Wine Quality Prediction using Machine Learning

Submitted by

Name	: Mishkat Sultana
ID	: 19201028
Name	: Nusrat Jahan
ID	: 19201071
Name	: Tahsina Tajrim Oishi
ID	: 20101394
Name	: Md. Shahriar Rahman Anuvab
ID	: 21101094

Submitted to : Monirul Haque & Benjir Islam Alvee

Group : 11

Introduction:

A lot of people across the world keep the best quality wine for their special occasions. Starting from some certain nations to individuals, wine is considered as a symbol of celebration. But the first question that comes to mind is what is so special about wine. A quote says, "Drinking good wine with good food in good company is one of life's most civilized pleasures." And all these are dependent solely on the quality of the wine itself. Over the years it has become a talked topic and people are doing research to find the difference between the quality and how to identify a very fine wine. And to know what makes a good wine, some machine learning techniques have been used. Numerous studies have been carried out in an effort to identify the qualities of different categories of wine that can have the greatest impact in any occasion or celebration. As a lot of money is invested in this luxury product, with this study, people can get an approximate idea of what they are purchasing. By analyzing wine data that is used to categorize whether or a wine is good or bad.

Motivation:

Wine is thought to be a happy sign. It has grown in popularity over time, and a study is currently being conducted to ascertain the variations in wine quality and how to identify a truly exceptional wine. To ascertain the qualities that produce a good wine, some machine learning techniques have also been used. Numerous studies have been carried out in an effort to identify the traits of various wine categories that can have the biggest influence on any occasion or celebration. According to this study, consumers can basically understand what they are buying by looking at the wine data that is used to categorize wines into good or terrible categories because of the high cost of this premium item.

Objectives:

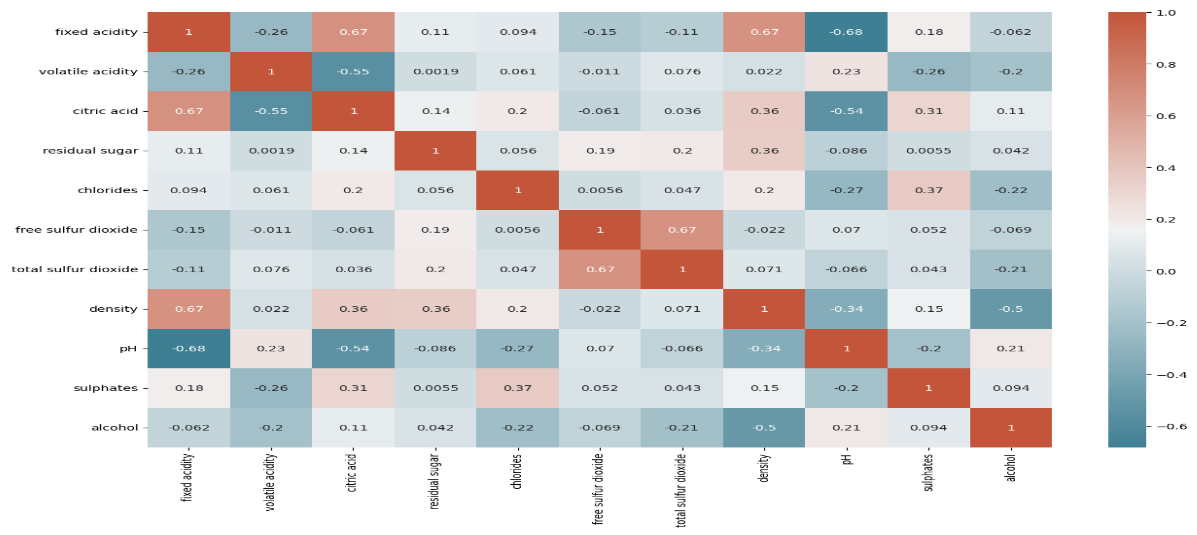
The following are the project's objectives:

1. To test various classification techniques to determine which produces the highest accuracy.
2. To identify the characteristics of a high-quality wine that are most indicative.

Methodology:

Dataset Description:

The dataset, which is from ongoing wine quality research among residents of the Massachusetts town of Framingham, is accessible to everyone on the Kaggle website. Different libraries have been used to execute different functions. The numerical calculation has been done by numpy and panda is being used in order to store the dataset and manipulate it. The seaborn function is used in plotting the values in the dataset. More than 1600 records and 12 attributes are included in the wine information it offers. The attributes include: fixed acidity, volatile acidity, citric acid, residual sugar, chlorides, free sulfur dioxide, total sulfur dioxide, density, pH, sulfates, alcohol and quality with which it is possible to predict the overall quality of wine. 1600 data points have been used here. There are 11 features in this dataset and 2 classes. The data collection is created as a data frame using the Pandas library in Python and is in the csv (Comma Separated Value) format.



Pre – processing technique applied:

Data preprocessing is done to change raw data into a more usable and suitable format for analysis. Libraries like pandas, numpy, seaborn, and train_test_split are used during this process. A popular preprocessing step is standard scaling, which entails centering the data by subtracting the mean and scaling by the standard deviation (y = (x-mean)/standard_deviation). A dummy variable, which represents category data numerically and can only take on the values of 0 or 1, is likewise utilized in this situation. Scaling aids in addressing the problems associated with variable scales, which can cause models to be unstable, perform poorly, and be sensitive to input values. Commonly, relevant fields that use the standard scaling method, also known as center scaling

Dataset splitting:

Using the train_test_split() tool from scikit-learn, a stratified split has been used with a test set size of 0.2 (or 20% of the data). The stratify argument makes sure that the split preserves the label variable's class distribution, which is needed for stratification. To guarantee that the same split can be replicated if necessary, the random_state argument is used.

After dividing the data, feature scaling is carried out on both the xTrain and xTest data using the MinMaxScaler() function, with the scaling parameters determined using only the xTrain data. Some machine learning algorithms are sensitive to the scale of the input features, so the MinMaxScaler() function scales each feature to a specified range (by default, between 0 and 1).

The per-feature minimum and maximum are then printed out by the code after scaling, which is helpful for verifying that the scaling was carried out properly.

Model Training:

Here, a dataset has been used to train and assess three distinct classifiers. First, the original data is divided into training and testing sets, and then a Random Forest classifier is trained on it using MinMaxScaler to scale the features. The model's test set score is then computed, and it is kept in the variable rf_score. The original data set is then separated into training and testing sets using train_test_split(), and a new data set is formed by removing the "quality" column. The same MinMaxScaler object that was fitted to the initial training set is then used to scale the new data. The new training set is used to train a new Random Forest classifier, and the variable new_rf_score is used to calculate and save the classifier's performance on the new test set. The results of the two Random Forest classifiers, one trained on the original data and the other on the new data set produced by removing the "quality" column, are then compared in a bar plot. The plot demonstrates that training the Random Forest classifier on the original data set as opposed to the new data set results in a higher score for the classifier.

Following that, the effectiveness of three different classification models was assessed: Logistic Regression, Decision Tree, and Random Forest. The code initially generates an instance of the matching classifier for each model before using the fit() method to fit the model to the training set of data. Using the predict() method, the models are applied to the test data after training.The scikit-learn library's classification_report() function is then used to create a report summarizing the performance of each model. For each class, this report offers measures like precision, recall, f1-score, and support. The evaluation results for each model, along with the random forest model's score, are then printed to the console.

Models Applied:

Random Forests Classification:

Multiple decision trees make up the machine learning model known as random forest. Random forest is intended to be less sensitive than decision trees, which are very sensitive to training data and have a propensity to overfit. It accomplishes this by bootstrapping, which involves randomly choosing rows from the dataset, to produce new datasets from the original data. The number of rows in each new dataset is the same as in the original data, but only a portion of the features are used for training. The majority vote of the forecasts from each individual tree is used to calculate the prediction for a new data point as it is passed through the random forest. Aggregation is the name given to this process of integrating the outcomes.

Due to the use of bootstrapping and random feature selection, random forest is regarded as "random". By guaranteeing that each tree in the model is trained on a slightly distinct set of data, bootstrapping helps to reduce sensitivity. By lowering the association between trees, random feature selection lowers sensitivity even further. Each tree would have comparable decision nodes and behave similarly if every feature were incorporated in every tree, which would increase the variance of the model. To avoid this, the average setting for the number of features in each tree is the square root or logarithm of the total number of features in the original dataset. Overall, random forest is a potent machine learning model that can increase prediction accuracy and decrease overfitting by mixing the outcomes of multiple decision trees.

Decision tree Classification :

The decision tree algorithm is a popular machine learning algorithm used for both classification and regression tasks. It is a supervised learning algorithm that uses a tree-like structure to represent all the possible outcomes of a decision based on the input features.The decision tree algorithm works by recursively splitting the data into smaller subgroups based on the input features until a stopping criterion is met. The splitting process involves choosing the best feature and threshold value that maximizes the information gain or minimizes the impurity of the data at each level of the tree.Once the tree is built, it can be used to predict the class label of a new instance by traversing the tree from the root to the leaf node based on the input features of the instance. For our project we used tree number n=50.

The decision tree algorithm is interpretable, as the decision tree structure can be visualized and understood by domain experts. It can be used for both binary and multi-class classification, where in binary classification, the decision tree splits the data into two subgroups based on a binary decision, while in multi-class classification, the decision tree creates a hierarchical structure that splits the data into multiple subgroups based on different conditions.

The decision tree algorithm has several advantages, such as its ability to handle both categorical and numerical data, and its ability to handle missing values. It can also be used for feature selection, as it automatically selects the most relevant features for the classification task.

Overall, the decision tree algorithm is a powerful and versatile machine learning algorithm that can be used for a variety of classification and regression tasks.

Logistic Regression :

Logistic regression is a popular machine learning algorithm used for binary classification tasks. It works by modeling the probability of a binary outcome (e.g., yes or no, true or false) as a function of the input features.

The logistic regression algorithm works by fitting a logistic curve to the data, which is a sigmoid function that maps any input value to a probability between 0 and 1. The logistic curve is determined by a set of coefficients that are learned from the training data.

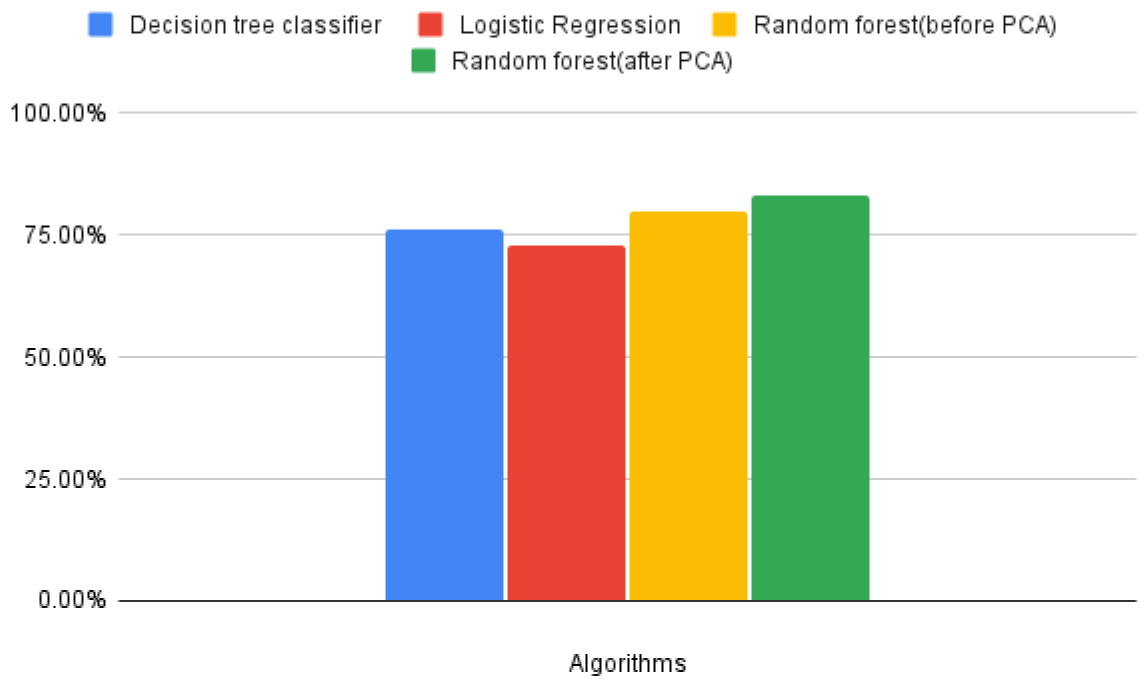
During training, the logistic regression algorithm adjusts the coefficients to minimize the difference between the predicted probabilities and the actual binary outcomes. This is typically done using a maximum likelihood estimation approach or gradient descent optimization.Once the model is trained, it can be used to predict the probability of a binary outcome for a new instance based on its input features. The predicted probability can then be converted to a binary decision based on a chosen threshold value.

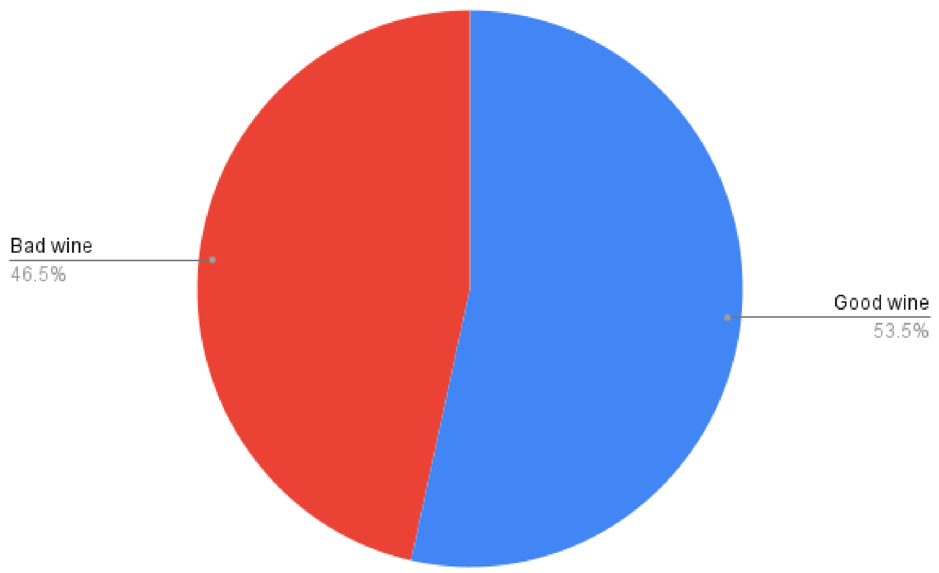
One of the advantages of logistic regression is its simplicity and interpretability, as the coefficients can be directly interpreted as the impact of each input feature on the predicted probability. It can also handle both categorical and numerical features and works well with small datasets.

Result:

After using a machine learning approach for both training and testing, we discover that the Random Forest Classifier has higher accuracy when compared to other algorithms. The accuracy_score function is used to calculate accuracy, and the comparison is displayed below.

Algorithm	Accuracy
Random Forest Classifier	80%(Before PCA) 83% (After PCA)
Decision Tree Classifier	76%
Logistic Regression Classifier	73%





After applying the models, the percentage of good wine is 46.5% and the percentage of bad wine is 53.5% where the number of the good wine is greater than the bad wine.

Conclusion:

This study makes use of the wine dataset to forecast wine quality based on its physicochemical characteristics. To balance the dataset during the data preprocessing stage and improve the model's performance, we first employed oversampling. Next, we search for traits that can produce more accurate prediction outcomes. In order to do this, we created correlation matrices and rated the attributes based on how highly correlated they were. The model's performance is enhanced after the sample datasets, which are balancing datasets, are applied. In general, the classification model performed better when unnecessary features from the datasets were removed. As a result, choosing the right features and balancing the data in the classification algorithms might enhance the model's performance.

Future Work:

Changes to the algorithm or the data may be made in the future to increase the classifier's precision. Utilizing the boosting algorithm on the more precise method is one advised strategy. Additionally, using different performance metrics and machine learning algorithms can help with better results comparison. This study can help the wine industry produce better wine products by predicting wine quality based on specific characteristics.

References:

For theoretical study-

https://towardsdatascience.com/predicting-wine-quality-with-several-classification-techniques-179038ea6434

https://www.diva-portal.org/smash/get/diva2:1574730/FULLTEXT01.pdf

https://www.analyticsvidhya.com/blog/2021/04/wine-quality-prediction-using-machine-learning/

For Dataset-

http://www3.dsi.uminho.pt/pcortez/wine/