.   : 1.000
##  1st Qu.: 0.00   1st Qu.:0.0000   1st Qu.:0.000   1st Qu.: 7.000
##  Median : 0.00   Median :0.0000   Median :0.000   Median : 9.000
##  Mean   : 5.81   Mean   :0.2527   Mean   :0.457   Mean   : 8.584
##  3rd Qu.: 6.00   3rd Qu.:1.0000   3rd Qu.:1.000   3rd Qu.:11.000
##  Max.   :30.00   Max.   :1.0000   Max.   :1.000   Max.   :13.000
##    Education        Income
##  Min.   :1.000   Min.   :1.000
##  1st Qu.:4.000   1st Qu.:4.000
##  Median :5.000   Median :6.000
##  Mean   :4.921   Mean   :5.698
##  3rd Qu.:6.000   3rd Qu.:8.000
##  Max.   :6.000   Max.   :8.000
```

```r
# Rename the column
colnames(diabetes)[colnames(diabetes) == "Diabetes_binary"] <- "Has_Diabetes"

#0=no diabetes, 1=yes diabetes, also chance to factor
diabetes <- diabetes %>%
  mutate(Has_Diabetes = factor(Has_Diabetes,
                               levels = c(0, 1),
                               labels = c('No', 'Yes')))

qual_vars <- c( "BMI", "GenHlth", "MentHlth", "PhysHlth", "Age", "Education","Income")
par(mfrow = c(3,3))

for (var in qual_vars) {
  hist(diabetes[[var]], main = var, xlab = var, col = "skyblue", breaks = 20)
}
par(mfrow = c(1, 1))
```



```
#computed matrix of correlation of variables
cor(diabetes_data)

##                       Diabetes_binary      HighBP    HighChol     CholCheck
## Diabetes_binary           1.00000000  0.38151555  0.28921281  0.1153816171
## HighBP                    0.38151555  1.00000000  0.31651485  0.1032832913
## HighChol                  0.28921281  0.31651485  1.00000000  0.0859813978
## CholCheck                 0.11538162  0.10328329  0.08598140  1.0000000000
## BMI                       0.29337274  0.24101856  0.13130949  0.0456475209
## Smoker                    0.08599896  0.08743830  0.09339831 -0.0043305151
## Stroke                    0.12542678  0.12905987  0.09978619  0.0225293810
## HeartDiseaseorAttack      0.21152340  0.21075039  0.18118664  0.0434971444
## PhysActivity             -0.15866560 -0.13610217 -0.09045316 -0.0082493633
## Fruits                   -0.05407656 -0.04085216 -0.04738362  0.0173838322
## Veggies                  -0.07929315 -0.06662374 -0.04283626  0.0003492577
## HvyAlcoholConsump        -0.09485314 -0.02702989 -0.02544298 -0.0271461602
## AnyHealthcare             0.02319075  0.03576444  0.03153180  0.1068004249
## NoDocbcCost               0.04097657  0.02651701  0.03319927 -0.0626687433
## GenHlth                   0.40761160  0.32053981  0.23777820  0.0592134125
## MentHlth                  0.08702877  0.06429371  0.08388071 -0.0106602375
## PhysHlth                  0.21308102  0.17392173  0.14261035  0.0345401017
## DiffWalk                  0.27264601  0.23478391  0.16204341  0.0444303666
## Sex                       0.04441286  0.04081925  0.01732446 -0.0079912226
## Age                       0.27873807  0.33813193  0.24033775  0.1017432891
## Education                -0.17048063 -0.14164264 -0.08438612 -0.0086948352
## Income                   -0.22444871 -0.18765683 -0.10777657  0.0075504178
##                                BMI      Smoker      Stroke
## Diabetes_binary       0.293372745  0.085998964  0.125426785
## HighBP                0.241018561  0.087438297  0.129059872
## HighChol              0.131309487  0.093398312  0.099786191
## CholCheck             0.045647521 -0.004330515  0.022529381
```

```
## BMI                  1.000000000  0.011551350  0.022930877
## Smoker               0.011551350  1.000000000  0.064658397
## Stroke               0.022930877  0.064658397  1.000000000
## HeartDiseaseorAttack  0.060354865  0.124417535  0.223393786
## PhysActivity        -0.170936077 -0.079823258 -0.079984782
## Fruits              -0.084505333 -0.074810805 -0.008996297
## Veggies             -0.056527598 -0.029925651 -0.047601204
## HvyAlcoholConsump    -0.058231697  0.077835011 -0.023394552
## AnyHealthcare        -0.013417050 -0.012938633  0.006483591
## NoDocbcCost           0.065831621  0.035798890  0.036198325
## GenHlth               0.267887547  0.152416015  0.189446857
## MentHlth              0.104681505  0.091256623  0.087303439
## PhysHlth              0.161862095  0.120697740  0.164487793
## DiffWalk              0.246093615  0.119788712  0.192265923
## Sex                   0.000827232  0.112125182  0.003822095
## Age                  -0.038648250  0.105423816  0.123879343
## Education            -0.100232772 -0.140966077 -0.073926229
## Income               -0.124878283 -0.104725449 -0.136577289
##                    HeartDiseaseorAttack PhysActivity       Fruits
## Diabetes_binary             0.21152340 -0.158665605 -0.054076556
## HighBP                      0.21075039 -0.136102169 -0.040852164
## HighChol                    0.18118664 -0.090453163 -0.047383624
## CholCheck                   0.04349714 -0.008249363  0.017383832
## BMI                         0.06035486 -0.170936077 -0.084505333
## Smoker                      0.12441753 -0.079823258 -0.074810805
## Stroke                      0.22339379 -0.079984782 -0.008996297
## HeartDiseaseorAttack        1.00000000 -0.098223332 -0.019435823
## PhysActivity               -0.09822333  1.000000000  0.133812914
## Fruits                     -0.01943582  0.133812914  1.000000000
## Veggies                    -0.03631473  0.149322340  0.238605287
## HvyAlcoholConsump          -0.03712977  0.019110751 -0.033245681
```

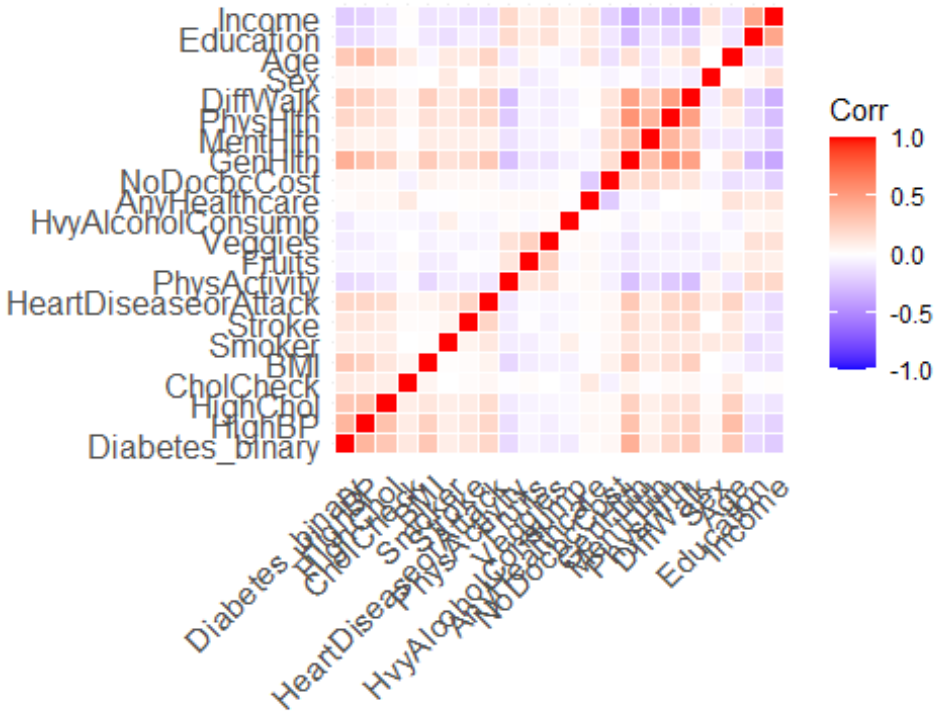
```
## AnyHealthcare              0.01568748  0.027088786  0.029385153
## NoDocbcCost                0.03602872 -0.063301661 -0.045842586
## GenHlth                    0.27586779 -0.273548068 -0.098686958
## MentHlth                   0.07505658 -0.130089507 -0.062102443
## PhysHlth                   0.19841575 -0.234499600 -0.048572126
## DiffWalk                   0.23261057 -0.276868259 -0.050784243
## Sex                        0.09816137  0.051752891 -0.088723101
## Age                        0.22187778 -0.100752963  0.061095654
## Education                 -0.09655874  0.190271208  0.098714685
## Income                    -0.14674841  0.196551353  0.079009161
##                        Veggies HvyAlcoholConsump AnyHealthcare  NoDocbcCost
## Diabetes_binary     -0.0792931456     -0.094853140   0.023190749  0.040976573
## HighBP              -0.0666237364     -0.027029886   0.035764442  0.026517009
## HighChol            -0.0428362633     -0.025442979   0.031531795  0.033199272
## CholCheck            0.0003492577     -0.027146160   0.106800425 -0.062668743
## BMI                 -0.0565275980     -0.058231697  -0.013417050  0.065831621
## Smoker              -0.0299256507      0.077835011  -0.012938633  0.035798890
## Stroke              -0.0476012036     -0.023394552   0.006483591  0.036198325
## HeartDiseaseorAttack -0.0363147331     -0.037129769   0.015687481
0.036028720
## PhysActivity         0.1493223396      0.019110751   0.027088786 -0.063301661
## Fruits               0.2386052867     -0.033245681   0.029385153 -0.045842586
## Veggies              1.0000000000      0.022090472   0.029152204 -0.037146195
## HvyAlcoholConsump    0.0220904721      1.000000000  -0.013483953
0.009682557
## AnyHealthcare        0.0291522037     -0.013483953   1.000000000 -0.221657573
## NoDocbcCost         -0.0371461952      0.009682557  -0.221657573  1.000000000
## GenHlth             -0.1157953191     -0.058796275  -0.033059848  0.169514873
## MentHlth            -0.0523590658      0.015626077  -0.049850018  0.193876516
## PhysHlth            -0.0668960023     -0.036257020  -0.003285153  0.157450521
## DiffWalk            -0.0840715623     -0.049293977   0.008113322  0.127110921
```

```
## Sex                -0.0526037770       0.014164398  -0.006561785 -0.048186618
## Age                -0.0188934247      -0.057705293   0.136974925 -0.129839183
## Education           0.1525118977       0.036279476   0.106600752 -0.096988754
## Income              0.1548987621       0.064095269   0.130491525 -0.198171295
##                      GenHlth    MentHlth     PhysHlth     DiffWalk
## Diabetes_binary      0.40761160  0.08702877  0.213081019  0.272646006
## HighBP               0.32053981  0.06429371  0.173921734  0.234783906
## HighChol             0.23777820  0.08388071  0.142610352  0.162043410
## CholCheck            0.05921341 -0.01066024  0.034540102  0.044430367
## BMI                  0.26788755  0.10468151  0.161862095  0.246093615
## Smoker               0.15241601  0.09125662  0.120697740  0.119788712
## Stroke               0.18944686  0.08730344  0.164487793  0.192265923
## HeartDiseaseorAttack 0.27586779  0.07505658  0.198415751  0.232610574
## PhysActivity        -0.27354807 -0.13008951 -0.234499600 -0.276868259
## Fruits              -0.09868696 -0.06210244 -0.048572126 -0.050784243
## Veggies             -0.11579532 -0.05235907 -0.066896002 -0.084071562
## HvyAlcoholConsump   -0.05879627  0.01562608 -0.036257020 -0.049293977
## AnyHealthcare       -0.03305985 -0.04985002 -0.003285153  0.008113322
## NoDocbcCost          0.16951487  0.19387652  0.157450521  0.127110921
## GenHlth              1.00000000  0.31507689  0.552756671  0.476638836
## MentHlth             0.31507689  1.00000000  0.380271665  0.251488672
## PhysHlth             0.55275667  0.38027167  1.000000000  0.487975799
## DiffWalk             0.47663884  0.25148867  0.487975799  1.000000000
## Sex                 -0.01455531 -0.08920416 -0.045956764 -0.082248325
## Age                  0.15562433 -0.10174603  0.084851718  0.195264987
## Education           -0.28542031 -0.10700508 -0.159316811 -0.202589922
## Income              -0.38296939 -0.21906953 -0.279325669 -0.343245002
##                          Sex         Age    Education       Income
## Diabetes_binary      0.044412858  0.278738066 -0.170480635 -0.224448715
## HighBP               0.040819253  0.338131930 -0.141642639 -0.187656832
## HighChol             0.017324456  0.240337752 -0.084386118 -0.107776574
```

```
## CholCheck            -0.007991223  0.101743289 -0.008694835  0.007550418
## BMI                  0.000827232 -0.038648250 -0.100232772 -0.124878283
## Smoker               0.112125182  0.105423816 -0.140966077 -0.104725449
## Stroke               0.003822095  0.123879343 -0.073926229 -0.136577289
## HeartDiseaseorAttack  0.098161367  0.221877780 -0.096558736 -0.146748413
## PhysActivity         0.051752891 -0.100752963  0.190271208  0.196551353
## Fruits               -0.088723101  0.061095654  0.098714685  0.079009161
## Veggies              -0.052603777 -0.018893425  0.152511898  0.154898762
## HvyAlcoholConsump     0.014164398 -0.057705293  0.036279476  0.064095269
## AnyHealthcare        -0.006561785  0.136974925  0.106600752  0.130491525
## NoDocbcCost          -0.048186618 -0.129839183 -0.096988754 -0.198171295
## GenHlth              -0.014555307  0.155624332 -0.285420312 -0.382969385
## MentHlth             -0.089204159 -0.101746035 -0.107005084 -0.219069532
## PhysHlth             -0.045956764  0.084851718 -0.159316811 -0.279325669
## DiffWalk             -0.082248325  0.195264987 -0.202589922 -0.343245002
## Sex                  1.000000000 -0.002314598  0.043563652  0.159654284
## Age                  -0.002314598  1.000000000 -0.107127338 -0.130139619
## Education            0.043563652 -0.107127338  1.000000000  0.460565473
## Income               0.159654284 -0.130139619  0.460565473  1.000000000
```

```
cor_matrix1 <- cor(diabetes_data)
```

```
# Create a square correlation plot
ggcorrplot(cor_matrix1, type="full", outline.col = "white", lab_size = 3)
```




```
#logistic regression with diabetes as the response and all variables as predictors.
set.seed(123)
diabetes.fit1<-glm(Has_Diabetes~., data=diabetes,family=binomial)
summary(diabetes.fit1)

##
## Call:
## glm(formula = Has_Diabetes ~ ., family = binomial, data = diabetes)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -3.5606  -0.8050  -0.0186   0.8388   2.9678
##
## Coefficients:
##                       Estimate Std. Error z value Pr(>|z|)
## (Intercept)         -6.865139   0.124479 -55.151  < 2e-16 ***
## HighBP               0.735319   0.019738  37.255  < 2e-16 ***
```

```
## HighChol              0.587302   0.018861  31.139  < 2e-16 ***
## CholCheck             1.360832   0.081318  16.735  < 2e-16 ***
## BMI                   0.075617   0.001573  48.057  < 2e-16 ***
## Smoker               -0.001680   0.018875  -0.089 0.929075
## Stroke                0.162029   0.040917   3.960 7.50e-05 ***
## HeartDiseaseorAttack  0.252675   0.028435   8.886  < 2e-16 ***
## PhysActivity         -0.033131   0.021294  -1.556 0.119726
## Fruits               -0.034507   0.019592  -1.761 0.078185 .
## Veggies              -0.061051   0.023333  -2.616 0.008884 **
## HvyAlcoholConsump    -0.749713   0.048762 -15.375  < 2e-16 ***
## AnyHealthcare         0.060813   0.047164   1.289 0.197264
## NoDocbcCost           0.018988   0.034083   0.557 0.577452
## GenHlth               0.584617   0.011447  51.071  < 2e-16 ***
## MentHlth             -0.004360   0.001285  -3.394 0.000688 ***
## PhysHlth             -0.008321   0.001192  -6.981 2.94e-12 ***
## DiffWalk              0.114941   0.025861   4.445 8.81e-06 ***
## Sex                   0.267261   0.019150  13.956  < 2e-16 ***
## Age                   0.152242   0.003909  38.950  < 2e-16 ***
## Education            -0.036815   0.010221  -3.602 0.000316 ***
## Income               -0.058807   0.005193 -11.325  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 98000  on 70691  degrees of freedom
## Residual deviance: 72388  on 70670  degrees of freedom
## AIC: 72432
##
## Number of Fisher Scoring iterations: 5
```

#Deviance residuals measure how well the model fits the data. Results showed the values are extremely close to zero, indicating a very good fit. The residuals are at the scale of 10-6, which suggests that the model fits the data very well.

#The estimated coefficients for the estimates are very close to zero, and the standard errors are very large. This might indicate a problem with the model fitting process, leading to unreliable coefficient estimates.

```
logdiabetes.prob1 <- predict(diabetes.fit1, type='response')
logdiabetes.pred1 <- rep("No", length(logdiabetes.prob1))
logdiabetes.pred1[logdiabetes.prob1 > 0.5] <- "Yes"
table(logdiabetes.pred1, diabetes$Has_Diabetes)

##
## logdiabetes.pred1    No   Yes
##               No  25747  8211
##               Yes  9599 27135

mean(logdiabetes.pred1 == diabetes$Has_Diabetes)

## [1] 0.748062
```

##accuracy (9599+8211/70692)=0.2519

```
set.seed(123)
diabetes.fit2<-
glm(Has_Diabetes~HighBP+HighChol+CholCheck+BMI+Stroke+HeartDiseaseorAttack
+HvyAlcoholConsump+GenHlth+MentHlth+PhysHlth+DiffWalk+Sex+Age+Education+In
come, data=diabetes,family=binomial)
summary(diabetes.fit2)

##
## Call:
## glm(formula = Has_Diabetes ~ HighBP + HighChol + CholCheck +
##     BMI + Stroke + HeartDiseaseorAttack + HvyAlcoholConsump +
##     GenHlth + MentHlth + PhysHlth + DiffWalk + Sex + Age + Education +
##     Income, family = binomial, data = diabetes)
```

```
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -3.5752  -0.8058  -0.0187   0.8391   2.9631
##
## Coefficients:
##                          Estimate Std. Error z value Pr(>|z|)
## (Intercept)             -6.898268   0.117343 -58.787  < 2e-16 ***
## HighBP                   0.736729   0.019731  37.338  < 2e-16 ***
## HighChol                 0.588973   0.018833  31.273  < 2e-16 ***
## CholCheck                1.362709   0.080973  16.829  < 2e-16 ***
## BMI                      0.076006   0.001567  48.499  < 2e-16 ***
## Stroke                   0.163951   0.040898   4.009 6.10e-05 ***
## HeartDiseaseorAttack     0.251249   0.028390   8.850  < 2e-16 ***
## HvyAlcoholConsump       -0.747502   0.048521 -15.406  < 2e-16 ***
## GenHlth                  0.587506   0.011394  51.564  < 2e-16 ***
## MentHlth                -0.004193   0.001278  -3.281  0.00103 **
## PhysHlth                -0.008209   0.001188  -6.908 4.93e-12 ***
## DiffWalk                 0.121640   0.025667   4.739 2.15e-06 ***
## Sex                      0.272678   0.018855  14.462  < 2e-16 ***
## Age                      0.152047   0.003815  39.857  < 2e-16 ***
## Education               -0.040883   0.010084  -4.054 5.03e-05 ***
## Income                  -0.060075   0.005110 -11.757  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 98000  on 70691  degrees of freedom
## Residual deviance: 72406  on 70676  degrees of freedom
## AIC: 72438
```

```
##
## Number of Fisher Scoring iterations: 5

logdiabetes.prob2 <- predict(diabetes.fit2, type='response')
logdiabetes.pred2 <- rep("No", length(logdiabetes.prob2))
logdiabetes.pred2[logdiabetes.prob2 > 0.5] <- "Yes"
table(logdiabetes.pred2, diabetes$Has_Diabetes)

##
## logdiabetes.pred2    No   Yes
##               No  25735  8203
##               Yes  9611 27143

mean(logdiabetes.pred2 == diabetes$Has_Diabetes)

## [1] 0.7480054
```

#accuracy (9611+8203/70692)=.2519

```
set.seed(123)
diabetes.fit3<-
glm(Has_Diabetes~HighBP+HighChol+CholCheck+BMI+Stroke+HeartDiseaseorAttack
+HvyAlcoholConsump+DiffWalk+Sex+Age+Education+Income,
data=diabetes,family=binomial)
summary(diabetes.fit3)

##
## Call:
## glm(formula = Has_Diabetes ~ HighBP + HighChol + CholCheck +
##     BMI + Stroke + HeartDiseaseorAttack + HvyAlcoholConsump +
##     DiffWalk + Sex + Age + Education + Income, family = binomial,
##     data = diabetes)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
```

```
## -3.5844  -0.8460  -0.0364   0.8795   2.9872
##
## Coefficients:
##                          Estimate Std. Error z value Pr(>|z|)
## (Intercept)            -5.089225   0.107726 -47.242  < 2e-16 ***
## HighBP                  0.843605   0.019179  43.987  < 2e-16 ***
## HighChol                0.655742   0.018326  35.781  < 2e-16 ***
## CholCheck               1.364495   0.079060  17.259  < 2e-16 ***
## BMI                     0.083613   0.001546  54.070  < 2e-16 ***
## Stroke                  0.278507   0.040316   6.908 4.91e-12 ***
## HeartDiseaseorAttack    0.438769   0.027741  15.817  < 2e-16 ***
## HvyAlcoholConsump      -0.760512   0.047425 -16.036  < 2e-16 ***
## DiffWalk                0.468283   0.022933  20.419  < 2e-16 ***
## Sex                     0.290649   0.018344  15.844  < 2e-16 ***
## Age                     0.139370   0.003621  38.491  < 2e-16 ***
## Education              -0.098466   0.009793 -10.055  < 2e-16 ***
## Income                 -0.103088   0.004898 -21.046  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 98000  on 70691  degrees of freedom
## Residual deviance: 75396  on 70679  degrees of freedom
## AIC: 75422
##
## Number of Fisher Scoring iterations: 4

logdiabetes.prob3 <- predict(diabetes.fit3, type='response')
logdiabetes.pred3 <- rep("No", length(logdiabetes.prob3))
```

```
logdiabetes.pred3[logdiabetes.prob3 > 0.5] <- "Yes"
table(logdiabetes.pred3, diabetes$Has_Diabetes)

##
## logdiabetes.pred3    No   Yes
##               No  25324  8715
##               Yes 10022 26631

mean(logdiabetes.pred3 == diabetes$Has_Diabetes)

## [1] 0.7349488
```

#accuracy (10022+8715/70692)=.2650

```
#10 predictors
set.seed(123)
diabetes.fit4<-
glm(Has_Diabetes~HighBP+HighChol+CholCheck+BMI+HeartDiseaseorAttack+HvyAlc
oholConsump+DiffWalk+Sex+Age+Income, data=diabetes,family=binomial)
summary(diabetes.fit4)

##
## Call:
## glm(formula = Has_Diabetes ~ HighBP + HighChol + CholCheck +
##     BMI + HeartDiseaseorAttack + HvyAlcoholConsump + DiffWalk +
##     Sex + Age + Income, family = binomial, data = diabetes)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -3.5922  -0.8485  -0.0357   0.8800   2.9739
##
## Coefficients:
##                       Estimate Std. Error z value Pr(>|z|)
## (Intercept)         -5.467770   0.100492  -54.41   <2e-16 ***
```

```
## HighBP                0.853025   0.019141   44.56   <2e-16 ***
## HighChol              0.659567   0.018301   36.04   <2e-16 ***
## CholCheck             1.365979   0.079078   17.27   <2e-16 ***
## BMI                   0.083715   0.001544   54.21   <2e-16 ***
## HeartDiseaseorAttack  0.468502   0.027433   17.08   <2e-16 ***
## HvyAlcoholConsump    -0.764269   0.047410  -16.12   <2e-16 ***
## DiffWalk              0.491874   0.022773   21.60   <2e-16 ***
## Sex                   0.295194   0.018315   16.12   <2e-16 ***
## Age                   0.140889   0.003613   39.00   <2e-16 ***
## Income               -0.125240   0.004452  -28.13   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 98000  on 70691  degrees of freedom
## Residual deviance: 75546  on 70681  degrees of freedom
## AIC: 75568
##
## Number of Fisher Scoring iterations: 4

logdiabetes.prob4 <- predict(diabetes.fit4, type='response')
logdiabetes.pred4 <- rep("No", length(logdiabetes.prob3))
logdiabetes.pred4[logdiabetes.prob4 > 0.5] <- "Yes"
table(logdiabetes.pred4, diabetes$Has_Diabetes)

##
## logdiabetes.pred4    No   Yes
##              No  25269  8723
##              Yes 10077 26623

mean(logdiabetes.pred4 == diabetes$Has_Diabetes)
```

```
## [1] 0.7340576
```

#accuracy (10077+8723/70692)=.2659

#?

```
set.seed(123)
sample_index <- sample(1:nrow(diabetes_data), 0.7 * nrow(diabetes_data))
train_data <- diabetes_data[sample_index, ]
test_data <- diabetes_data[-sample_index, ]

diabetes_logistic_model <- glm(Diabetes_binary
~HighBP+HighChol+CholCheck+BMI+HeartDiseaseorAttack+HvyAlcoholConsump+Diff
Walk+Sex+Age+Income, data = train_data, family = binomial)
summary(diabetes_logistic_model)

##
## Call:
## glm(formula = Diabetes_binary ~ HighBP + HighChol + CholCheck +
##     BMI + HeartDiseaseorAttack + HvyAlcoholConsump + DiffWalk +
##     Sex + Age + Income, family = binomial, data = train_data)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -3.5845  -0.8485   0.0669   0.8795   2.7735
##
## Coefficients:
##                       Estimate Std. Error z value Pr(>|z|)
## (Intercept)         -5.445568   0.118690  -45.88   <2e-16 ***
## HighBP               0.849901   0.022910   37.10   <2e-16 ***
## HighChol             0.679885   0.021882   31.07   <2e-16 ***
## CholCheck            1.345375   0.092904   14.48   <2e-16 ***
## BMI                  0.083125   0.001842   45.13   <2e-16 ***
## HeartDiseaseorAttack 0.463268   0.032827   14.11   <2e-16 ***
```

```
## HvyAlcoholConsump    -0.760461   0.056440  -13.47   <2e-16 ***
## DiffWalk              0.497009   0.027188   18.28   <2e-16 ***
## Sex                   0.279603   0.021893   12.77   <2e-16 ***
## Age                   0.140438   0.004309   32.59   <2e-16 ***
## Income               -0.121662   0.005315  -22.89   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 68599  on 49483  degrees of freedom
## Residual deviance: 52876  on 49473  degrees of freedom
## AIC: 52898
##
## Number of Fisher Scoring iterations: 4

diabetes_logistic_model <- glm(Diabetes_binary ~HighBP + HighChol + Age, data =
train_data, family = binomial)
logistic_probabilities <- predict(diabetes_logistic_model, newdata = test_data, type =
"response")
logistic_predictions <- ifelse(logistic_probabilities > 0.5, 1, 0)


table(logistic_predictions, test_data$Diabetes_binary)

##
## logistic_predictions    0    1
##                    0 7002 2840
##                    1 3607 7759

mean(logistic_predictions == test_data$Diabetes_binary)

## [1] 0.6960109
```

##error rate (3607+2840/21208)=0.3040

```
set.seed(123)
lda_model <-
lda(Diabetes_binary~HighBP+HighChol+CholCheck+BMI+HeartDiseaseorAttack+HvyAl
coholConsump+DiffWalk+Sex+Age+Income, data = train_data)

# Predictions on the test data
lda_pred <- predict(lda_model, newdata = test_data)$class

# Create a contingency table
confusion_table <- table(lda_pred, test_data$Diabetes_binary)
print(confusion_table)

##
## lda_pred    0    1
##        0 7484 2526
##        1 3125 8073

mean(lda_pred== test_data$Diabetes_binary)

## [1] 0.7335439
```

##accuracy (3125+2526/21208)=0.2664

```
set.seed(123)
naive_bayes_model <-
naiveBayes(Diabetes_binary~HighBP+HighChol+CholCheck+BMI+HeartDiseaseorAtta
ck+HvyAlcoholConsump+DiffWalk+Sex+Age+Income, data = train_data)
naive_bayes_pred <- predict(naive_bayes_model, test_data)
naive_bayes_class <- as.factor(naive_bayes_pred)
table(naive_bayes_class, test_data$Diabetes_binary)

##
## naive_bayes_class    0    1
```

```
##              0 6711 1926
##              1 3898 8673

mean(naive_bayes_class== test_data$Diabetes_binary)

## [1] 0.7253866
```

##accuracy (3898+1926/21208)=0.2746

```
set.seed(123)
qda_model <-
qda(Diabetes_binary~HighBP+HighChol+CholCheck+BMI+HeartDiseaseorAttack+Hvy
AlcoholConsump+DiffWalk+Sex+Age+Income, data = train_data)

# Predictions on the test data
qda_pred <- predict(qda_model, newdata = test_data)$class

# Create a contingency table
confusion_table <- table(qda_pred, test_data$Diabetes_binary)
print(confusion_table)

##
## qda_pred    0    1
##        0 5901 1361
##        1 4708 9238

mean(qda_pred== test_data$Diabetes_binary)

## [1] 0.7138344
```

##accuracy (4708+1361/21208)=0.2861

```
library(class)

set.seed(123)
predictors <- c("HighBP", "HighChol", "CholCheck", "BMI", "HeartDiseaseorAttack",
```

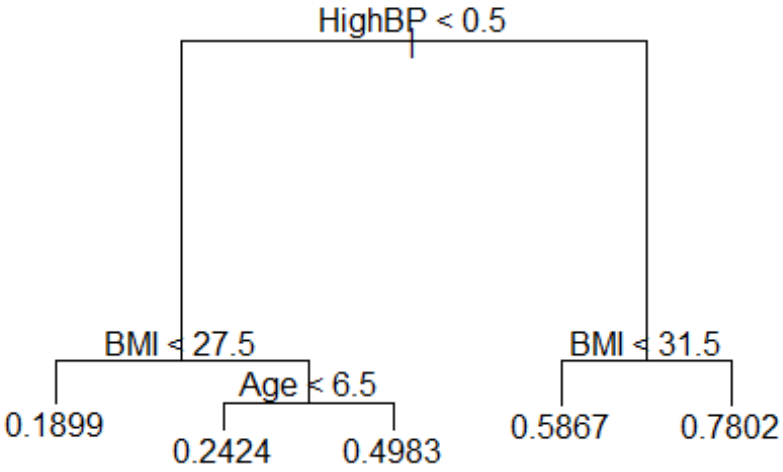
```
"HvyAlcoholConsump", "DiffWalk", "Sex", "Age", "Income")

knn_model <- knn(train_data[, predictors], test_data[, predictors],
train_data$Diabetes_binary, k = 5)



confusion_table_knn <- table(knn_model, test_data$Diabetes_binary)
print(confusion_table_knn)

##
## knn_model    0    1
##         0 7247 2668
##         1 3362 7931

accuracy_knn <- mean(knn_model == test_data$Diabetes_binary)
print(paste("Accuracy for kNN:", accuracy_knn))

## [1] "Accuracy for kNN: 0.715673330818559"
```

##error rate (3362+2668/21208)=0.2843

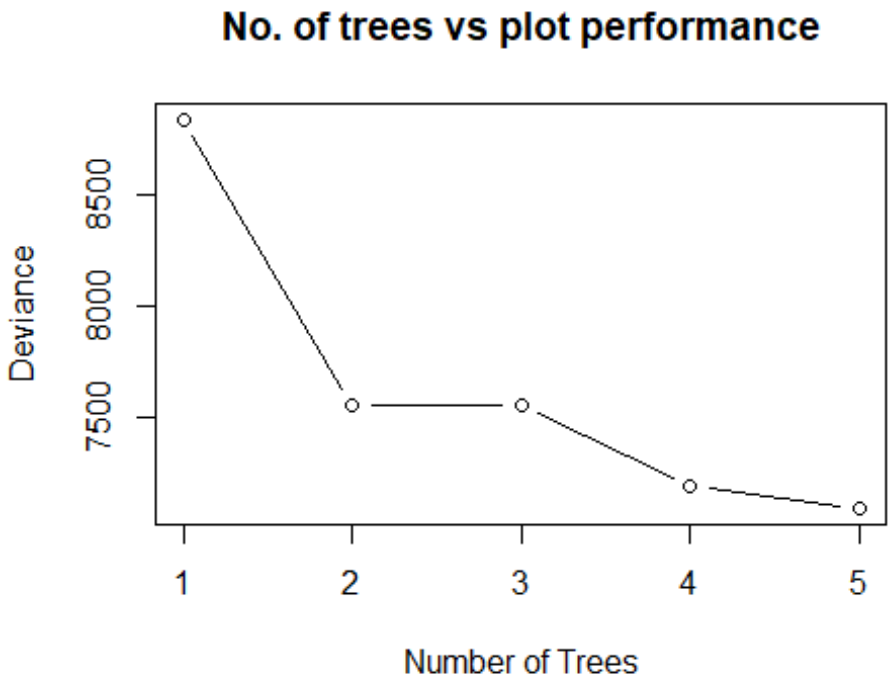
```
library(class)

set.seed(123)
predictors <- c("HighBP", "HighChol", "CholCheck", "BMI", "HeartDiseaseorAttack",
"HvyAlcoholConsump", "DiffWalk", "Sex", "Age", "Income")

knn_model <- knn(train_data[, predictors], test_data[, predictors],
train_data$Diabetes_binary, k = 10)



confusion_table_knn <- table(knn_model, test_data$Diabetes_binary)
print(confusion_table_knn)
```

```
##
## knn_model    0    1
##         0 7239 2527
##         1 3370 8072

accuracy_knn <- mean(knn_model == test_data$Diabetes_binary)
print(paste("Accuracy for kNN:", accuracy_knn))

## [1] "Accuracy for kNN: 0.721944549226707"

set.seed(123)
train <- sample(1:nrow(diabetes_data), nrow(diabetes_data)/2)
diabetes.train <- diabetes_data[train, ]
diabetes.test <- diabetes_data[-train, ]
y.test <- diabetes.test$Diabetes_binary

diabetes.tree <-
tree(Diabetes_binary~HighBP+HighChol+CholCheck+BMI+HeartDiseaseorAttack+Hvy
AlcoholConsump+DiffWalk+Sex+Age+Income, data = diabetes.train)
summary(diabetes.tree)

##
## Regression tree:
## tree(formula = Diabetes_binary ~ HighBP + HighChol + CholCheck +
##     BMI + HeartDiseaseorAttack + HvyAlcoholConsump + DiffWalk +
##     Sex + Age + Income, data = diabetes.train)
## Variables actually used in tree construction:
## [1] "HighBP" "BMI"    "Age"
## Number of terminal nodes:  5
## Residual mean deviance:  0.2007 = 7092 / 35340
## Distribution of residuals:
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
## -0.7802 -0.2424  0.2198  0.0000  0.4133  0.8101
```

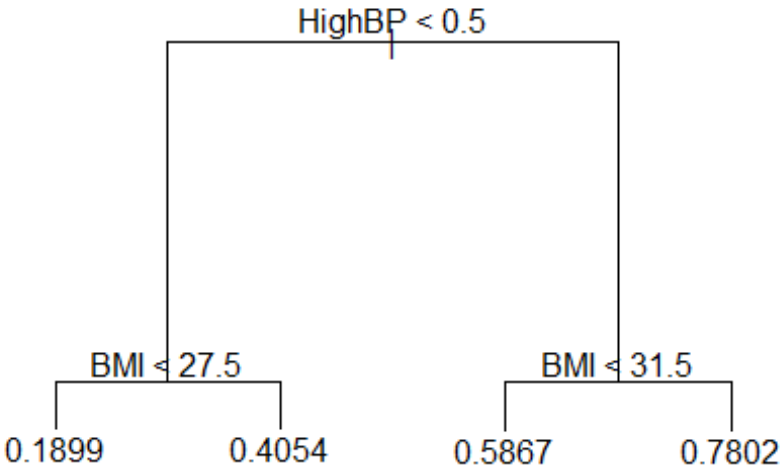
```
plot(diabetes.tree)
text(diabetes.tree, pretty=0)
```




```
yhat <- predict(diabetes.tree, newdata=diabetes.test)
mean((yhat - y.test)^2)

## [1] 0.2011998

cv.diabetes <- cv.tree(diabetes.tree)


plot(cv.diabetes$size, cv.diabetes$dev, type='b', main='No. of trees vs plot
performance', xlab='Number of Trees', ylab='Deviance')
```




```
prune.diabetes <- prune.tree(diabetes.tree, best=4)
plot(prune.diabetes)
text(prune.diabetes, pretty=0)
```




```
yhat <- predict(prune.diabetes, diabetes.test)
mean((yhat-y.test)^2)

## [1] 0.2038963
```