IBM Employee Attrition Prediction Analysis Report

1. Introduction

The wine industry is highly competitive and saturated with many different varieties and blends for the consumer to choose from. As such, many wine rating websites have popped up such as Wine Enthusiast and Vivino, which help customers separate the gems from the chaff. For a winery, it's important for them to invest in wines that are going to get high ratings so that they can recoup their investments. This project will use Red Wine Quality Data Set, available on the UCI machine learning repository https://archive.ics.uci.edu/ml/datasets/wine+quality/winequality-red.csv
to perform binary classification for wine quality using ML Algorithm
In this report, we summarized the findings and insights gained from analyzing the red wine dataset from UCI machine learning repository.
In this project we used Logistic Regression and Random Forest Classifier to predict red wine quality. We also showed each models' performance on the test data.

2. Dataset Analysis and Preprocessing

Dataset Description:

Wine Quality dataset is publicly available for research in the University of California, Irvine Machine Learning repository created by Paulo Cortez (Univ. Minho), Antonio Cerdeira, Fernando Almeida, Telmo Matos and Jose Reis (CVRVV) in 2009. This repository has two datasets of red and white wine samples which consists of inputs includes objective tests (e.g. PH values) and the output is based on sensory data (median of at least 3 evaluations made by wine experts). Each expert graded the wine quality between 0 (very bad) and 10 (very excellent).

The original data set has 1599 rows and 12 columns. The dataset is clean, with no missing values, no duplicates.

Input variables (based on physicochemical tests):

1. Alcohol: the amount of alcohol in wine
2. Volatile acidity: are high acetic acid in wine which leads to an unpleasant vinegar taste
3. Sulphates: a wine additive that contributes to SO2 levels and acts as an antimicrobial and antioxidant
4. Citric Acid: acts as a preservative to increase acidity (small quantities add freshness and flavor to wines)
5. Total Sulfur Dioxide: is the amount of free + bound forms of SO2
6. Density: sweeter wines have a higher density
7. Chlorides: the amount of salt in the wine
8. Fixed acidity: are non-volatile acids that do not evaporate readily
9. pH: the level of acidity
10. Free Sulfur Dioxide: it prevents microbial growth and the oxidation of wine

11. Residual sugar: is the amount of sugar remaining after fermentation stops. The key is to have a perfect balance between — sweetness and sourness (wines > 45gram/liters are sweet)

Output variable (based on sensory data): 12 - quality (score between 0 and 10)

We performed some transformation to the target variable “quality” in order to perform binary classification.

We categorized wines with score higher than 6 as high quality (hi_quality = 1) and wines with score less than 6 were categorized as low grade(hi_quality =0). And we renamed the column from “quality” to hi_quality”

It is important to note that there is an imbalance in the dataset with only 13.57% of the data were hiqh quality red wines, and 86.43% were low quality. This class imbalance is taken into consideration when splitting the data into training and testing data by using stratification. We also relied more on the metrics that are not heavily skewed by class imbalance, like F1-score, precision, and recall.

Data Exploration:

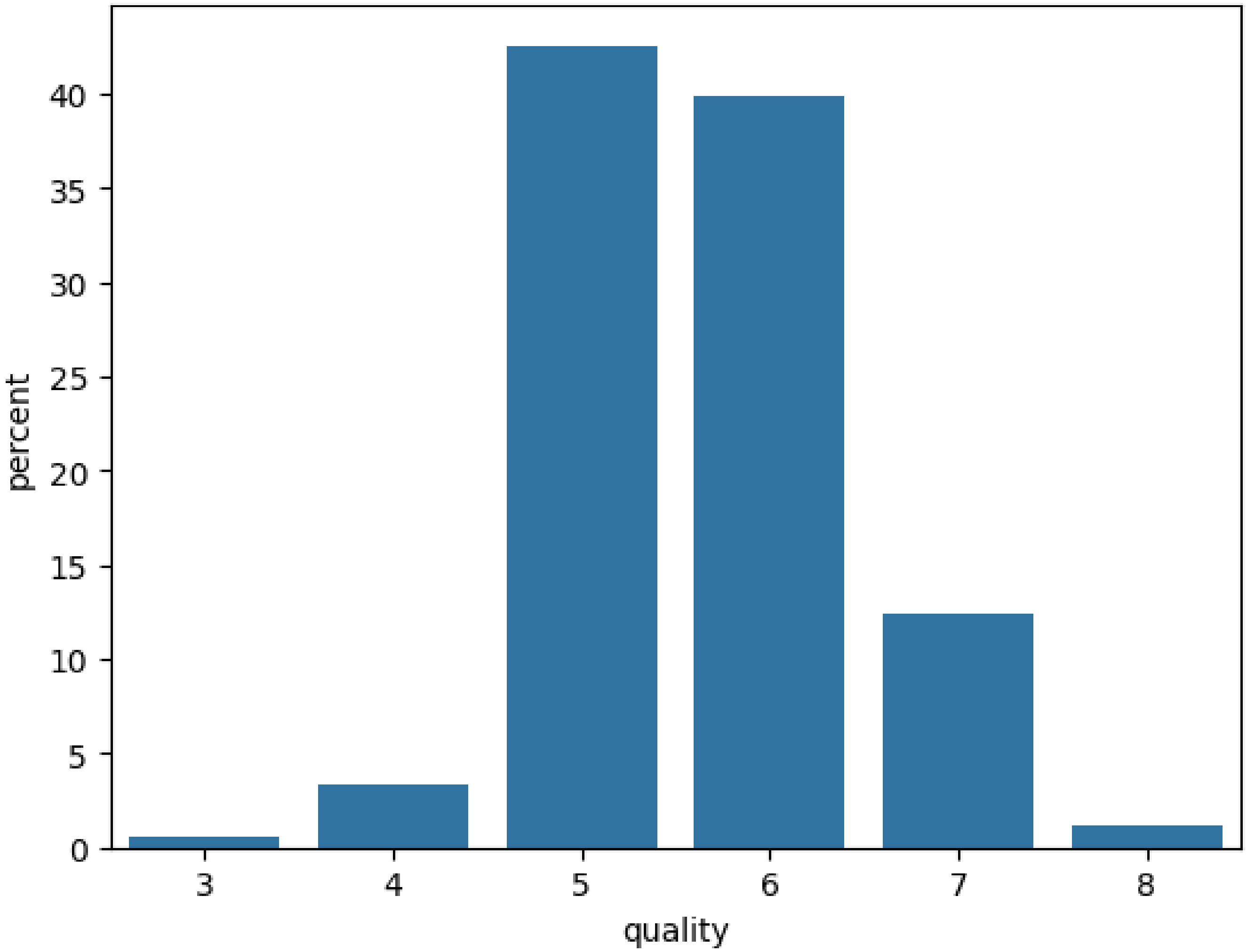
We explored the dataset to understand its structure, features, and distribution. This involved examining descriptive statistics, checking for missing values, and visualizing relationships between variables. Fortunately, our small dataset is pretty clean with no duplicates or missing values.

Preprocessing Steps:

We handled missing values, encoded categorical variables using one-hot encoding, and split the dataset into training and testing sets for model development.

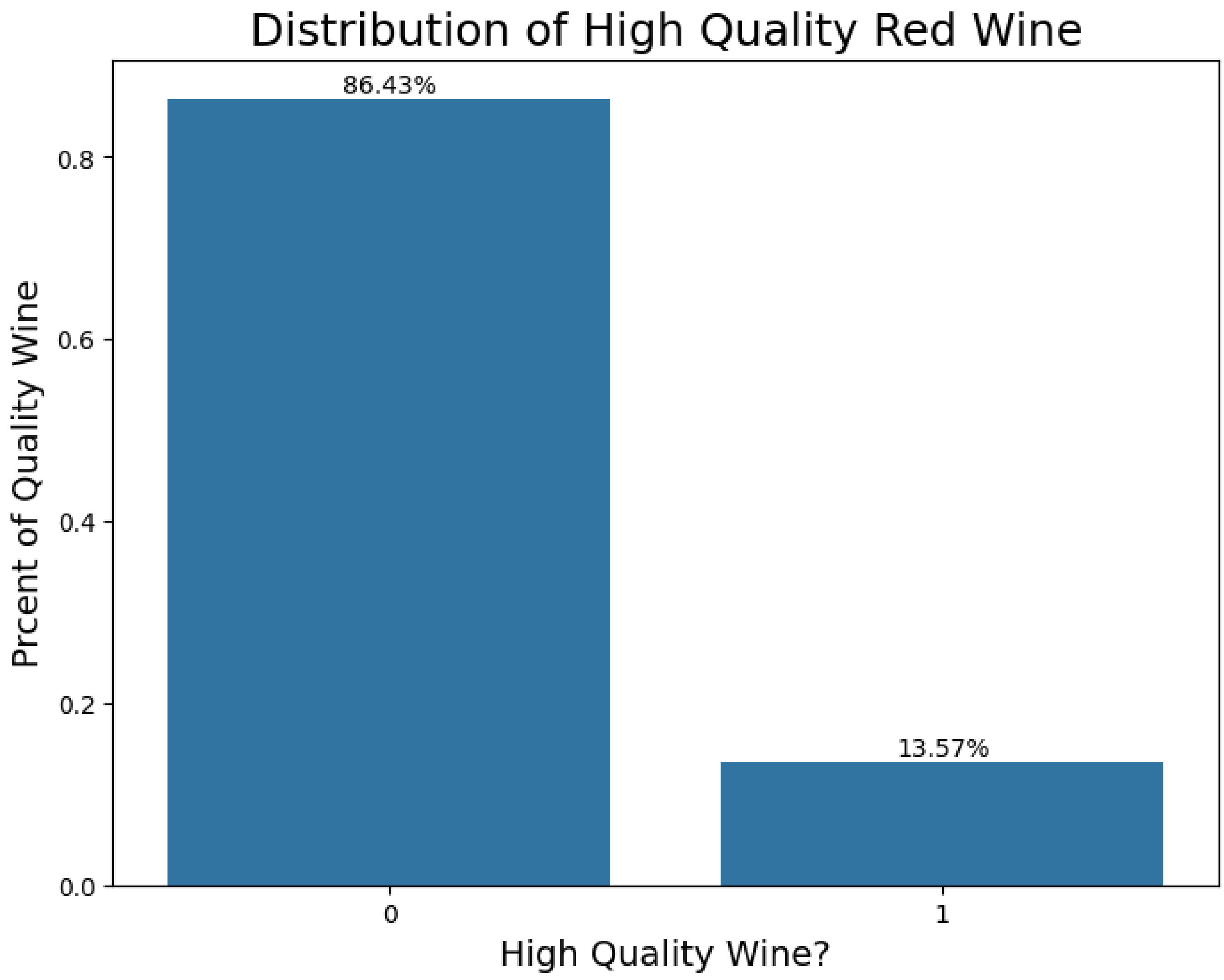
2.1 Exploratory Data Analysis (EDA) and Discovery

First, we created a univariate chart of our target variable, Quality, to look at its distribution.



Our dataset consists mostly of mediocre wine quality.

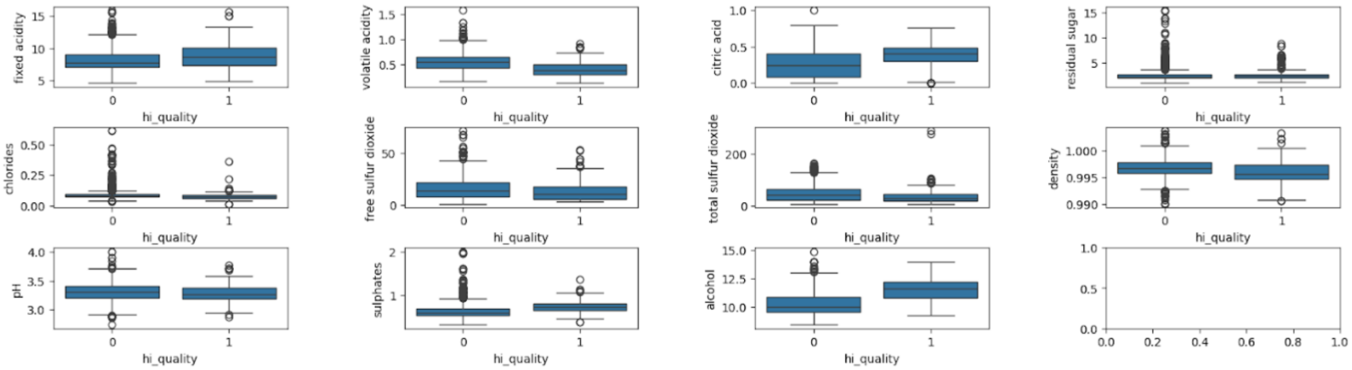
We will create 2 bins for the quality column. Originally the score was from 3 to 8. Instead we will categorize wines with score higher than 6 as high quality (hi_quality = 1) and wines with score less than 6 will be categorized as low grade(hi_quality =0). And we renamed the column from “quality” to hi_quality”
Then let’s look at the distribution again.



As can be seen above, only 13.57% of our wines are high quality. We have an imbalance dataset her

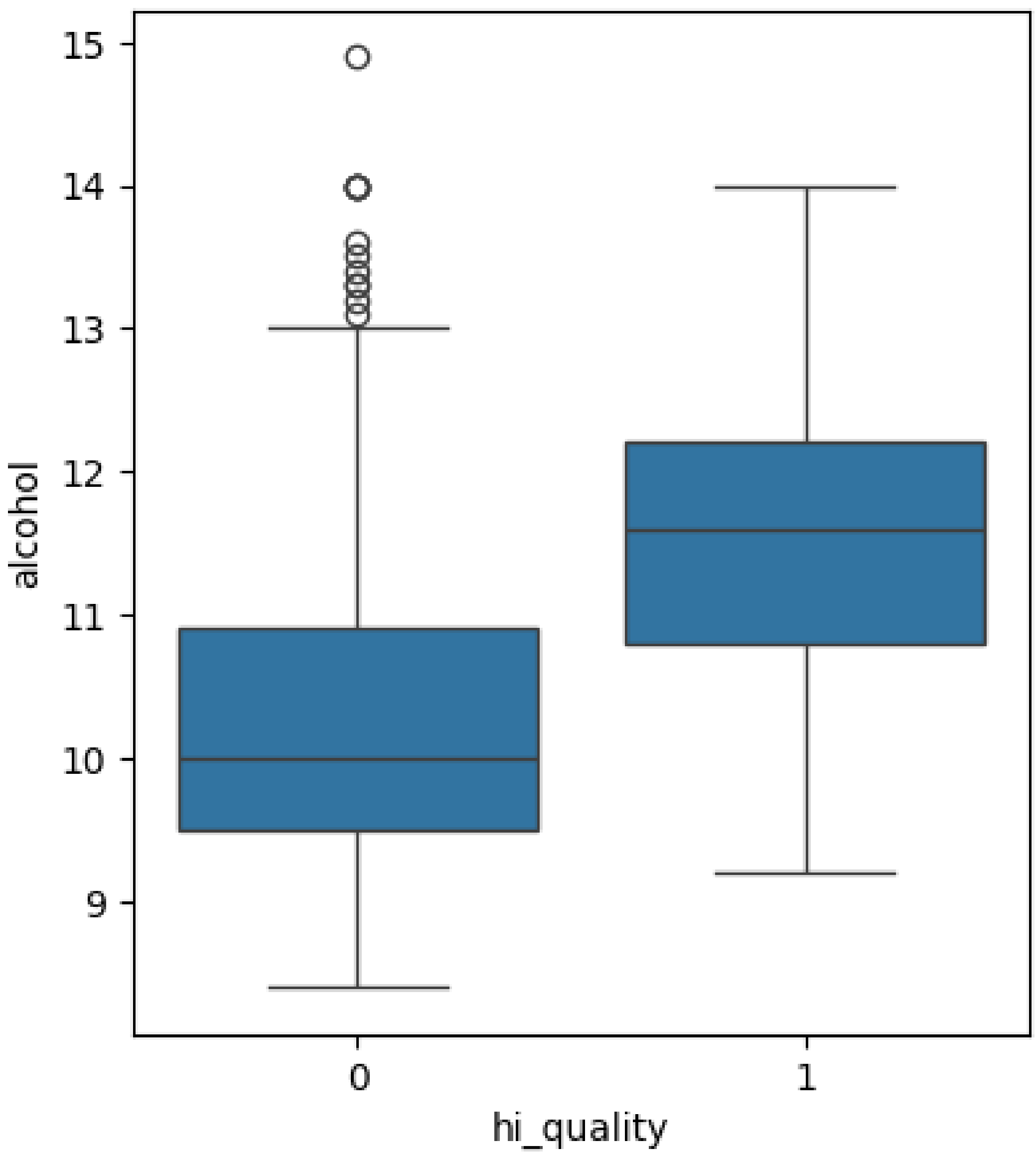
Then we created various bivariate and multivariate charts to see how each feature relates to the target variable, hi_quality.

Let’s look at some boxplots to see how each feature related to target variable “hi_quality”



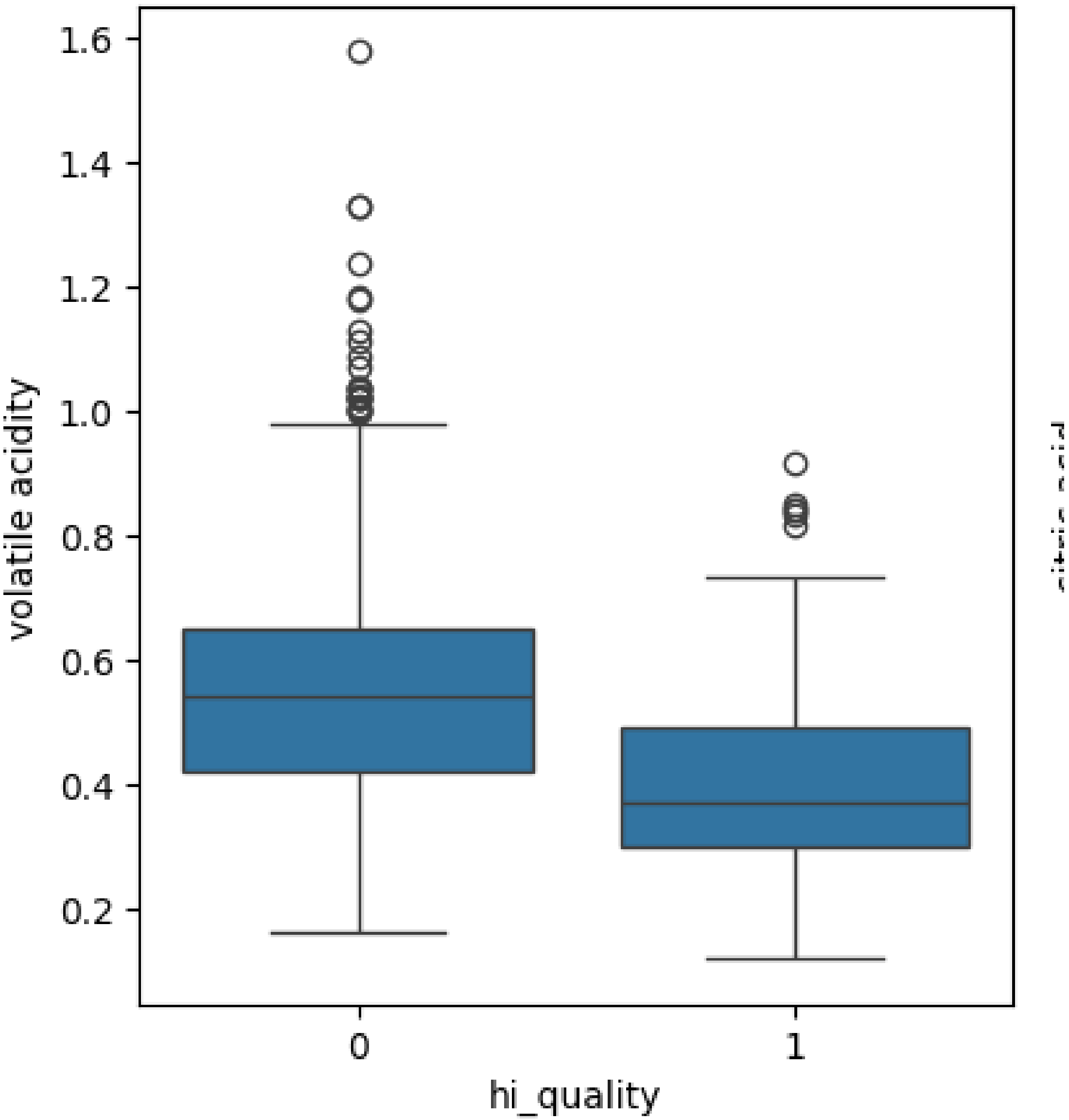
Let’s look at some boxplots to compare our most related features to target variable “hi_quality”

Figure 2a: **High Quality vs Alcohol**



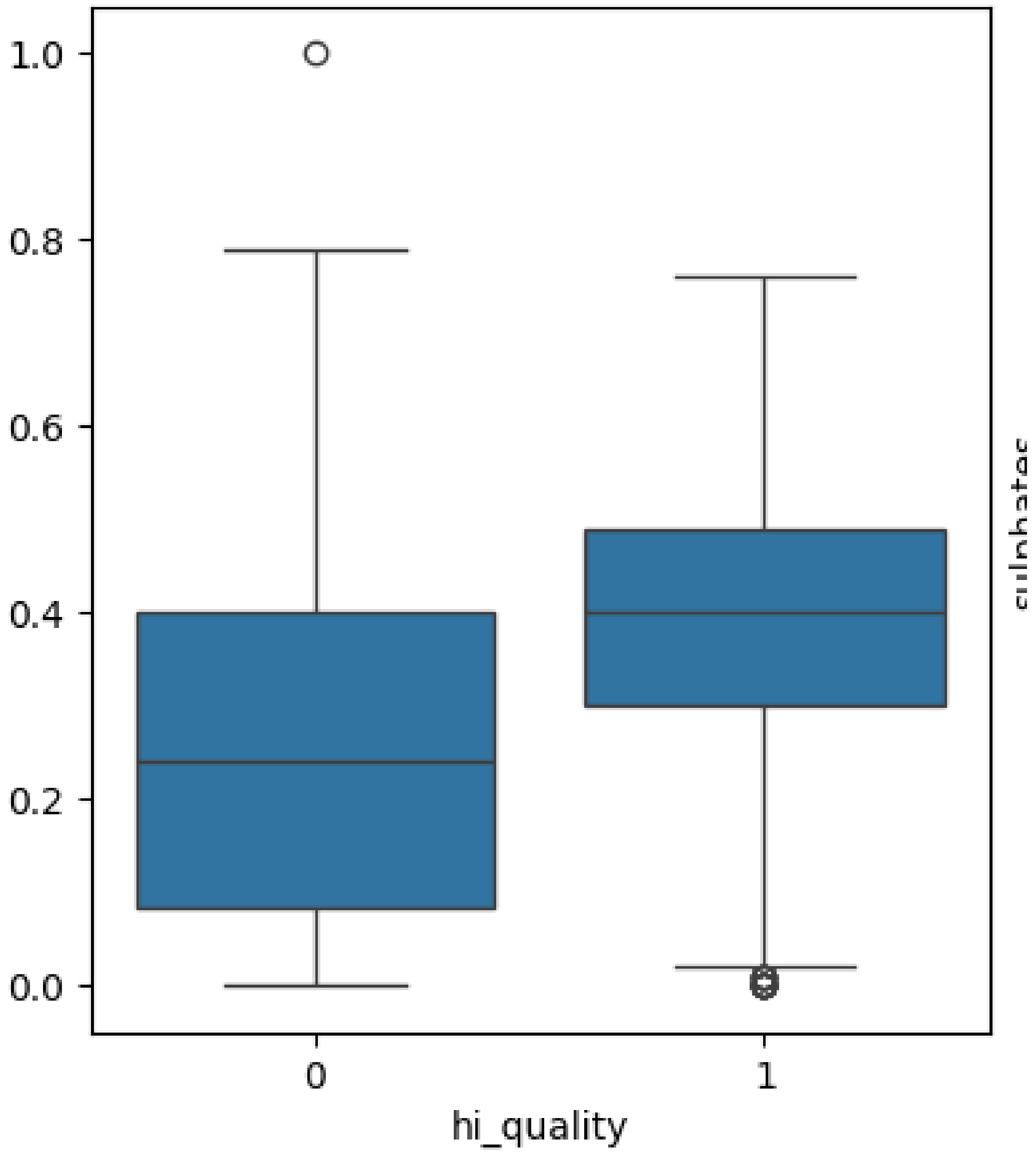
High quality wines have about 1.5% more alcohol on average than low quality wines. While it's unclear exactly why this might be the case, it may simply be due to general taste preferences of the raters lean more towards wine that is alcoholic. This was shown to be statistically significant via a t-test (p-value < .001)

Figure 2b: High Quality vs volatile acidity



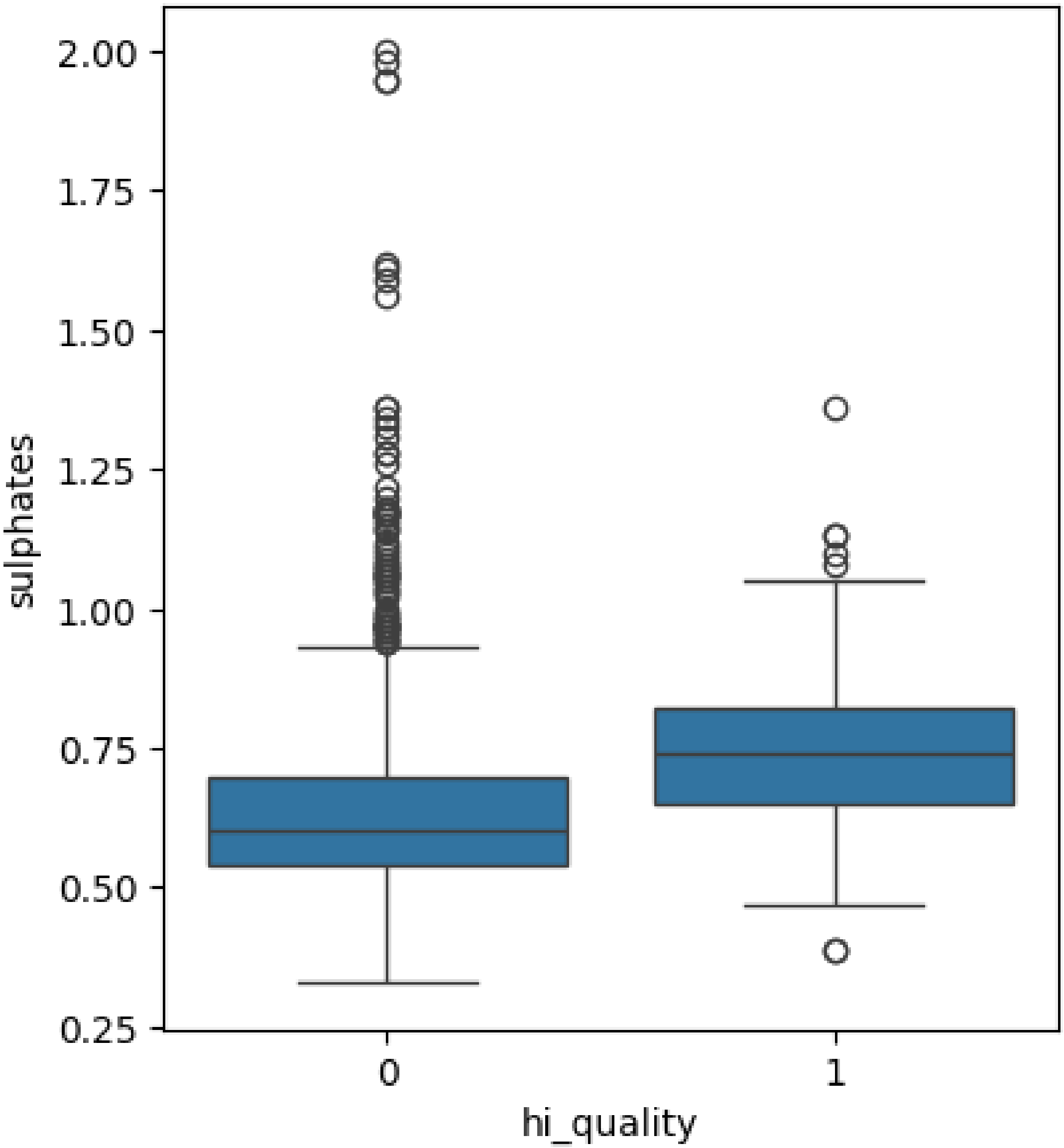
There is an inverse relationship between volatile acidity and wine quality : Lower level of volatile acidity makes better wine. Volatile acids, similar to acetic acid, give a sour taste to our wine, degrading its quality. Hence we see a negative correlation here

Figure 2c: **Quality vs citric acid**



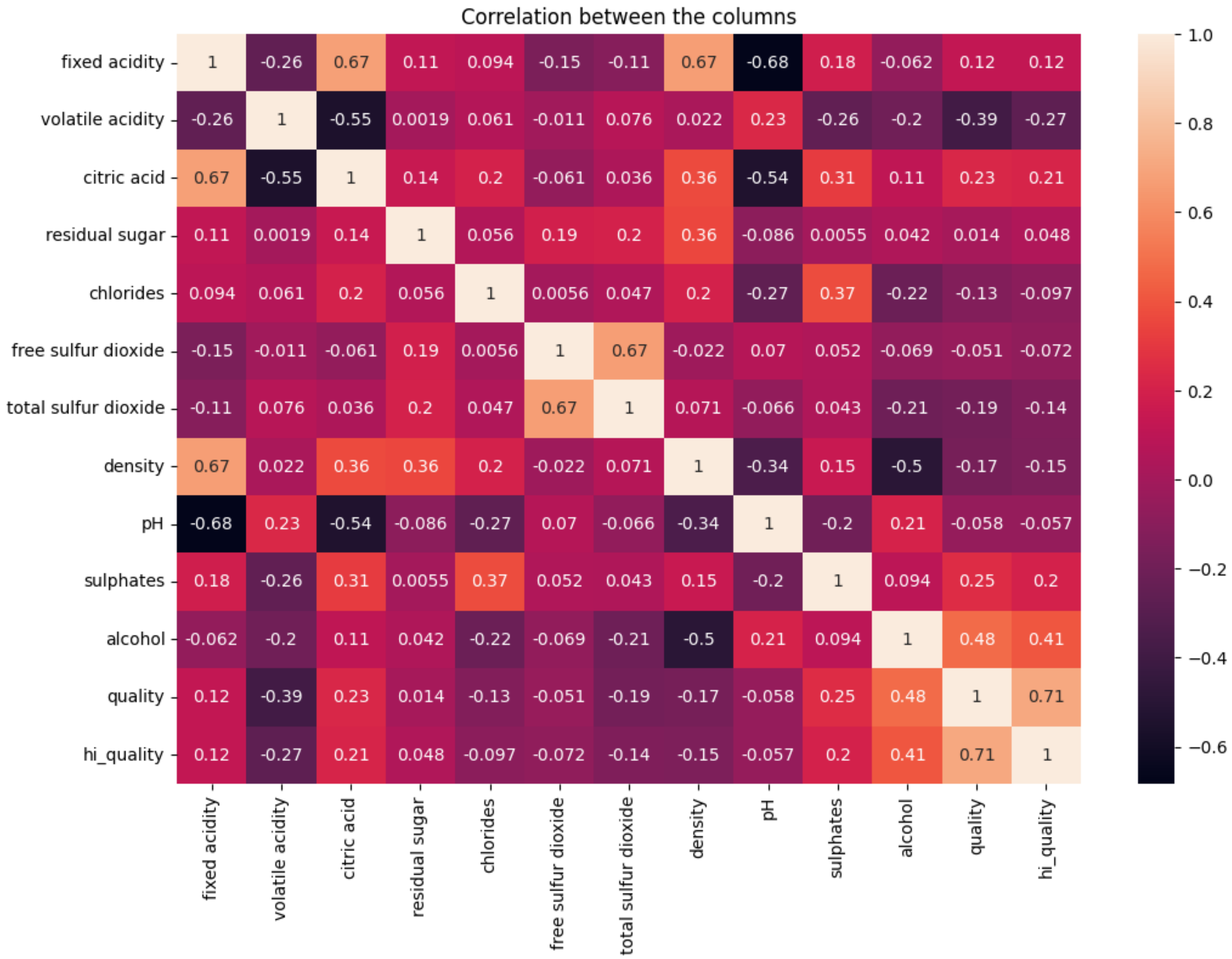
Higher quality wines tend to have more citric acid. Citric acid can give the wine a "fresh" taste, enhancing its flavor. Citric acid is often used as a stabilizer in food and beverages.

Figure 2d: **High Quality vs sulphates**



High quality wines have more sulphates than low quality wine

Dependent Correlation: We created a correlation matrix heatmap to see how the dependent variables relate to our target variable "hi_quality".



The heatmap shows that pH and fixed acidity has inverse correlation, as well as citric acid and volatile acidity Multicollinearity: There is a strong positive correlation between fixed acidity and density (0.67). But we will leave it as it is. we will not drop any column in this dataset

Some columns like citric acid, alcohol, sulphates are strongly and positively correlated whereas a lot of columns like volatile acidity,density,total sulphur dioxide have negative correlation which indicates that these columns do not have a big impact on the target column.
Max correlation: alcohol ; Min Correlation: volatile acidity

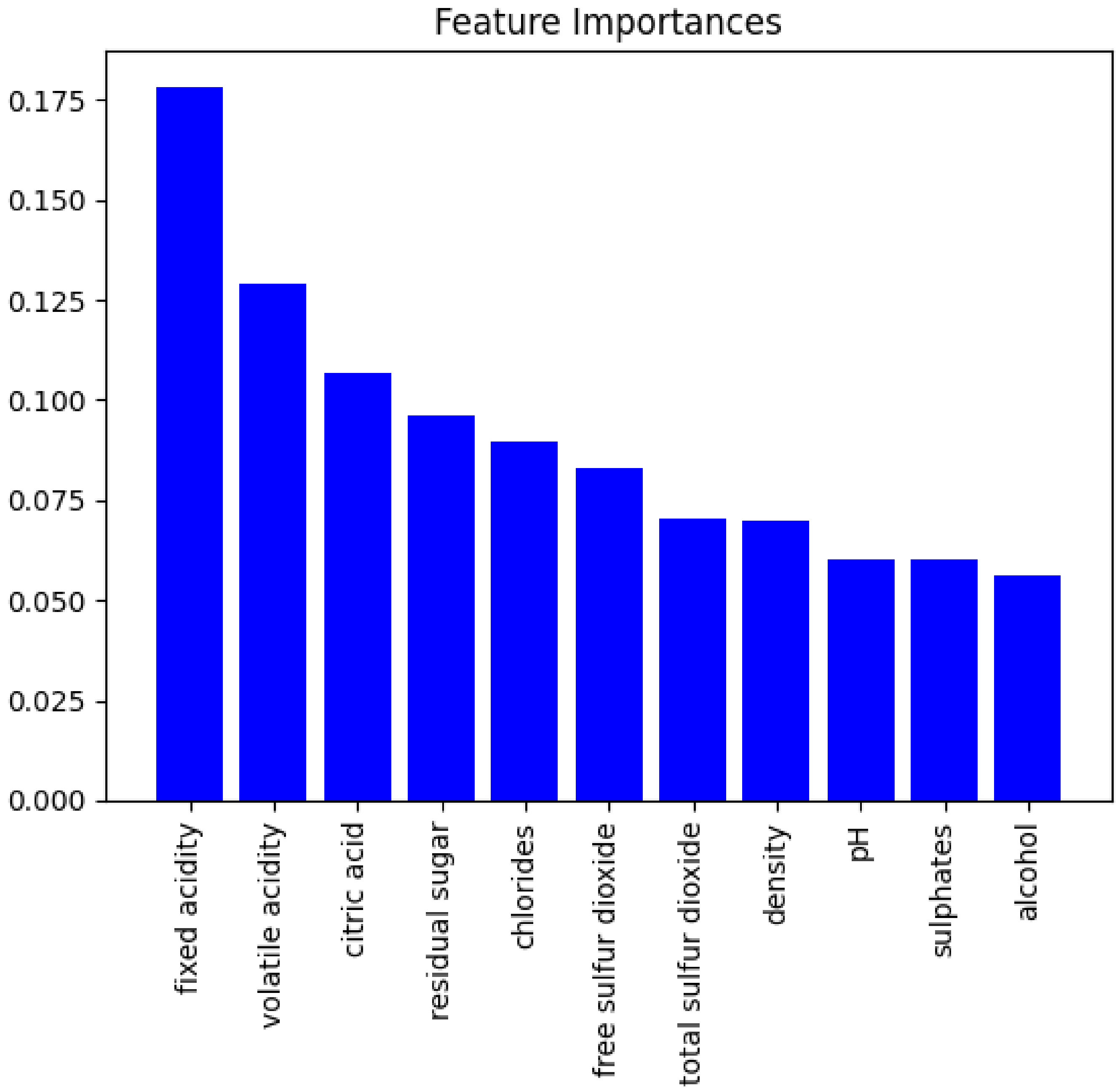
```
wine_df.corr()['quality'].sort_values()
```

```
volatile acidity       -0.390558
total sulfur dioxide   -0.185100
density                -0.174919
chlorides              -0.128907
pH                     -0.057731
```

```
free sulfur dioxide    -0.050656
residual sugar          0.013732
fixed acidity           0.124052
citric acid             0.226373
sulphates               0.251397
alcohol                 0.476166
hi_quality              0.710196
quality                 1.000000
Name: quality, dtype: float64
```

we can see alcohol has strong correlation with hi_quality

Feature importances: The top 10 most important feature according to Random Forest Classifier



The top 10 most important feature according to Random Forest Classifier:
fixed acidity, volatile acidity, citric acid, residual sugar, chlorides, free sulfur dioxide, total sulfur dioxide, density, ph, sulphates, alcohol

Data Preprocessing

We scale our data using Standard Scaler,which applies a linear transformation to our features such that their mean is zero and their variance is one. This is especially useful since the data has varying scales and it prevents the algorithms like linear regression from making assumptions that the data has a Gaussian distribution.

---

3. Model Development

We experimented with two machine learning algorithms for discrete binary classification, including Logistic Regression and Random Forest. We evaluated each model's performance using metrics such as accuracy, precision, recall, F1-score as well as AUC-ROC score on the test data. Optimization techniques like hyperparameter tuning were applied to improve model performance.

Model Performance Metrics: Model performance metrics are used to evaluate the performance of a machine learning model. Because the dataset appears to be imbalanced, with positive class (hi_quality=1) as low as 13.57%%, selecting the right model performance measure is critical. As a result, model accuracy alone cannot determine the robustness of a machine learning model. Based on a confusion matrix created for training dataset predictions:

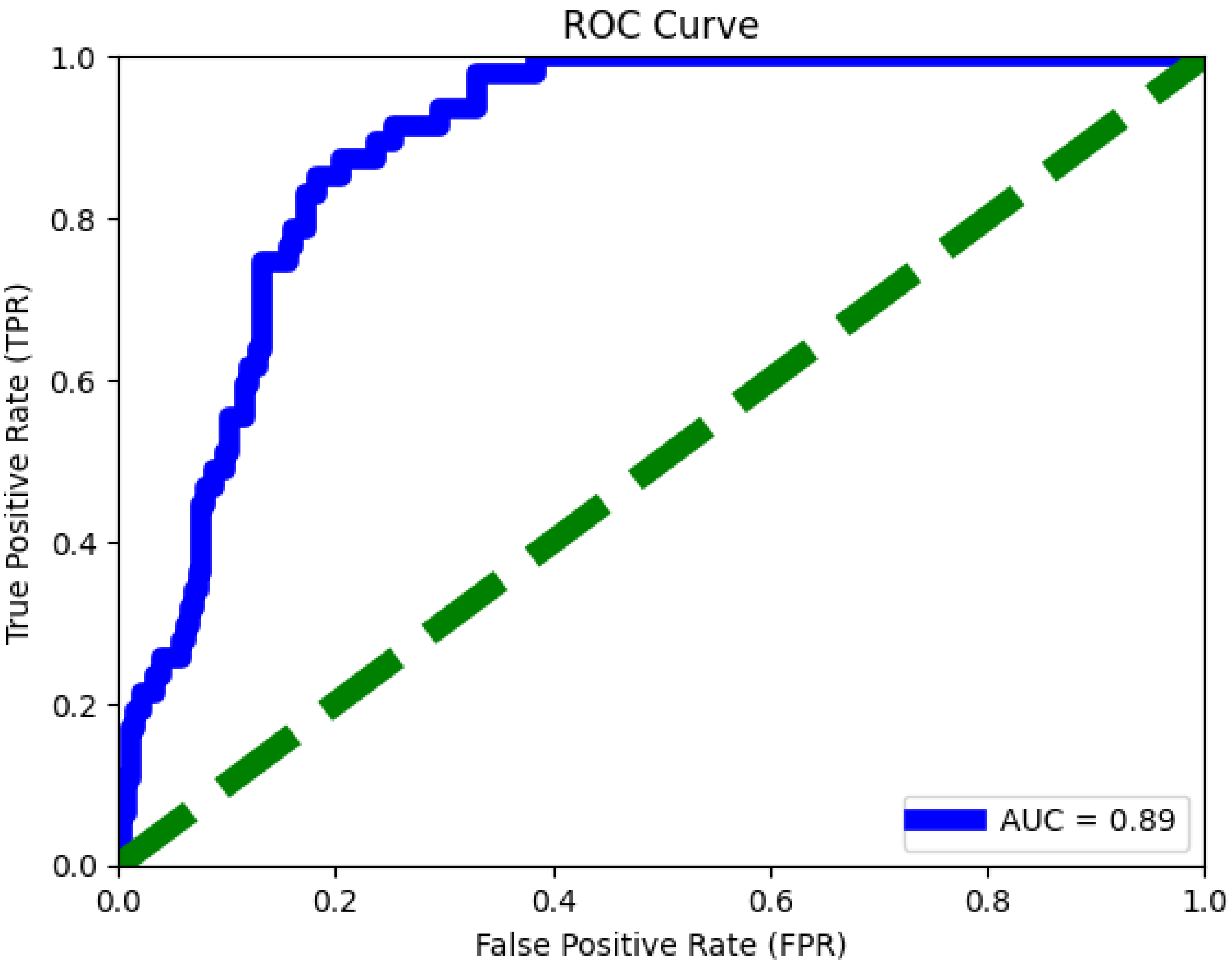
- **Accuracy** is defined as the number of correct predictions made by the machine learning model divided by the total number of datapoints. The best accuracy is 100 percent, which indicates that all predictions are correct. Given our dataset's conversion rate of 13.57%, accuracy is not a valid measure of model performance. Even if all of our predictions are incorrect, our model's accuracy would still be 86.43%. As a result, additional model performance measures are included.

Logistic Regression: A logistic regression model is used for predicting classes using the probability of the target variable. Unlike linear regression, which uses expected values of the response model, logistic regression uses the probability or odds of the response variable to model based on the combination of values taken by the predictors. This model uses the sigmoid function that maps predicted values to probabilities. It works well on linearly separable classes with easy implementation, making it a popular choice for classification problems.

There are two types of logistic regression models for classification: binary and multinomial. Binary logistic regression requires a dependent variable with only two possible outcomes whereas a multinomial requires three or more outcomes. In this case, the dataset is working with binary logistic regression since the target variable is binary (1 or 0). Logistic regression is applicable to this problem since we want to predict the probabilities and classify the red wine quality into two categories based on the explanatory variables. For the solver in the logistic regression model, liblinear is picked since it supports both L1 and L2 regularization.

We first tried to build a logistic regression model, which is specifically designed for binary classification problems and is the most straightforward model in our case. And we also performed hyperparameter tuning via GridSearchCV to improve the result of our model.

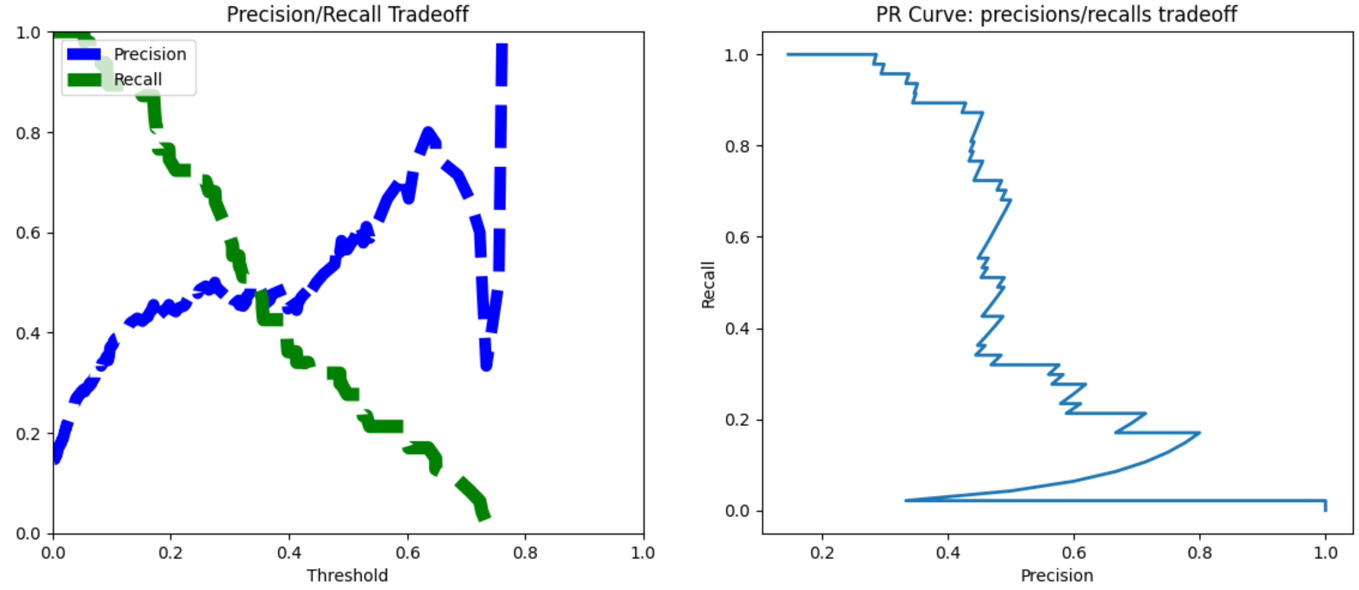
```
Best Parameters for Logistic Regression: {'C': 0.01, 'penalty': 'l2'}

Accuracy Tuned for Logistic Regression: 0.846875

Confusion Matrix Tuned for Logistic Regression:
 [[245  28]
 [ 21  26]]

Classification Report Tuned for Logistic Regression:
               precision    recall  f1-score   support

           0       0.92      0.90      0.91       273
           1       0.48      0.55      0.51        47

    accuracy                           0.85       320
   macro avg       0.70      0.73      0.71       320
weighted avg       0.86      0.85      0.85       320
```



```
AUC-ROC Score: 0.8871483126802275
```

The tuned Logistic Regression model has an F1-score of 0.51 and AUC=0.89 for hi_quality= 1. Considering only a small percentage of red wines are high quality, the Linear Regression model is doing a good job in predicting attrition. We have an imbalanced dataset. The F1 score is generally better than the area under the curve (AUC) for imbalanced datasets when the minority class is of interest. The F1 score is the harmonic mean of precision and recall, which balances the importance of both metrics. It's a more robust evaluation metric than accuracy because it gives a fair representation of a model's performance despite class imbalance.

AUC The AUC is a single value that summarizes a model's overall performance. It's useful for comparing the performance of multiple models. However, the AUC and ROC curve may not be well-suited for imbalanced problems because they may be biased toward the majority class. The accuracy score may also be less useful here as there is lesser emphasis on the minority class (hi_quality = 0) with better results in the majority class. With the relatively small sample and lack of longitudinal data, these scores are pretty decent in aiding the company to predict red wine quality

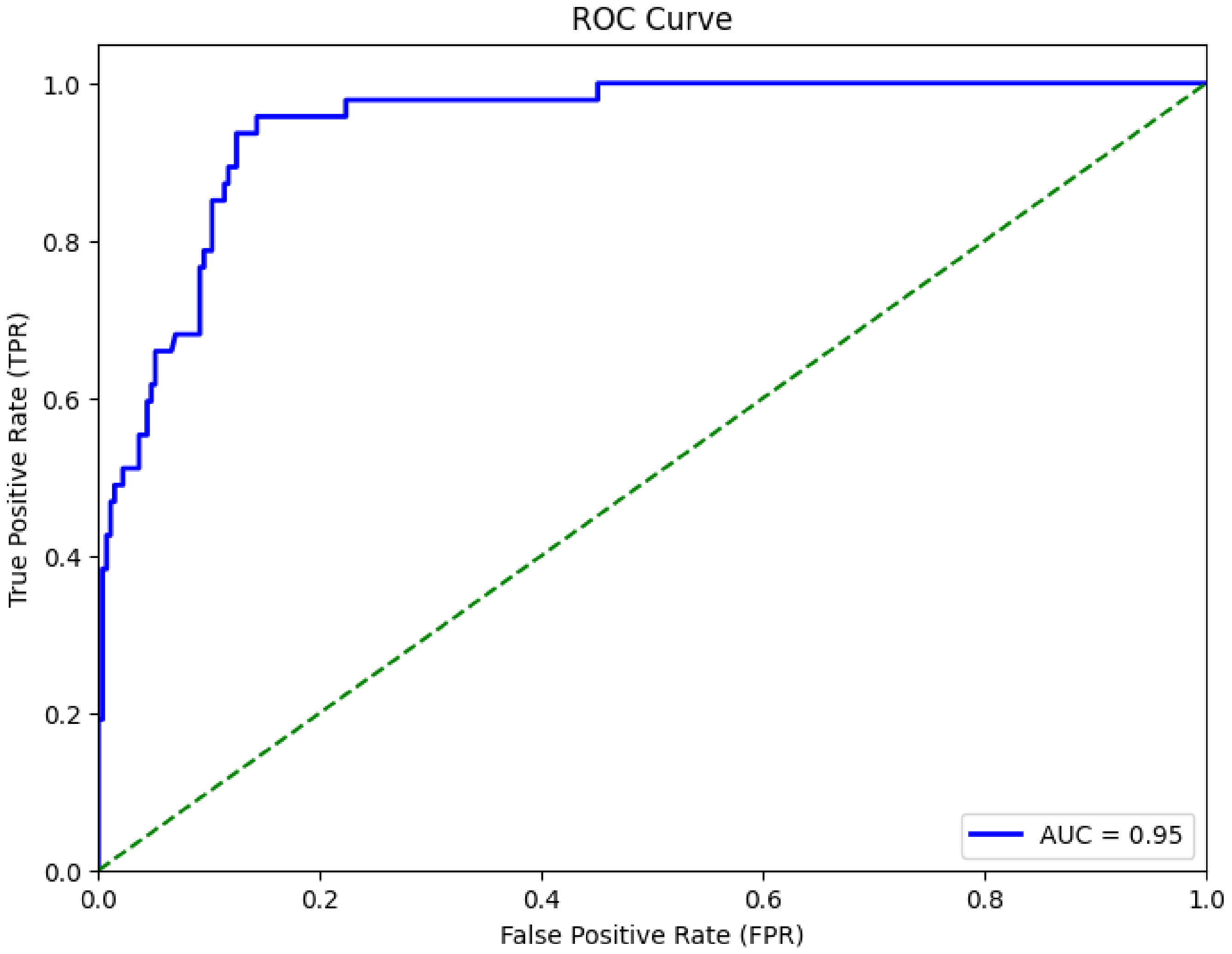


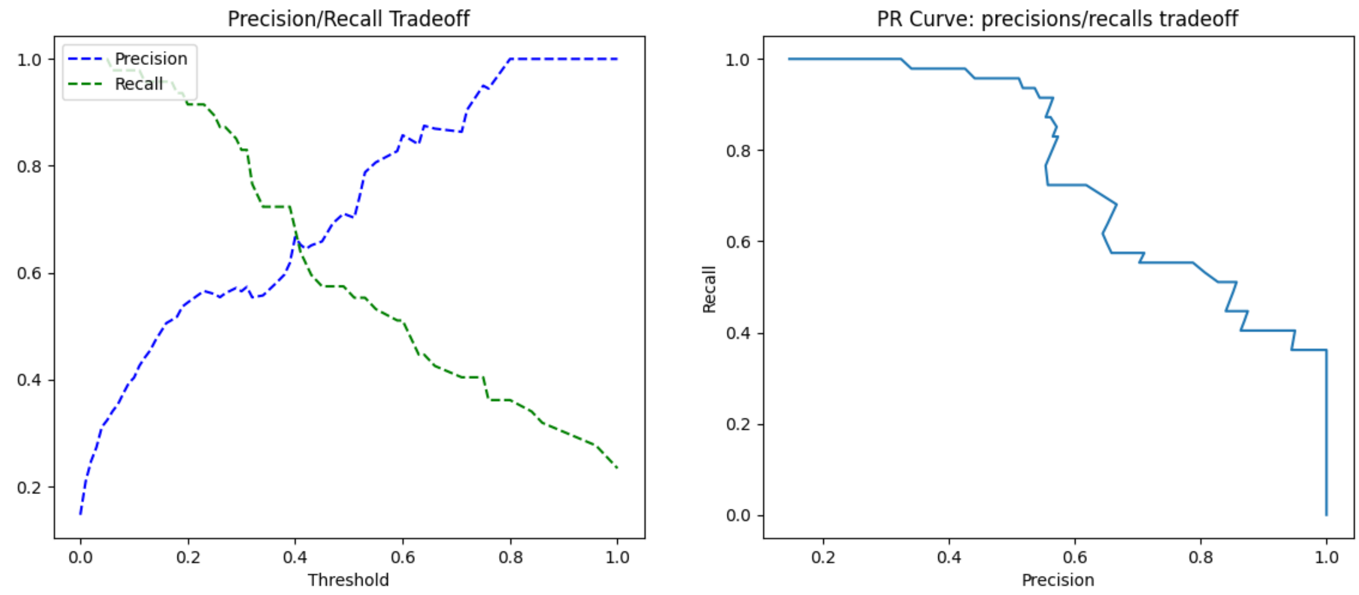
Next, we examined the data through a different model - Random Forest. Considering the numerous correlations across the features, the low F1 score in the logistic regression may be attributed to more non-linear and complex relationships across features and Attrition. There may also be more noisy data in this case as our hyperparameter tuning did not pose much benefit to the linear regression. This prompted us to utilize the Random Forest model to see if we can deal better with the possible presence of complex relationships and noisy data.

RANDOM FOREST: A Random Forest Classifier model is a meta estimator that fits a number of decision tree classifiers on various sub-samples of the dataset and uses averaging to improve the predictive accuracy and control over-fitting. Trees in the forest use the best split strategy, i.e., equivalent to passing splitter="best" to the underlying DecisionTreeRegressor. The sub-sample size is controlled with the max_samples parameter if bootstrap=True (default), otherwise, the whole dataset is used to build each tree. Here we went straight to tune the model by performing hyperparameter tuning via GridSearchCV to improve the result.

```
Fitting 3 folds for each of 81 candidates, totalling 243 fits
```

```
Best Parameters (Random Forest): {'max_depth': 15, 'min_samples_leaf': 1, 'min_sampl
es_split': 2, 'n_estimators': 500}
Accuracy (Tuned Random Forest): 0.89375
Confusion Matrix (Tuned Random Forest):
 [[255  18]
 [ 16  31]]
Classification Report (Tuned Random Forest):
               precision    recall  f1-score   support

           0       0.94      0.93      0.94       273
           1       0.63      0.66      0.65        47

    accuracy                           0.89       320
   macro avg       0.79      0.80      0.79       320
weighted avg       0.90      0.89      0.89       320
```





The tuned Random Forest model (with StratifiedKFold) has an F1-score of 0.65 and the AUC-ROC score of 0.95. The hyperparameter tuned Random Forest model has higher F1-score of 0.65 (the regular RF has F1-score of 0.62) and the AUC-ROC score is about the same 0.9485620762216507 vs 0.9457174031642117

Our Random Forest model perform much better than the Logistic Regression model, suggesting there is a complex, non-linear relationship between the features and hi quality target variable.

Discussion:

Metrics selection:
The default scoring for classification is accuracy (correct classification count / total count). However, we are facing a skewed/imbalance classes classification problem, so accuracy may not the best measure. We also have precision, recall, and F1, which are more suitable for this case.

However, it is good to have a single real number evaluation metric in order to compare different models objectively. We want to be confident that the wine will be of superior quality so the customer will not be disappointed. In this scenario, minimizing false positives (labeling a mediocre wine as high quality) is more important than potentially missing a few good wines (false negatives), which means precision is more important than recall.

However, we will not state that "no wine is worthy", either.

It seems at first for small brewery that precision is more important than recall

Our solution is to adjust the threshold in our calculation of precision, recall, and F1 score.

Adjusting the threshold in precision-recall-F1 score calculation is important because it allows us to directly control the balance between precision and recall, optimizing the F1 score for our specific use case by finding the "sweet spot" where both metrics are adequately balanced depending on whether we prioritize identifying all positive cases (high recall) or minimizing false positives (high precision). .

Key points about threshold adjustment:

- **Impact on Precision and Recall:**

A higher threshold leads to higher precision (fewer false positives) but lower recall (more false negatives), while a lower threshold increases recall but decreases precision.

- **Importance for Imbalanced Datasets:**

  In datasets where one class is significantly smaller than the other, adjusting the threshold can be crucial to accurately evaluate the model's performance on the minority class.

- **Precision-Recall Curve:**

  Visualizing the precision-recall curve helps identify the optimal threshold by showing how precision and recall change as the threshold varies.

- **F1 Score Optimization:**
  By adjusting the threshold, you can find the point where the F1 score (harmonic mean of precision and recall) is maximized, which represents the best balance between both metrics for your application.