Wine Quality Prediction

Arnav Agrawal
IIIT Delhi
arnav22097@iiitd.ac.in

Aryan Singla
IIIT Delhi
aryan22112@iiitd.ac.in

Harsh Rajput
IIIT Delhi
harsh22201@iiitd.ac.in

Kshitij Gupta
IIIT Delhi
kshitij22257@iiitd.ac.in

Abstract

We build a learning system for multi-class classification problems that consists of standard tabular data preprocessing methods, various general training scripts for classical Machine Learning methods as well as Deep and Shallow Learning architectures along with options for regularization techniques and normalization. This framework allows visualisation and comparison of evaluation graphs for the purpose of hyper-parameter tuning using the WandB software. The goal is to create a simple and interactive interface for processing data, training models, evaluating, inferencing, and tuning hyperparameters. Code for this pipeline is available on https://github.com/Boltnav/Wine-Quality-Prediction with set-up instructions provided.

1. Introduction

Wine quality assessment is an important factor maintaining consistency and value in the wine industry. Accurate prediction of wine quality, based on physicochemical features, helps wine producers ensure that their product meets certain standards. Traditional methods of wine quality assessment are time-consuming and subjective. This project develops an *end-to-end automated machine learning pipeline* to predict wine quality based on features such as acidity, alcohol content, and residual sugar.

Wine quality is rated on a discrete scale from 3 to 8, making this an ordinal classification problem. The objective is to use various machine learning and deep learning techniques, specifically classification models, to predict these scores. Additionally, this project explores MLOps practices for managing the model lifecycle.

2. Related Works

Previous studies on wine quality prediction have applied a variety of machine learning models [1]. Methods like Ridge Regression, Support Vector Machine, Gradient Boosting Regressor, and multi-layer Artificial Neural Network, with non-linear models such as Random Forest and Gradient Boosting consistently outperforming linear models. These ensemble methods capture complex feature interactions and non-linearities, which are essential for improving prediction accuracy. The analysis showed that Gradient Boosting Regressor surpassed all other models' performance.

Also, [2] utilized Random Forests and Extreme Gradient Boosting. Using these two ML approaches, they learned the top three features from a total of eleven features which were chosen using Dimensionality Reduction methods. They displayed several visualizations that showed that only these three features were relevant to the target class, and that they could reduce the feature space and not lose out on many important feature directions of the vectors. They employed the XGBoost model and Random Forests to demonstrate this feature importance based selection.

3. Dataset details

3.1. Dataset Acquisition

The Wine Quality dataset was sourced from the Kaggle Machine Learning Repository. It contains physicochemical features of red and white wine samples, with quality scores ranging from 3 to 8. The dataset contains 12 attributes related to the wine's composition, including fixed acidity, volatile acidity, residual sugar, chlorides, and alcohol content.

3.2. Data Preprocessing

Initial exploration revealed no missing values in the dataset. The features were standardized using **Standard-**

Scaler to ensure equal contribution to the models. The dataset showed a class imbalance, where most wines were rated between 5 and 6. For Data Augmentation, the simplest approach involves duplicating examples in the minority class, although these examples don't add any new information to the model. Instead, new examples can be synthesized from the existing examples, applying **SMOTE** (Synthetic Minority Over-sampling Technique) to generate synthetic samples for the minority classes and balance the dataset.

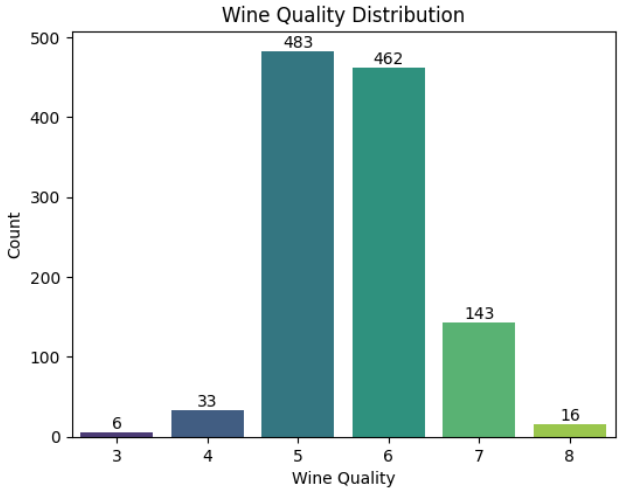

Figure 1. Class imbalance in the dataset.

4. Methodology and Model Details

4.1. Machine Learning algorithms

We implemented both regression and classification models to predict wine quality. However, wine quality prediction is inherently a classification problem, as the target variable is a discrete quality score (3 to 8). Below is a summary of the models used:

- **Linear Regression**: Predicts continuous approximations of wine quality scores. This method is suboptimal because it predicts continuous values that must be rounded, introducing errors.
- **Ridge and Lasso Regression**: Add regularization to avoid overfitting, but face the same limitations as Linear Regression for classification tasks.
- **Logistic Regression**: A linear classifier that models the probability of each class but struggles with non-linear relationships.
- **Random Forest**: An ensemble method that builds multiple decision trees to capture non-linear relationships in the data. Performs well due to its ability to model complex feature interactions.
- **Gradient Boosting**: Builds decision trees sequentially, correcting errors from the previous model. Provides high accuracy, comparable to Random Forest.

Model	Accuracy	F1 Score	MAE	R^2 Score
Linear Regression	-	-	0.4773	0.3171
Lasso Regression	-	-	0.608	0.0431
Ridge Regression	-	-	0.4721	0.329
Logistic Regression	0.4017	0.3998	-	-
Random Forest	0.7031	0.6893	-	-
Gradient Boosting	0.69	0.6841	-	-

Figure 2. Comparison of model performance.

4.1.1 Regression Models on Classification Problems

Regression models like **Linear Regression**, **Ridge Regression**, and **Lasso Regression** were applied, but their continuous predictions required rounding, leading to poor classification results.

Why Regression Fails:

- **Continuous Output**: Regression models predict continuous values that must be rounded, introducing errors.
- **Lack of Decision Boundaries**: Regression models do not define clear boundaries between classes, which is crucial for classification tasks.
- **Linear Assumptions**: These models assume linear relationships between features and target, which is not the case with wine quality data.

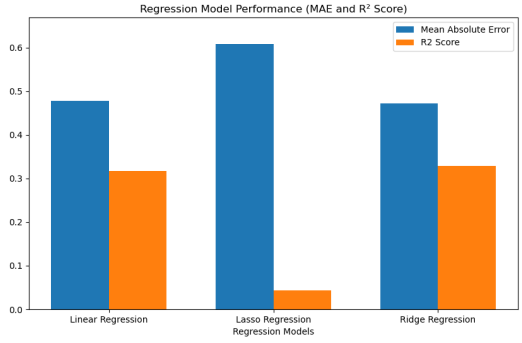

Figure 3. Regression models perform poorly on classification tasks.

4.1.2 Classification Model Results

Classification models performed significantly better, with **Random Forest** and **Gradient Boosting** outperforming logistic regression due to their ability to capture non-linear patterns.

Classification Model Performance:

- **Logistic Regression**: Accuracy = 40.17%, F1 Score = 0.3998.
- **Random Forest**: Accuracy = 70.31%, F1 Score = 0.6893.
- **Gradient Boosting**: Accuracy = 69.00%, F1 Score = 0.6841.

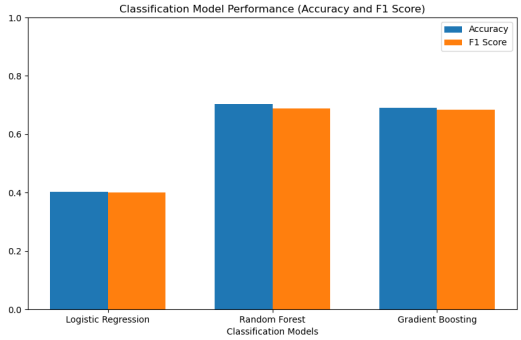

Figure 4. Performance of classification models.

We compare how quickly models converge by analyzing training loss over epochs:

4.2. Deep Learning Models

We started of by using the original dataset on a shallow Neural Network involving only a singular hidden layer of 64 neurons. Initially, this proved to be a naive attempt, as we saw observed 99% train accuracy but only 55% validation and 20% test accuracies. This was clear evidence of overfitting, as the validation accuracy didn't increase and the validation loss didn't decrease for atleast a few 100 epochs. We performed early stopping for the same, and the model ran for about 20 epochs before converging with about 60% validation and test accuracies.

This was clear evidence of class imbalance and a full roadblock on the idea of using the unedited original data. Hence, we performed further experiments on the dataset we obtained from applying the Synthetic Minority Over-Sampling Technique.

Also, a side note- we train on 250 epochs in all our runs, as any number either end i.e. 100 or 500 led to the model not converging or overfitting, respectively.

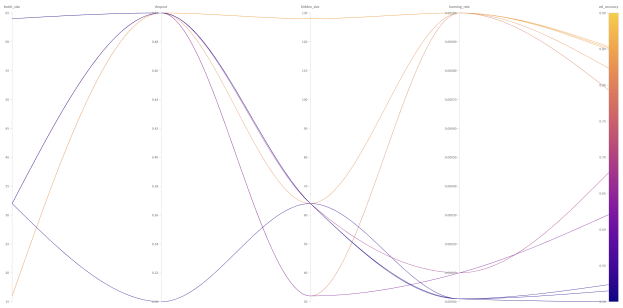

Figure 5. Sweep Results

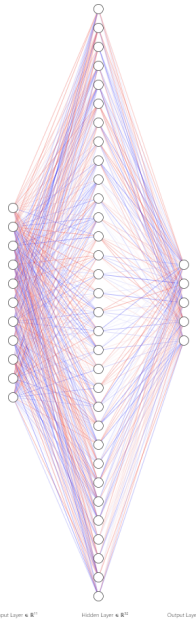

Figure 6. Single Hidden Layer Architecture

- **Shallow Neural Network**: Beginning with a simple single-hidden layer architecture 6, this gave us 85% accuracy on the test set upon using a "sweep" across a bunch of hyperparameters as shown in 1, with train accuracy touching 87%. This is good, but considering this is the most vanilla of Neural Networks, it can get better.
- **Deep Neural Network**: Introducing 2 *more* hidden layers of 32 or 64 or 128 (on the basis of the grid sweep) neurons each, we saw an improvement of the best test accuracy by 1%. The iterations in which the model had higher number of neurons in the hidden layers i.e. 128 neurons tended to show better convergence.
- **Neural Network with Dropout**: Using an architecture with 2 hidden layers, we introduced the concept of dropout to our model in which certain neurons of layers are de-activated i.e. the activation output of these

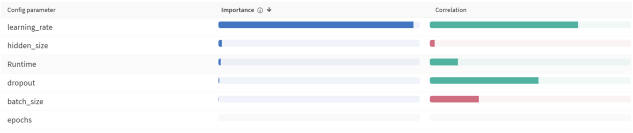

Figure 7. Hyper-parameter Importance

neurons is set to 0 with an arbitrary probability p. Now, for deciding this p, we added a "dropout" parameter to the WandB based Sweep Configuration, with possible values of 0.3 and 0.5. There was a very clear of higher validation accuracy to a higher value of dropout probability, implying that a high number of neurons were unnecessary and a sparser model in terms of width was better 5. There were many other inferences that we could make from 5, and they have been mentioned in the Results section.

- **Neural Network with Batch Normalization layers**: Introducing Batch Normalization led to a drop in accuracy to 84.5%. It appeared to regularize the model excessively, reducing its ability to fit the training data. This suggests that Batch Normalization may not be beneficial for such shallower networks or such smaller datasets.

Table 1. Sweep Configuration Parameters

Parameter	Description	Values
learning_rate	Learning rate values to search	{0.001, 0.0001, 0.00001}
hidden_size	Hidden layer sizes to search	{32, 64, 128}
batch_size	Batch sizes to search	{16, 32, 64}
dropout	Dropout rates to search	{0.3, 0.5}

4.3. MLOps Pipeline

We implemented an MLOps pipeline to manage the model lifecycle, including data ingestion, preprocessing, model training, and evaluation. Experiment tracking was handled through version control, ensuring reproducibility.

5. Results and Analysis

Regression models struggled with predicting discrete quality scores. After rounding, the performance was significantly worse compared to classification models.

- **Feature Distribution**: The data shows a significant imbalance in the target variable (`quality`), with the majority of samples clustered around `quality` scores of 5 and 6. This indicated a potential bias in the dataset that needed addressing to prevent model favoritism toward majority classes. Also, the effectiveness of SMOTE indicates that the dataset is sensitive to class imbalances, and synthetic sampling can improve model generalization by creating a more evenly distributed training set. This highlights the importance of robust data preprocessing for imbalanced datasets.

- **Learning Rate**: The learning rate is the most influential hyperparameter, as shown by its high importance in determining model performance. This indicates that small adjustments to the learning rate can significantly affect convergence speed and overall accuracy. *Fewer gradient jumps are required to arrive at some loss minima.*

- **Hidden Layer size**: Larger hidden layer sizes (e.g., 128 neurons) showed better convergence and accuracy, especially in deeper networks. This suggests that the dataset benefits from higher model capacity to capture complex patterns.

- **Batch Normalization Regularization**: Batch Normalization, while useful in deeper networks, over-regularized the simpler architectures, leading to reduced accuracy. This suggests the dataset and model architecture did not benefit from the feature normalization during training.

- **Feature Interaction**: Nonlinear models like Random Forest and Gradient Boosting outperformed linear models, indicating that interactions between features play a crucial role in determining wine quality.

6. Conclusion

In this project, we explored both regression and classification models for predicting wine quality. Classification models, particularly ensemble methods like Random Forest and Gradient Boosting, performed significantly better than regression models. Non-linear relationships between features and discrete target classes could only be handled by these Classification models.

For a small dataset like the one used in this study, several key observations were made regarding the behavior of the neural networks. Increasing the number of hidden layers significantly improved model performance, as deeper architectures were better equipped to capture the underlying complex patterns. Additionally, larger learning rates demonstrated better convergence during training, underscoring the importance of tuning this parameter effectively, especially for smaller datasets where fewer iterations are required to generalize well. Interestingly, the introduction of Batch Normalization led to a decline in performance, suggesting that this technique, while beneficial for larger

and more complex datasets, may overly regularize models trained on small datasets, reducing their ability to effectively learn the data distribution.

We successfully implemented a basic pipeline to manage the model lifecycle, covering data ingestion, preprocessing, model training, and evaluation. The initial experiment tracking was also integrated with version control, ensuring reproducibility and experiment transparency. By building a scalable framework, we can continuously improve the model's accuracy and reliability while minimizing manual intervention and reducing time-to-deployment.

Any new dataset can be easily trained, analyzed, and potentially deployed. This could be useful for getting your hands dirty on new models or new datasets. All the user has to do is collect data and define the model architecture.

6.1. Contribution

- **Arnav Agrawal:** Preprocessing, Model Implementation, Eval, Inference
- **Aryan Singla:** EDA, Inference
- **Harsh Rajput:** Preprocessing, Eval
- **Kshitij Gupta:** Preprocessing, Model implementation

7. References

References

[1] K. R. Dahal, J. N. Dah-al, H. Banjade, and S. Gaire. Prediction of wine quality using machine learning algorithms. *Open Journal of Statistics*, 11:278–289, 2021.

[2] K. Jain, K. Kaushik, S.K. Gupta, et al. Machine learning-based predictive modelling for the enhancement of wine quality. *Scientific Reports*, 13(1):17042, 2023.