# Data 622 - Homework 3

Leticia Salazar

December 3, 2023

- Read the following articles:
  - https://www.hindawi.com/journals/complexity/2021/5550344/
  - https://www.ncbi.nlm.nih.gov/pmc/articles/PMC8137961/
- Search for academic content (at least 3 articles) that compare the use of decision trees vs SVMs in your current area of expertise.
- Perform an analysis of the dataset used in Homework #2 using the SVM algorithm.
- Compare the results with the results from previous homework.
- Answer questions, such as:
  - Which algorithm is recommended to get more accurate results?
  - Is it better for classification or regression scenarios?
  - Do you agree with the recommendations?
  - Why?

**Load Libraries:** Below are the libraries used to complete this assignment

```r
library(tidyverse) # data prep
```

```
FALSE -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
FALSE v dplyr     1.1.3     v readr     2.1.4
FALSE v forcats   1.0.0     v stringr   1.5.0
FALSE v ggplot2   3.4.3     v tibble    3.2.1
FALSE v lubridate 1.9.3     v tidyr     1.3.0
FALSE v purrr     1.0.2
FALSE -- Conflicts ------------------------------------------ tidyverse_conflicts() --
FALSE x dplyr::filter() masks stats::filter()
FALSE x dplyr::lag()    masks stats::lag()
FALSE i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become er
```

```r
library(skimr) # data prep
library(rpart) # decision tree package
library(rpart.plot) # decision tree display package
library(knitr) # kable function for table
library(tidyr) # splitting data
library(ggplot2) # graphing
library(hrbrthemes) # chart customization
```

```
FALSE NOTE: Either Arial Narrow or Roboto Condensed fonts are required to use these themes.
FALSE       Please use hrbrthemes::import_roboto_condensed() to install Roboto Condensed and
FALSE       if Arial Narrow is not on your system, please see https://bit.ly/arialnarrow
```

```r
library(gridExtra) # layering charts
```

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

```r
library(stringr) # data prep
library(tidymodels) # predictions
```

```
FALSE -- Attaching packages -------------------------------------- tidymodels 1.1.1 --
FALSE v broom        1.0.5     v rsample      1.2.0
FALSE v dials        1.2.0     v tune         1.1.2
FALSE v infer        1.0.5     v workflows    1.1.3
FALSE v modeldata    1.2.0     v workflowsets 1.0.1
FALSE v parsnip      1.1.1     v yardstick    1.2.0
FALSE v recipes      1.0.8
FALSE -- Conflicts ----------------------------------------- tidymodels_conflicts() --
FALSE x gridExtra::combine() masks dplyr::combine()
FALSE x scales::discard()    masks purrr::discard()
FALSE x dplyr::filter()      masks stats::filter()
FALSE x recipes::fixed()     masks stringr::fixed()
FALSE x dplyr::lag()         masks stats::lag()
FALSE x dials::prune()       masks rpart::prune()
FALSE x yardstick::spec()    masks readr::spec()
FALSE x recipes::step()      masks stats::step()
FALSE * Use tidymodels_prefer() to resolve common conflicts.
```

```r
library(corrplot) # correlation plot
```

```
FALSE corrplot 0.92 loaded
```

```r
library(randomForest) # for the random forest
```

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

```
FALSE     combine
FALSE
FALSE The following object is masked from 'package:ggplot2':
FALSE
FALSE     margin
```

```r
library(caret) # confusion matrix
```

```
FALSE Loading required package: lattice
FALSE
FALSE Attaching package: 'caret'
FALSE
FALSE The following objects are masked from 'package:yardstick':
FALSE
FALSE     precision, recall, sensitivity, specificity
FALSE
FALSE The following object is masked from 'package:purrr':
FALSE
FALSE     lift
```

```r
library("e1071") #svm
```

```
FALSE
FALSE Attaching package: 'e1071'
FALSE
FALSE The following object is masked from 'package:tune':
FALSE
FALSE     tune
FALSE
FALSE The following object is masked from 'package:rsample':
FALSE
FALSE     permutations
FALSE
FALSE The following object is masked from 'package:parsnip':
FALSE
FALSE     tune
```

**Load Data:** The data chosen in homework 2 was from Kaggle.com called Red Wine Quality. The data set is included in my GitHub and read into R.

| fixed.acidity | volatile.acidity | citric.acid | residual.sugar | chlorides | free.sulfur.dioxide | total.sulfur.dioxide | density | pH | sulphates | alcohol | quality |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 7.4 | 0.70 | 0.00 | 1.9 | 0.076 | 11 | 34 | 0.9978 | 3.51 | 0.56 | 9.4 | 5 |
| 7.8 | 0.88 | 0.00 | 2.6 | 0.098 | 25 | 67 | 0.9968 | 3.20 | 0.68 | 9.8 | 5 |
| 7.8 | 0.76 | 0.04 | 2.3 | 0.092 | 15 | 54 | 0.9970 | 3.26 | 0.65 | 9.8 | 5 |
| 11.2 | 0.28 | 0.56 | 1.9 | 0.075 | 17 | 60 | 0.9980 | 3.16 | 0.58 | 9.8 | 6 |
| 7.4 | 0.70 | 0.00 | 1.9 | 0.076 | 11 | 34 | 0.9978 | 3.51 | 0.56 | 9.4 | 5 |
| 7.4 | 0.66 | 0.00 | 1.8 | 0.075 | 13 | 40 | 0.9978 | 3.51 | 0.56 | 9.4 | 5 |

**The Data:**

Based on the description from Kaggle, the two datasets are related to red and white variants of the Portuguese "Vinho Verde" wine. For more details, consult: http://www.vinhoverde.pt/en/ or the reference [Cortez et al., 2009]. Due to privacy and logistic issues, only physicochemical (inputs) and sensory (the output) variables are available (e.g. there is no data about grape types, wine brand, wine selling price, etc.).

These datasets can be viewed as classification or regression tasks. The classes are ordered and not balanced (e.g. there are many more normal wines than excellent or poor ones). Outlier detection algorithms could be used to detect the few excellent or poor wines. Also, we are not sure if all input variables are relevant. So it could be interesting to test feature selection methods.

Input variables (based on physicochemical tests):

1 - fixed acidity

2 - volatile acidity

3 - citric acid

4 - residual sugar

5 - chlorides

6 - free sulfur dioxide

7 - total sulfur dioxide

8 - density

9 - pH

10 - sulphates

11 - alcohol

Output variable (based on sensory data):

12 - quality (score between 0 and 10)

**Data Exploration:**

Using the `skimr` library we can obtain a quick summary statistic of the dataset. It has 1599 values with 12 variables all numeric and no missing variables.

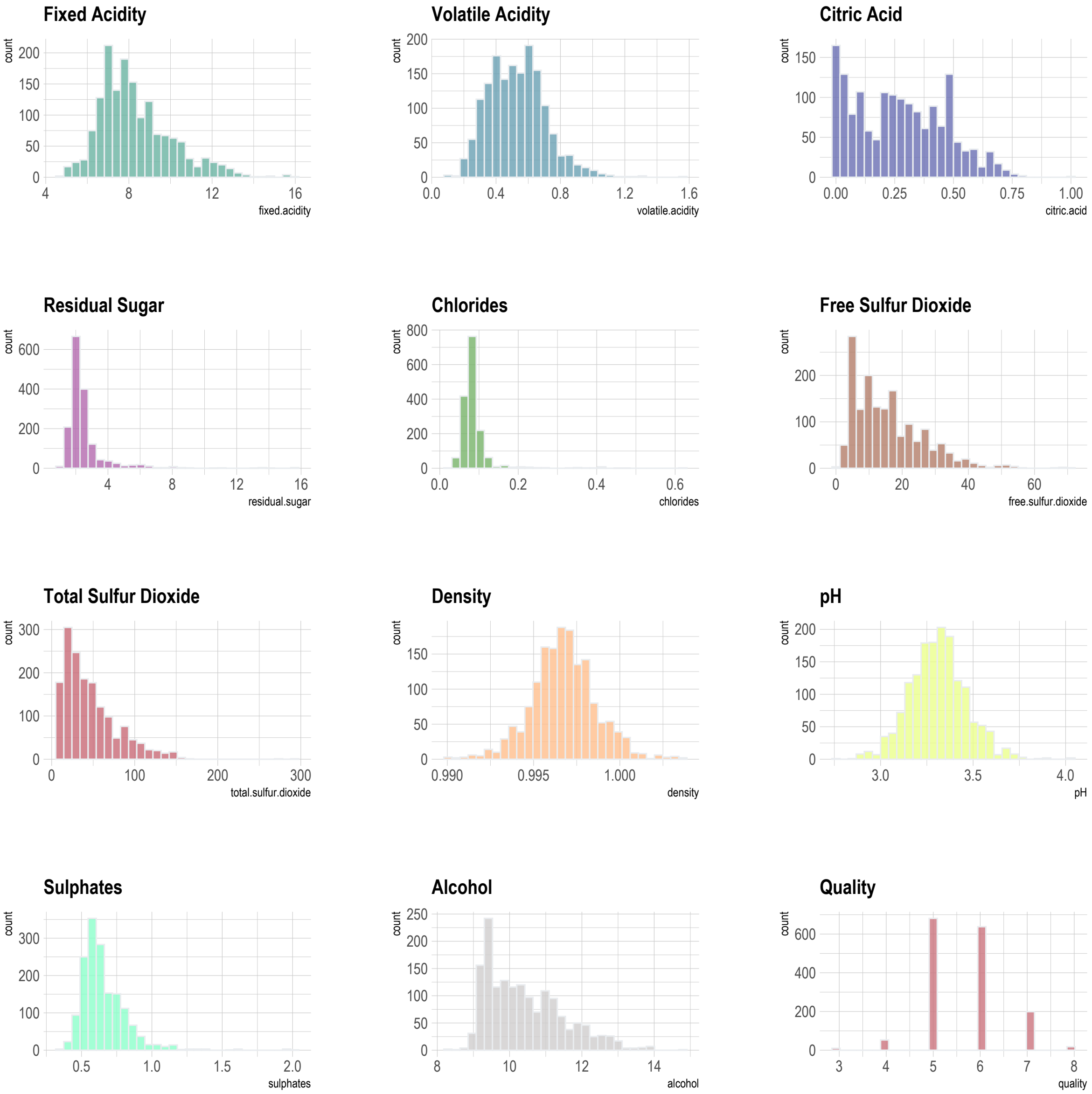
Table 2: Data summary

| | |
|---|---|
| Name | wine_df |
| Number of rows | 1599 |
| Number of columns | 12 |
| Column type frequency: | |
| numeric | 12 |
| Group variables | None |

**Variable type: numeric**

| skim_variable | n_missing | complete_rate | mean | sd | p0 | p25 | p50 | p75 | p100 | hist |
|---|---|---|---|---|---|---|---|---|---|---|
| fixed.acidity | 0 | 1 | 8.32 | 1.74 | 4.60 | 7.10 | 7.90 | 9.20 | 15.90 | |
| volatile.acidity | 0 | 1 | 0.53 | 0.18 | 0.12 | 0.39 | 0.52 | 0.64 | 1.58 | |
| citric.acid | 0 | 1 | 0.27 | 0.19 | 0.00 | 0.09 | 0.26 | 0.42 | 1.00 | |
| residual.sugar | 0 | 1 | 2.54 | 1.41 | 0.90 | 1.90 | 2.20 | 2.60 | 15.50 | |
| chlorides | 0 | 1 | 0.09 | 0.05 | 0.01 | 0.07 | 0.08 | 0.09 | 0.61 | |
| free.sulfur.dioxide | 0 | 1 | 15.87 | 10.46 | 1.00 | 7.00 | 14.00 | 21.00 | 72.00 | |
| total.sulfur.dioxide | 0 | 1 | 46.47 | 32.90 | 6.00 | 22.00 | 38.00 | 62.00 | 289.00 | |
| density | 0 | 1 | 1.00 | 0.00 | 0.99 | 1.00 | 1.00 | 1.00 | 1.00 | |
| pH | 0 | 1 | 3.31 | 0.15 | 2.74 | 3.21 | 3.31 | 3.40 | 4.01 | |
| sulphates | 0 | 1 | 0.66 | 0.17 | 0.33 | 0.55 | 0.62 | 0.73 | 2.00 | |
| alcohol | 0 | 1 | 10.42 | 1.07 | 8.40 | 9.50 | 10.20 | 11.10 | 14.90 | |
| quality | 0 | 1 | 5.64 | 0.81 | 3.00 | 5.00 | 6.00 | 6.00 | 8.00 | |

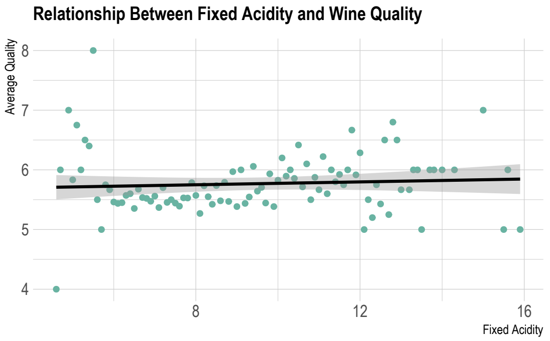
**Let's take a look at the distributions of the data set:**

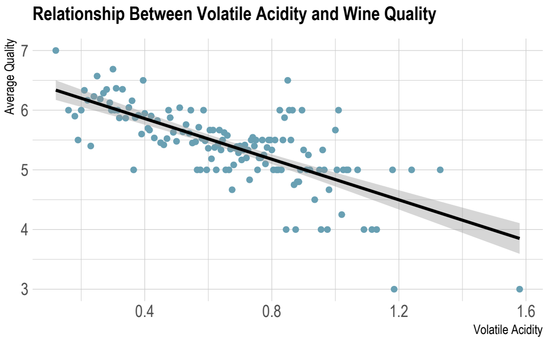


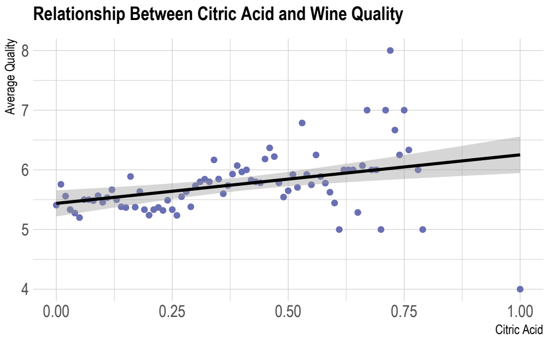
### Some notes on the visualizations above:

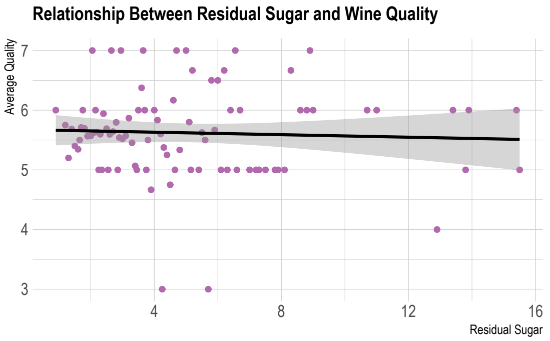
- Most of the distributions for the variables are right skewed with the exception of Density and pH
- Density and pH have more of a normal distribution
- Citric Acid has a more uniform distribution

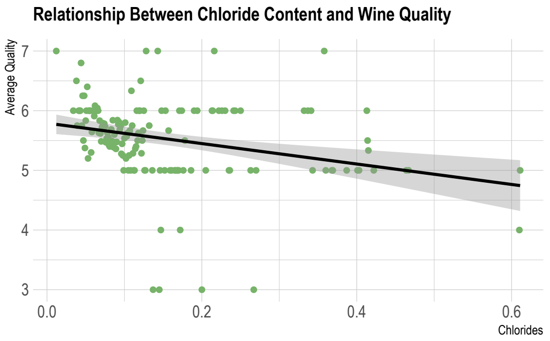
**Let's check if there's any relationships between the variables against the quality of the wine:**

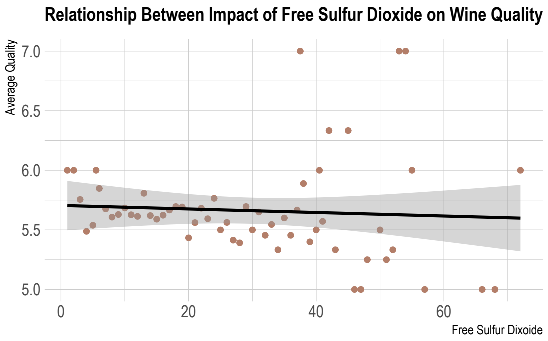

Relationship Between Fixed Acidity and Wine Quality

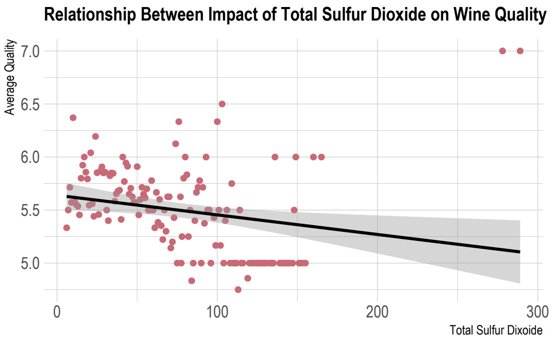

Relationship Between Volatile Acidity and Wine Quality

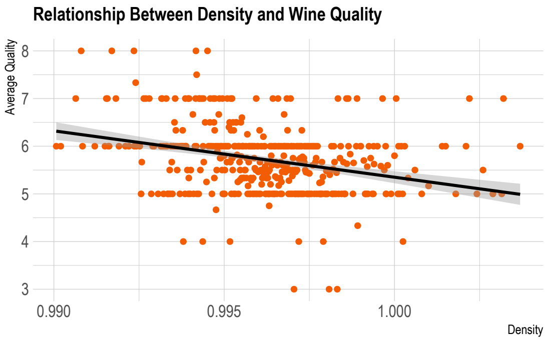

Relationship Between Citric Acid and Wine Quality

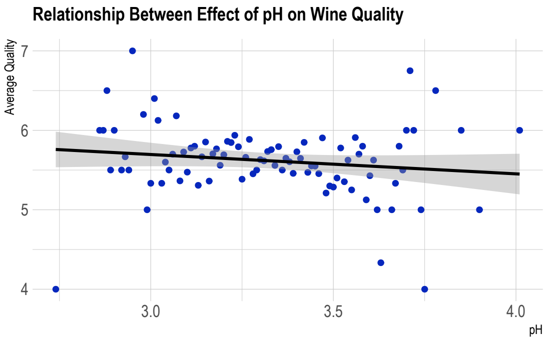

Relationship Between Residual Sugar and Wine Quality

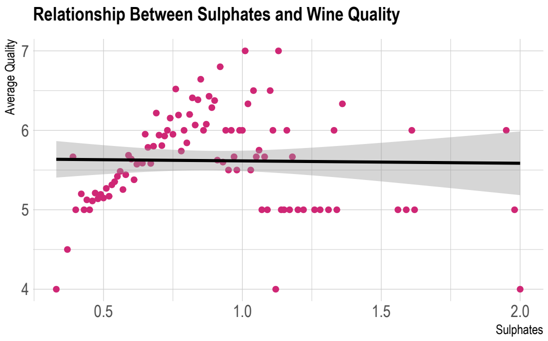

Relationship Between Chloride Content and Wine Quality

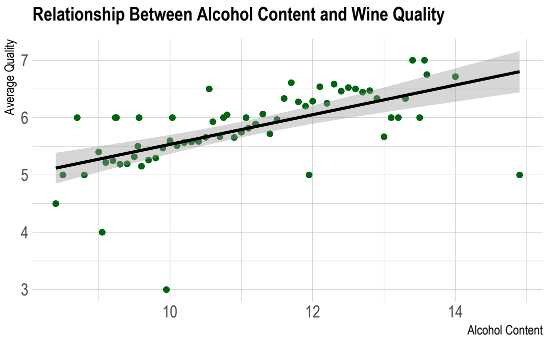












### Key takeaways from the scatterplot:

- There is no correlation between a wine's residual sugar and its quality rating.
- There's no visible relationship between chloride content, free sulfur dioxide, and wine quality.
- Wines containing higher levels of total sulfur dioxide are not consistently rated as low quality wines and don't provide a reliable indicator of wine quality.
- There is a slight negative relationship between a wine's density and it's quality rating. Higher density wines tend to have a slightly lower quality rating.
- There is very little to no correlation between pH and wine quality.

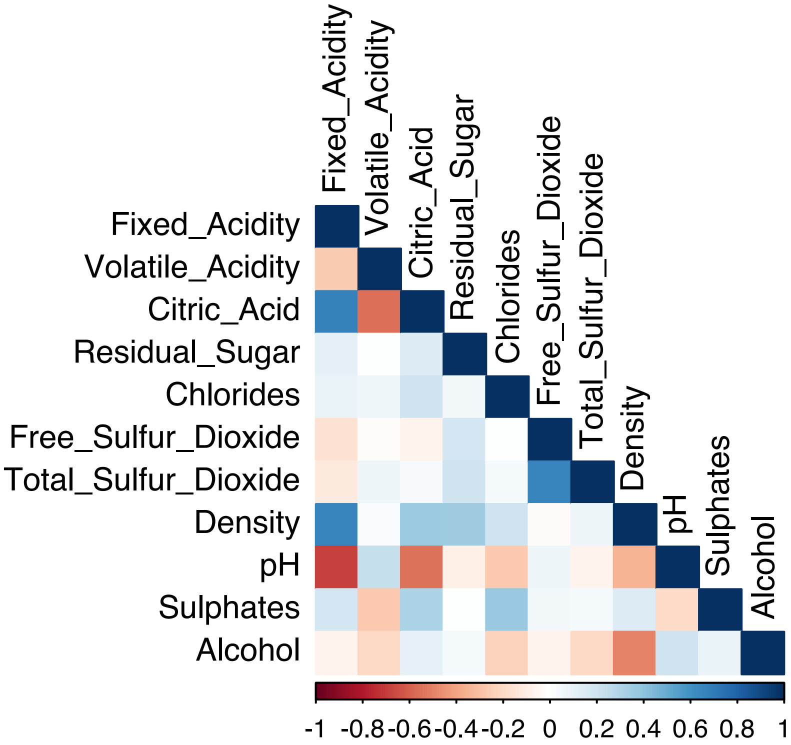
- There is a slight positive relationship between alcohol content and wine quality. The higher the alcohol content, the higher the average of the wine quality.

**Data Preparation:**

Now that I’ve visualized the data I want to do one minor change to the columns. Most of the columns have a “.” and I’m changing it to an “_“. I’ll also be converting the column `Quality` to factor. Since there’s no missing values there’s not much more to prepare the data.

| Fixed_Acidity | Volatile_Acidity | Citric_Acid | Residual_Sugar | Chlorides | Free_Sulfur_Dioxide | Total_Sulfur_Dioxide | Density | pH | Sulphates | Alcohol | Quality |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 7.4 | 0.70 | 0.00 | 1.9 | 0.076 | 11 | 34 | 0.9978 | 3.51 | 0.56 | 9.4 | 5 |
| 7.8 | 0.88 | 0.00 | 2.6 | 0.098 | 25 | 67 | 0.9968 | 3.20 | 0.68 | 9.8 | 5 |
| 7.8 | 0.76 | 0.04 | 2.3 | 0.092 | 15 | 54 | 0.9970 | 3.26 | 0.65 | 9.8 | 5 |
| 11.2 | 0.28 | 0.56 | 1.9 | 0.075 | 17 | 60 | 0.9980 | 3.16 | 0.58 | 9.8 | 6 |
| 7.4 | 0.70 | 0.00 | 1.9 | 0.076 | 11 | 34 | 0.9978 | 3.51 | 0.56 | 9.4 | 5 |
| 7.4 | 0.66 | 0.00 | 1.8 | 0.075 | 13 | 40 | 0.9978 | 3.51 | 0.56 | 9.4 | 5 |

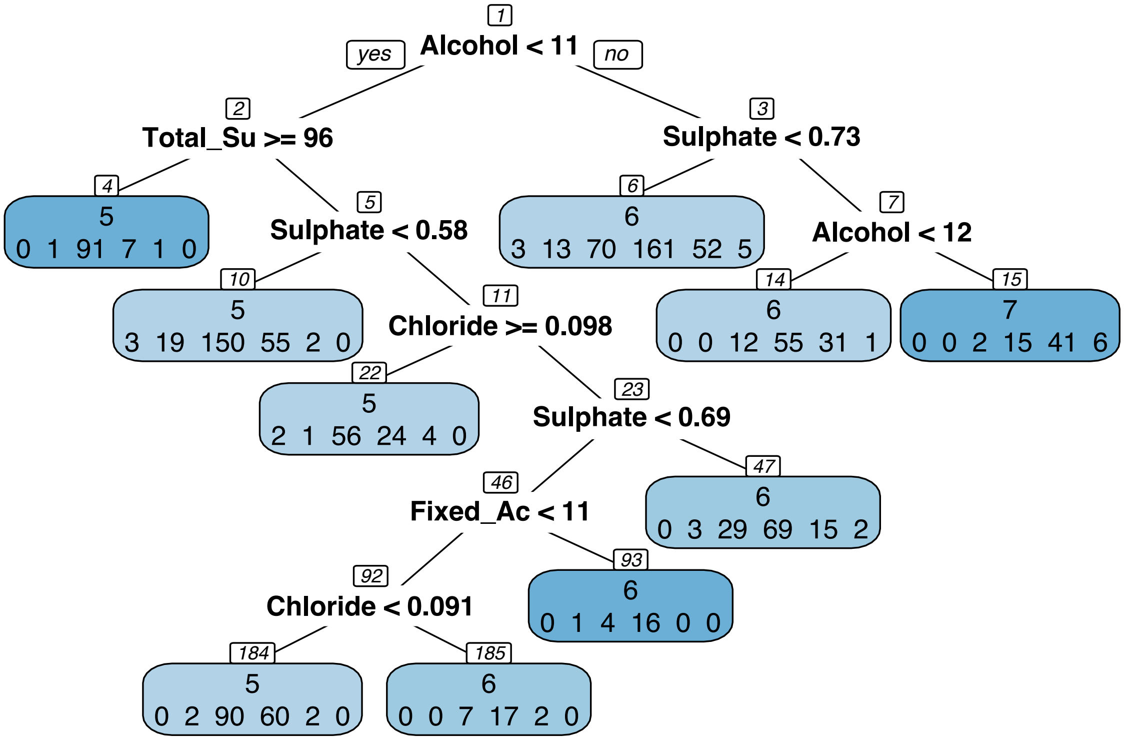
The correlation plot below is measuring the degree of linear relationship within the dataset. The values in which this is measured falls between -1 and +1, with +1 being a strong positive correlation and -1 a strong negative correlation. The darker the dot the more strongly correlated (whether positive or negative). From the results below, there’s a strong positive correlation with citric acid, density and fixed acidity as well as free sulfur dioxide and total sulfur dioxide. Negative strong correlations are only seen with fixed acidity and pH, citric acid and volatile acidy, citric acid and pH, and density and alcohol.



**Model Building Decision Tree and Random Forest:**

Building from the previous homework I am recreating the Decision Trees and Random Forest models. If you recall, I had some issues displaying the confusion matrix for both models so I have improved on this to hopefully get a better accuracy of the models and be able to compare it with the support vector machines (SVM).

The first decision tree is between `Quality` and the whole data set and started off by doing the cross validations setup by using the 75:25 ratio. Below is the decision tree created:



Then we test the model using the validation dataset. The results are seen in the confusion matrix and statistics output:

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   3   4   5   6   7   8
##          3   0   0   0   0   0   0
##          4   0   0   0   0   0   0
##          5   2   6 132  61   4   0
##          6   0   7  38  87  34   3
##          7   0   0   0  11  11   1
##          8   0   0   0   0   0   0
##
## Overall Statistics
##
##                Accuracy : 0.5793
##                  95% CI : (0.5291, 0.6284)
##     No Information Rate : 0.4282
##     P-Value [Acc > NIR] : 1.021e-09
##
##                   Kappa : 0.3004
##
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
```

```
##
##                      Class: 3 Class: 4 Class: 5 Class: 6 Class: 7 Class: 8
## Sensitivity          0.000000  0.00000   0.7765   0.5472  0.22449  0.00000
## Specificity          1.000000  1.00000   0.6784   0.6555  0.96552  1.00000
## Pos Pred Value            NaN      NaN   0.6439   0.5148  0.47826      NaN
## Neg Pred Value       0.994962  0.96725   0.8021   0.6842  0.89840  0.98992
## Prevalence           0.005038  0.03275   0.4282   0.4005  0.12343  0.01008
## Detection Rate       0.000000  0.00000   0.3325   0.2191  0.02771  0.00000
## Detection Prevalence 0.000000  0.00000   0.5164   0.4257  0.05793  0.00000
## Balanced Accuracy    0.500000  0.50000   0.7274   0.6013  0.59500  0.50000
```

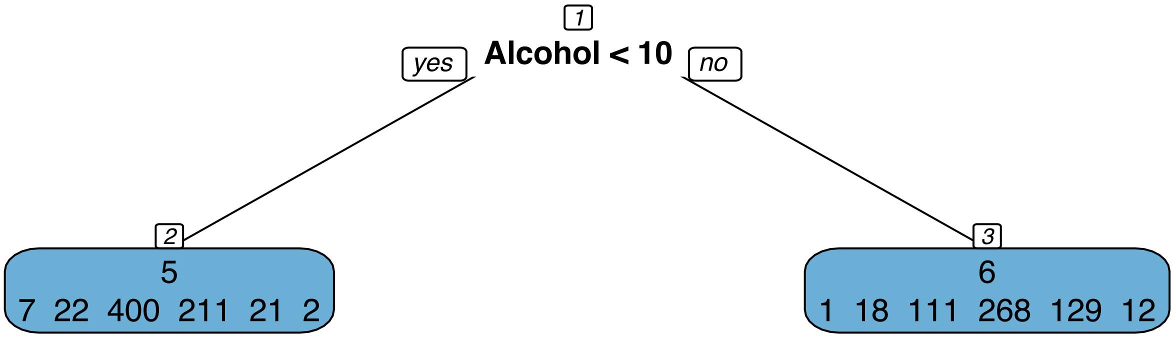
Let's look at the contribution of each variable:

| | Overall |
|---|---:|
| Alcohol | 108.070244 |
| Chlorides | 17.151769 |
| Citric_Acid | 28.034564 |
| Density | 42.102816 |
| Fixed_Acidity | 36.316930 |
| Free_Sulfur_Dioxide | 6.357231 |
| pH | 6.169872 |
| Residual_Sugar | 2.400345 |
| Sulphates | 82.234256 |
| Total_Sulfur_Dioxide | 53.237707 |
| Volatile_Acidity | 77.456655 |

and we check the accuracy which is 58% (previous accuracy was 57.4%):

| | x |
|---|---:|
| Accuracy | 0.5793451 |

**Switching Variables:** We were also asked to switch variables and create a second decision tree. I looked at the relationship between `Quality` and `Density`, `pH`, and `Alcohol` that yield an accuracy of 57%. Upon making changes this accuracy went down. Below is the output of this decision tree.



Same as before, we create the confusion matrix and statistics for the second decision tree:

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   3   4   5   6   7   8
##          3   0   0   0   0   0   0
##          4   0   0   0   0   0   0
##          5   0   8 132  67   5   0
##          6   2   5  38  92  44   4
##          7   0   0   0   0   0   0
##          8   0   0   0   0   0   0
##
## Overall Statistics
##
##                Accuracy : 0.5642
##                  95% CI : (0.5139, 0.6136)
##     No Information Rate : 0.4282
##     P-Value [Acc > NIR] : 3.518e-08
##
##                   Kappa : 0.2547
##
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
```

```
##                      Class: 3 Class: 4 Class: 5 Class: 6 Class: 7 Class: 8
## Sensitivity          0.000000  0.00000   0.7765   0.5786   0.0000  0.00000
## Specificity          1.000000  1.00000   0.6476   0.6092   1.0000  1.00000
## Pos Pred Value            NaN      NaN   0.6226   0.4973      NaN      NaN
## Neg Pred Value       0.994962  0.96725   0.7946   0.6840   0.8766  0.98992
## Prevalence           0.005038  0.03275   0.4282   0.4005   0.1234  0.01008
## Detection Rate       0.000000  0.00000   0.3325   0.2317   0.0000  0.00000
## Detection Prevalence 0.000000  0.00000   0.5340   0.4660   0.0000  0.00000
## Balanced Accuracy    0.500000  0.50000   0.7120   0.5939   0.5000  0.50000
```

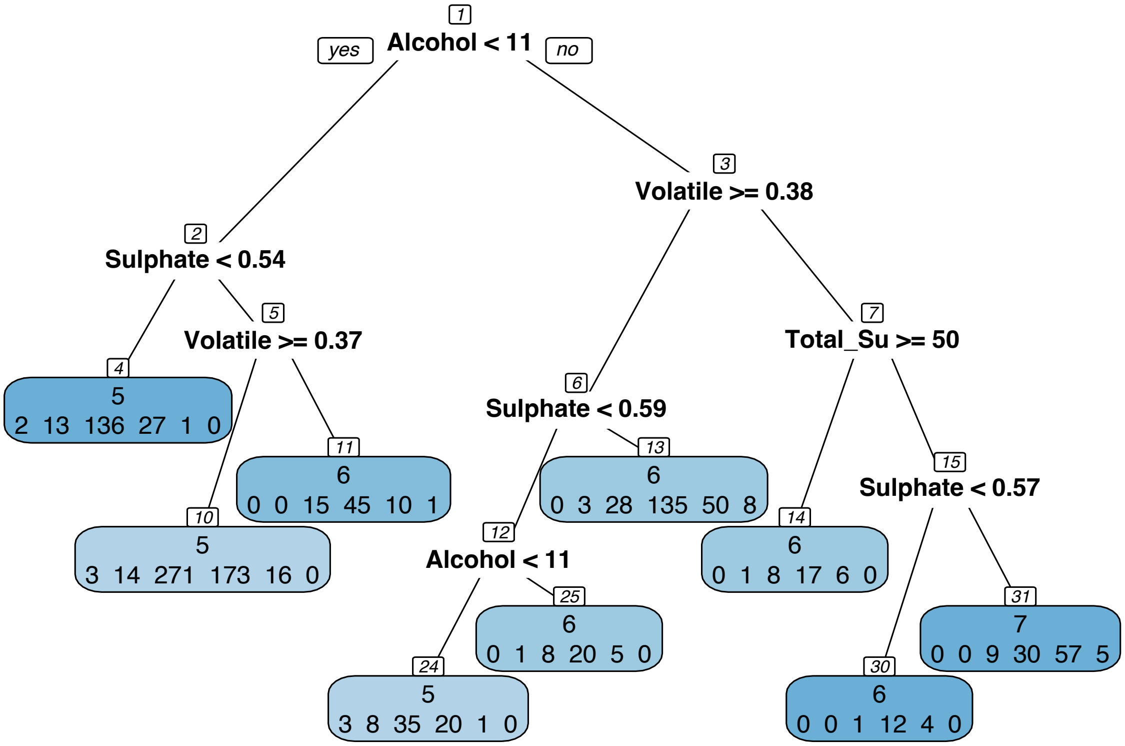
Let’s look at the contribution of each variable for the second dataset:

| | Overall |
|---|---:|
| Alcohol | 69.422698 |
| Density | 25.855801 |
| pH | 2.874687 |

and now for the accuracy of 56.4% which is lower than the first decision tree:

| | x |
|---|---:|
| Accuracy | 0.5642317 |

**Switching Variables Again** From the variable contribution in the first decision tree, I decided to create a third decision tree composed of `Quality`, `Alcohol`, `Sulphates`, `Volatile_Acidity`, and `Total_Sulfur_Dioxide` and view the changes in the model accuracy. Same as before, I created a new datasets from the original choosing only the variables above and followed the same steps to create this final decision tree.



The confusion matrix and statistics for the third decision tree:

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   3   4   5   6   7   8
##          3   0   0   0   0   0   0
##          4   0   0   0   0   0   0
##          5   2   8 146  78   8   0
##          6   0   5  22  68  23   3
##          7   0   0   2  13  18   1
##          8   0   0   0   0   0   0
##
## Overall Statistics
##
##                Accuracy : 0.5844
##                  95% CI : (0.5342, 0.6333)
##     No Information Rate : 0.4282
##     P-Value [Acc > NIR] : 2.896e-10
##
##                   Kappa : 0.3145
##
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
```

```
##                      Class: 3 Class: 4 Class: 5 Class: 6 Class: 7 Class: 8
## Sensitivity          0.000000  0.00000   0.8588   0.4277  0.36735  0.00000
## Specificity          1.000000  1.00000   0.5771   0.7773  0.95402  1.00000
## Pos Pred Value            NaN      NaN   0.6033   0.5620  0.52941      NaN
## Neg Pred Value       0.994962  0.96725   0.8452   0.6703  0.91460  0.98992
## Prevalence           0.005038  0.03275   0.4282   0.4005  0.12343  0.01008
## Detection Rate       0.000000  0.00000   0.3678   0.1713  0.04534  0.00000
## Detection Prevalence 0.000000  0.00000   0.6096   0.3048  0.08564  0.00000
## Balanced Accuracy    0.500000  0.50000   0.7180   0.6025  0.66068  0.50000
```

Let's look at the contribution of each variable for the third dataset:

| | Overall |
|---|---|
| Alcohol | 110.52522 |
| Sulphates | 85.19619 |
| Total_Sulfur_Dioxide | 71.14540 |
| Volatile_Acidity | 83.16022 |

and now for the accuracy of 58.4% which is higher than the first and second decision tree models:

| | x |
|---|---|
| Accuracy | 0.5843829 |

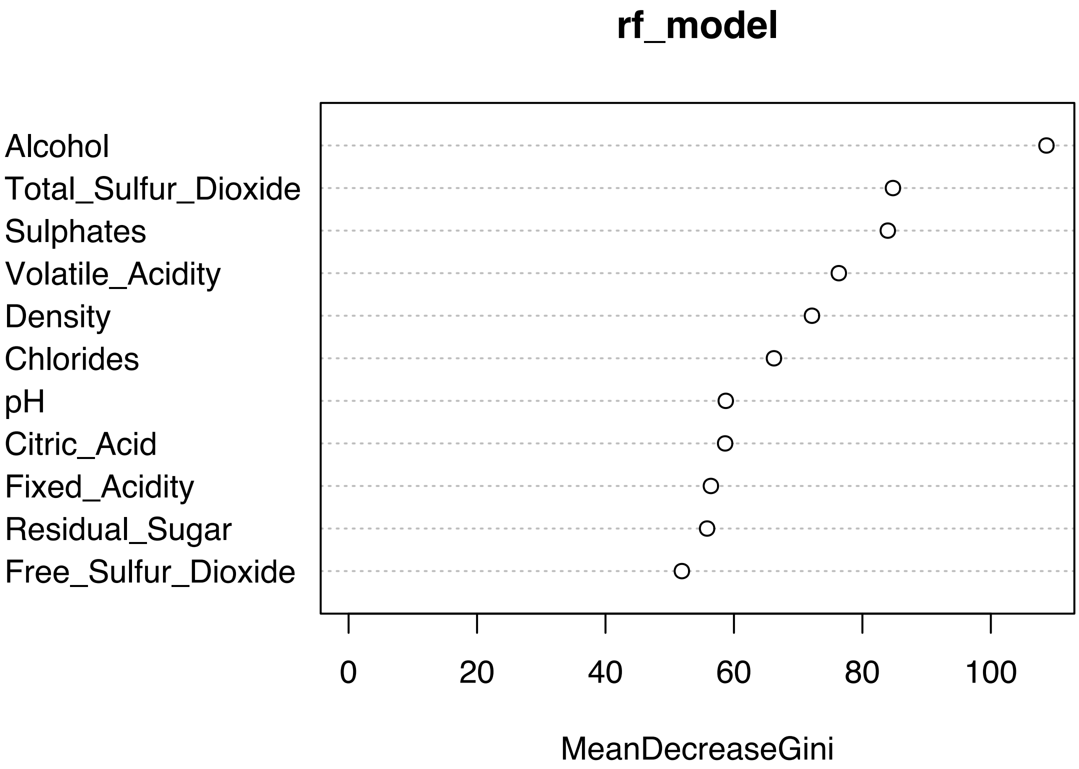
**Random Forest** For a second recap: we now create a random forest model for the dataset. A Random Forest is an ensemble learning technique in machine learning that combines multiple decision trees to make accurate predictions. It works by creating a collection of decision trees, each trained on a bootstrapped dataset (randomly sampled with replacement) from the original data and considering only a subset of features at each split. The final prediction in a classification task is determined by a majority vote of the individual trees, while in a regression task, it's an average of their predictions. Random Forests are valued for their high accuracy, resistance to overfitting, and the ability to assess feature importance.

For the random forest model, I first chose the first decision tree as it had a higher accuracy compared to the second model. Create the random forest model using the training data and then applying it to the validation data. A new addition to this model is that now I will create a second random forest model with the third decision tree model and make the comparison. Below are the results:

```
##
## Call:
##  randomForest(formula = Quality ~ ., data = train)
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 3
##
##         OOB estimate of  error rate: 31.86%
## Confusion matrix:
##   3 4   5   6  7 8 class.error
## 3 0 0   7   1  0 0   1.0000000
## 4 0 0  27  13  0 0   1.0000000
## 5 1 1 411  93  5 0   0.1956947
## 6 0 1 116 334 27 1   0.3027140
## 7 0 0   9  66 74 1   0.5066667
## 8 0 0   0   8  6 0   1.0000000
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   3   4   5   6   7   8
##          3   0   1   0   0   0   0
##          4   0   0   0   0   0   0
##          5   2   7 148  44   1   0
##          6   0   4  22 104  18   2
##          7   0   1   0  11  30   1
##          8   0   0   0   0   0   1
##
## Overall Statistics
##
##                Accuracy : 0.7128
##                  95% CI : (0.6656, 0.7569)
##     No Information Rate : 0.4282
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.5349
##
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: 3 Class: 4 Class: 5 Class: 6 Class: 7 Class: 8
## Sensitivity          0.000000  0.00000   0.8706   0.6541  0.61224 0.250000
## Specificity          0.997468  1.00000   0.7621   0.8067  0.96264 1.000000
## Pos Pred Value       0.000000      NaN   0.7327   0.6933  0.69767 1.000000
## Neg Pred Value       0.994949  0.96725   0.8872   0.7773  0.94633 0.992424
## Prevalence           0.005038  0.03275   0.4282   0.4005  0.12343 0.010076
## Detection Rate       0.000000  0.00000   0.3728   0.2620  0.07557 0.002519
## Detection Prevalence 0.002519  0.00000   0.5088   0.3778  0.10831 0.002519
## Balanced Accuracy    0.498734  0.50000   0.8164   0.7304  0.78744 0.625000
```

From the random forest model we can create a variable importance plot which shows each variable and how important it is in classifying the data. From the plot below we note that `Alcohol`, `Total_Sulfur_Dixoide` and `Sulphates` are among the top variables that play a significant role in the classification of the quality of the wine.



Numerically, we can see the same result below:

| | Overall |
|---|---|
| Fixed_Acidity | 56.41020 |
| Volatile_Acidity | 76.32992 |
| Citric_Acid | 58.62591 |
| Residual_Sugar | 55.81955 |
| Chlorides | 66.23051 |
| Free_Sulfur_Dioxide | 51.89288 |
| Total_Sulfur_Dioxide | 84.75798 |
| Density | 72.14677 |
| pH | 58.72748 |
| Sulphates | 83.95732 |
| Alcohol | 108.65654 |

Lastly, I check the accuracy on the validation data with the results of 71.3% accuracy seen below:

```
##  Accuracy
## 0.7128463
```

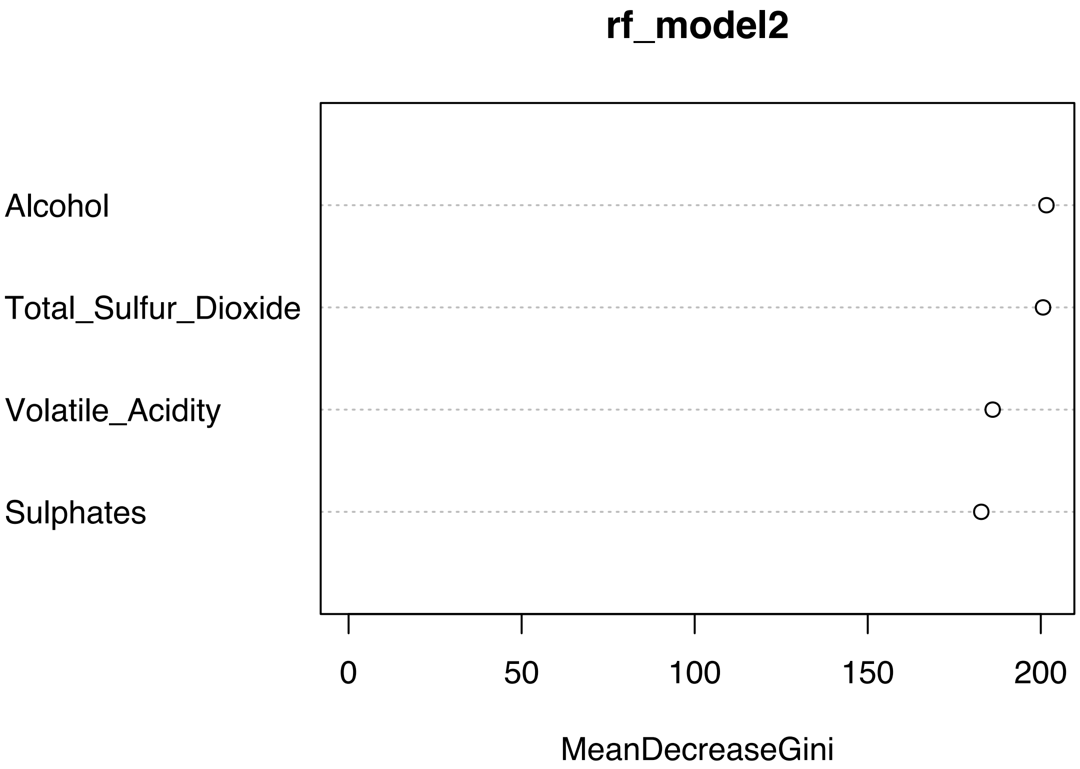
**Second Random Forest Model** Now to create the second random forest with the third dataset using the variables `Quality`, `Alcohol`, `Volatile_Acidity`, `Sulphates`, and `Total_Sulfur_Dioxide`. The results are below:

```
##
## Call:
##  randomForest(formula = Quality ~ Alcohol + Volatile_Acidity +      Sulphates + Total_Sulfur_Dioxide
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 2
##
##         OOB estimate of  error rate: 34.03%
## Confusion matrix:
##   3 4   5   6  7 8 class.error
## 3 0 1   6   1  0 0   1.0000000
## 4 0 2  26  12  0 0   0.9500000
## 5 0 5 400 103  3 0   0.2172211
## 6 0 1 119 317 41 1   0.3382046
## 7 0 0  11  65 73 1   0.5133333
## 8 0 0   0   2 11 1   0.9285714
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   3   4   5   6   7   8
##          3   0   0   0   0   0   0
##          4   0   0   0   0   0   0
##          5   1   7 134  28   4   0
##          6   1   6  35 116  17   2
##          7   0   0   1  15  28   1
##          8   0   0   0   0   0   1
##
## Overall Statistics
##
##                Accuracy : 0.7028
##                  95% CI : (0.6552, 0.7473)
##     No Information Rate : 0.4282
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.5204
##
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: 3 Class: 4 Class: 5 Class: 6 Class: 7 Class: 8
## Sensitivity          0.000000  0.00000   0.7882   0.7296  0.57143 0.250000
## Specificity          1.000000  1.00000   0.8238   0.7437  0.95115 1.000000
## Pos Pred Value            NaN      NaN   0.7701   0.6554  0.62222 1.000000
## Neg Pred Value       0.994962  0.96725   0.8386   0.8045  0.94034 0.992424
## Prevalence           0.005038  0.03275   0.4282   0.4005  0.12343 0.010076
## Detection Rate       0.000000  0.00000   0.3375   0.2922  0.07053 0.002519
## Detection Prevalence 0.000000  0.00000   0.4383   0.4458  0.11335 0.002519
## Balanced Accuracy    0.500000  0.50000   0.8060   0.7366  0.76129 0.625000
```

From the random forest model we created, we can create a variable importance plot which shows each variable and how important it is in classifying the data. From the plot below we note that `Alcohol` and

Total_Sulfur_Dioxide are among the top variables that play a significant role in the classification of the quality of the wine.



Numerically, we can see the same result below:

| | Overall |
|---|---|
| Alcohol | 201.6277 |
| Volatile_Acidity | 186.0988 |
| Sulphates | 182.8026 |
| Total_Sulfur_Dioxide | 200.6466 |

Lastly, we check on the validation data's accuracy of the second model with the results of 70.3% accuracy seen below:

```
##  Accuracy
## 0.7027708
```

**Model Building SVM:**

A Support Vector Machine (SVM) is a supervised machine learning algorithm used for classification and regression tasks. It is particularly effective for classification tasks in which the goal is to divide data points into different classes based on their features. Due to their effectiveness in handling high-dimensional data and their ability to perform well with relatively small datasets SVM is used in various fields.

We were asked to create an SVM algorithm with the same data used and make the comparison. To start the algorithm, we follow similar criteria to the decision tree and random forest model by setting up the cross validation set-up, create the prediction and confusion matrix and lastly it's accuracy. Results are below:

The confusion matrix for SVM:

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   3   4   5   6   7   8
##          3   0   0   0   0   0   0
##          4   0   0   0   0   0   0
##          5   2   9 138  51   1   0
##          6   0   3  31  99  32   3
##          7   0   1   1   9  16   1
##          8   0   0   0   0   0   0
##
## Overall Statistics
##
##                Accuracy : 0.6373
##                  95% CI : (0.5878, 0.6847)
##     No Information Rate : 0.4282
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.4005
##
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: 3 Class: 4 Class: 5 Class: 6 Class: 7 Class: 8
## Sensitivity          0.000000  0.00000   0.8118   0.6226  0.32653  0.00000
## Specificity          1.000000  1.00000   0.7225   0.7101  0.96552  1.00000
## Pos Pred Value            NaN      NaN   0.6866   0.5893  0.57143      NaN
## Neg Pred Value       0.994962  0.96725   0.8367   0.7380  0.91057  0.98992
## Prevalence           0.005038  0.03275   0.4282   0.4005  0.12343  0.01008
## Detection Rate       0.000000  0.00000   0.3476   0.2494  0.04030  0.00000
## Detection Prevalence 0.000000  0.00000   0.5063   0.4232  0.07053  0.00000
## Balanced Accuracy    0.500000  0.50000   0.7671   0.6664  0.64602  0.50000
```

The summary of the SVM results:

```
##   3   4   5   6   7   8
##   0   0 201 168  28   0
```

The accuracy of this SVM is 63.7% for the original data set:

```
##  Accuracy
## 0.6372796
```

**Second SVM algorithm** Decided to do a second SVM algorithm to check for any changes in accuracy, these results are below.

The confusion matrix for the second SVM:

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   3   4   5   6   7   8
##          3   0   0   0   0   0   0
##          4   0   0   0   0   0   0
##          5   2   7 121  48   1   0
##          6   0   6  49 108  37   1
##          7   0   0   0   3  11   3
##          8   0   0   0   0   0   0
##
## Overall Statistics
##
##                Accuracy : 0.6045
##                  95% CI : (0.5545, 0.653)
##     No Information Rate : 0.4282
##     P-Value [Acc > NIR] : 1.246e-12
##
##                   Kappa : 0.3396
##
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: 3 Class: 4 Class: 5 Class: 6 Class: 7 Class: 8
## Sensitivity          0.000000  0.00000   0.7118   0.6792  0.22449  0.00000
## Specificity          1.000000  1.00000   0.7445   0.6092  0.98276  1.00000
## Pos Pred Value            NaN      NaN   0.6760   0.5373  0.64706      NaN
## Neg Pred Value       0.994962  0.96725   0.7752   0.7398  0.90000  0.98992
## Prevalence           0.005038  0.03275   0.4282   0.4005  0.12343  0.01008
## Detection Rate       0.000000  0.00000   0.3048   0.2720  0.02771  0.00000
## Detection Prevalence 0.000000  0.00000   0.4509   0.5063  0.04282  0.00000
## Balanced Accuracy    0.500000  0.50000   0.7281   0.6442  0.60362  0.50000
```

The summary of the SVM results:

```
##   3   4   5   6   7   8
##   0   0 179 201  17   0
```

The accuracy of the second SVM is 60.5% for the original data set which is lower than the first SVM accuracy.

```
## Accuracy
## 0.604534
```

**Comparison of models:** Lastly, let's do a model comparison for Decision Tree, Random Forest and SVM:

```
##             Model  Accuracy
## 2 Random Forest 0.7128463
## 3           SVM 0.6372796
## 1 Decision Tree 0.5843829
```

**Conclusion:**

**Decision Tree:** The Decision Tree model using the rpart algorithm achieved an accuracy of 58.4%. The confusion matrix revealed a limited ability to predict wine quality, particularly for classes 3, 4, and 8, where the sensitivity was low.

**Random Forest:** The Random Forest model outperformed the Decision Tree with an accuracy of 71.3%. The confusion matrix showed improved predictions across all classes compared to the Decision Tree, resulting in better specificity but still had a low sensitivity in classes 3 and 4.

**Support Vector Machine (SVM):** The SVM model achieved an accuracy of 63.7%. While it showed high specificity for most classes, it struggled with low sensitivity in classes 3, 4, and 8.

Overall, I'd recommend Random Forest as the algorithm of choice for this dataset or similar for more accurate results since it outperformed decision tree by almost 20% and SVM by 11%. Random Forest is a versatile algorithm that performs well in both classification and regression scenarios. Its ability to handle high-dimensional data, deal with non-linear relationships, and reduce overfitting makes it a popular choice across a wide range of machine learning applications. Keep in mind the selection between using Random Forest for classification or regression often depends on the specific nature of the problem and the characteristics of the dataset being analyzed.

**Academic Content:**

- Detecting Credit Card Fraud by Decision Trees and Support Vector Machines

The article discusses credit card fraud detection using decision trees and Support Vector Machines (SVMs) in response to the escalating fraud rates causing substantial financial losses globally. While preventive measures like CHIP&PIN exist, they often fail to curb prevalent fraud types like virtual POS terminal or mail order credit card fraud. As a result, fraud detection becomes crucial. The study compares the effectiveness of SVM and decision tree-based models for credit card fraud detection using real datasets.

It emphasizes the challenges of fraud detection due to limited transaction data, constantly changing fraudulent behavior, limited collaboration on fraud detection ideas, lack of available datasets for benchmarking, highly skewed data with minimal fraudulent instances, and constantly evolving fraudulent behaviors.

The study employs decision tree algorithms (C5.0, C&RT, CHAID) and different SVM methods (polynomial, sigmoid, linear, RBF kernels) to build models based on different ratios of fraudulent to normal records. The performance of these models is assessed using accuracy rates on training and testing datasets.

Results show that decision tree models generally outperform SVM models, especially in catching fraudulent transactions. Although SVM models initially tend to overfit training data, their performance improves with larger datasets but still lags behind decision tree models in identifying fraudulent transactions.

- Credit Card Fraud Detection using Decision Tree and Random Forest

The article discusses the importance of secure credit card fraud detection systems in the era of technological advancement and increased online shopping. It highlights the benefits of online shopping, particularly the convenience and time-saving aspects, along with the popularity of credit card payments. However, it addresses the significant concern of rising fraudulent credit card transactions, causing financial losses for both banks and customers.

The paper explores the application of various machine learning algorithms for credit card fraud detection, including Naïve Bayes, Logistic Regression, SVM, Decision Trees, Random Forest, Genetic Algorithm, J48, and AdaBoost. These algorithms are utilized to analyze datasets and accurately identify fraudulent transactions.

- Demand forecasting with color parameter in retail apparel industry using artificial neural networks (ANN) and support vector machines (SVM) methods

The article explores demand forecasting in the retail apparel industry, specifically focusing on the impact of including color details in predictive models. The study aims to improve sales forecasting accuracy by utilizing artificial intelligence (AI) techniques, such as artificial neural networks (ANN) and support vector machines (SVM). These models are used to predict sales while considering various factors like weather, gender, special days, and, notably, color details of products.

The importance of demand forecasting is highlighted due to its significant impact on a company's success. Inaccurate predictions can lead to reduced sales, loss of reputation, and income. Traditional forecasting methods often require complete data, while AI systems can handle missing data and process large datasets more effectively.

The study conducts demand forecasting using ANN and SVM models across different datasets involving nine products separately and one combined dataset. It compares the models' performance by considering the root mean square error (RMSE). The results indicate that ANN outperformed SVM in seven out of ten datasets without color details, while their performances were similar for datasets including color details.

Additionally, the article provides theoretical information about ANN and SVM. Practical applications of these models using R programming language on sales data from a textile retailer are discussed. The conclusion highlights the growing significance of accurate demand forecasting in shaping business strategies, especially in an industry characterized by rapid changes in customer demand.

**References:**

1. Sahin, Yusuf & Duman, Ekrem. (2011). Detecting Credit Card Fraud by Decision Trees and Support Vector Machines. IMECS 2011 - International MultiConference of Engineers and Computer Scientists 2011. 1. 442-447.

2. Shah, D., & Kumar Sharma, L. (2023). Credit Card Fraud Detection using Decision Tree and Random Forest. ITM Web of Conferences, 53, 2012-. https://doi.org/10.1051/itmconf/20235302012

3. İlker Güven, Fuat Şimşir, Demand forecasting with color parameter in retail apparel industry using artificial neural networks (ANN) and support vector machines (SVM) methods, Computers & Industrial Engineering, Volume 147,2020,106678,ISSN 0360-8352,https://doi.org/10.1016/j.cie.2020.106678. (https://www.sciencedirect.com/science/article/pii/S0360835220304125)