Analysis and Prediction of Wine Type and Quality

Group 5:
Farhan Hasan
Gabriela Pittari

Introduction

Wine is an alcoholic beverage produced by the fermentation of the juice of fruits, usually grapes. Grapes are naturally chemically balanced to ferment completely without requiring extra sugars, acids, enzymes or other nutrients. The process of fermentation generates chemical processes that modify the grapes' structure and attributes.

The following study presents an analysis of some wine properties. We are going to base part of our study in analyzing the differences of the attributes by wine color: red or white. Then we will also dedicate our study to predict the wine quality.

Objectives

1. To predict if a wine sample is a red or white wine.
2. To predict the quality of a wine sample, classifying it in low, medium, or high.

Data Preparation

We based our study using the dataset from Kaggle.com Wine Quality, which originally comes from the UCI Machine Learning Repository. The data set contains 6497 rows.

Attributes

The wine is composed of many elements which contribute to their quality. It's worth understanding what these features mean in the wine. Here's a small overview of them:

Type

The wine color comes from the color of the grape: red wine is made with red and black grapes, while white wine is made with white grapes. For the white wine production, seeds and skins are removed while the red wine production does not necessarily involve this. This difference in the process affects the taste and color, making the red wine richer and the white wine lighter and fruitier. This dataset has the **type** attribute which identifies the wine color.

Acidity

Acidity in wine is very important as it affects the taste. A wine that has a high acidity can be defined as zippy, bright, racy, angular, lively, crispy, fresh, firm. It's ideal to pair with food. While low acidity will be described as fat, stale, soft. Acidity is the most important attribute for sparkling wines.

Apart from modifying the taste in the wine it also allows it to live longer as it performs as a natural preservative for the wine structure.
There are different types of acids:

- **Fixed acidity:** Comes from the acids found in grapes. The fixed acidity is expressed in g(tartaric acid)/dm3 in the dataset.
- **Volatile acidity:** is caused by the presence of acetic acid. If there's a small amount of volatile acidity it enhances the wine smell. When the volatile acidity is too high, it adds a vinegary flavor and makes the wine quality decrease. The volatile acidity is expressed in g(acetic acid)/dm3 in the dataset.
- **Citric acid:** is introduced as a supplement during the wine fermentation process and it helps boost acidity overall. It's also used for stabilization purposes. The citric acid is expressed in g/dm3 in the dataset.

In the wine world the **pH** (potential of Hydrogen) is a numeric scale to specify the acidity of the wine. All wines lie on the acidic side of the pH spectrum, and most range from 2.5 to about 4.5 pH (a drink with pH=7 is neutral, water).

Residual Sugar

The sweetness in wine comes from the sugars in the grape juice: glucose and fructose. The different levels of these elements contribute to the **residual sugar**. Other sources of sugar can also be added during the fermentation process like corn sugar, yeast. The residual sugar is a measure of the amount of sugar solids at the end of the wine production process. The residual sugar is expressed in g/dm3 in the dataset.

Chlorides

Chlorides is the amount of salt in wine. It comes from different aspects of the grapes and environment conditions in which they have grown. The chlorides are expressed in g(sodium chloride)/dm3 in the dataset.

Sulphites

The sulphites in wine occur during its fermentation process. The sulphites in wine are inversionally proportional with the acidity and color. Residual sugar is directionally related to the amount of sulphites to prevent fermentation. Sulphites are naturally found but can also be added during the process. They help protect the wine from undesirable microbes and oxidation. There are multiple ways to measure the sulphites in wine: free sulfur dioxide, total sulfur dioxide and sulfates.

- **Free sulfur dioxide:** an excessive amount of free sulfur dioxide can be noticeable to consumers, hiding the wine's aromas and inhibiting its oxygenation while the breathing process. In high concentrations it contributes to sharp, bitter, metallic flavor. Sulfur dioxide is mainly used for killing harmful bacteria but at the same time preserving the quality and freshness. The free sulfur dioxide is expressed in mg/dm3 in the dataset.
- **Total sulfur dioxide**: is added to kill bacteria and preserve the wine quality. An excessive amount of this dioxide can damage the wine by adding undesirable odor. The total sulfur dioxide is expressed in mg/dm3 in the dataset.
- **Sulphates**: are mineral salts containing sulfur. Sulfates are essential for wine production. It contributes to the wine aroma and flavor. The sulfates are expressed in g(potassium sulphate)/dm3 in the dataset.

Alcohol

The **alcohol** is the result of yeast converting sugar in ethanol during the fermentation process. Alcohol levels affect wine taste, higher alcohol will make a stronger oilier taste while lower alcohol levels will make the wine lighter. The alcohol is expressed in % vol in the dataset.

Density

Density is a comparison of the weight of a specific volume of wine to an equivalent volume of water. It is generally used as a measure of the conversion of sugar to alcohol. The density is expressed in g/cm3 in the dataset.

Quality

Finally the **quality** attribute will score how good or bad the wine is, being its lower value 0 and highest 10.

Other important aspects to note about the attributes:

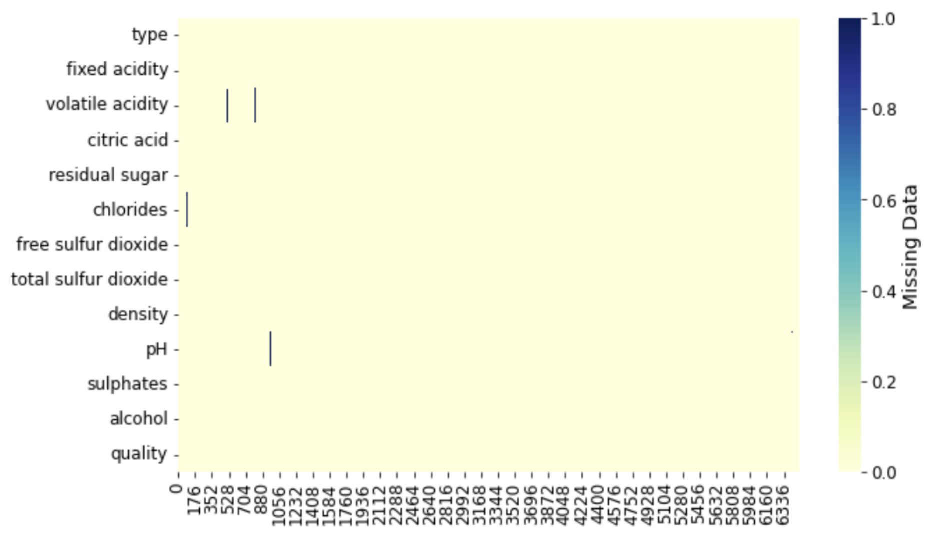
- The studied dataset does not contain information about grape types, wine brand or price.
- The classes are ordered and not balanced (e.g. there are much more normal wines than excellent or poor ones).
- Outlier detection algorithms could be used to detect the few excellent or poor wines.

Data Quality

Understanding the completeness of the data along the set was simple through the .info() method.

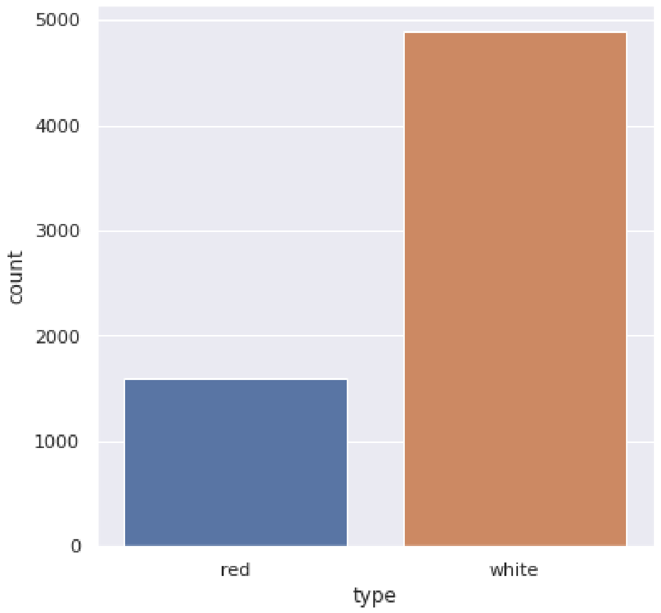
```
0   type                  6497 non-null   object
1   fixed acidity         6487 non-null   float64
2   volatile acidity      6489 non-null   float64
3   citric acid           6494 non-null   float64
4   residual sugar        6495 non-null   float64
5   chlorides             6495 non-null   float64
6   free sulfur dioxide   6497 non-null   float64
7   total sulfur dioxide  6497 non-null   float64
8   density               6497 non-null   float64
9   pH                    6488 non-null   float64
10  sulphates             6493 non-null   float64
11  alcohol               6497 non-null   float64
12  quality               6497 non-null   int64
```

Through this table it is possible to confirm that the type attribute is the only categorical variable, the quality is an integer and the rest of attributes are float types. The type variable was converted into a categorical variable and the missing rows were first studied a bit more to understand the percentage of missing data:



All missing values are a very low percentage in numerical features. We filled the null values with the mean for each attribute.

The wine types are red and white wine which we plan to predict based on the ingredients, but unfortunately the dataset is unbalanced. The data for red wine is only 25% of the complete dataset hence we needed to take that into consideration.



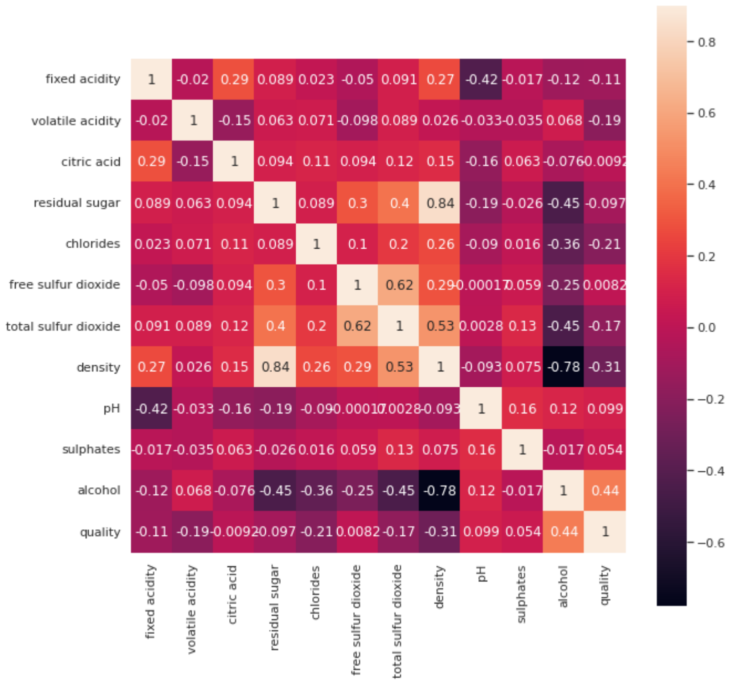
Data Analysis

The data analysis was made mainly by comparing the wine attributes for the wine type: red or white.

	key_0	Red Wine Stats	White Wine Stats
0	fixed acidity	8.320721	6.856121
1	volatile acidity	0.527620	0.278340
2	citric acid	0.271175	0.334244
3	residual sugar	2.538806	6.392862
4	chlorides	0.087467	0.045783
5	free sulfur dioxide	15.874922	35.308085
6	total sulfur dioxide	46.467792	138.360657
7	density	0.996747	0.994027
8	pH	3.310748	3.188246
9	sulphates	0.657919	0.489851
10	alcohol	10.422983	10.514267
11	quality	5.636023	5.877909

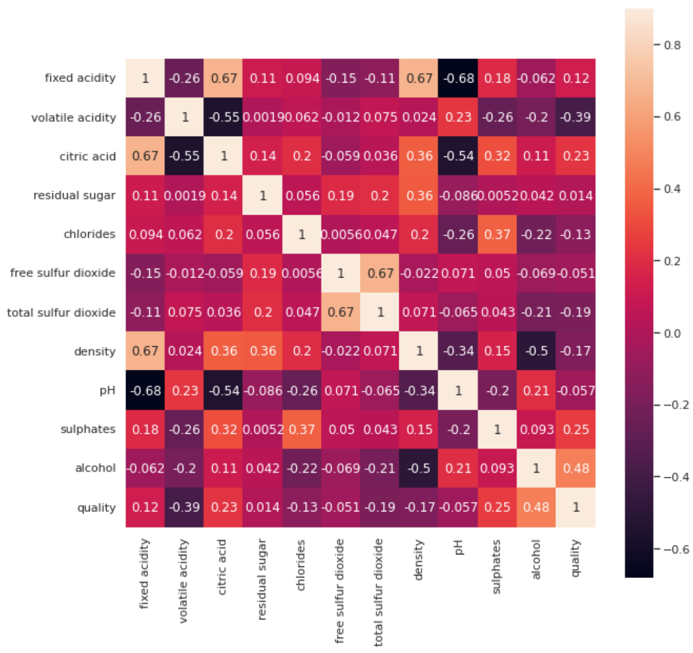
The fastest way to get a sense of which features are important to the wine quality is through a correlation matrix. We created two different correlation matrices, one for white wine and one for red wine.

For the red wine alcohol, density and volatile acidity seemed to have strong correlations with the wine quality whereas the white also has density as a strong indicator. The correlation matrix gives us a quick insight into the stronger features hence we can generate more useful graphs with these features.



	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	pH	sulphates	alcohol	quality
fixed acidity	1	-0.02	0.29	0.089	0.023	-0.05	0.091	0.27	-0.42	-0.017	-0.12	-0.11
volatile acidity	-0.02	1	-0.15	0.063	0.071	-0.098	0.089	0.026	-0.033	-0.035	0.068	-0.19
citric acid	0.29	-0.15	1	0.094	0.11	0.094	0.12	0.15	-0.16	0.063	-0.076	-0.0092
residual sugar	0.089	0.063	0.094	1	0.089	0.3	0.4	0.84	-0.19	-0.026	-0.45	-0.097
chlorides	0.023	0.071	0.11	0.089	1	0.1	0.2	0.26	-0.09	0.016	-0.36	-0.21
free sulfur dioxide	-0.05	-0.098	0.094	0.3	0.1	1	0.62	0.29	-0.00017	0.059	-0.25	0.0082
total sulfur dioxide	0.091	0.089	0.12	0.4	0.2	0.62	1	0.53	0.0028	0.13	-0.45	-0.17
density	0.27	0.026	0.15	0.84	0.26	0.29	0.53	1	-0.093	0.075	-0.78	-0.31
pH	-0.42	-0.033	-0.16	-0.19	-0.09	-0.00017	0.0028	-0.093	1	0.16	0.12	0.099
sulphates	-0.017	-0.035	0.063	-0.026	0.016	0.059	0.13	0.075	0.16	1	-0.017	0.054
alcohol	-0.12	0.068	-0.076	-0.45	-0.36	-0.25	-0.45	-0.78	0.12	-0.017	1	0.44
quality	-0.11	-0.19	-0.0092	-0.097	-0.21	0.0082	-0.17	-0.31	0.099	0.054	0.44	1

Fig: Red Wine



	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	pH	sulphates	alcohol	quality
fixed acidity	1	-0.26	0.67	0.11	0.094	-0.15	-0.11	0.67	-0.68	0.18	-0.062	0.12
volatile acidity	-0.26	1	-0.55	0.0019	0.062	-0.012	0.075	0.024	0.23	-0.26	-0.2	-0.39
citric acid	0.67	-0.55	1	0.14	0.2	-0.059	0.036	0.36	-0.54	0.32	0.11	0.23
residual sugar	0.11	0.0019	0.14	1	0.056	0.19	0.2	0.36	-0.086	0.0052	0.042	0.014
chlorides	0.094	0.062	0.2	0.056	1	0.0056	0.047	0.2	-0.26	0.37	-0.22	-0.13
free sulfur dioxide	-0.15	-0.012	-0.059	0.19	0.0056	1	0.67	-0.022	0.071	0.05	-0.069	-0.051
total sulfur dioxide	-0.11	0.075	0.036	0.2	0.047	0.67	1	0.071	-0.065	0.043	-0.21	-0.19
density	0.67	0.024	0.36	0.36	0.2	-0.022	0.071	1	-0.34	0.15	-0.5	-0.17
pH	-0.68	0.23	-0.54	-0.086	-0.26	0.071	-0.065	-0.34	1	-0.2	0.21	-0.057
sulphates	0.18	-0.26	0.32	0.0052	0.37	0.05	0.043	0.15	-0.2	1	0.093	0.25
alcohol	-0.062	-0.2	0.11	0.042	-0.22	-0.069	-0.21	-0.5	0.21	0.093	1	0.48
quality	0.12	-0.39	0.23	0.014	-0.13	-0.051	-0.19	-0.17	-0.057	0.25	0.48	1

Fig: White Wine

We shall categorize the wine quality into Low = (3, 4), Medium = (5,6) and High = (7,8,9) so that it enables us to see more clearly how different features vary across the different qualities.

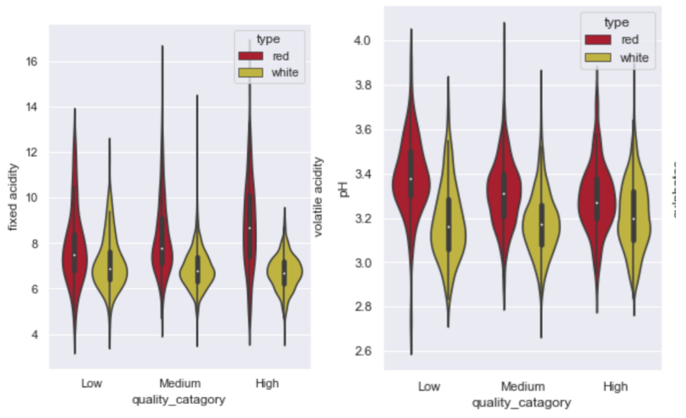
Fixed Acidity and pH

Fixed acidity comes from the main ingredient of wines themselves, the grapes. Cool climate grapes give high acidity and results in a more sour taste and warm climate grapes give less acidity.

For both types of wine and across the wine quality bins, the fixed acidity is spread out around a value of 7 to 8.

One would presume that these should play a bigger role in the wine quality, but it matters not what acidity it starts with but more on how the acidity is balanced later with other ingredients.

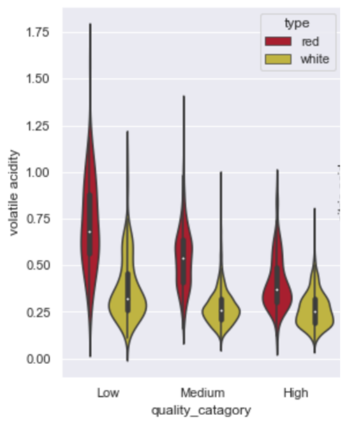
The pH also does not vary much across different wine quality. So this is because pH is more the defining acidity strength of a solution in terms of power of hydrogen. It does not tell us anything about what flavors of acidity and in what blend etc.

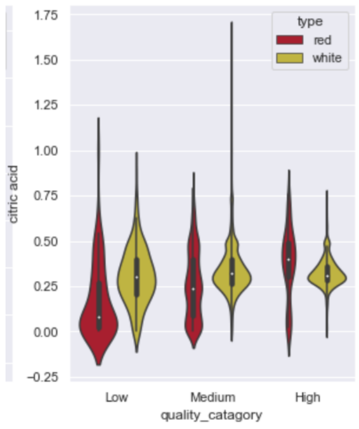


Volatile Acidity and Citric Acid

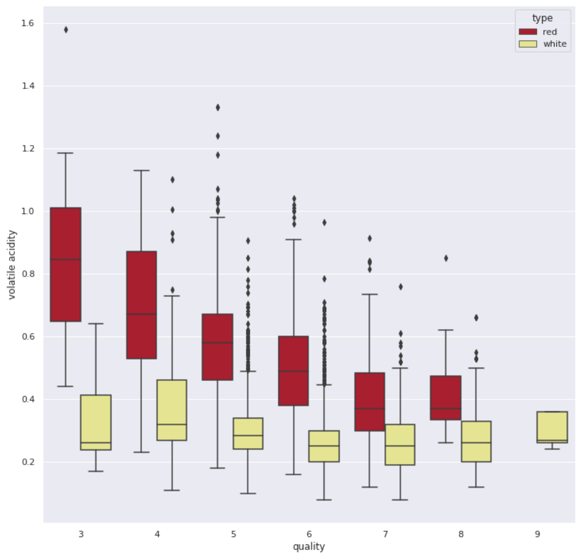
The volatile acidity is distilled out from the wine as too much of it can lead to undesirable taste hence we can see more of its presence in lower quality wine. For white wine there isn't much change in the quality.

Citric acid is added separately to give the wine more freshness, we can see the effect of higher citric acid on red wine leading to better quality. But in white wine we don't see much difference.



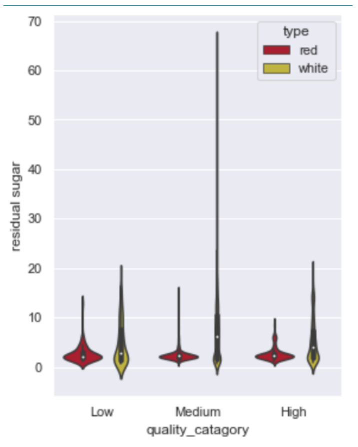


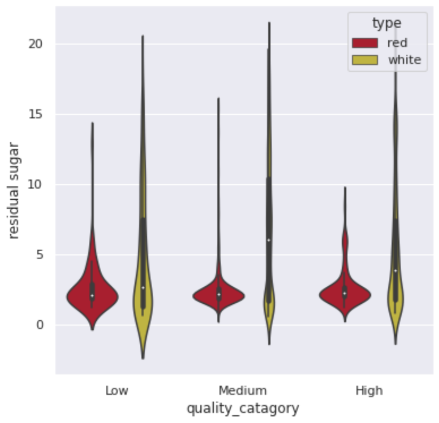
Volatile acidity is a very strong indicator for red wine, it is very evident once we check against the wine quality numbers. However the same cannot be said about white wine.



Residual Sugar

The residual sugar does not seem to have an obvious effect on the wine quality, what we do notice is that for medium quality white wine the sugar level is much broader. On the left we have the violin plot with outliers and on the right is without outliers.

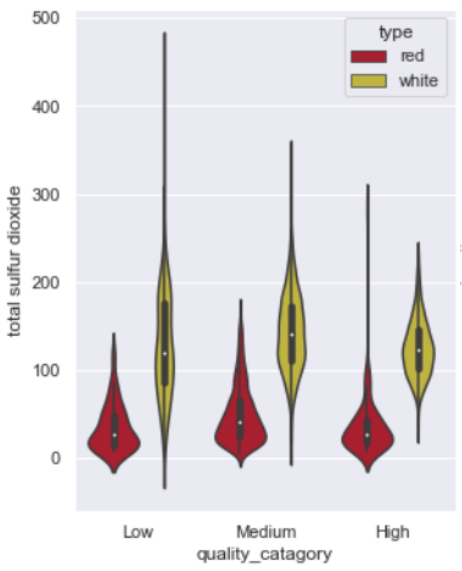




Total Sulfur Dioxide

As mentioned before, too much sulphur dioxide can kill the good yeast and give an undesirable odour, so basically the least amount of sulphur dioxide without making the wine pungent will give us a good wine.

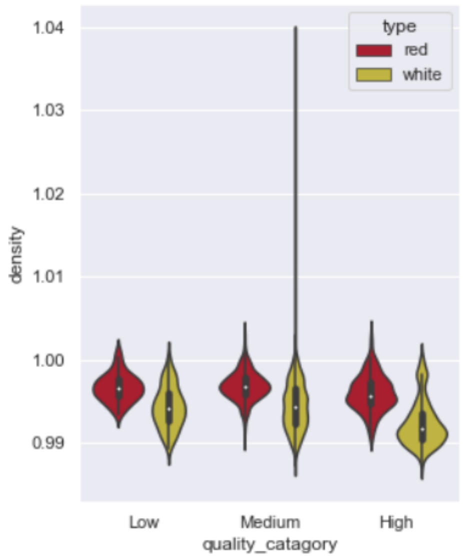
We can see from the graph that for high quality white wine the range of sulfur dioxide used is much narrower.

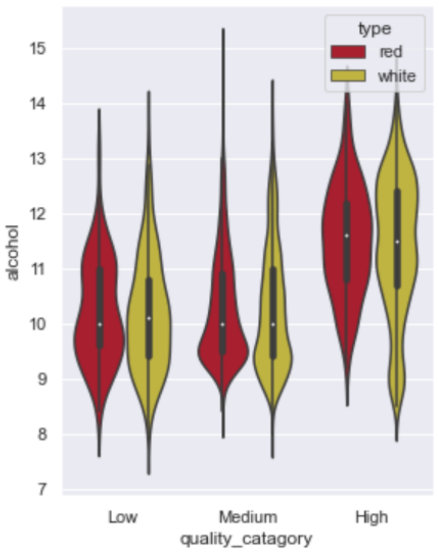


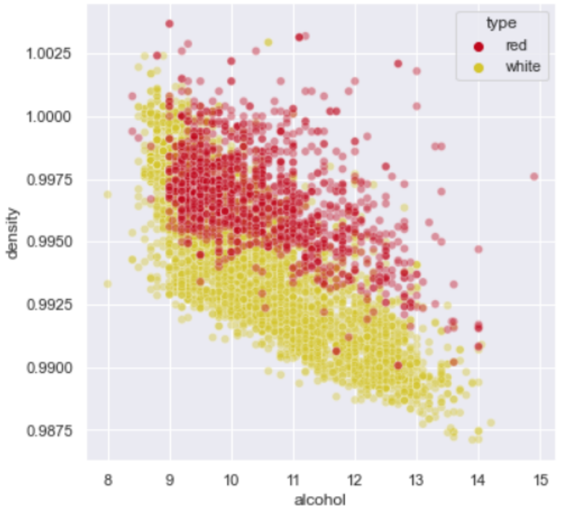
Density and Alcohol

The density of wine seems to be a relevant factor for only white wine. So the density of a wine keeps decreasing during the fermentation process which converts the wine into ethanol and carbon dioxide.

The alcohol is produced during the fermentation process and is definitely a vital feature for wine. For both red and white wine we see a stark difference in alcohol percentage for the different wine qualities.





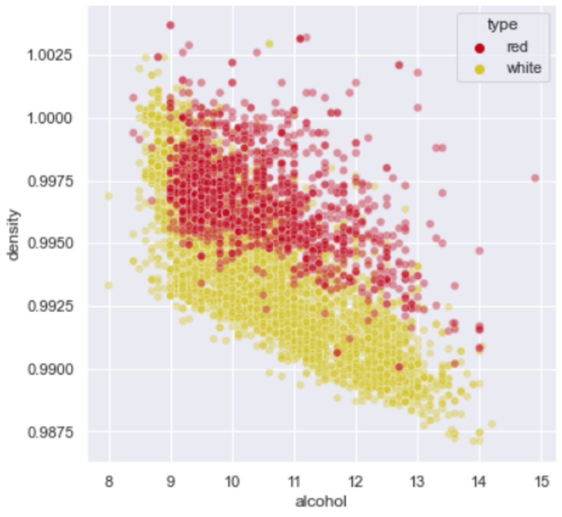


Now that we are aware of the important features for wine quality we can bring them together to get a more overall idea.

Here below we can see that the lower white wine quality tends more towards a higher volatile acidity and higher sulfur dioxide. The better wine quality mostly tends to have the higher wine quality.

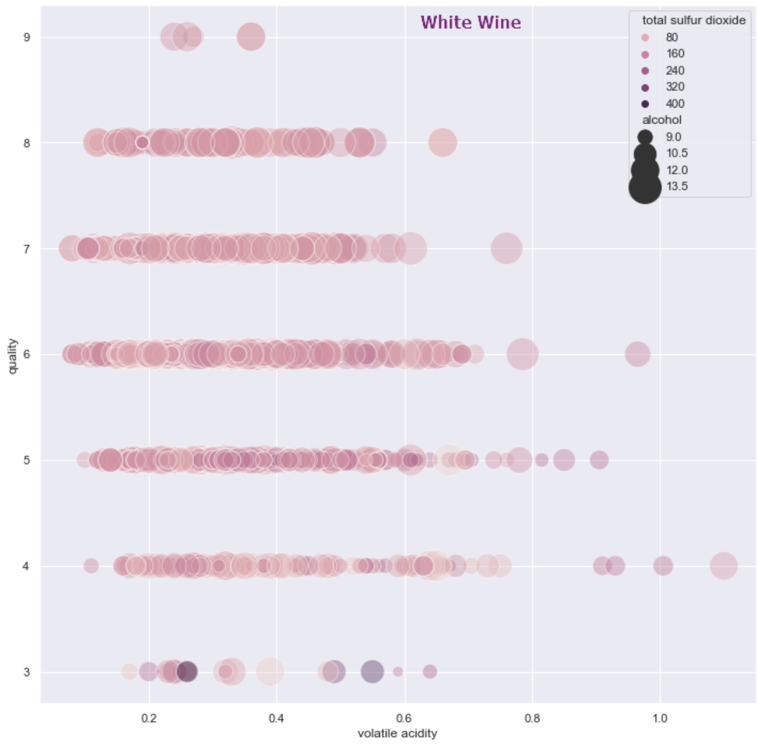
Density and Alcohol

During the fermentation process ethanol and CO2 is released and density is used to track this process. More fermentation leads to lower density and more alcohol which is supported by the scatterplot regardless of wine type.

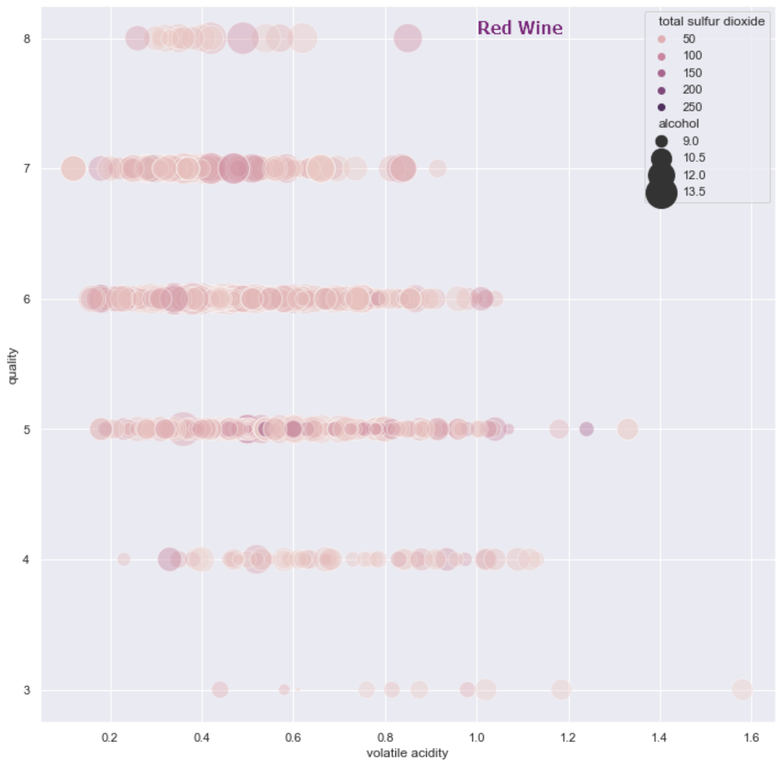


Important Features

Now that we have probed into the important features separately, let us see how they come together. Below you can see for higher quality wine the bubbles are light colored and big, meaning they have low sulfur dioxide and high alcohol, whereas the lower quality ones have darker and smaller bubbles indicating



For red wine the volatile acidity has a stronger impact on the quality. However higher sulfur dioxide does not necessarily mean lower quality wine although the correlation matrix shows otherwise. The percentage of alcohol is high for better wine across both red and white wine.



Model Design

After analyzing our dataset features we decided to focus on the wine Type and the wine Quality. We evaluated multiple models to predict both features.

We decided to explore different classifiers to be able to predict both the wine type and the quality. The data was split by keeping 30% of it for testing our models and 70% for training. Then, we first proceeded to scale numerical values by using the MinMaxScaler. Our dataset for the type analysis only contains the categorical variable *type* therefore we didn't need to implement an encoder on this analysis. On the contrary, for the Quality analysis we did apply One Hot Encoder for the categorical variable Type. Later, for doing the modeling we implemented a pipeline that would first scale the data and then fit the model on it. Each model parameter was tuned by using GridSearch for both analyses.

DecisionTreeClassifier

We first explored the *DecisionTreeClassifier* as we wanted to have a good understanding of the F1 Score results along with the balancing techniques without compromising much the time for computation, also our dataset is very small so we could expect good predictions from this model.

We decided to tune the following hyperparameters:

- *max_depth* parameter which helps us understand how deep the tree can be, as the deeper it is the more information is captured about the data.
- *min_samples_split*, gives us an understanding of the minimum number of samples required to split an internal node. The higher this value is the more constrained the tree becomes. We tuned from 1% to 25%.
- *criterion*: evaluates different options for the split of features. *Gini* impurity measures the frequency at which any element of the dataset will be mislabelled when it's randomly labeled, while *Entropy* indicates the disorder of the features with the target. We decided to explore both types for our criterion hyperparameter.

As mentioned before, during our analysis phase we found that there was a significant difference between the amount of White wine and Red wine samples, therefore we also dedicated some time to use balancing techniques with one of the classifiers to understand if a better distribution of the data would give us better predictions. The balancing techniques used were SMOTE, ADASYN and we also dedicated some time to understand if DecisionTreeClassifier along with PCA would give us good predictions.

As only measuring the Decision Tree Classifier was maybe too short an exploration, we also decided to test our data against other models: KNNeighborsClassifier, RandomForestClassifier, AdaBoostClassifier, and GradientClassifier. We tuned all of them and once we got the best hyperparameters for each one of them we decided to investigate if using the methodology of ensemble, it would help us get a better prediction. We used the best parameters of the classifiers listed above along with LogisticRegression as inputs for the StackingClassifier and then we calculated the y_predicted value for each of the models individually and then we also calculated the predicted for the Stacking Classifier.

KNNeighborsClassifier

This classifier was selected as we didn’t want to make assumptions about underlying data. For this model we decided to tune the following parameters:

- *metric*: the metric options we evaluated were 'minkowski', 'euclidean', and 'manhattan'. These options represent the distance while building the KNN model.
- *n_neighbors*: are the number of nearest neighbors, we used different ranges for this parameter.
- *weights*: *uniform* or *distance*. *Uniform* does not consider the distance between the new data point and its neighbors, they all have equal influence over the prediction. While *distance* assigns higher weights to closer training examples.

RandomForestClassifier

We decided to analyze our model with this classifier to get a deeper analysis after the decision tree modeling as it decorrelates the trees with the introduction of splitting them on random subset of features. For this model we decided to tune the following parameters:

- *max_leaf_nodes*: helps us restrict the growth of the tree as it sets the max splitting of the nodes. We evaluated from None to 50.
- *n_estimators*: simply represents the number of trees in the forest. We evaluated from 1 to 300 to start and then we tuned further.

AdaBoost Classifier

The advantage of this classifier is that it adapts based on the result of the weak classifiers, it gives more weight to the misclassified observations of the last weak learner therefore we considered that trying this classifier to our model would give us better insight. For this model we decided to tune the following parameters:

- *algorithm*: SAMME.R and SAMME (Stagewise Additive Modeling) learning algorithms. We tried both options during our GridSearch process.
- *learning_rate*: represents the contribution of each model to the weights. The lowest the rate the slower the model trains as the weights will be increased or decreased to a small degree. We tried values from 0.1 to 1
- *n_estimators*: number of models to iteratively train. We tried values from 1 to 300.

Gradient Boosting Classifier

We decided to try this classifier as it is known for being more accurate compared to other models as with each tree added, the model becomes more expressive. For this model we decided to tune the following parameters:

- *learning_rate*: we tried values between 0.1 to 0.4 as a low learning rate could improve the performance of the model.
- *max_depth*: different values help to control over fitting as higher depth allows the model to learn relations specific to a particular sample. We tuned values from 1 to 10.
- *max_features*: this parameter represents the number of features to consider while searching for a best split. We used values between 1 to 10 as per our number of features.

- *n_estimators*: number of models to iteratively train. We tried values from 1 to 300.

Support Vector Machine

SVM or SVC is known for its performance in imbalanced datasets, another good advantage is that it does not overfit. For this model we decided to tune the following parameters:

- *kernel*: values evaluated 'poly', 'rbf', 'sigmoid', 'linear'. This hyperparameter will help us understand the data distribution.

Logistic Regression

We decided to use logistic regression as the last step used for the final estimator of the stacking classifier. This classifier provides a simple learning algorithm and for low dimensional datasets it's less prone to overfitting. Before using it in the classifier, we decided to tune the hyperparameters so we could achieve a more precise result in the stacking classifier. For this model we decided to tune the following parameters:

- *solver*: values newton-cg, lbfgs, liblinear, sag, saga. Each algorithm solves different aspects of the processing such as memory, datasets size, dimension sizes.
- *penalty*: values l1, l2, elastic net, none. The goal of this hyperparameter is to reduce the generalization error and regulate overfitting. Some penalties will not support all solvers.

Stacking Classifier

The stacking classifier will take multiple models and return a better prediction as the intention is that all of them combined will give a more accurate result. We took the best hyperparameters after tuning each of the previously mentioned classifiers and we used them to fit the new classifier model, and then predict the value. We then proceeded to compare the F1-Score for each model independently and with the ensemble method.

Model Evaluation

There were different options when deciding how to evaluate the success of the models used to predict the wine Type and Quality. The *accuracy score* given allows us to look at correctly classified observations. Another way of measuring the performance of a model is through the F1-Score which balances the *precision* and *recall* values on the positive cases. F1 allows us to have better predictions when False Negatives and False positives are present.

As mentioned earlier, our dataset contained imbalanced classes, therefore we decided to first understand the impact of the imbalance classes in the prediction quality for both *Accuracy* and *F1-Score.*

The SKLearn library provides the *accuracy_score* function that returns the calculation for a given model. The library also contains a *classification_report* function that gives us a chart with the F1-score per class and the average F1 scores.

The average F1 scores split into **Micro** average and **Macro** average. The **Micro average** one computes the global average F1-Score by counting the sums of True Positives, False Negatives and False Positives (micro F1 = accuracy or micro-recall). The **Macro average** uses the arithmetic mean of all the per-class F1 Scores; it treats all classes equally regardless of their support values.

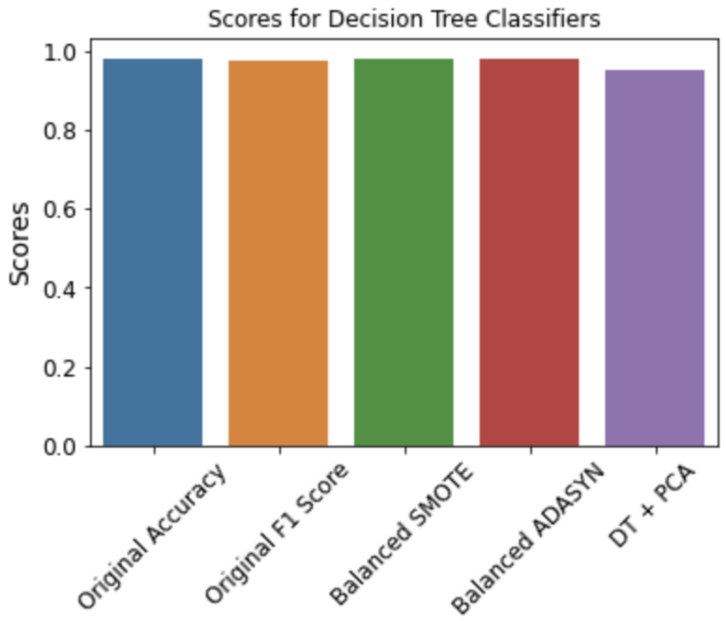
We needed to understand which of all these scores was the most adequate for our modeling. Given that our dataset is imbalanced, the F1 Macro score seems to be a good option as the performance of the model won't be affected even if our classes are skewed, meaning that all classes are equally important. To arrive at this conclusion we also decided to run predictions for the same model and compare their accuracy and F1-macro score. We only ran this comparison with the first analysis (Wine Type) for a single classifier: Decision Tree Classifier.

Wine Type

The results obtained for the wine Type were very good. Most of the models gave at least 0.98 F1-Score.

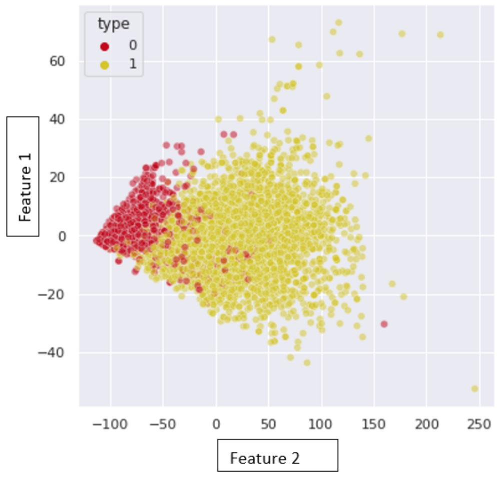
Decision Tree Classifier

For the decision tree classifier we explored both Accuracy and F1 Scores along with the balancing techniques (SMOTE, ADASYN). Below a chart with the different values:

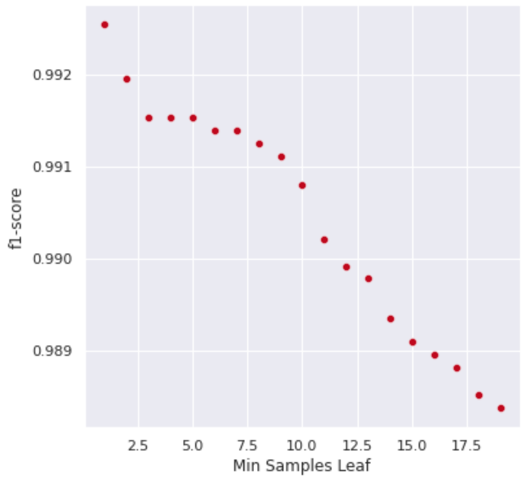


As we can see the scores are very similar for all cases, therefore balancing the data did not represent an important improvement for this classifier.

We attempted to bring all features down to just two features through PCA but we found that there are still many overlaps between red and white wine. This would make it difficult to differentiate between the classes as shown below.



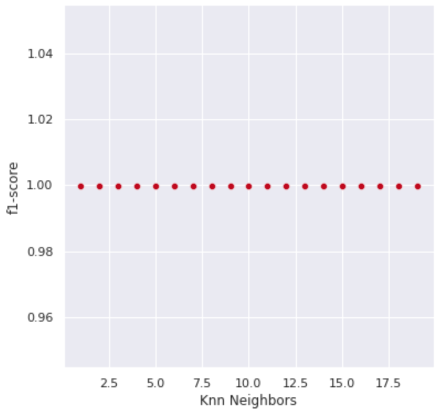
Once we got hold of the best parameters, we kept every parameter constant except minimum samples leaf so that we can see how the f1-score can be improved.



We found that for lower values of min samples leaf the f1 score improved.

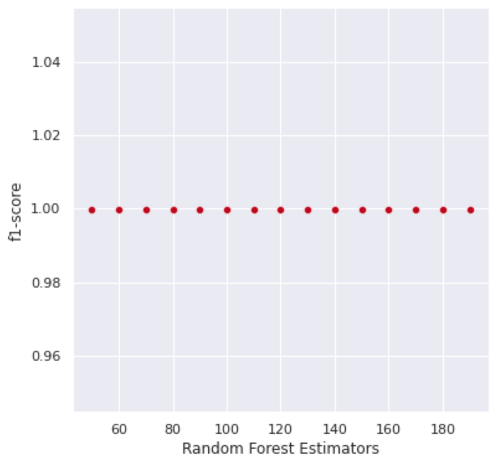
KNNeighbors Classifier

For further tuning we kept most hyperparameters constant except the n_neighbors. The f1 score stayed the same at 0.99971 regardless of the number of neighbors.



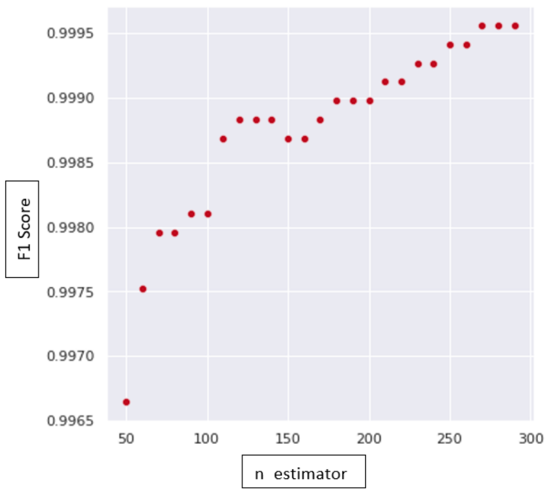
Random Forest Classifier

For further tuning we kept most hyperparameters constant except the n_neighbors. The f1 score stayed the same at 0.999 regardless of the number of estimators.



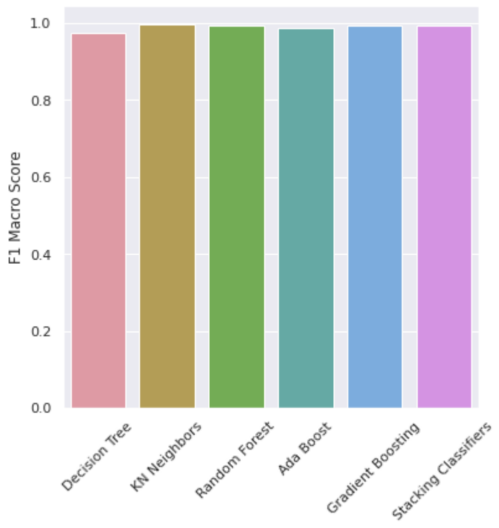
Ada Boost Classifier

For further tuning we adjusted the n_estimators and found higher values to favor higher f1-score.



Stacking Classifier

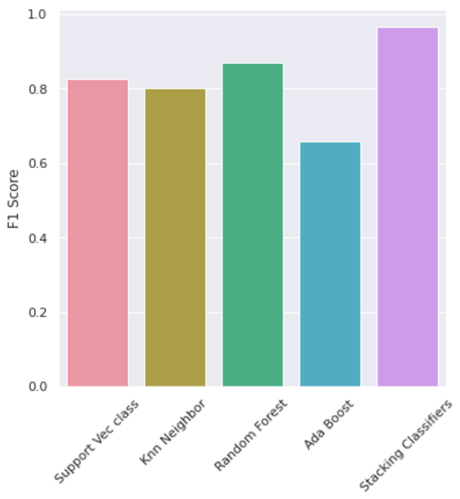
The stacking classifier returned similar results as the rest of the models.The scores returned by the classifiers are very similar. The score was very high for all iterations (0.978 - 0.995). The tuning of the hyperparameters allowed us to increase the F1 Score. The Stacking Classifier returned a similar result (0.994). On Appendix A, Table 2 we are able to find a more detailed description of the classifiers, tuned parameters and F1 Scores.



Wine Type F1 Score for all evaluated classifiers

Wine Quality

For the wine quality the models returned a different performance as for the wine type. Classifiers returned F1 Score values between 0.66 and 0.85, the Stacking Classifier returned a much better F1 Score (0.984) than any of the other classifiers individually. The classifier which performed the worst was the Ada Boost Classifier. On Appendix A, Table 3 we are able to find a more detailed description of the classifiers, tuned parameters and F1 Scores.

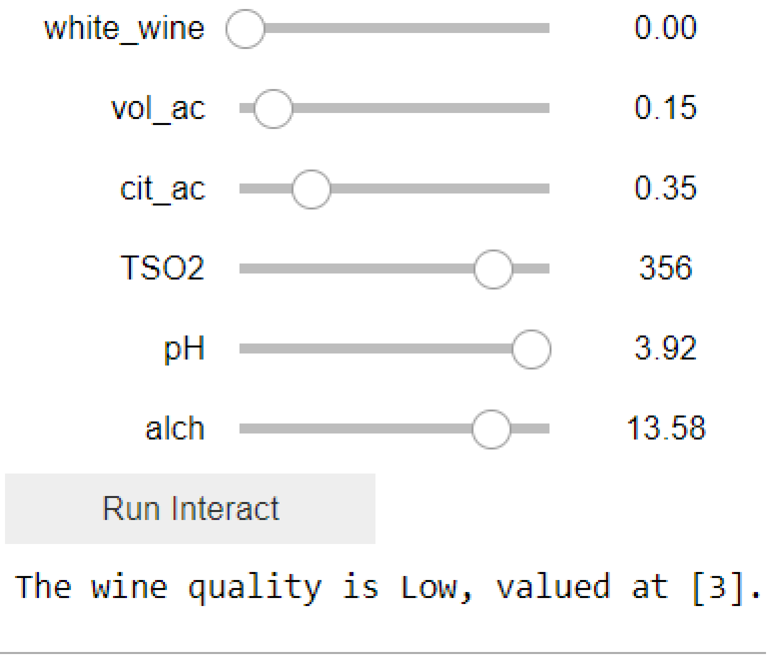


Wine Quality F1 Score for all evaluated classifiers

Wine brewer

There are so many features that affect the wine quality that it is hard to get a sense of what leads to a good wine. If we are able to constantly play with the wine ingredients and see the wine quality ourselves then we would learn better. This led to us making an interactive widget that allows anyone to tune the wine ingredients and understand better what differentiates a good wine from bad.

The wine brewer uses our best performing machine learning model to give an accurate prediction on the quality.



white_wine 0.00
vol_ac 0.15
cit_ac 0.35
TSO2 356
pH 3.92
alch 13.58

```
The wine quality is Low, valued at [3].
```

Conclusions

Using the wine dataset we have been able to predict both the wine type and wine quality using various machine learning models. We had to clean the data for any null values by imputing with the mean. The dataset had a low percentage of missing data hence the quality of the dataset was pretty good.

Although we had few missing data, as an improvement we could have grouped the data into red and white wine and imputed the missing data with the corresponding wine's average properties.

The dataset was mildly unbalanced with white wine consisting of 75% of the data hence any prediction for the wine quality is biased towards white wine. We did attempt to use an adaptive synthetic sampling approach to balance out the wine classes, but it did not lead to a significant improvement in classifier score.

Principal component analysis was also experimented with for the data but red and white wine had many overlaps hence it was difficult to differentiate using limited features. We investigated the importance of each feature for wine quality using correlation matrices and analyzed further using violin plots to get a sense of the spread of data for each wine quality catagory.

The dataset had features like sulfur dioxide which ranged from 6 to 400 and chlorides which ranged from 0.009 to 0.6, hence we scaled the data using MinMaxScaler.

For our modeling performance indicator we used F1 macro score as it does not get affected due to class imbalances. The F1 score came out to be in the high 0.9 for most models hence we chose these strong models for our ensemble and resulted in a stacked classifier which performed better than the individual classifiers.

Appendix A

Wine Type Decision Tree Classifiers Table

Classifier	**Hyperparameters**	**F1 Macro Score**
DecisionTreeClassifier	'criterion': 'gini', 'max_depth': 20, 'min_samples_split': 0.01	0.984 (Accuracy)
DecisionTreeClassifier	'criterion': 'gini', 'max_depth': 20, 'min_samples_split': 0.008	0.978 (F1 Macro Score)
DecisionTreeClassifier SMOTE=1	'criterion': 'entropy', 'max_depth': 20, 'min_samples_split': 0.008	0.984 (F1 Macro Score)
DecisionTreeClassifier ADASYN=0.4	'criterion': 'entropy', Max_depth': 10, 'min_samples_split': 0.008	0.984 (F1 Macro Score)
DecisionTreeClassifier PCA=2 features	'criterion': 'entropy', 'max_depth': 5, 'min_samples_split': 0.001	0.955 (F1 Macro Score)

Table 1. Wine Type Decision Tree F1 Scores

Wine Type Classifiers Table

Classifier	Hyperparameters	F1 Macro Score
Decision Tree Classifier	‘criterion': 'gini', 'max_depth': 20, 'min_samples_split': 0.008	0.978
KNNeighbors Classifier	'metric': 'manhattan', 'n_neighbors': 6, 'weights': 'distance'	0.994
Random Forest Classifier	'max_leaf_nodes': None, 'n_estimators': 75,	0.992
Ada Boost Classifier	‘algorithm': 'SAMME.R', 'learning_rate': 0.45, 'n_estimators': 120,	0.994
Gradient Boost Classifier	'learning_rate': 0.1, 'max_depth': 4, 'max_features': 4,	0.995

	'n_estimators': 500	
Logistic Regression	'penalty': 'none', 'solver': 'sag'	0.993
Stacking Classifier	Using all parameters above for models KNN, RFC, AdaBoost, GradientBoost and Logistic Regression	0.994

Table 2. Wine Type Classifiers F1 Scores

Wine Quality Classifiers Table

Classifier	Hyperparameters	F1 Macro Average Score
SVC	'svcclf__gamma': 'auto', 'svcclf__kernel': 'linear'	0.701
KNNeighbors classifier	'knnclf__metric': 'manhattan', 'knnclf__n_neighbors': 4, 'knnclf__weights': 'distance'	0.739
Random Forest Classifier	'rfclf__max_leaf_nodes': None, 'rfclf__n_estimators': 100	0.731
Ada Boost Classifier	'adaclf__algorithm': 'SAMME', 'adaclf__learning_rate': 1.2, 'adaclf__n_estimators': 300	0.561
Stacking Classifier	Using all parameters above for models SVC, KNN, Random Forest, AdaBoost	0.984

Table 3. Wine Quality Classifiers F1 Scores