COSC 5557: Practical Machine Learning

Algorithm Selection

Abiodun Awosola

2024-03-02

Introduction:

The aim of this study is to determine the most suitable machine learning algorithm for predicting wine quality based on a dataset of wine attributes. We will employ a variety of machine learning algorithms commonly used for regression tasks, including logistic regression, support vector machines (SVM), decision trees, random forests, and Naïve Bayes classifier. By comparing the performance of these algorithms on the Wine Quality dataset, we aim to identify the algorithm(s) that produce the most accurate predictions.

Experimental Setup:

For this experiment, we will utilize Python and R programming languages along with several libraries such as pandas, scikit-learn, and matplotlib for Python; and tidyverse, knitr, DataExplore for R. These libraries provide functions for data manipulation, machine learning algorithms, and visualization.

We will begin by loading the White Wine Quality dataset into pandas `DataFrame` and R `data.frame`.

Dataset Description:

The White Wine Quality dataset contains physicochemical properties of white variants of the Portuguese "Vinho Verde" wine. Each observation represents a wine sample, and the features include various chemical properties such as acidity, pH, alcohol content, etc. The target variable is the quality rating of the white wine, which is a score between 0 and 10.

The dataset consists of approximately 4900 observations. There are no missing values in the dataset.

Table 1: First 8 Rows of White Wine Data

	1	2	3	4	5	6	7	8
fixed.acidity	7.000	6.300	8.1000	7.2000	7.2000	8.1000	6.2000	7.000

	1	2	3	4	5	6	7	8
volatile.acidity	0.270	0.300	0.2800	0.2300	0.2300	0.2800	0.3200	0.270
citric.acid	0.360	0.340	0.4000	0.3200	0.3200	0.4000	0.1600	0.360
residual.sugar	20.700	1.600	6.9000	8.5000	8.5000	6.9000	7.0000	20.700
chlorides	0.045	0.049	0.0500	0.0580	0.0580	0.0500	0.0450	0.045
free.sulfur.dioxide	45.000	14.000	30.0000	47.0000	47.0000	30.0000	30.0000	45.000
total.sulfur.dioxide	170.000	132.000	97.0000	186.0000	186.0000	97.0000	136.0000	170.000
density	1.001	0.994	0.9951	0.9956	0.9956	0.9951	0.9949	1.001
pH	3.000	3.300	3.2600	3.1900	3.1900	3.2600	3.1800	3.000
sulphates	0.450	0.490	0.4400	0.4000	0.4000	0.4400	0.4700	0.450
alcohol	8.800	9.500	10.1000	9.9000	9.9000	10.1000	9.6000	8.800
quality	6.000	6.000	6.0000	6.0000	6.0000	6.0000	6.0000	6.000

Table 2: Summary of White Wine Data

	V1	V2	V3	V4	V5	V6
fixed.acidity	Min. : 3.800	1st Qu.: 6.300	Median : 6.800	Mean : 6.855	3rd Qu.: 7.300	Max. :14.200
volatile.acidity	Min. :0.0800	1st Qu.:0.2100	Median :0.2600	Mean :0.2782	3rd Qu.:0.3200	Max. :1.1000
citric.acid	Min. :0.0000	1st Qu.:0.2700	Median :0.3200	Mean :0.3342	3rd Qu.:0.3900	Max. :1.6600
residual.sugar	Min. : 0.600	1st Qu.: 1.700	Median : 5.200	Mean : 6.391	3rd Qu.: 9.900	Max. :65.800
chlorides	Min. :0.00900	1st Qu.:0.03600	Median :0.04300	Mean :0.04577	3rd Qu.:0.05000	Max. :0.34600
free.sulfur.dioxide	Min. : 2.00	1st Qu.: 23.00	Median : 34.00	Mean : 35.31	3rd Qu.: 46.00	Max. :289.00
total.sulfur.dioxide	Min. : 9.0	1st Qu.:108.0	Median :134.0	Mean :138.4	3rd Qu.:167.0	Max. :440.0
density	Min. :0.9871	1st Qu.:0.9917	Median :0.9937	Mean :0.9940	3rd Qu.:0.9961	Max. :1.0390
pH	Min. :2.720	1st Qu.:3.090	Median :3.180	Mean :3.188	3rd Qu.:3.280	Max. :3.820
sulphates	Min. :0.2200	1st Qu.:0.4100	Median :0.4700	Mean :0.4898	3rd Qu.:0.5500	Max. :1.0800
alcohol	Min. : 8.00	1st Qu.: 9.50	Median :10.40	Mean :10.51	3rd Qu.:11.40	Max. :14.20
quality	Min. :3.000	1st Qu.:5.000	Median :6.000	Mean :5.878	3rd Qu.:6.000	Max. :9.000

Table 3: Data summary

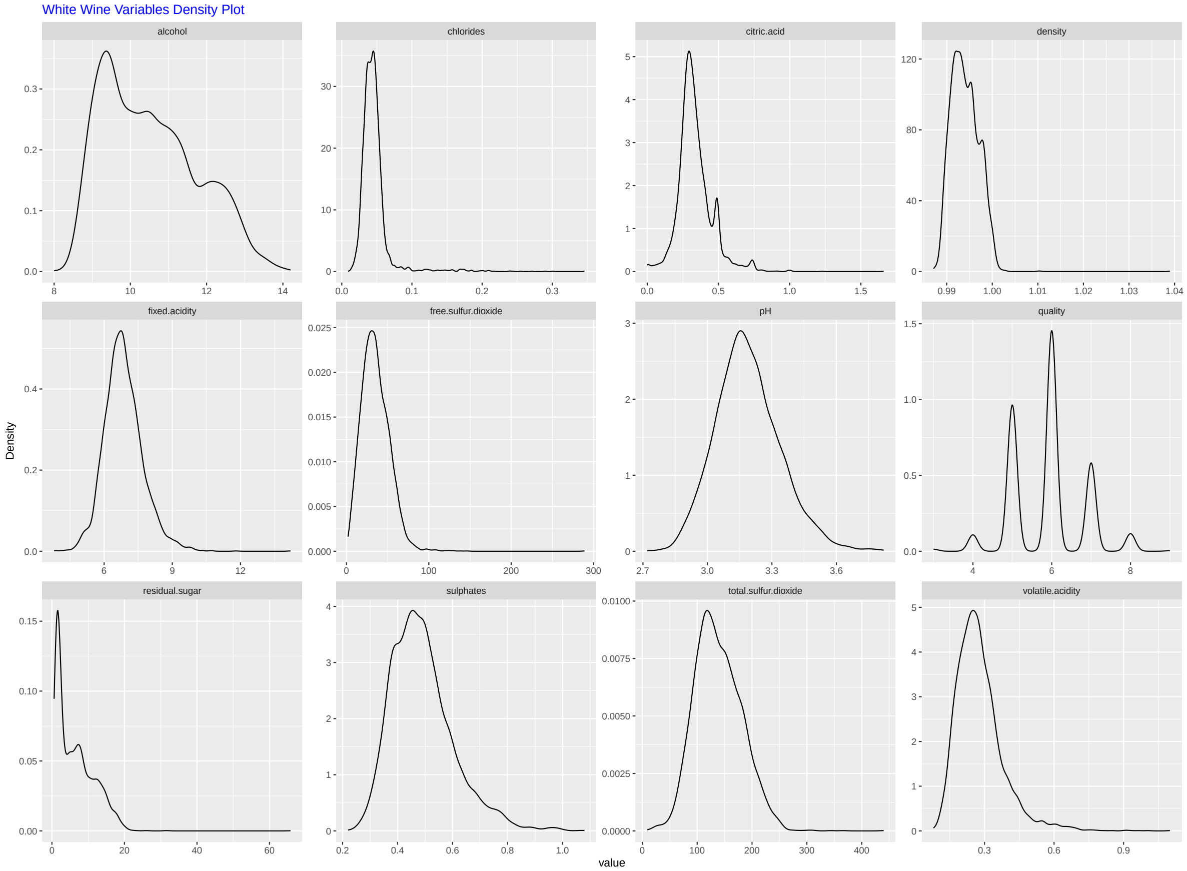
Name	wine_data
Number of rows	4898
Number of columns	12
Column type frequency:	
numeric	12
Group variables	None

Variable type: numeric

skim_variable	n_missing
fixed.acidity	0
volatile.acidity	0
citric.acid	0
residual.sugar	0
chlorides	0
free.sulfur.dioxide	0
total.sulfur.dioxide	0
density	0
pH	0
sulphates	0
alcohol	0
quality	0

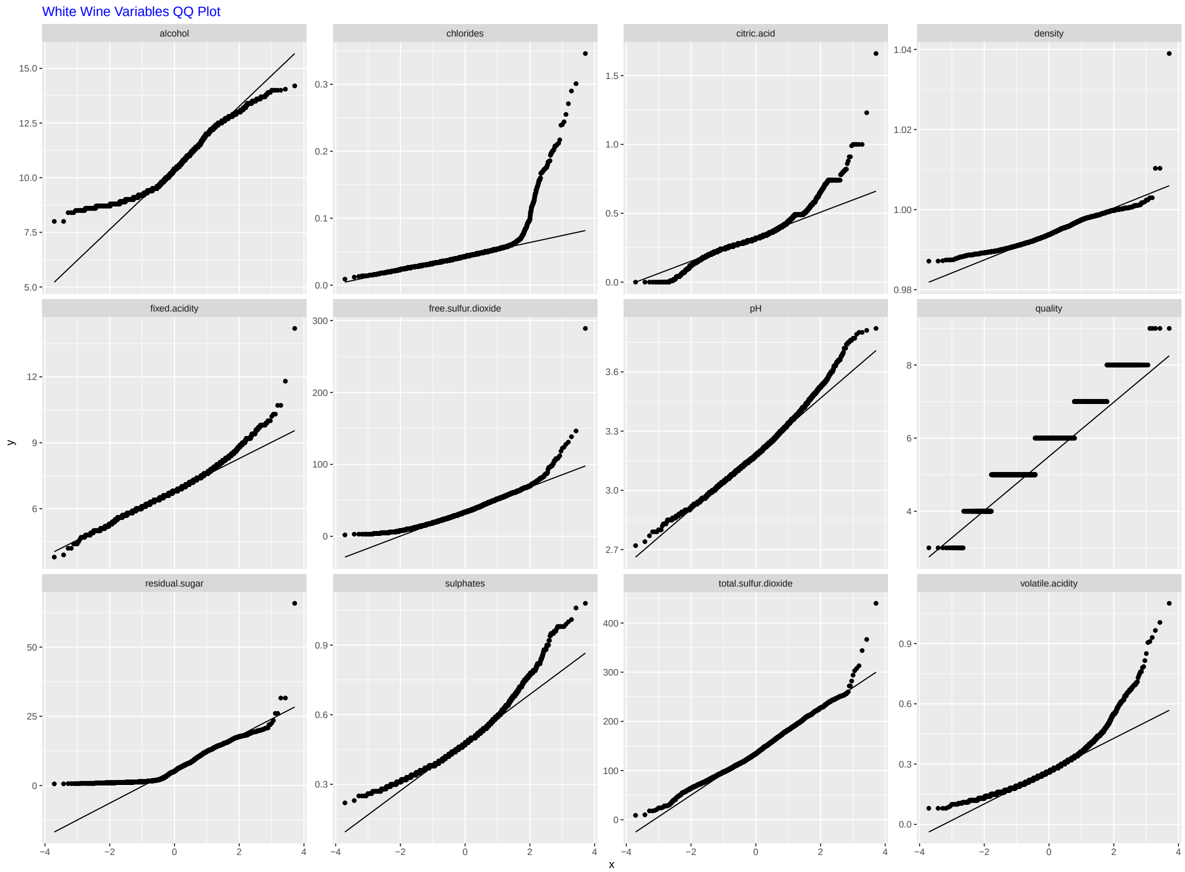
Density Plots

The density plots show white wine variables distribution.



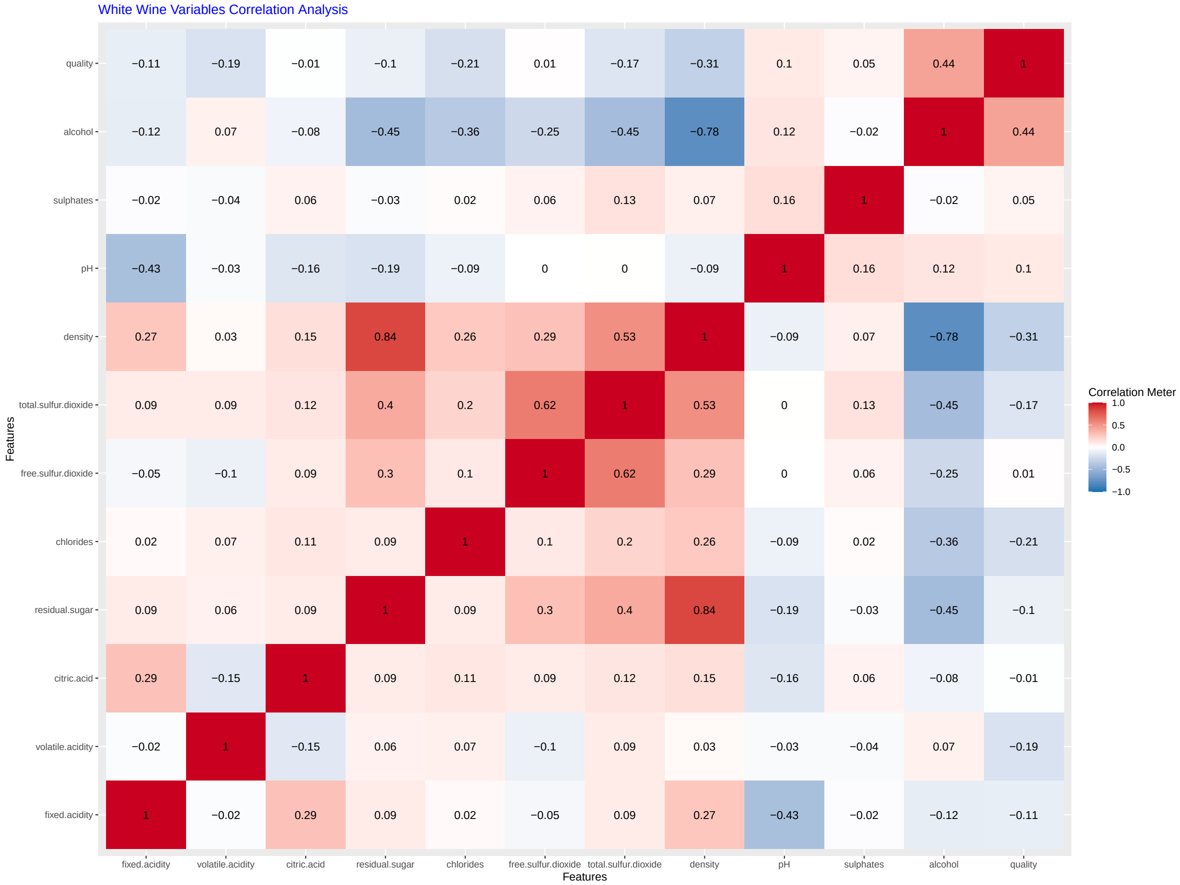
QQ Plots

QQ plots show tendency of variables to be normally distributed. Most points of a variable need to be on the 45-degree straight line for the variable to be normally distributed.



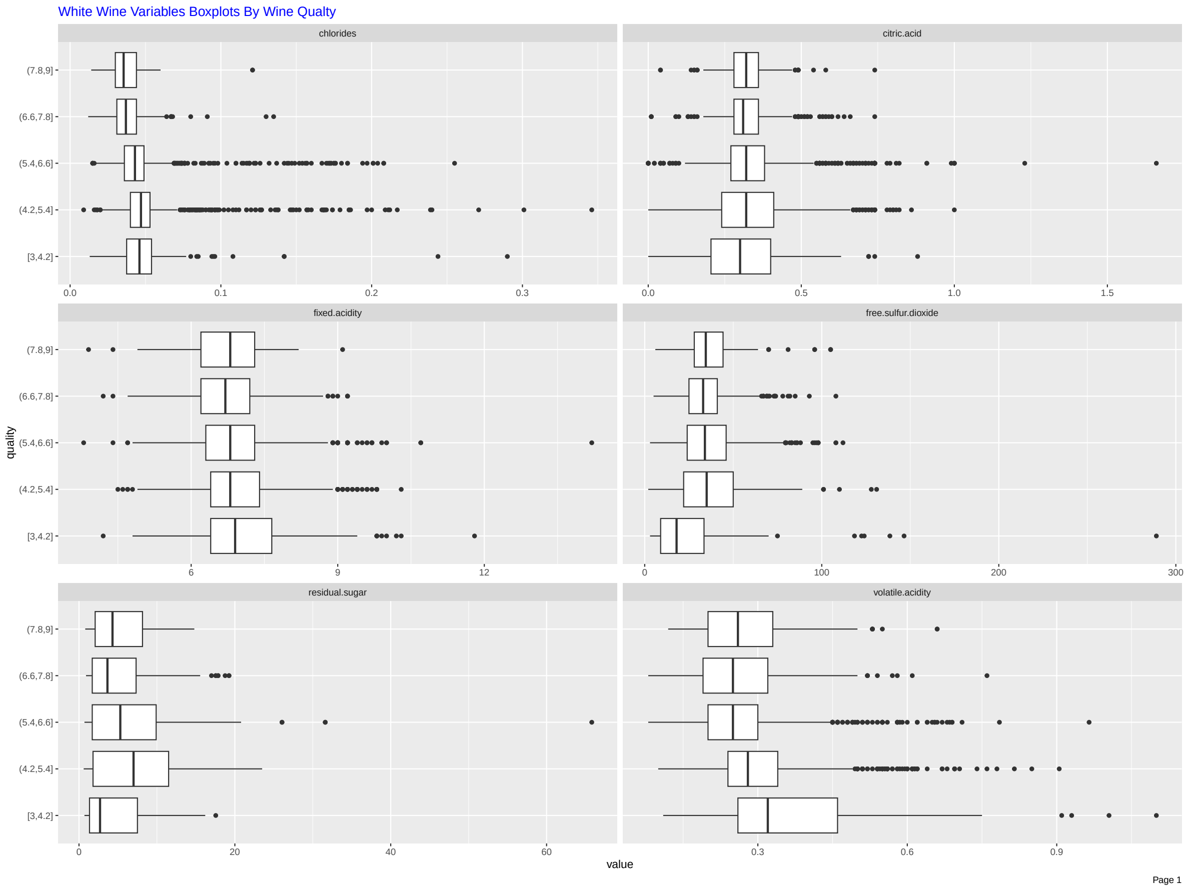
Correlation Plot

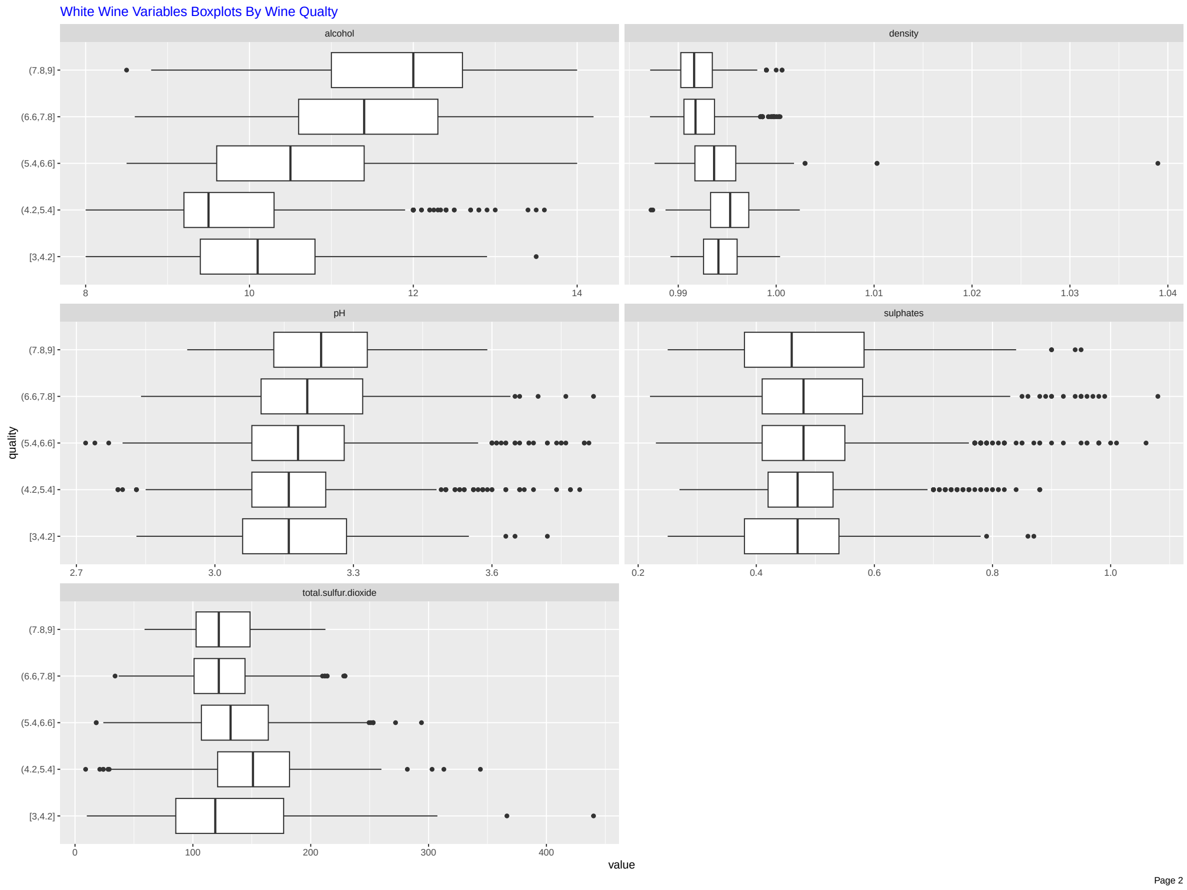
The correlation plot shows the linear relation between the variables.



Boxplots

The boxplots show white wine variables distribution by wine `quality`.


White Wine Variables Boxplots By Wine Qualty
chlorides
citric.acid
fixed.acidity
free.sulfur.dioxide
residual.sugar
volatile.acidity
quality
(7.8,9]
(6.6,7.8]
(5.4,6.6]
(4.2,5.4]
[3,4.2]
0.0
0.1
0.2
0.3
0.0
0.5
1.0
1.5
6
9
12
0
100
200
300
0
20
40
60
0.3
0.6
0.9
value
Page 1



The dots after the whiskers, (the horizontal end lines) represent outliers.

Checking for null values

```
fixed acidity           0
volatile acidity        0
citric acid             0
residual sugar          0
chlorides               0
free sulfur dioxide     0
total sulfur dioxide    0
density                 0
pH                      0
sulphates               0
alcohol                 0
quality                 0
dtype: int64
```

Data types in Python.

```
fixed acidity           float64
volatile acidity        float64
citric acid             float64
residual sugar          float64
chlorides               float64
free sulfur dioxide     float64
total sulfur dioxide    float64
density                 float64
pH                      float64
sulphates               float64
alcohol                 float64
quality                   int64
dtype: object
```

Training and Test Datasets

Next, we will split the dataset into training and testing sets using an 80-20 split, respectively. Each machine learning algorithm is trained on the training set and the performance is evaluated on the test dataset.

```
   fixed acidity  volatile acidity  citric acid  ...  sulphates  alcohol  quality
0             31                36           36  ...         22        5        6
1             23                42           34  ...         26       12        6
2             43                38           40  ...         21       24        6
3             34                28           32  ...         17       21        6
4             34                28           32  ...         17       21        6

[5 rows x 12 columns]

X_train shape: (3918, 11)

X_test shape: (980, 11)

y_train shape: (3918, 1)

y_test shape: (980, 1)
```

Data Preprocessing

Here is effort to preprocess the data by standardizing the features to have zero mean and unit variance. This preprocessing step is essential for algorithms sensitive to feature scaling, such as SVM.

We will build all six classification models and compare their accuracy scores. The machine learning algorithms considered for this experiment are as follows:

1. Logistic Regression
2. KNN Classifier
3. Support Vector Machine, (SVR)
4. Decision Tree Regression
5. Random Forest Regression
6. Naïve Bayes Classifier

For each algorithm, we will use default `hyperparameters` provided by `scikit-learn`. However, if necessary, we may perform `hyperparameter` tuning using techniques such as grid search or random search to further optimize the model performance.

Cross–validation

The purpose of cross–validation is to test the ability of a machine learning model to predict new data. It is also used to flag problems like overfitting or selection bias and gives insights on how the model will generalize to an independent dataset.

To test the ability of a machine learning model to predict new data, we will employ k-fold `cross-validation` with k=5. This technique partitions the dataset into k equal-sized folds, where each fold serves as the testing set once while the remaining k-1 folds are used for training.

1. Logistic Regression

Logistic regression is a supervised learning algorithm used for binary classification tasks. It models the probability of the input belonging to a particular class using the logistic function, which maps input features to probabilities between 0 and 1. It assumes a linear relationship between the independent variables and the logarithm of the odds of the dependent variable. Logistic regression is widely used due to its simplicity, interpretability, and efficiency in processing large datasets.

```
LogisticRegression(max_iter=1000, random_state=42)


Train Score: 0.5428757135037923


Test Score: 0.5183673469387755
```

2. K-Nearest Neighbors (KNN) Classifier

KNN is a simple and intuitive algorithm used for both classification and regression tasks. It classifies a new data point by assigning it the majority class label among its k nearest neighbors in the feature space, where k is a `hyperparameter` chosen by the user. KNN operates on the principle that similar data points tend to belong to the same class. It is a non-parametric algorithm, which means it does not make explicit assumptions about the underlying data distribution.

```
KNeighborsClassifier()
```

```
Train Score: 0.5523197018270911
```

```
Test Score: 0.5479591836734694
```

3. Support Vector Classifier

SVM is a powerful supervised learning algorithm used for classification and regression tasks. In the case of regression, it is referred to as Support Vector Regression (SVR). SVR aims to find the `hyperplane` that best fits the data while maximizing the margin between different classes or, in the case of regression, minimizing the error between the predicted and actual values. It works by transforming the input data into a higher-dimensional space using a kernel function, allowing it to find a linear separation or regression boundary.

```
SVC(random_state=42)
```

```
Train Score: 0.5707013840018766
```

```
Test Score: 0.5591836734693878
```

4. Decision Tree Classifier

Decision tree regression is a non-parametric supervised learning algorithm used for regression tasks. It works by recursively partitioning the feature space into regions, with each region associated with a specific prediction value. The decision tree learns from the data by selecting the feature that best splits the dataset at each node based on certain criteria (e.g., minimizing variance or mean squared error). Decision trees are easy to interpret and visualize, making them useful for understanding the decision-making process of the model.

```
DecisionTreeClassifier(random_state=42)
```

```
Train Score: 0.5806506893945318
```

```
Test Score: 0.5510204081632653
```

5. Random Forest Classifier

Random forest is an ensemble learning algorithm that builds multiple decision trees and combines their predictions to improve accuracy and reduce overfitting. In the case of regression, it aggregates the predictions of individual decision trees to produce a final prediction. Random forest introduces randomness both in the selection of features at each node and the bootstrap sampling of the training data. This randomness helps to decorrelate the individual trees, leading to more robust and generalizable predictions.

```
RandomForestClassifier(random_state=42)
```

```
Train Score: 0.6574741965751818
```

```
Test Score: 0.673469387755102
```

6. Naïve Bayes Classifier

Naïve Bayes is a probabilistic classifier based on Bayes' theorem with the assumption of independence between features. Despite its simplicity, it is effective in many real-world classification tasks, especially in text categorization and spam filtering. Naïve Bayes calculates the probability of each class given the input features and selects the class with the highest probability as the predicted class label. Despite its simplifying assumptions, Naïve Bayes can perform surprisingly well, particularly on datasets with high dimensionality and sparse features.

```
GaussianNB()
```

```
Train Score: 0.4517694242447937
```

```
Test Score: 0.45816326530612245
```

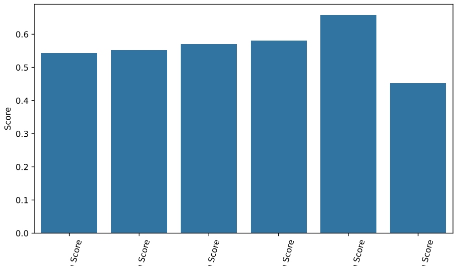
Select Algorithm Train and Test Accuracy Scores

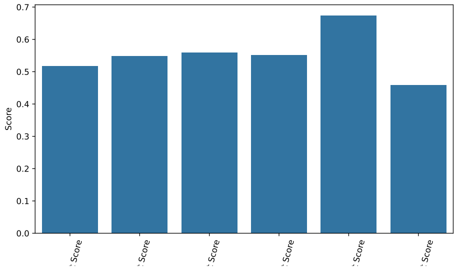
Tables and plots are used to visualize the performance of each algorithm and compare their effectiveness in predicting the white wine quality.

```
                                 Score
Logistic Train Score          0.542876
KNN Train Score               0.552320
SVM Train Score               0.570701
Decision Tree Train Score     0.580651
Random Forest Train Score     0.657474
Gaussian NB Train Score       0.451769
```

```
                                Score
Logistic Test Score          0.518367
KNN Test Score               0.547959
SVM Test Score               0.559184
Decision Tree Test Score     0.551020
Random Forest Test Score     0.673469
Gaussian NB Test Score       0.458163
```

Visualizing the scores





Conclusion:

Random Forest Regression algorithm is the best machine learning algorithm for predicting the white wine quality.

Code:

The code used for data preprocessing, model training, and evaluation will be provided in a separate Quarto Document file for reproducibility.