*Summary of findings:*

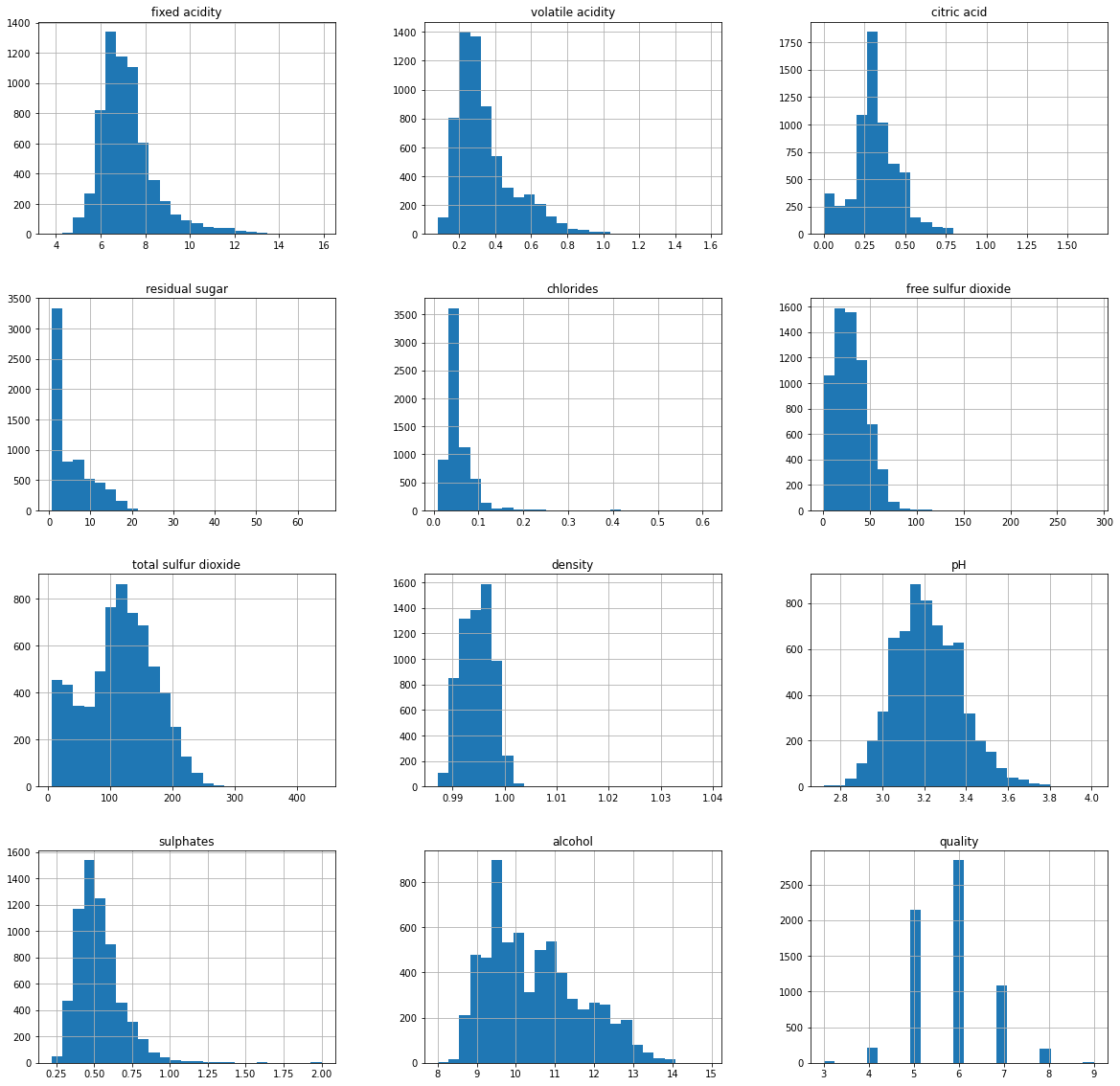
*Dataset:*

‘Vinho Verde’ wine samples available on: <http://archive.ics.uci.edu/ml/datasets/Wine+Quality>

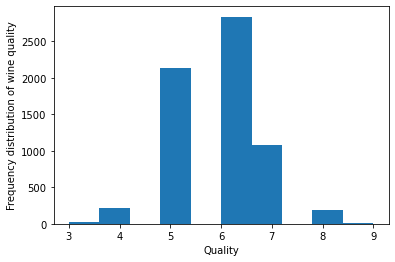
1. *Missing values:*

This consolidated dataset would have no missing values.

2. *Distribution of all the attributes in the consolidated dataset:*

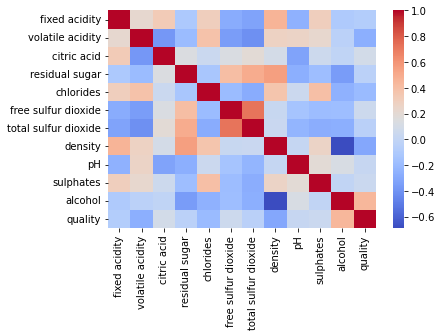
We can see that all the attributes are following different distributions, also have different ranges and scale. We can also see that there would be no outliers, which would need actioning.

3. *Distribution of our attribute of interest, the ‘quality’ attribute:*



It’s interesting to note that 6053 (2836 + 2138 + 1079) entries out of the total 6497 entries, about 93%, would have a quality rating of either 5 / 6 or 7.

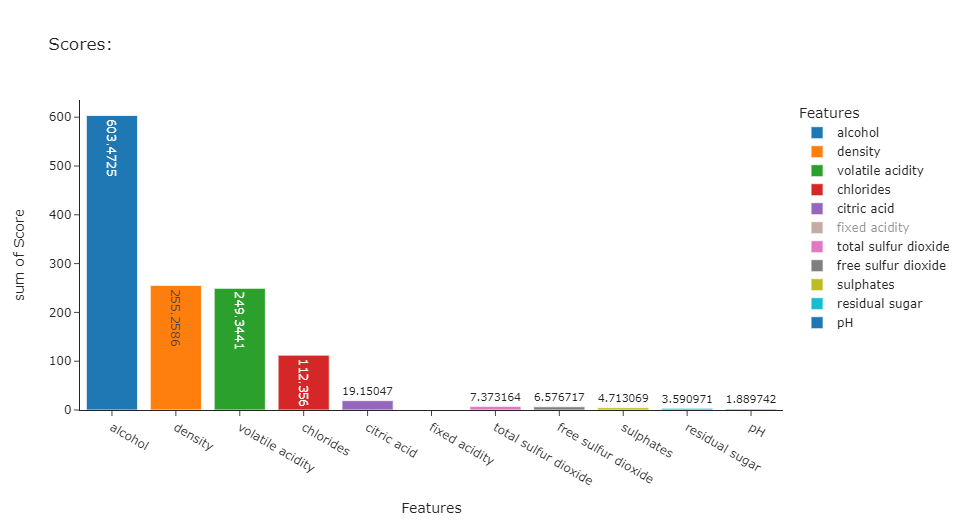
4. *Correlations between the attributes:*



With respect to the quality attribute, strongest correlations seen between: the quality / alcohol, quality / density, quality / volatile acidity and quality / chlorides pairs.

As regards the other set of attributes, strongest correlations seen between: the free sulfur dioxide / total sulfur dioxide, alcohol / density, density / residual sugar, residual sugar / total sulfur dioxide and density / fixed acidity pairs.

5. *ANOVA ('analysis of variance') findings:*



Based on the above scores, we see that the 'alcohol', 'density', 'volatile acidity' and 'chlorides' attributes would be impacting quality the most.

These results are also consistent with the results of the correlation matrix.

The other interesting thing, is that the scores of the other seven physicochemical features, would be very low as compared to these four attributes, and we can omit the lower score features and use a subset of these four high score features for our predictive model building tasks.

6. *Accuracy scores of the implemented algorithms:*

*6.1. For the 80:20 train: test split:*

|  |  |
| --- | --- |
| **Algorithm:** | **Accuracy:** |
| Decision Tree implementation: using all features ('criterion': 'gini', 'max\_depth': 9, 'max\_features': 8) | 0.768461538 |
| Decision Tree implementation: using the subset of four selected features: ('criterion': 'gini', 'max\_depth': 6, 'max\_features': 2) | 0.753846154 |
| Random Forest implementation: using all features | 0.828461538 |
| Random Forest implementation: using the subset of four selected features | 0.813076923 |
| k-Nearest Neighbour (k-NN) implementation: using all features: ('metric': 'minkowski', 'n\_neighbors': 6, 'weights': 'distance') | 0.792307692 |
| k-Nearest Neighbour (k-NN) implementation: using the subset of four selected features: ('metric': 'chebyshev', 'n\_neighbors': 6, 'weights': 'distance') | 0.800769231 |
| Support Vector Machine (SVM) implementation: using all features | 0.77 |
| Support Vector Machine (SVM) implementation: using the subset of four selected features | 0.756923077 |

*6.2. For the 75:25 train: test split:*

|  |  |
| --- | --- |
| **Algorithm:** | **Accuracy:** |
| Decision Tree implementation: using all features ('criterion': 'gini', 'max\_depth': 9, 'max\_features': 6) | 0.744615385 |
| Decision Tree implementation: using the subset of four selected features: ('criterion': 'gini', 'max\_depth': 7, 'max\_features': 2) | 0.737846154 |
| Random Forest implementation: using all features | 0.836307692 |
| Random Forest implementation: using the subset of four selected features | 0.799384615 |
| k-Nearest Neighbour (k-NN) implementation: using all features: ('metric': 'minkowski', 'n\_neighbors': 6, 'weights': 'distance') | 0.792 |
| k-Nearest Neighbour (k-NN) implementation: using the subset of four selected features: ('metric': 'minkowski', 'n\_neighbors': 5, 'weights': 'distance') | 0.787076923 |
| Support Vector Machine (SVM) implementation: using all features | 0.764307692 |
| Support Vector Machine (SVM) implementation: using the subset of four selected features | 0.749538462 |

*6.3. For the 70:30 train: test split:*

|  |  |
| --- | --- |
| **Algorithm:** | **Accuracy:** |
| Decision Tree implementation: using all features ('criterion': 'gini', 'max\_depth': 8, 'max\_features': 5) | 0.730769231 |
| Decision Tree implementation: using the subset of four selected features: ('criterion': 'gini', 'max\_depth': 7, 'max\_features': 4) | 0.728205128 |
| Random Forest implementation: using all features | 0.819487179 |
| Random Forest implementation: using the subset of four selected features | 0.789230769 |
| k-Nearest Neighbour (k-NN) implementation: using all features: ('metric': 'minkowski', 'n\_neighbors': 6, 'weights': 'distance') | 0.788717949 |
| k-Nearest Neighbour (k-NN) implementation: using the subset of four selected features: ('metric': 'chebyshev', 'n\_neighbors': 6, 'weights': 'distance') | 0.779487179 |
| Support Vector Machine (SVM) implementation: using all features | 0.751794872 |
| Support Vector Machine (SVM) implementation: using the subset of four selected features | 0.734358974 |

After reviewing the model performance of all the implemented iterations, we see that the accuracy scores, when using the full set of features and when using the subset of the four selected features, is almost the same, for all the four algorithms. This would be consistent with our ANOVA (analysis of variance) findings, where we see that the scores of the other seven physicochemical features is very low as compared to these four features, hence a much lower impact on the ‘quality’ attribute.

The accuracy scores would be between the ranges, 72.820% and 83.630%. The lowest, 72.820%, for the Decision Tree implementation, using the subset of four selected features, for the 70:30, train: test split and the highest, 83.630%, for the Random Forest implementation, using all features in the sample set, for the 75:25, train: test split.

All the four implemented algorithms provide robust performance. Amongst them, the Random Forest algorithm, using all features, gives us the best overall performance for all the three train: test variations.