SC1015 MINI PROJECT

Wine Quality Prediction using Machine Learning

By FR1-Team 2

Ayushmaan Kumar Yadav - N2400848H
Yu Huajia - U2322589F
Diego Claude - N2402496D

Problem Statement

Explores different ways to predict wine quality using physicochemical properties of red wine

The original dataset contains integer quality scores ranging from 3 to 8

Each team member framed a unique machine learning problem based on this dataset

Dataset

- Source: UCI Wine Quality Dataset
- File used: winequality-red.csv (https://www.kaggle.com/datasets/yasserh/wine-quality-dataset/data)
- Target: quality
- Features: 11 physicochemical test results per wine (e.g., alcohol, pH, sulphates)

Methodology

1. Data Cleaning & Preprocessing
2. Exploratory Data Analysis (EDA)
3. Modeling
4. Upsampling Techniques
5. Evaluation

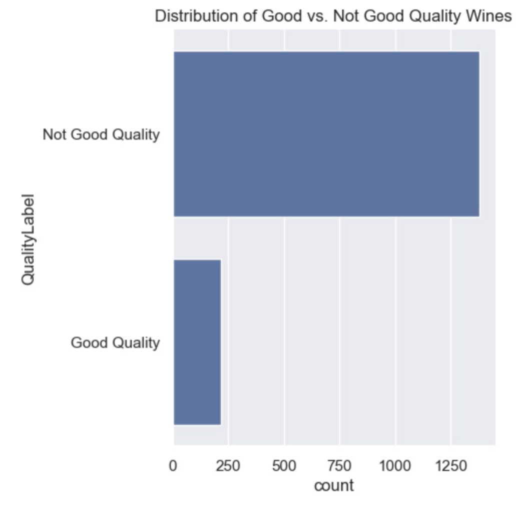
Approach 1

Explored red wine quality prediction as a binary classification problem, where each wine sample was categorized into **2** quality levels:

- **Not good <7**
- **Good ≥7**

Distribution of Quality

- Not Good: 1382 samples (86%)
- Good: 217 (14%)

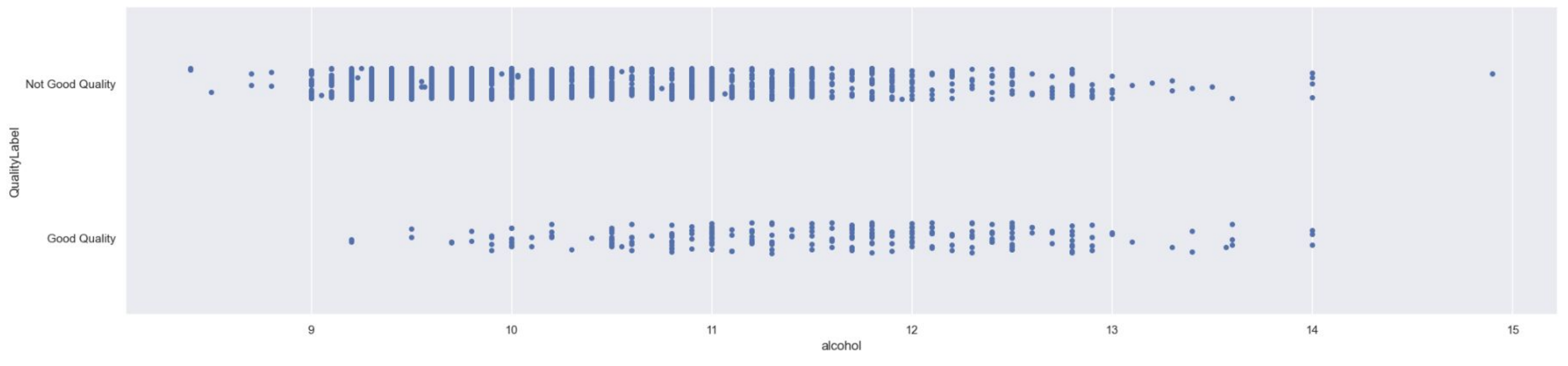


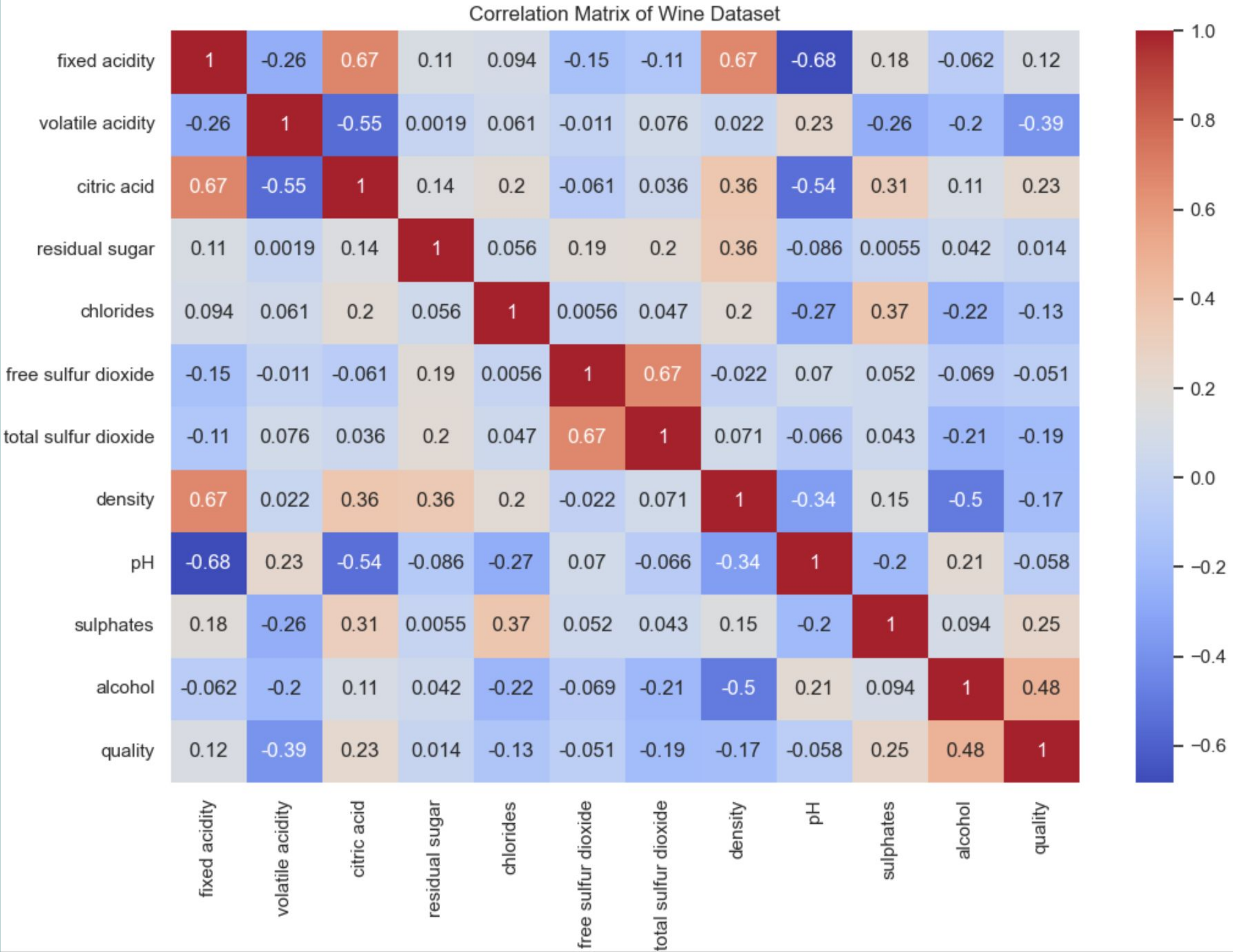
- Reframed wine quality as a **binary task**:
- Practical for consumer-facing or industrial quality filtering
- Simplifies modeling but introduces **heavy class imbalance**
- Targets real-world use: "Is this wine worth recommending?"

Feature Correlation & Insights

What Influences Quality?

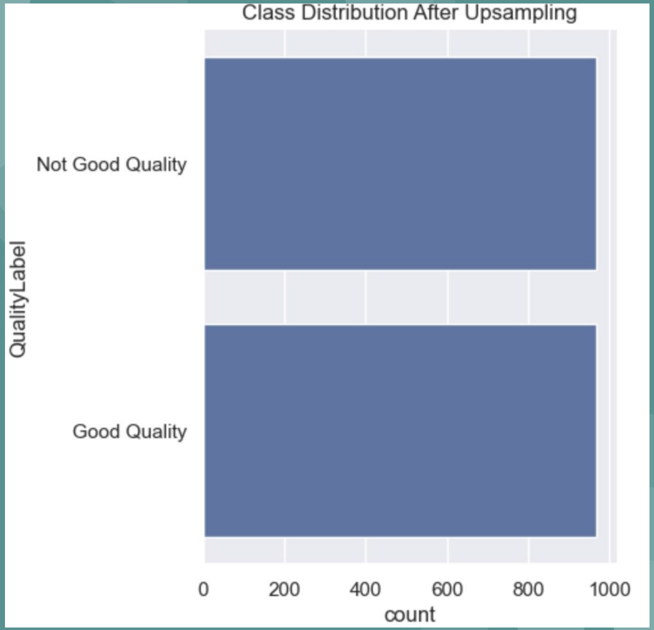
- **Alcohol** has strong positive correlation with "Good" wines
- **Volatile acidity** negatively affects perceived quality
- EDA showed higher alcohol content often means higher score
- Feature selection informed by **heatmaps and histograms**




Correlation Matrix of Wine Dataset
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 0.061 -0.011 0.076 0.022 0.23 -0.26 -0.2 -0.39
citric acid 0.67 -0.55 1 0.14 0.2 -0.061 0.036 0.36 -0.54 0.31 0.11 0.23
residual sugar 0.11 0.0019 0.14 1 0.056 0.19 0.2 0.36 -0.086 0.0055 0.042 0.014
chlorides 0.094 0.061 0.2 0.056 1 0.0056 0.047 0.2 -0.27 0.37 -0.22 -0.13
free sulfur dioxide -0.15 -0.011 -0.061 0.19 0.0056 1 0.67 -0.022 0.07 0.052 -0.069 -0.051
total sulfur dioxide -0.11 0.076 0.036 0.2 0.047 0.67 1 0.071 -0.066 0.043 -0.21 -0.19
density 0.67 0.022 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 -0.27 0.07 -0.066 -0.34 1 -0.2 0.21 -0.058
sulphates 0.18 -0.26 0.31 0.0055 0.37 0.052 0.043 0.15 -0.2 1 0.094 0.25
alcohol -0.062 -0.2 0.11 0.042 -0.22 -0.069 -0.21 -0.5 0.21 0.094 1 0.48
quality 0.12 -0.39 0.23 0.014 -0.13 -0.051 -0.19 -0.17 -0.058 0.25 0.48 1
fixed acidity
volatile acidity
citric acid
residual sugar
chlorides
free sulfur dioxide
total sulfur dioxide
density
pH
sulphates
alcohol
quality
1.0
0.8
0.6
0.4
0.2
0.0
−0.2
−0.4
−0.6

Train/Test Split & Scaling

- Used **80/20 stratified split** to preserve class ratio
- Applied **StandardScaler** to normalize features
- Preprocessing done before both models (with and without upsampling)
- Ensures fair feature weighting and robust comparison



First Model: Random Forest Imbalanced

- Trained on original (imbalanced) dataset
- **High accuracy (~88%)**, but **very low recall for “Good” class**
- Model overfits to majority “Not Good” wines
- Highlights **why accuracy is misleading** for imbalanced data

```
Train Data
Accuracy  :       0.9249329758713136

TPR Train (Sensitivity/Recall) :        0.5266666666666666
TNR Train (Specificity)        :        0.9865841073271414

FPR Train (False Positive Rate):        0.013415892672858616
FNR Train (False Negative Rate):        0.47333333333333333
```

Balancing classes with random oversampling

- Randomly **duplicated “Good” wine samples** to match “Not Good”
- Prevented bias during model training
- Easy but effective way to balance binary classes
- Performed **only on the training set** to avoid data leakage

```
Original class distribution:
quality
3     10
4     53
5    681
6    638
7    199
8     18
Name: count, dtype: int64

Balanced class distribution:
quality
3    681
4    681
5    681
6    681
7    681
8    681
Name: count, dtype: int64
```

Final Model – Balanced vs Unbalanced

- Retrained same model on balanced data
- **Improved recall and F1** for “Good” class
- Accuracy stable, but **model is fairer and more generalizable**
- Demonstrates how simple resampling improves real-world performance

```
Test Data
Accuracy  :     0.8384879725085911

TPR Test (Sensitivity/Recall) :  0.8904109589041096
TNR Test (Specificity)        :  0.7862068965517242

FPR Test (False Positive Rate):  0.21379310344827587
FNR Test (False Negative Rate):  0.1095890410958904
```

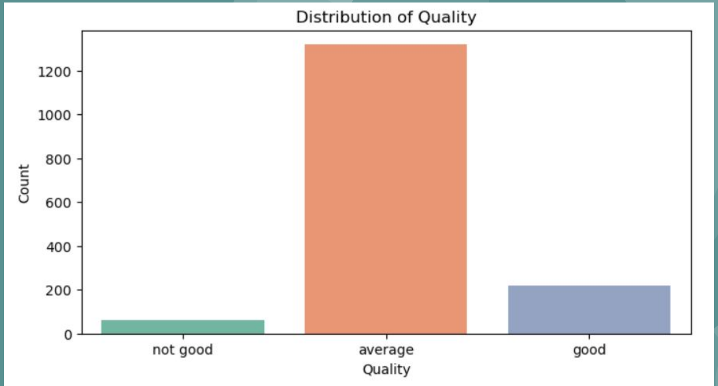
Approach 2

Explored red wine quality prediction as a multi-class classification problem, where each wine sample was categorized into **3** quality levels:

- **Not good (≤4)**
- **Average (5 – 6)**
- **Good (≥7)**

Distribution of Quality

- Not Good: 63 samples (4%)
- Average: 1,319 (82%)
- Good: 217 (14%)

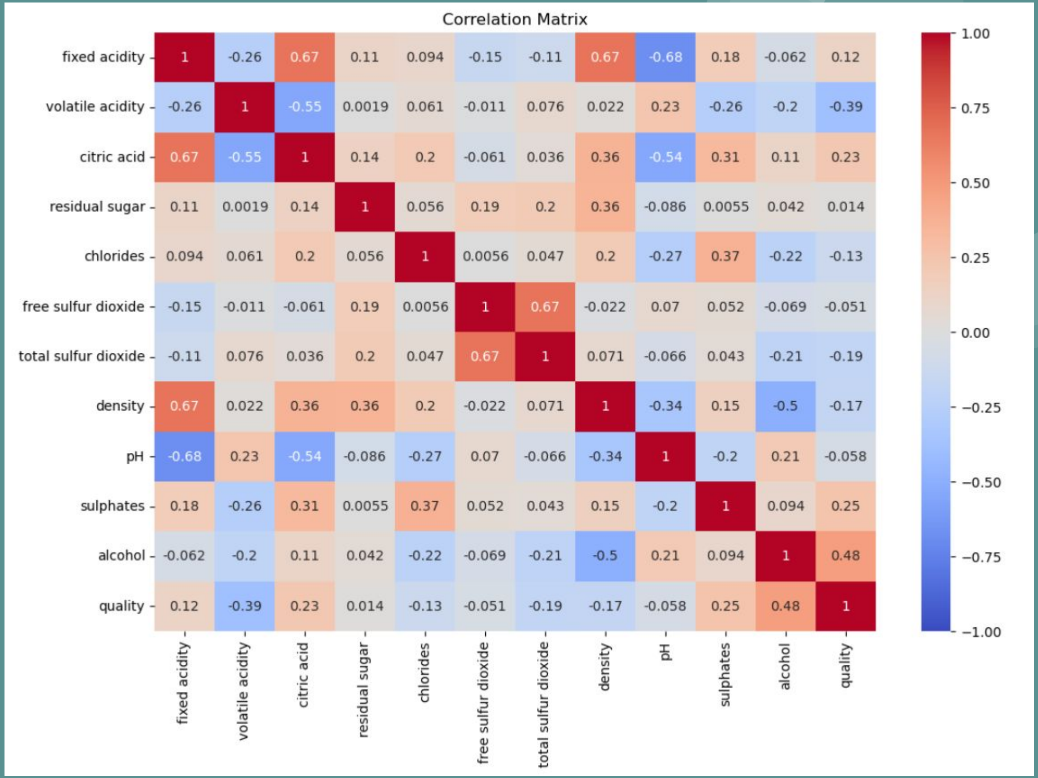


- A wine rated 5 (average) differs significantly from one rated 3 (not good) or 8 (good)
- The minority classes (‘Bad’ and ‘Good’) constitute just 18% of observations

Heavy skews towards “Average” wines → severe Class-Imbalance

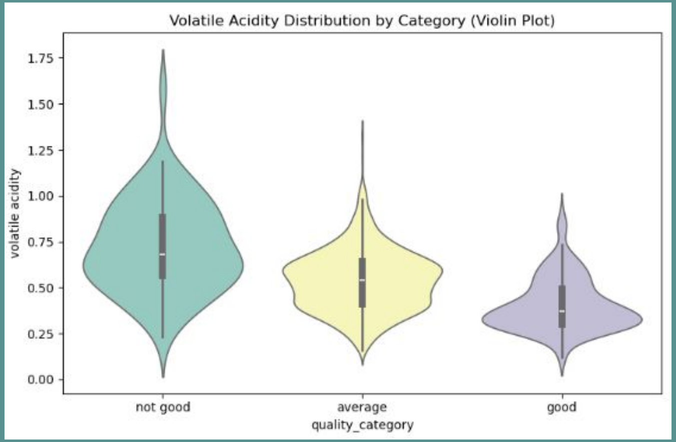
Feature Distribution

Correlation Matrix

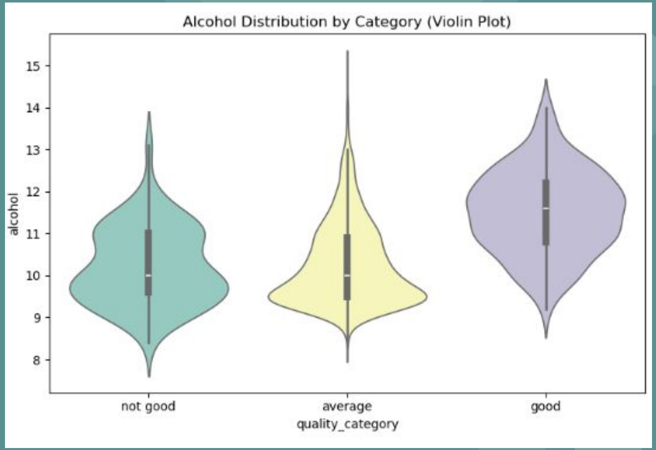


Feature Distribution

↓ **Volatile acidity** indicates better flavor



↑ **Alcohol** correlates with higher quality



Train & Test Split

Split data 80/20 (stratified to preserve class ratios)

- 80% train
- 20% test

Standardized features (StandardScaler)

- mean = 0
- Standard Deviation = 1

Baseline Models (No Upsampling)

Models trained on imbalanced data:

- Random Forest: accuracy of **87.50%**, macro f1 of **0.54**
- XGBoost: accuracy of **86.56%**, macro f1 of **0.58**

=== Random Forest (No Upsampling) ===

Accuracy: 0.8750

	precision	recall	f1-score	support
average	0.90	0.95	0.93	264
good	0.69	0.67	0.68	43
not good	0.00	0.00	0.00	13
accuracy			0.88	320
macro avg	0.53	0.54	0.54	320
weighted avg	0.84	0.88	0.86	320

=== XGBoost (No Upsampling) ===

Accuracy: 0.8656

	precision	recall	f1-score	support
average	0.91	0.93	0.92	264
good	0.69	0.72	0.70	43
not good	0.17	0.08	0.11	13
accuracy			0.87	320
macro avg	0.59	0.58	0.58	320
weighted avg	0.85	0.87	0.86	320

Addressing Class Imbalance (SMOTE)

SMOTE mechanics:

- Finds k-nearest neighbors for each minority sample
- Interpolates new synthetic points along the feature vectors

Effect on data:

- ‘Not Good’ increases from 13 ➔ ~264 samples
- ‘Good’ increases from 43 ➔ ~264 samples

Baseline Models (Upsampled)

Models trained on upsampled data:

- Random Forest: accuracy of **83.75%**, HIGHER macro f1 of **0.65**
- XGBoost: accuracy of **84.69%**, HIGHER macro f1 of **0.61**

```
=== Random Forest (Upsampled) ===
Accuracy: 0.8375

Classification Report:
              precision    recall  f1-score   support

     average       0.95      0.85      0.90       264
        good       0.58      0.88      0.70        43
    not good       0.31      0.38      0.34        13

    accuracy                           0.84       320
   macro avg       0.61      0.71      0.65       320
weighted avg       0.87      0.84      0.85       320
```

```
=== XGBoost (Upsampled) ===
Accuracy: 0.8469

Classification Report:
              precision    recall  f1-score   support

     average       0.93      0.88      0.90       264
        good       0.64      0.81      0.71        43
    not good       0.21      0.23      0.22        13

    accuracy                           0.85       320
   macro avg       0.59      0.64      0.61       320
weighted avg       0.86      0.85      0.85       320
```

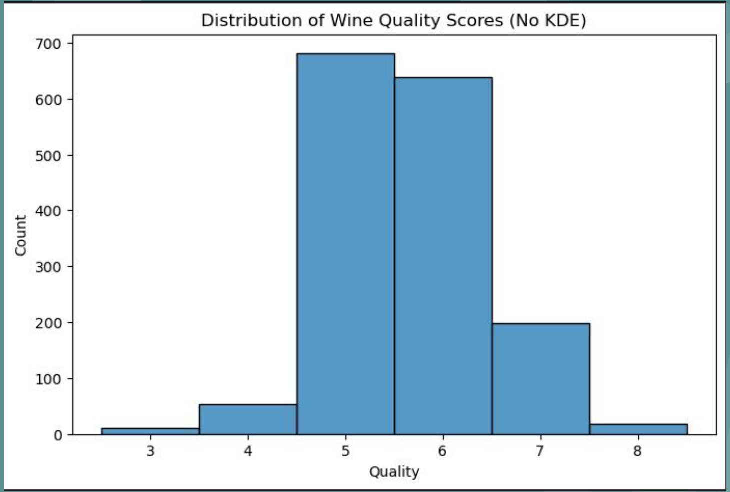
Final Model Comparison

Model	Accuracy	Macro F1	Weighted F1
Random Forest (Imbalanced)	**87.50%**	0.54	0.86
XGBoost (Imbalanced)	86.56%	**0.58**	0.86
Random Forest + SMOTE	83.75%	**0.65**	0.85
XGBoost + SMOTE	**84.69%**	0.61	0.85

Approach 3

Multi-Class Wine Quality Classification (Scores 3 to 8)

- Predicted exact wine quality scores (3 to 8) using multi-class classification
- More fine-grained than binary or 3-class setups
- Better reflects how wines are rated in real life



Modeling Steps:

1. **Started with Linear Regression**

   Poor R^2 and high RMSE → unsuitable for discrete, ordinal scores

```
Linear Regression Model Evaluation:
MAE: 0.503530441552438
RMSE: 0.6245199307980125
R^2: 0.4031803412796229
```

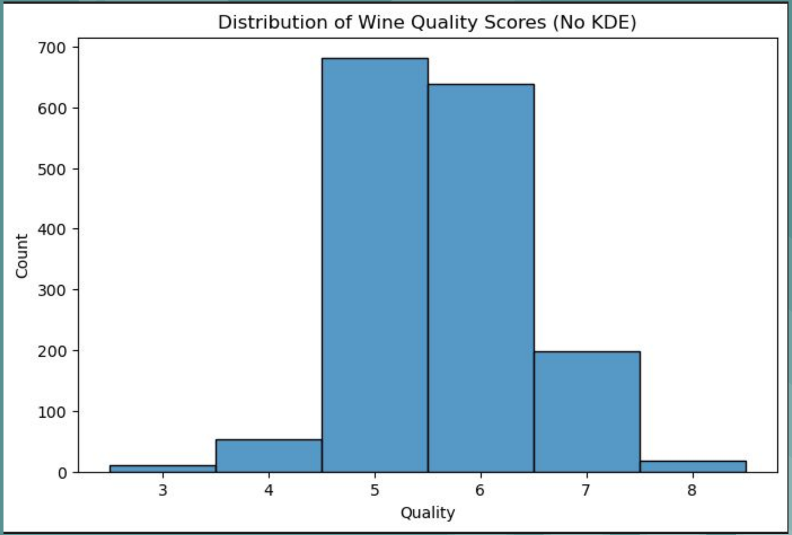
Modeling Steps:

2. Built XGBoost Classifier

Biased toward majority classes (5 & 6)

Baseline Accuracy: 0.6625

	precision	recall	f1-score	support
3	0.00	0.00	0.00	2
4	0.50	0.09	0.15	11
5	0.72	0.72	0.72	136
6	0.62	0.69	0.65	128
7	0.71	0.60	0.65	40
8	0.33	0.33	0.33	3
accuracy			0.66	320
macro avg	0.48	0.41	0.42	320
weighted avg	0.66	0.66	0.65	320



Modeling Steps:

3. Applied Random Oversampling

Better F1 for rare classes, but risk of overfitting

```
Original class distribution:
quality
3     10
4     53
5    681
6    638
7    199
8     18
Name: count, dtype: int64

Balanced class distribution:
quality
3    681
4    681
5    681
6    681
7    681
8    681
Name: count, dtype: int64
```

```
Accuracy on Original Imbalanced Test Set: 0.928125
              precision    recall  f1-score   support

           3       1.00      1.00      1.00         2
           4       0.85      1.00      0.92        11
           5       0.92      0.98      0.95       136
           6       0.98      0.84      0.91       128
           7       0.85      1.00      0.92        40
           8       1.00      1.00      1.00         3

    accuracy                           0.93       320
   macro avg       0.93      0.97      0.95       320
weighted avg       0.93      0.93      0.93       320
```

Modeling Steps:

3. Applied Random Oversampling

Better F1 for rare classes, but risk of overfitting

Accuracy on Original Imbalanced Test Set: 0.928125

20% of the Original Data

	precision	recall	f1-score	support
3	1.00	1.00	1.00	2
4	0.85	1.00	0.92	11
5	0.92	0.98	0.95	136
6	0.98	0.84	0.91	128
7	0.85	1.00	0.92	40
8	1.00	1.00	1.00	3
accuracy			0.93	320
macro avg	0.93	0.97	0.95	320
weighted avg	0.93	0.93	0.93	320

Accuracy on Full Original Imbalanced Dataset: 86.93%

100% of the Original Data

	precision	recall	f1-score	support
3	0.77	1.00	0.87	10
4	0.77	1.00	0.87	53
5	0.89	0.88	0.88	681
6	0.89	0.80	0.84	638
7	0.80	0.98	0.88	199
8	0.95	1.00	0.97	18
accuracy			0.87	1599
macro avg	0.84	0.95	0.89	1599
weighted avg	0.87	0.87	0.87	1599

Modeling Steps:

4. Switched to SMOTE (Synthetic Oversampling)

Created synthetic samples for balanced learning

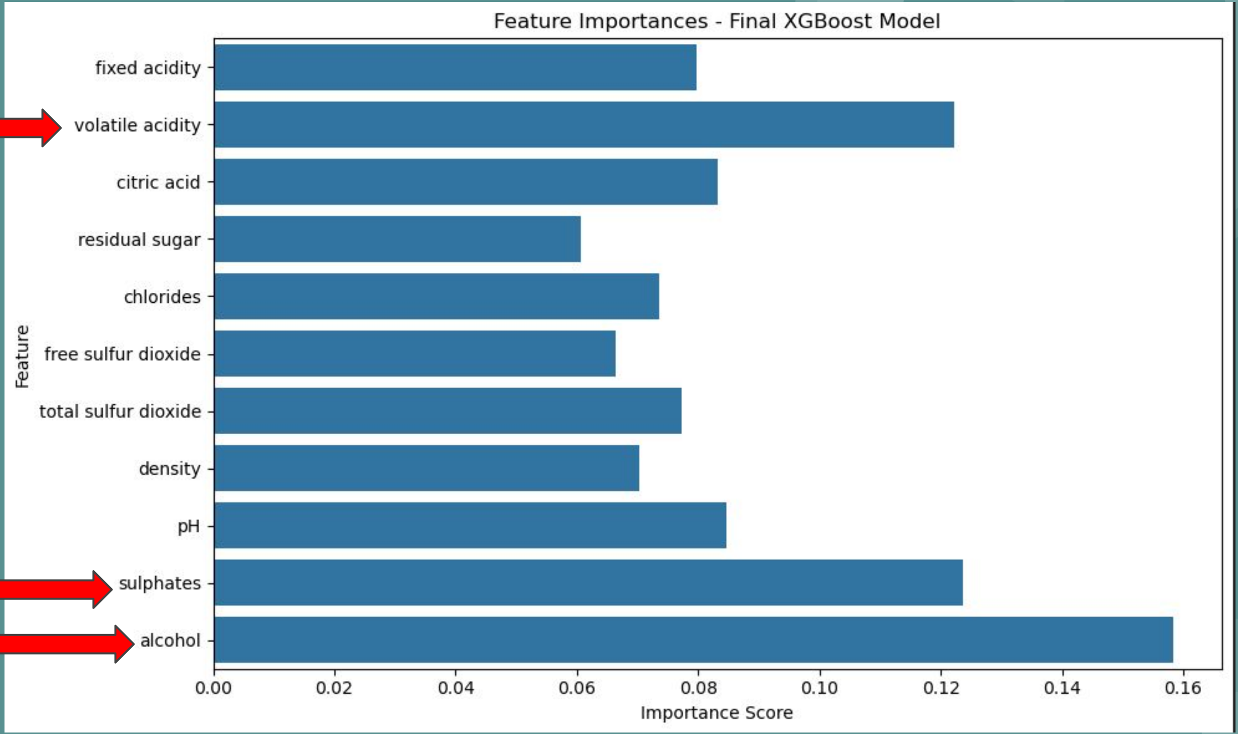
```
Balanced class distribution:
quality_encoded
0    681
1    681
2    681
3    681
4    681
5    681
Name: count, dtype: int64
```

```
Accuracy on Full Original Dataset (SMOTE model): 0.943089430894309
              precision    recall  f1-score   support

           3       0.71      1.00      0.83        10
           4       0.87      0.98      0.92        53
           5       0.96      0.95      0.96       681
           6       0.95      0.92      0.93       638
           7       0.90      0.97      0.93       199
           8       0.95      1.00      0.97        18

    accuracy                           0.94      1599
   macro avg       0.89      0.97      0.93      1599
weighted avg       0.94      0.94      0.94      1599
```

Additional Insights

Feature Importance for the final model

MOST INFLUENTIAL FACTORS



Conclusion

✅Problem 1 - Binary Classification ("Good" vs. "Not Good")

✅Problem 2 - Multi-class Classification ("Not Good", "Average", and "Good")

✅Problem 3 – Multi-class Classification (Quality 3–8)