A Highly Optimum, Efficient and Accurate Model to Predict Wine Quality using Multi-Class Classification

Eesha Tur Razia Babar
Department of Electrical Engineering and Computer Science
University of California, Irvine
ebabar@uci.edu

Michael Nguyen
Department of Computer Science
University of California, Irvine
michadn2@uci.edu

Dong Tran
Department of Computer Science
University of California, Irvine
donght@uci.edu

Abstract

The purpose of this report is to predict the quality of white wine using machine learning algorithms. In addition to this, we demonstrate our ability to explore and process the data, and to implement multiple machine learning models on given datasets to achieve optimum results. In this report, the method of multi-class classification is employed to distinguish classes from each other. A labeled dataset of 4898 instances was used to predict the quality of the wine based on provided 11 features. Multiple models were employed and tested on the labeled dataset from simple models, like K Nearest-Neighbors to more complex models such as Random Forest and Neural Networks. The performance of these models were compared based on their performance on a validation dataset. The best accuracy of 54% was obtained using the Random Forest model.

1 Introduction

The wine dataset which we used for this project consists of 4898 instances. The quality of wine is rated on a scale from zero to ten, considering zero as the worst and ten as the best. We utilized various libraries to help visualize and analyze the data and to explore the performance of multiple models on it. The pandas and NumPy library were used for statistical analysis of data, matplotlib, seaborn libraries were used to visualize features, and SDV (Synthetic Data Vault) to extend the provided dataset. In addition to this, the mltools, sklearn, and tensorflow neural network libraries were used to explore different models and test their performance on our dataset. In this report, we started with simple data exploration to understand the dataset in-depth, model exploration to train multiple models and to compare their performance, data processing and filtering to determine the impact of adding/dropping features and data points on performance. We also detailed performance validation to validate our models' performance and finally, adaption to under and over-fitting to obtain the optimal results.

2 Data Exploration

In this step, the dataset was analyzed and visualized in different ways in order to get more familiarity with it. This dataset has 11 features, 1 label, 4898 rows, and 12 columns.

Preprint. Under review.

2.1 Numerical Statistics

The data was initially treated as a data frame and its statistics for example mean, standard deviation, minimum and maximum values of features, and percentiles were calculated across columns. Similar statistics were calculated for labels and it was realized that there are 7 unique labels out of the 10. The minimum value of label (worst class) is 3 while the maximum is 9 (best class). Also, the 25th percentile of the label is class 5, while 50 and 75 percentiles belong to class label 6.

Table 1: Numerical statistics across columns

Feature	Mean	Standard deviation
Fixed acidity	6.864	0.84
Volatile acidity	0.27	0.10
Citric acid	0.33	0.12
Residual sugar	6.39	5.07
Chlorides	0.04	0.02
Free sulfur dioxide	35	17
Total sulfur dioxide	138	42
Density	0.99	0.002
pH	3.18	0.15
Sulphates	0.48	0.11
Alcohol	10.0	1.23

These statistics helped us to realize that Residual Sugar, Free Sulfur Dioxide and Total Sulfur Dioxide have the highest mean and standard deviation values.

2.2 Graphical Visualization

To better visualize the data, and understand feature distribution, multiple plots were used including histograms, scatter plots, and bar graphs.

Figure 1 shows the histogram of individual features while figure 2 shows the scatter plots of individual features.

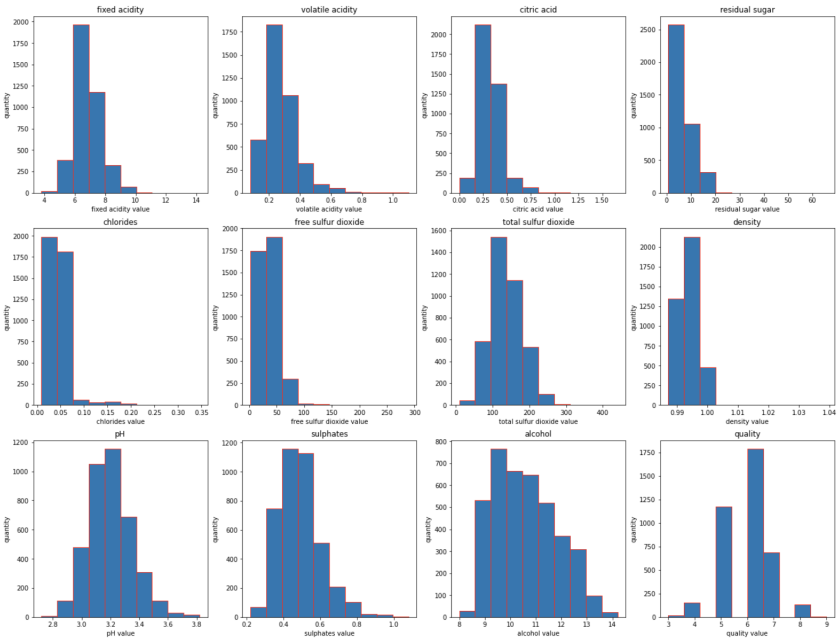

Figure 1: Histograms of individual features

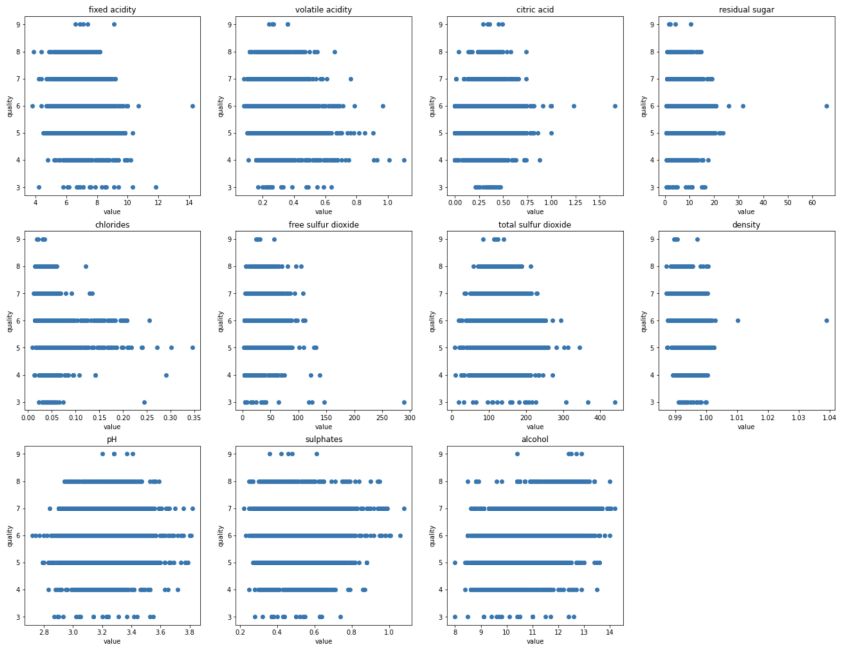

Figure 2: Scatter plots of individual features

3 Model Exploration

3.1 Phase 1

After data exploration and visualization, it was realized that the linear classification model will be very simple for our problem statement hence we started our work from the K Nearest-Neighbor algorithm classification model. In addition to this, we decided to use error as a performance metric for our dataset and to use ML tools for the initial phase of model exploration.

3.1.1 K Nearest-Neighbor

Initially, the data was divided into train and validation sets of 75% and 25%. Firstly, the K Nearest-Neighbor algorithm with different values of K and regularization was employed to see the impact on the result. Using KNN, after tuning multiple hyper-parameters, the minimum validation error was obtained with K = 50 and regularization = 1, as 0.42. 2-dimensional graphs were plotted to visualize the error as a function of K (number of nearest numbers) and alpha (regularization).

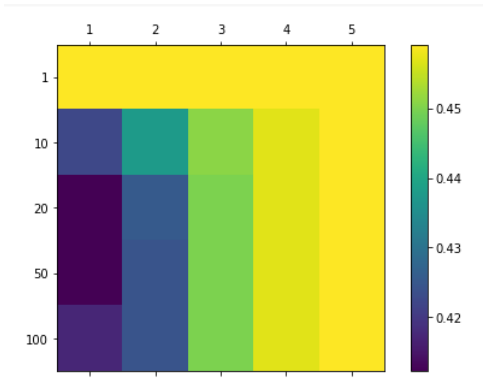

Figure 3: Validation error graph using K Nearest-Neighbor algorithm

3.1.2 Neural Networks

After the KNN and Decision Trees, the Neural Network model was employed. Multiple hyper-parameters were also tested for this model including several hidden layers, several nodes per layer, and activation functions including but not limited to htangent and logistics. The minimum error obtained using Neural Network was 0.44 through 2 hidden layers, 100 nodes per layer, and htangent activation function.

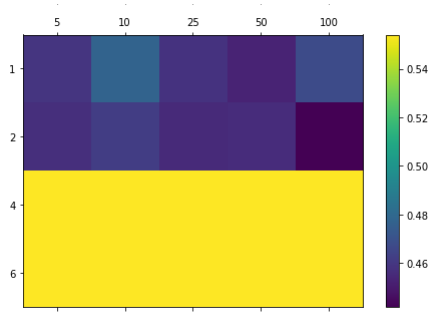

Figure 4: Validation error graph using Neural Networks with htangent activation function

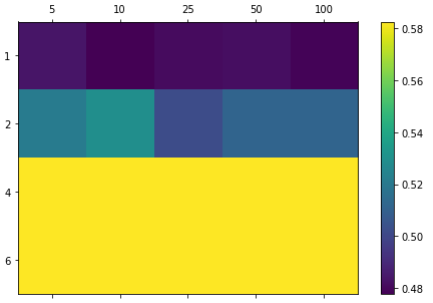

Figure 5: Validation error graph using Neural Networks with logistics activation function

3.1.3 Decision Tree and Random Forest

After the K Nearest-Neighbor model, the Decision Tree model was employed to the data and after tuning multiple hyper-parameters for example minimum parents, maximum depth, and minimum leaves, the minimum validation error of 0.47 was obtained after employing minimum parent as 45, maximum depth as 25 and nFeatures as 6. While after using a bag of 10 trees through a random forest, using the same hyper-parameters the minimum validation error obtained was 0.39.

Table 2: Model performance using error metric

Model	Minimum Error
K Nearest-Neighbor	0.42
Decision Trees	0.47
Neural Networks	0.44
Random Forest	0.39

Using the error as a performance metric it was realized that Random Forest has the best performance among all employed algorithms having a small error rate of 0.39. Also, this model is highly efficient since it is very fast compared to other models and used less system power.

3.2 2nd Phase

After initial model exploration through KNN, Decision Trees, Random Forest, and Neural Networks, we observed that accuracy metric is a more interpretable metric for performance than error metric. Since the (mean squared error) value can be greater than 1, therefore, it is not a feasible method to compare the performance of multiple models. Hence, during the second phase of our model exploration, we shifted towards an accuracy metric for performance comparison of multiple models from error metric. Also, we decided to explore other advanced libraries in addition to the ML tools.

3.2.1 Gaussian Mixture Model and Naive Bayes Classifier

Using GaussianNB from the sklearn library, we achieved an accuracy score of 0.48 on our training data and 0.45 on our validation data, respectively. With these scores, we determined the Naive Bayes model had very low accuracy and decided to shift our focus to other models.

3.2.2 Random Forest

With the Random Forest model from the sklearn library, we achieved an accuracy score of 0.54. For the hyper-parameters in this model, we mainly focused on n estimators (number of trees in the forest), max depth, and max features. Initially, we set n-estimators = 105, max-depth = 75, and max-features = sqrt(11). Given that the model trained extremely quickly, we increased n estimators from 1 to 105 which improved the accuracy score from 0.43 to 0.54. Any value above 105 resulted in the same accuracy score, but with a much slower computation time. The same reasoning was applied for the max depth.

Table 3: Model Performance using accuracy metric

Model	Highest Accuracy value
GaussianNB	0.45
Random Forest model	0.54

3.2.3 Binary Classification with Random Forests and Neural Networks

After testing our models on multi-classification problems we decided to check their performance on binary classification, which is simpler. For this, we set a threshold of seven, considering anything below seven as bad quality and seven or higher as good quality. Simplifying the class labels into binary, we achieved an accuracy score of 0.84. For hyper-parameters, we decided to choose the same values as we did with multi-class classification. With binary classification, we were able to achieve a much higher accuracy score with this model. Cross-validation was applied to verify this accuracy score. We applied splits of 5, 10, 15, and 20 to verify the accuracy, and after averaging the accuracy of all splits, obtained an accuracy of 0.82. With cross-validation, it is safe to assume that the original accuracy score was precise.

We also decided to test Neural Networks on binary classification. Instead of using the MLtools module to train the dataset, we used tensorflow, which significantly reduced training time by almost 96%. For this model, 3 layers with 100 nodes were used. Relu activation function was used instead of the htangent activation function. With a batch size of 32 and epoch of 100, we achieved an accuracy score of 0.80. This, also, was an improvement over multi-classification.

4 Data Processing and Filtering

We began data processing and filtering by removing 937 duplicates from the data. This resulted in a significantly reduced-sized dataset as only 3961 data points were remaining. There were not many outliers nor any amounts of NaN values in the dataset, so we did not remove any of them. However, there were two huge outliers in total and free sulfur dioxide features, with a standard deviation of 42 in total sulfur dioxide and 17 in free sulfur dioxide, and removing them did not make an improvement in accuracy. Using pandas series, we found that there were also no outlier features, as the most important feature had a score of 0.12, and the lowest score of 0.07, with alcohol being the most

important feature and fixed acidity being the lowest. We removed the lowest three important features and measured the accuracy of our models, but we realized this negatively impacted our performance. We then attempted using the SDV module to add more data in our model, which also did not improve the score, assuming there was not enough original data to create more quality data.

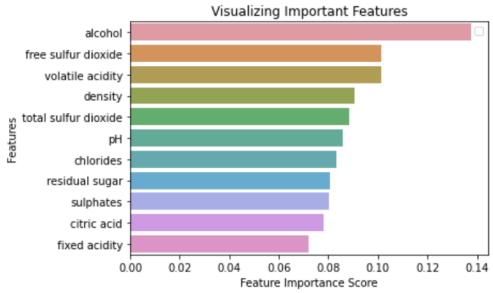

Figure 6: Histogram of Important Features

5 Performance Validation

Several methods were utilized to validate the performance of our models. Initially, we shuffled the data to ensure proper randomness, as well as reducing any biases, before splitting the data into training and testing. By splitting the data, we held out 20 percent of the data to verify that the predicted labels on our trained models matched those of test data. We measured the accuracy with the accuracy scoring function from sklearn, errors in our predictions, and used cross-validation to further validate our results. These methods were used to compare the predictions on our training data and testing data to adjust for any under-fitting or over-fitting. This allowed us to compare the performance of each model against another to determine which performed the best on the wine dataset.

6 Recommended Model and Final Accuracy

After comparing the performance of multiple models on our datasets in terms of accuracy and adapting to under-fitting/over-fitting, we found the best accuracy for multi-class classification using Random Forest method as 0.54. Since, there are 7 unique classes hence anything above 14% accuracy will be better than randomly choosing a class. Since our model's accuracy is 54%, it means that our model is good. Also, the Random Forest model is the fastest and uses fewer system resources as compared to other models because it is the best model we tried until now for multi-class classification. Hence, we will recommend this model for this dataset and specified problem.

7 Conclusion

During this project, we explored different methods of analyzing data, various machine learning algorithms and libraries, multiple performance metrics and methods of processing and filtering the data to get the best result using available resources (system memory, CPU, processing power, etc.) efficiently. We learned to handle a moderate-sized dataset, with multiple features and multiple class labels. It was a good practice which gave us hands-on experience.

8 Team Member Contributions

Eesha Tur Razia Babar was responsible for statistical data analysis, KNN, Decision Trees, Random Forest, and Neural Network model using ML tools. In addition to this, she was also responsible for writing her part of work in the report.

Michael Nguyen was responsible for statistical data analysis, Random Forest, Neural Network, conversion to binary classification, removing features, generating synthetic data, using advanced modules (sklearn, tensorflow, sdv), and his portion of the report.

Dong Tran was responsible for statistical data analysis, using modules sklearn and ML tools for Simple Decision Trees, semi-supervised models (Naive Bayes, Gaussian Mixture Model), Ada-Boosting and wrote his portion of the paper.