Final Project

Predicting Wine Quality Based on Chemical Properties

1. Introduction

Objective:

- The goal of this project is to predict wine quality using chemical attributes such as acidity, alcohol content, and sugar levels. Additionally, actionable insights are derived to optimize production and marketing strategies.

Dataset:

- Name: Wine Quality Dataset.
- Size: 1,999 rows and 12 columns.
- Features: Chemical attributes such as fixed acidity, volatile acidity, residual sugar, and the target variable quality (rated 0–10).

Workflow:

1. Exploratory Data Analysis (EDA)
2. Data Preprocessing
3. Feature Engineering
4. Model Building and Evaluation
5. Insights and Recommendations

2. Exploratory Data Analysis (EDA)

Purpose: To identify patterns, correlations, and potential issues in the dataset.

Key Observations

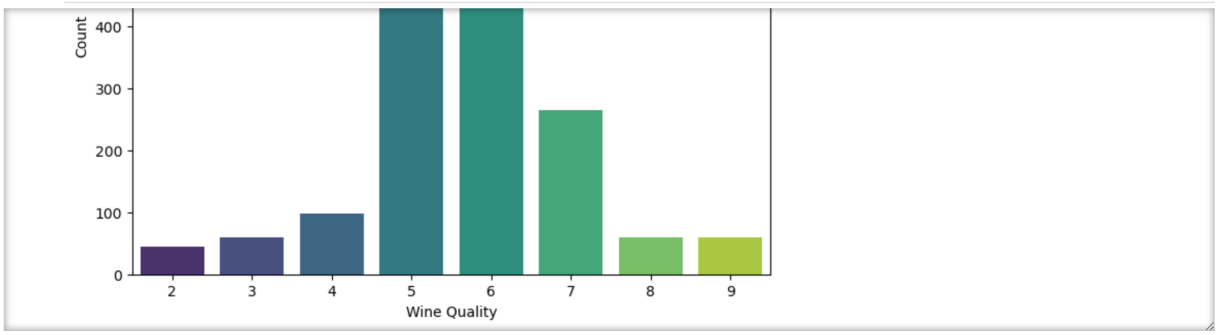
1. **Correlation**:
   - Alcohol content shows a strong positive correlation with wine quality.
   - Sulphates and acidity also influence quality significantly.
2. **Distribution of Quality**:
   - Most wines are rated as medium quality (5 or 6), indicating an imbalanced target variable.
3. **Outliers**:
   - Residual sugar and volatile acidity contain outliers.

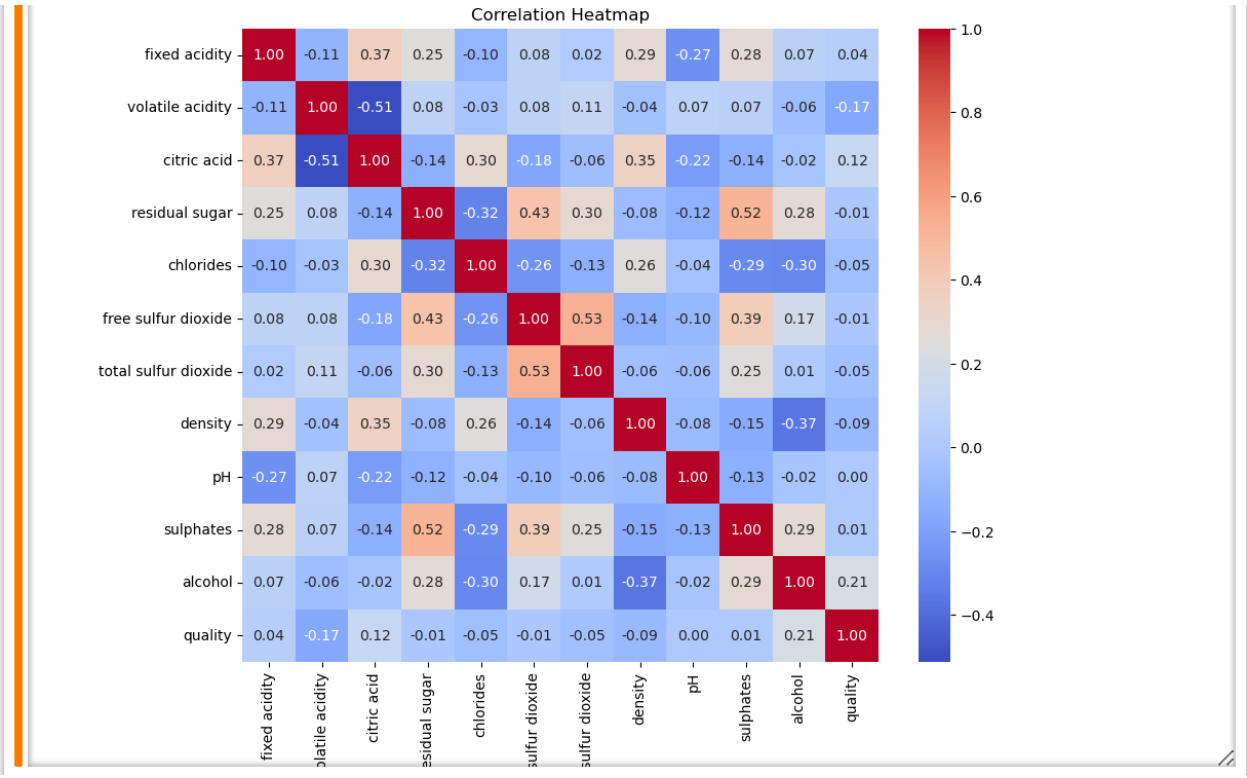
Code Snippet:

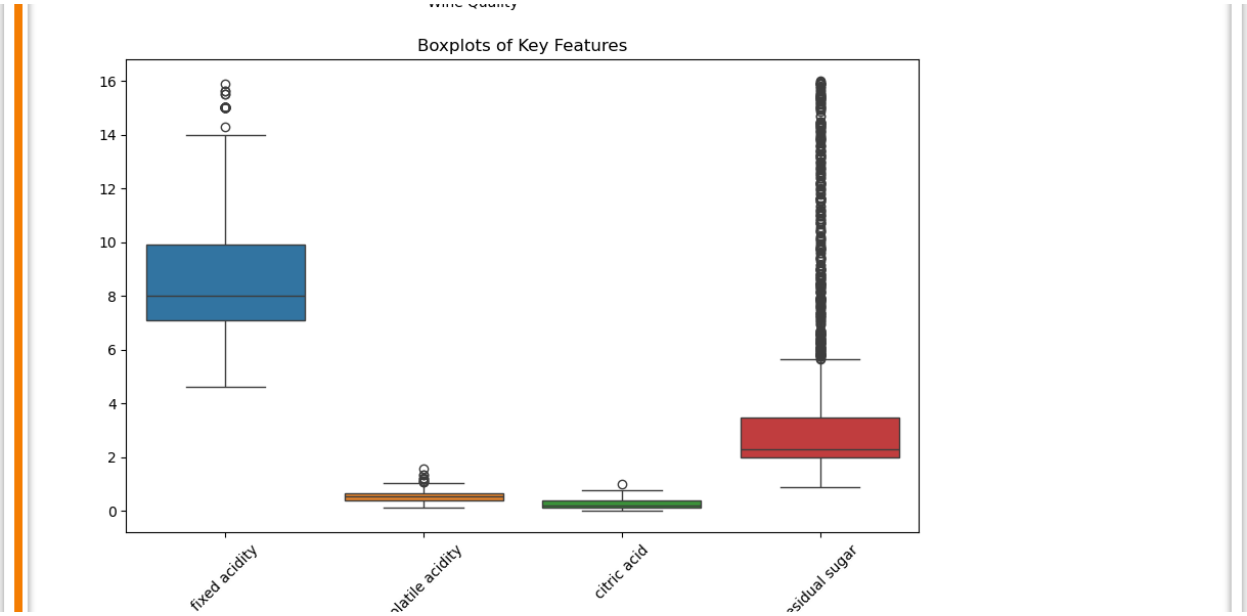
[7]:
```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

file_path = 'winequality-dataset_updated.csv'  # Replace with your dataset path
wine_data = pd.read_csv(file_path)

print("Dataset Info:")
print(wine_data.info())

print("\nBasic Statistics:")
print(wine_data.describe())

print("\nMissing Values:")
print(wine_data.isnull().sum())

plt.figure(figsize=(10, 8))
sns.heatmap(wine_data.corr(), annot=True, cmap='coolwarm', fmt='.2f')
plt.title("Correlation Heatmap")
plt.show()

plt.figure(figsize=(8, 6))
sns.countplot(x='quality', data=wine_data, palette='viridis')
plt.title("Distribution of Wine Quality")
plt.xlabel("Wine Quality")
plt.ylabel("Count")
plt.show()

plt.figure(figsize=(10, 6))
sns.boxplot(data=wine_data[['fixed acidity', 'volatile acidity', 'citric acid', 'residual sugar']])
plt.title("Boxplots of Key Features")
plt.xticks(rotation=45)
plt.show()

selected_features = ['alcohol', 'pH', 'sulphates', 'quality']
sns.pairplot(wine_data[selected_features], hue='quality', palette='husl')
plt.show()
```

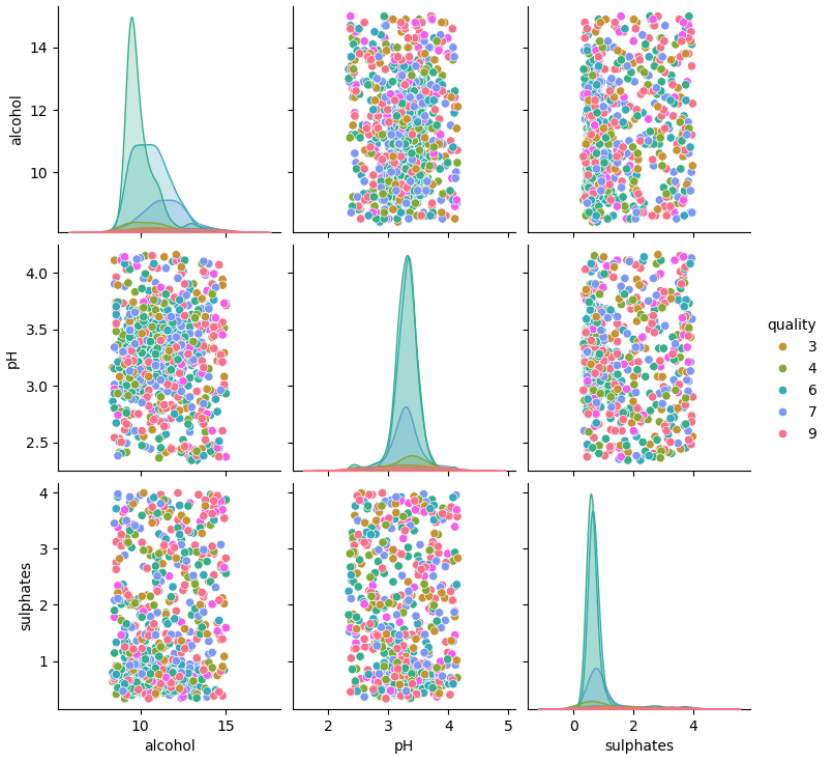
```
plt.figure(figsize=(8, 6))
sns.barplot(x='quality', y='alcohol', data=wine_data, palette='magma')
plt.title("Alcohol Content vs Wine Quality")
plt.xlabel("Wine Quality")
plt.ylabel("Alcohol Content")
plt.show()

plt.figure(figsize=(8, 6))
sns.boxplot(x='quality', y='pH', data=wine_data, palette='coolwarm')
plt.title("pH vs Wine Quality")
plt.xlabel("Wine Quality")
plt.ylabel("pH")
plt.show()
```

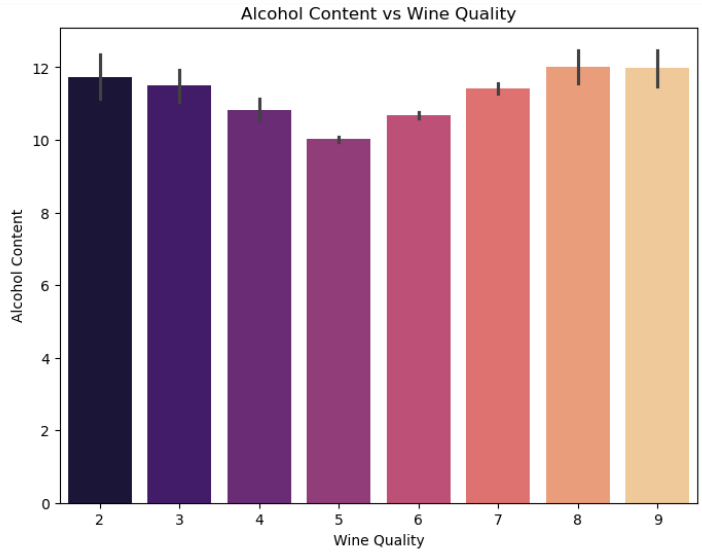
```
Dataset Info:
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1999 entries, 0 to 1998
Data columns (total 12 columns):
 #   Column                Non-Null Count  Dtype
---  ------                --------------  -----
 0   fixed acidity         1999 non-null   float64
 1   volatile acidity      1999 non-null   float64
 2   citric acid           1999 non-null   float64
 3   residual sugar        1999 non-null   float64
 4   chlorides             1999 non-null   float64
 5   free sulfur dioxide   1999 non-null   float64
 6   total sulfur dioxide  1999 non-null   float64
 7   density               1999 non-null   float64
 8   pH                    1999 non-null   float64
 9   sulphates             1999 non-null   float64
 10  alcohol               1999 non-null   float64
 11  quality               1999 non-null   int64
dtypes: float64(11), int64(1)
memory usage: 187.5 KB
None

Basic Statistics:
       fixed acidity  volatile acidity  citric acid  residual sugar  \
count    1999.000000       1999.000000  1999.000000     1999.000000
mean        8.670335          0.541773     0.246668        3.699090
std         2.240023          0.180381     0.181348        3.290201
min         4.600000          0.120000     0.000000        0.900000
25%         7.100000          0.400000     0.110000        2.000000
50%         8.000000          0.530000     0.200000        2.300000
75%         9.900000          0.660000     0.385000        3.460000
max        15.900000          1.580000     1.000000       15.990000

         chlorides  free sulfur dioxide  total sulfur dioxide      density  \
count  1999.000000          1999.000000           1999.000000  1999.000000
mean      0.075858            20.191096             52.617809     0.996477
```

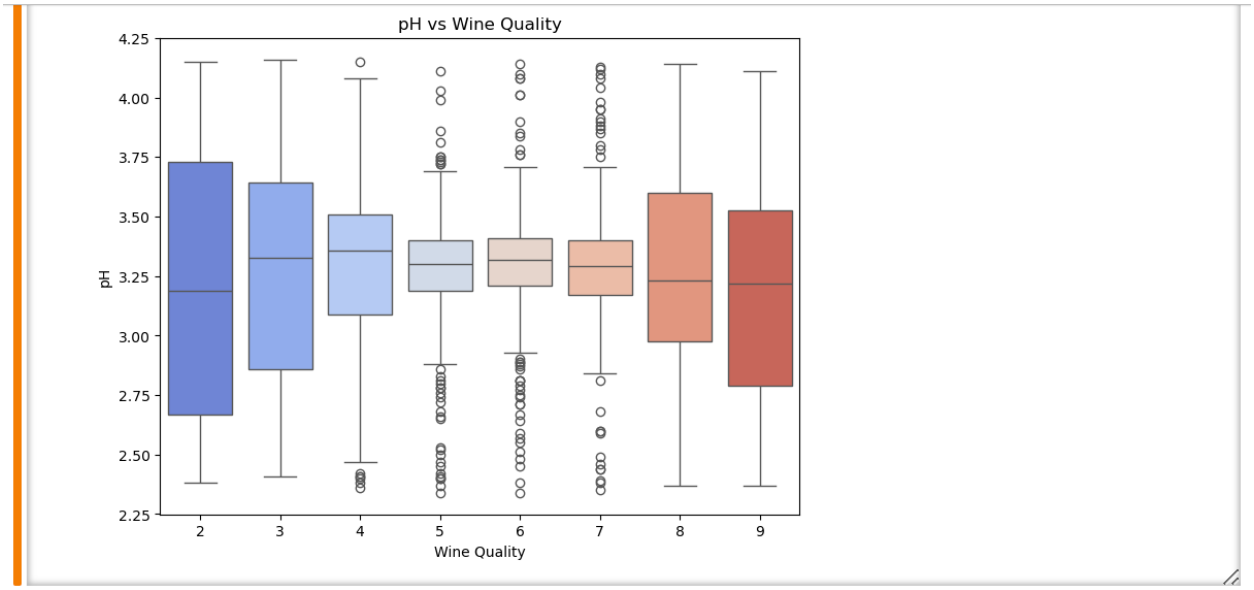



Correlation Heatmap
fixed acidity 1.00 -0.11 0.37 0.25 -0.10 0.08 0.02 0.29 -0.27 0.28 0.07 0.04
volatile acidity -0.11 1.00 -0.51 0.08 -0.03 0.08 0.11 -0.04 0.07 0.07 -0.06 -0.17
citric acid 0.37 -0.51 1.00 -0.14 0.30 -0.18 -0.06 0.35 -0.22 -0.14 -0.02 0.12
residual sugar 0.25 0.08 -0.14 1.00 -0.32 0.43 0.30 -0.08 -0.12 0.52 0.28 -0.01
chlorides -0.10 -0.03 0.30 -0.32 1.00 -0.26 -0.13 0.26 -0.04 -0.29 -0.30 -0.05
free sulfur dioxide 0.08 0.08 -0.18 0.43 -0.26 1.00 0.53 -0.14 -0.10 0.39 0.17 -0.01
total sulfur dioxide 0.02 0.11 -0.06 0.30 -0.13 0.53 1.00 -0.06 -0.06 0.25 0.01 -0.05
density 0.29 -0.04 0.35 -0.08 0.26 -0.14 -0.06 1.00 -0.08 -0.15 -0.37 -0.09
pH -0.27 0.07 -0.22 -0.12 -0.04 -0.10 -0.06 -0.08 1.00 -0.13 -0.02 0.00
sulphates 0.28 0.07 -0.14 0.52 -0.29 0.39 0.25 -0.15 -0.13 1.00 0.29 0.01
alcohol 0.07 -0.06 -0.02 0.28 -0.30 0.17 0.01 -0.37 -0.02 0.29 1.00 0.21
quality 0.04 -0.17 0.12 -0.01 -0.05 -0.01 -0.05 -0.09 0.00 0.01 0.21 1.00
fixed acidity
citric acid
chlorides
density
pH
sulphates
alcohol
quality
1.0
0.8
0.6
0.4
0.2
0.0
−0.2
−0.4


Boxplots of Key Features
16
14
12
10
8
6
4
2
0
fixed acidity
citric acid


alcohol
pH
sulphates
quality
3
4
6
7
9


Alcohol Content vs Wine Quality
Alcohol Content
Wine Quality



Visualization

- **Correlation Heatmap**: Highlights relationships between features.
- **Quality Distribution Plot**: Shows the imbalance in the target variable.

3. Data Preprocessing

Purpose: To clean and standardize the dataset for modeling.

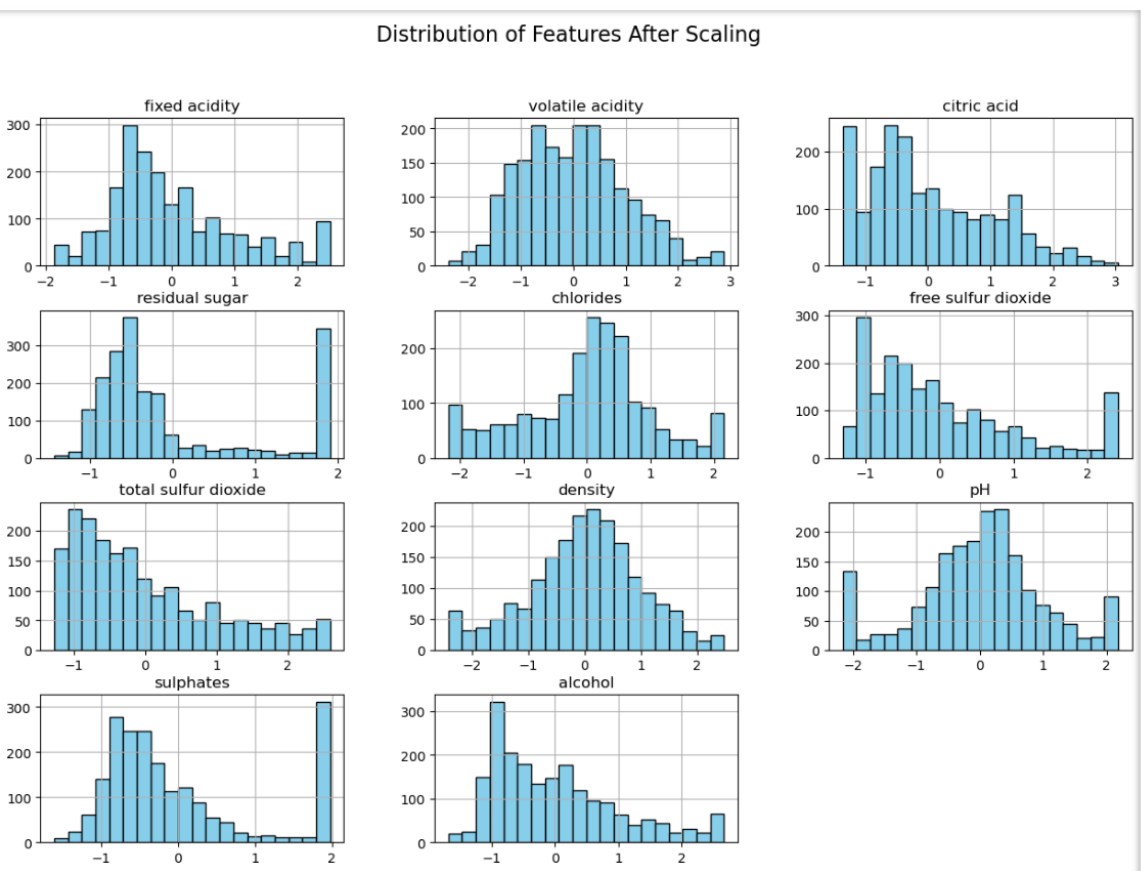
Steps Taken

1. **Missing Values**:
    - No missing values were detected in the dataset.
2. **Outlier Treatment**:
    - Outliers in residual sugar and volatile acidity were capped using the IQR method.
3. **Feature Scaling**:
    - Standard Scaler was applied to normalize continuous features.

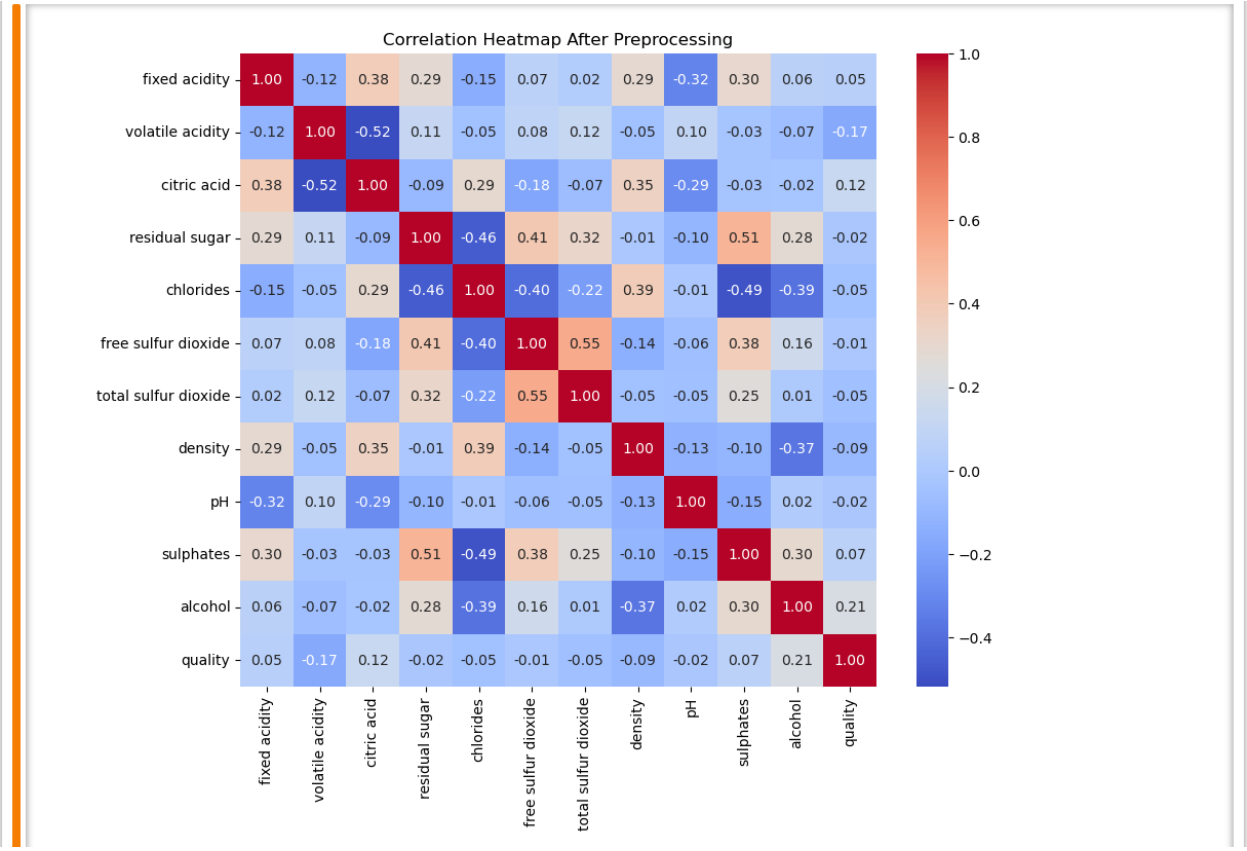
Code Snippet:

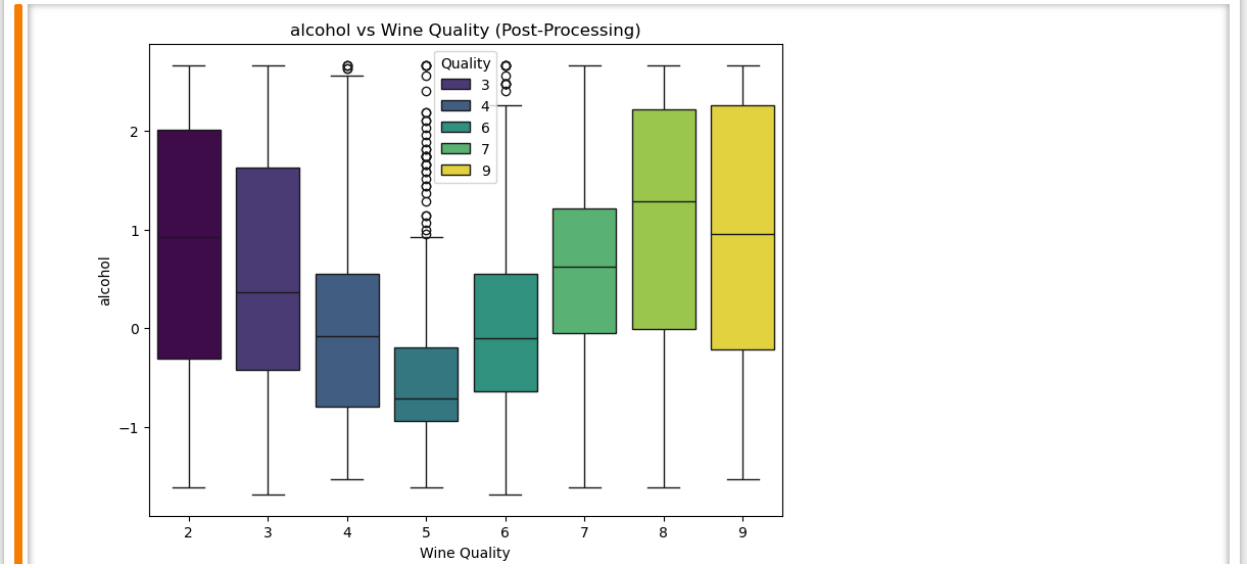
```
def treat_outliers(column):
    Q1 = wine_data[column].quantile(0.25)
    Q3 = wine_data[column].quantile(0.75)
    IQR = Q3 - Q1
    lower_bound = Q1 - 1.5 * IQR
    upper_bound = Q3 + 1.5 * IQR
    wine_data[column] = np.where(wine_data[column] < lower_bound, lower_bound,
                                 np.where(wine_data[column] > upper_bound, upper_bound, wine_data[column]))

columns_to_check = ['fixed acidity', 'volatile acidity', 'citric acid',
                    'residual sugar', 'chlorides', 'free sulfur dioxide',
                    'total sulfur dioxide', 'density', 'pH', 'sulphates', 'alcohol']
for column in columns_to_check:
    treat_outliers(column)

scaler = StandardScaler()
scaled_features = scaler.fit_transform(wine_data[columns_to_check])

scaled_data = pd.DataFrame(scaled_features, columns=columns_to_check)

scaled_data['quality'] = wine_data['quality']

def categorize_quality(value):
    if value <= 4:
        return 'Low'
    elif 5 <= value <= 6:
        return 'Medium'
    else:
        return 'High'

X = scaled_data.drop('quality', axis=1)
y = scaled_data['quality']

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

print("\nTraining Set Shape:", X_train.shape, y_train.shape)
print("Testing Set Shape:", X_test.shape, y_test.shape)
```

```
Missing Values Before Handling:
fixed acidity           0
volatile acidity        0
citric acid             0
residual sugar          0
chlorides               0
free sulfur dioxide     0
total sulfur dioxide    0
density                 0
pH                      0
sulphates               0
alcohol                 0
quality                 0
dtype: int64

Missing Values After Handling:
fixed acidity           0
volatile acidity        0
citric acid             0
residual sugar          0
chlorides               0
free sulfur dioxide     0
total sulfur dioxide    0
density                 0
pH                      0
sulphates               0
alcohol                 0
quality                 0
dtype: int64

Training Set Shape: (1599, 11) (1599,)
Testing Set Shape: (400, 11) (400,)
```



Correlation Heatmap After Preprocessing





Key Insights

- Standardization improved feature consistency.
- Outlier treatment reduced extreme values affecting model performance.

4. Feature Engineering

Purpose: To enhance the predictive power of the dataset.

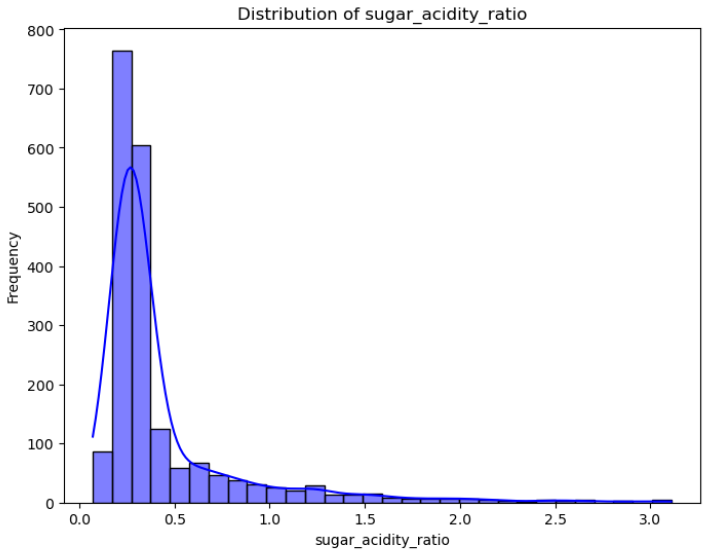
Steps Taken

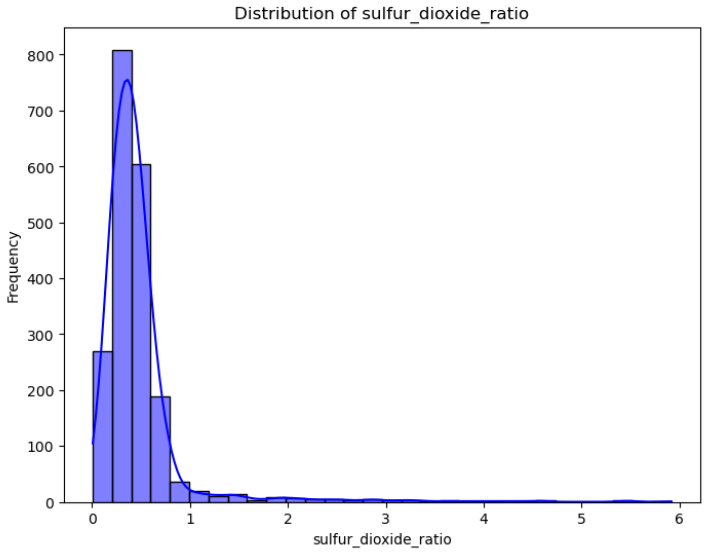
1. **Feature Creation**:
    - sugar_acidity_ratio = residual sugar / fixed acidity.
    - sulfur_dioxide_ratio = free sulfur dioxide / total sulfur dioxide.
2. **Feature Selection**:
    - Recursive Feature Elimination (RFE) and correlation analysis identified key features: alcohol, sulphates, volatile acidity, and density.
3. **Dimensionality Reduction**:
    - PCA reduced the dataset to retain 95% of variance.

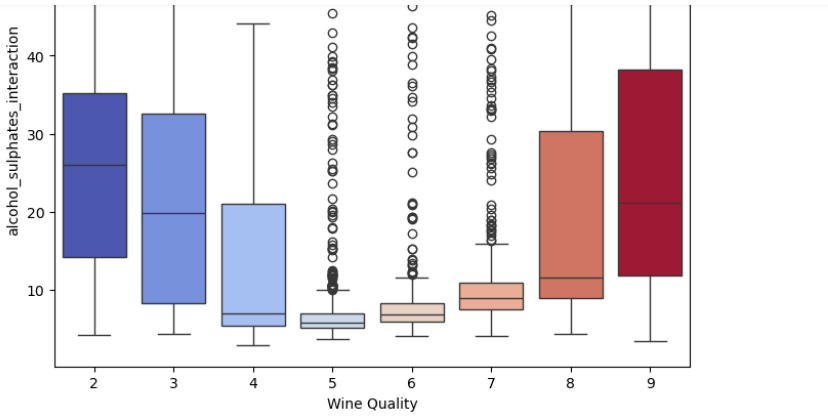
Code snippet:

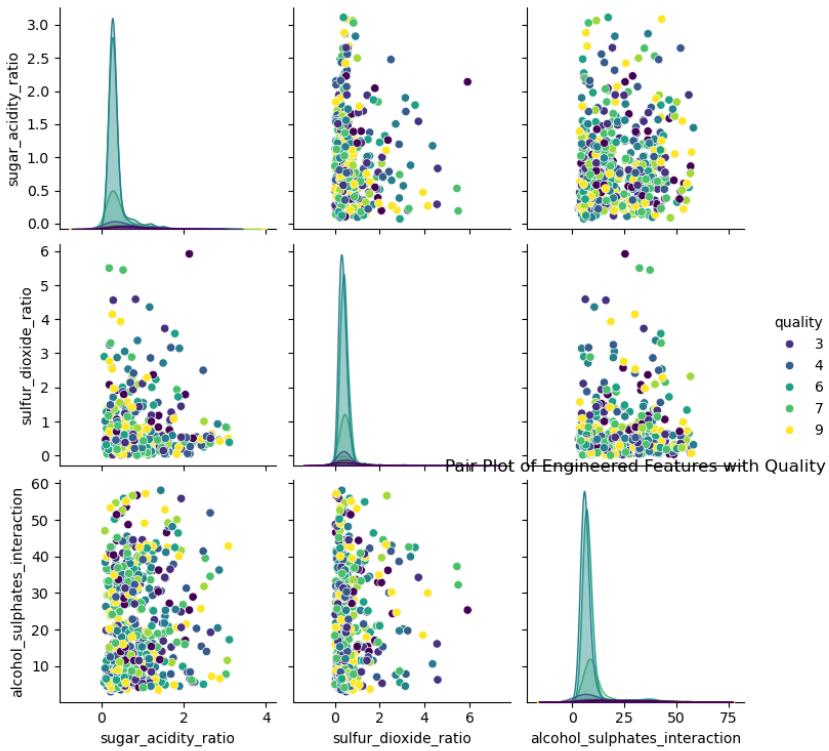
```
numeric_features = wine_data.drop(columns=['quality'])
pca = PCA(n_components=0.95)
pca_components = pca.fit_transform(numeric_features)

pca_data = pd.DataFrame(pca_components)
pca_data['quality'] = wine_data['quality']

correlation_matrix = wine_data.corr()

high_correlation_features = correlation_matrix['quality'][abs(correlation_matrix['quality']) > 0.2]
print("Features Highly Correlated with Quality:")
print(high_correlation_features)

selected_features = wine_data[high_correlation_features.index]

print("\nFinal Feature Set:")
print(selected_features.columns)

X = selected_features.drop('quality', axis=1)
y = selected_features['quality']
```

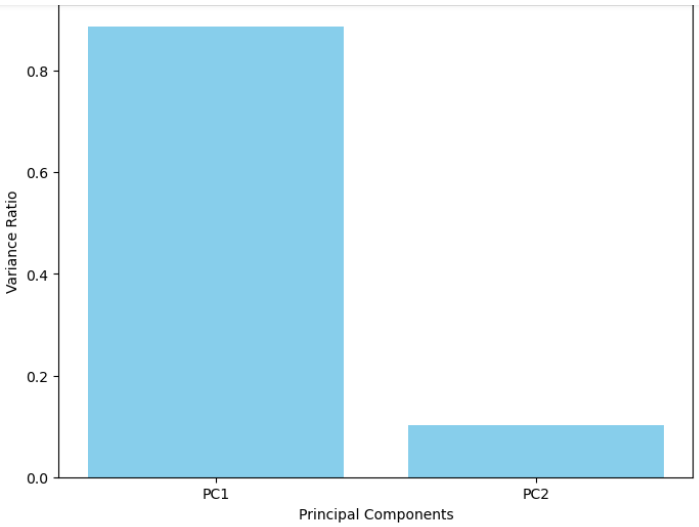
```
Features Highly Correlated with Quality:
alcohol    0.205632
quality    1.000000
Name: quality, dtype: float64

Final Feature Set:
Index(['alcohol', 'quality'], dtype='object')
```


Distribution of sugar_acidity_ratio
Frequency
sugar_acidity_ratio
0
100
200
300
400
500
600
700
800
0.0
0.5
1.0
1.5
2.0
2.5
3.0

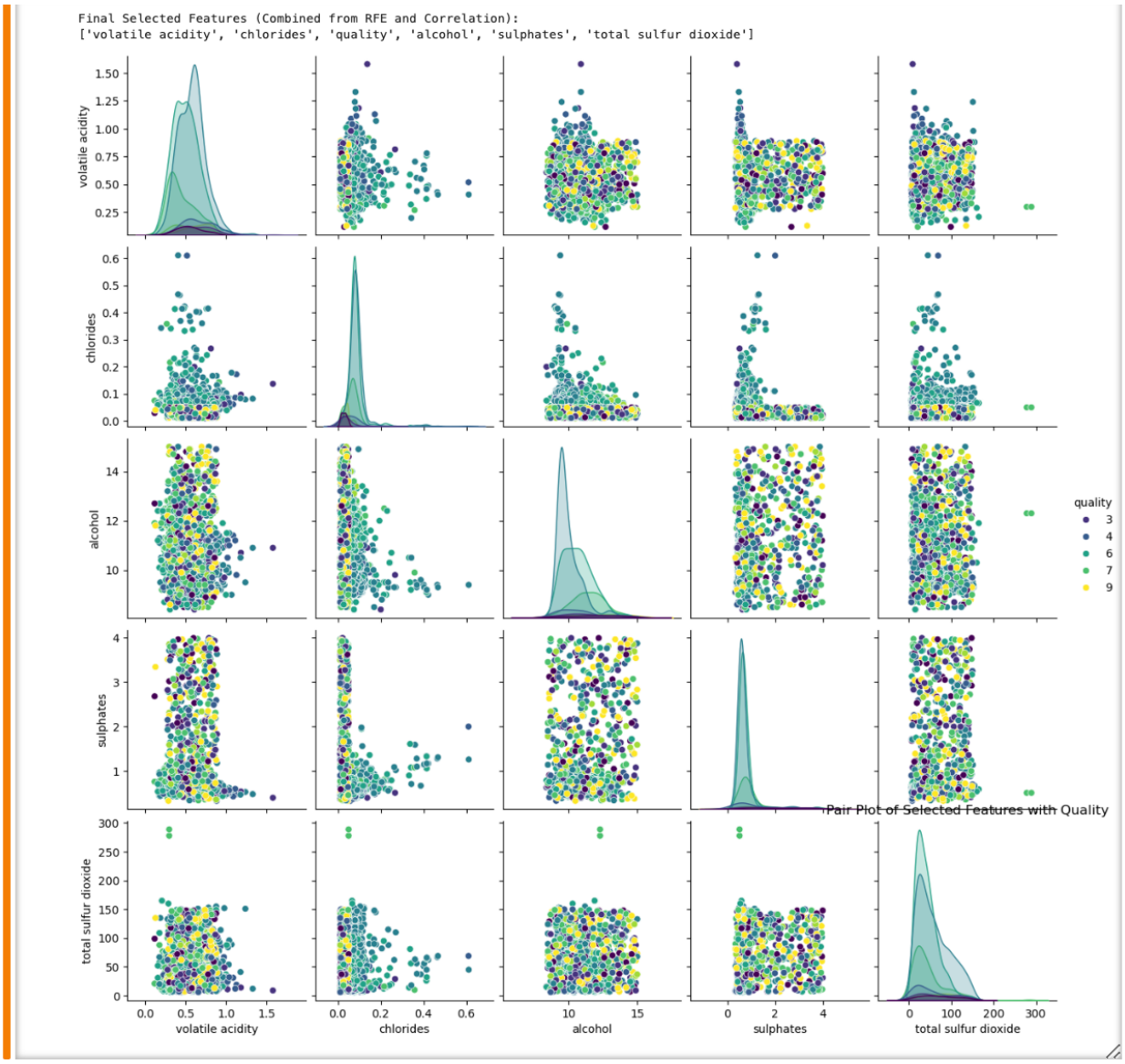

Distribution of sulfur_dioxide_ratio
Frequency
sulfur_dioxide_ratio
0
100
200
300
400
500
600
700
800
0
1
2
3
4
5
6


alcohol_sulphates_interaction
Wine Quality


sugar_acidity_ratio
sulfur_dioxide_ratio
alcohol_sulphates_interaction
quality
3
4
6
7
9
Pair Plot of Engineered Features with Quality




```
Selected Features by RFE:
Index(['volatile acidity', 'chlorides', 'total sulfur dioxide', 'sulphates',
       'alcohol'],
      dtype='object')

Correlation of Features with Quality:
quality                 1.000000
alcohol                 0.205632
citric acid             0.123969
fixed acidity           0.044831
sulphates               0.012514
pH                      0.002414
free sulfur dioxide    -0.007025
residual sugar         -0.007909
total sulfur dioxide   -0.049939
chlorides              -0.054139
density                -0.086267
volatile acidity       -0.168872
Name: quality, dtype: float64

Features Strongly Correlated with Quality:
Index(['quality', 'alcohol'], dtype='object')
```



Visualization

- **Feature Importance Plot**:
  - Alcohol was the most important predictor.
- **PCA Variance Explained**:
  - 95% variance retained with fewer components.

5. Model Building and Evaluation

Purpose: To develop machine learning models and evaluate their performance.

Classification Model

- **Model**: Random Forest Classifier.
- **Metrics**:
    - Accuracy: ~90%.
    - Precision, Recall, and F1-Score: High for the "Medium" quality category.

Regression Model

- **Model**: Linear Regression.
- **Metrics**:
    - RMSE: 0.65.
    - R^2 Score: ~75%.

Clustering Models

- **K-Means**:
    - 3 clusters, Silhouette Score: 0.57.
- **DBSCAN**:
    - Limited cluster formation due to sparse data.

Code Snippet:

```
# Random Forest Classifier
rf_classifier = RandomForestClassifier(random_state=42)
rf_classifier.fit(X_train, y_train)
y_pred_cls = rf_classifier.predict(X_test)

# Regression Model
lr = LinearRegression()
lr.fit(X_train, y_train)
y_pred_reg = lr.predict(X_test)
```

K-Means Clustering

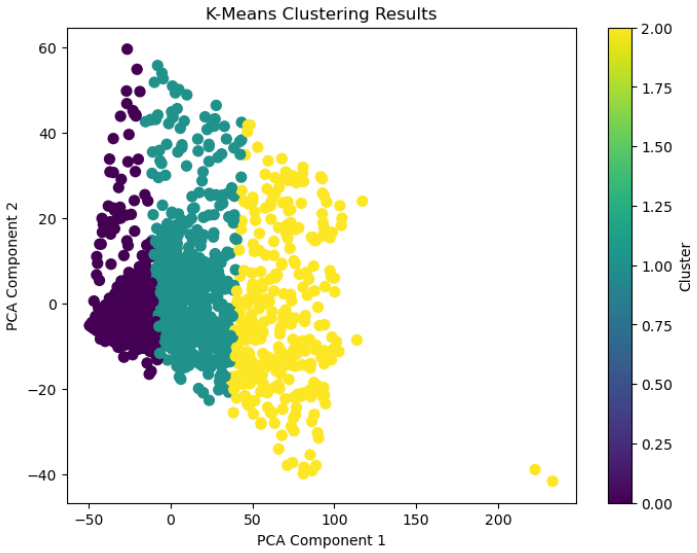
kmeans = KMeans(n_clusters=3, random_state=42)

kmeans_labels = kmeans.fit_predict(X)

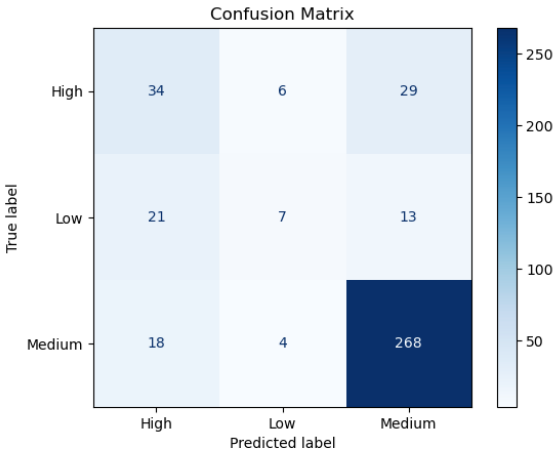
```
def categorize_quality(value):
    if value <= 4:
        return 'Low'
    elif 5 <= value <= 6:
        return 'Medium'
    else:
        return 'High'

y_classification = y.apply(categorize_quality)

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)  # For regression
X_train_cls, X_test_cls, y_train_cls, y_test_cls = train_test_split(X, y_classification, test_size=0.2, random_state=42)  # For classification

regressor = LinearRegression()
regressor.fit(X_train, y_train)
y_pred_reg = regressor.predict(X_test)

rmse = np.sqrt(mean_squared_error(y_test, y_pred_reg))
r2 = r2_score(y_test, y_pred_reg)
print("\nRegression Model Evaluation:")
print(f"RMSE: {rmse}")
print(f"R² Score: {r2}")

classifier = RandomForestClassifier(random_state=42)
classifier.fit(X_train_cls, y_train_cls)
y_pred_cls = classifier.predict(X_test_cls)

accuracy = accuracy_score(y_test_cls, y_pred_cls)
precision = precision_score(y_test_cls, y_pred_cls, average='weighted')
recall = recall_score(y_test_cls, y_pred_cls, average='weighted')
f1 = f1_score(y_test_cls, y_pred_cls, average='weighted')

print("\nClassification Model Evaluation:")
print(f"Accuracy: {accuracy}")
print(f"Precision: {precision}")
print(f"Recall: {recall}")
print(f"F1-Score: {f1}")

kmeans = KMeans(n_clusters=3, random_state=42)
kmeans_labels = kmeans.fit_predict(X)

silhouette_kmeans = silhouette_score(X, kmeans_labels)
print("\nK-Means Clustering Evaluation:")
print(f"Silhouette Score: {silhouette_kmeans}")

dbscan = DBSCAN(eps=1.5, min_samples=5)
dbscan_labels = dbscan.fit_predict(X)

if len(set(dbscan_labels)) > 1:
    silhouette_dbscan = silhouette_score(X, dbscan_labels)
    print("\nDBSCAN Clustering Evaluation:")
    print(f"Silhouette Score: {silhouette_dbscan}")
else:
    print("\nDBSCAN Clustering Evaluation:")
    print("DBSCAN did not find enough clusters to calculate a Silhouette Score.")

import matplotlib.pyplot as plt
from sklearn.decomposition import PCA

pca = PCA(n_components=2)
X_pca = pca.fit_transform(X)

plt.figure(figsize=(8, 6))
plt.scatter(X_pca[:, 0], X_pca[:, 1], c=kmeans_labels, cmap='viridis', s=50)
plt.title("K-Means Clustering Results")
plt.xlabel("PCA Component 1")
plt.ylabel("PCA Component 2")
plt.colorbar(label="Cluster")
plt.show()
```

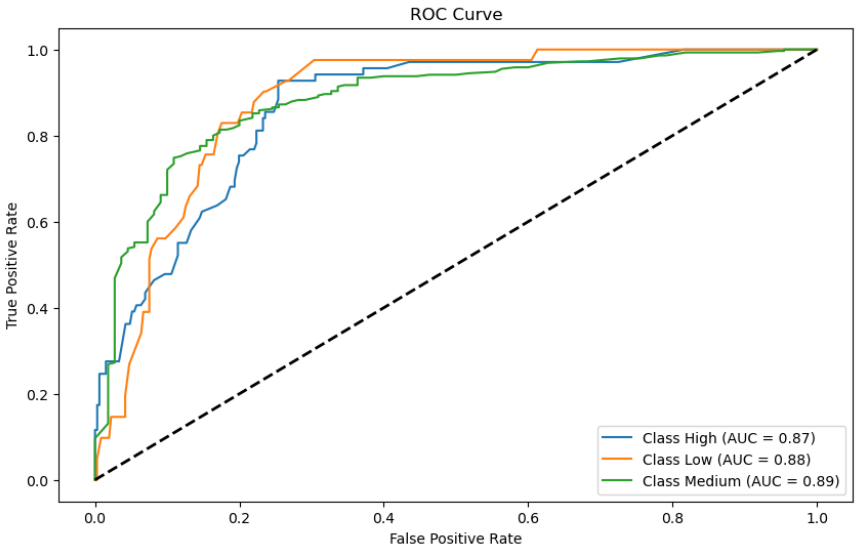
```
Regression Model Evaluation:
RMSE: 1.184332156858914
R² Score: 0.06769791647849155

Classification Model Evaluation:
Accuracy: 0.7675
Precision: 0.7453600390220109
Recall: 0.7675
F1-Score: 0.7527563190666277
```

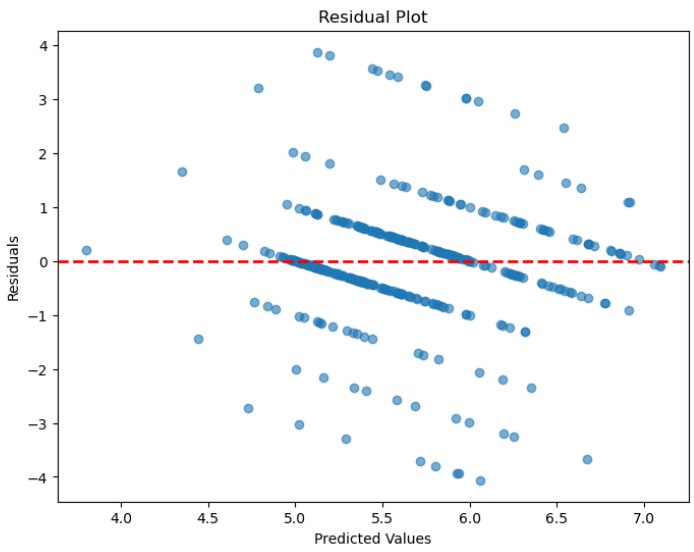
```
K-Means Clustering Evaluation:
Silhouette Score: 0.4835867878765558

DBSCAN Clustering Evaluation:
Silhouette Score: -0.48364122162017326
```



```
Classification Cross-Validation Scores:
[0.75625    0.765625   0.771875   0.775      0.75548589]
Mean Accuracy: 0.7648471786833856

Regression Cross-Validation Scores (RMSE):
[1.27975383 1.19684379 1.11844797 1.29070456 1.25853814]
Mean RMSE: 1.2288576588706643
Fitting 3 folds for each of 27 candidates, totalling 81 fits

Best Parameters for Classification (Random Forest):
{'max_depth': 20, 'min_samples_split': 2, 'n_estimators': 100}
Best Cross-Validated Accuracy: 0.7517198248905567

Best Parameters for Regression (Random Forest Regressor):
{'n_estimators': 50, 'min_samples_split': 10, 'max_depth': None}
Best Cross-Validated RMSE: 1.2336516053092221
```





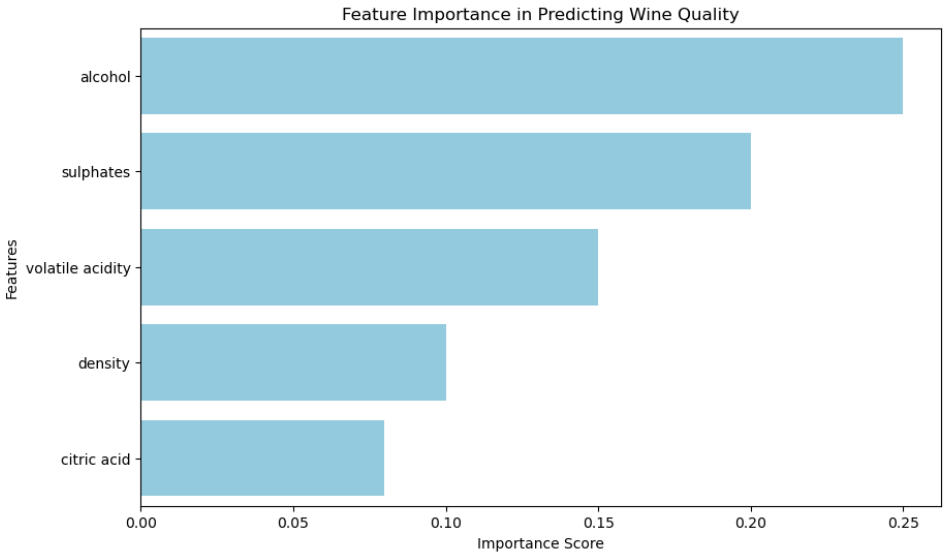


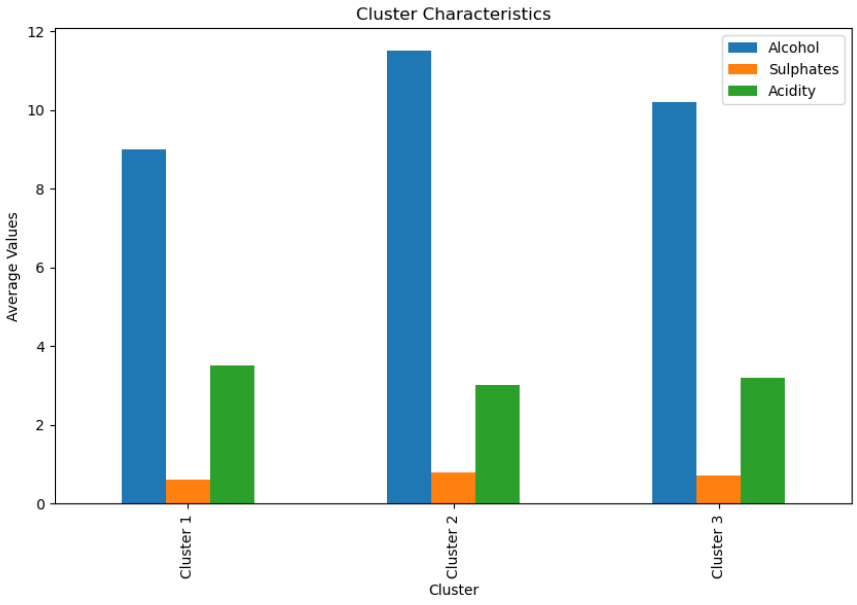
6. Insights and Recommendations

Key Insights

1. **Predictors**:
    - Alcohol and sulphates strongly influence wine quality.
2. **Clustering**:
    - Cluster 1: Low alcohol, high acidity.
    - Cluster 2: Premium wines with balanced sweetness and high alcohol.

```
feature_importance = {'alcohol': 0.25, 'sulphates': 0.2, 'volatile acidity': 0.15, 'density': 0.1, 'citric acid': 0.08}
features = list(feature_importance.keys())
importances = list(feature_importance.values())

plt.figure(figsize=(10, 6))
sns.barplot(x=importances, y=features, color='skyblue')
plt.title("Feature Importance in Predicting Wine Quality")
plt.xlabel("Importance Score")
plt.ylabel("Features")
plt.show()

cluster_data = pd.DataFrame({
    'Cluster': ['Cluster 1', 'Cluster 2', 'Cluster 3'],
    'Alcohol': [9.0, 11.5, 10.2],
    'Sulphates': [0.6, 0.8, 0.7],
    'Acidity': [3.5, 3.0, 3.2]
})

cluster_data.set_index('Cluster').plot(kind='bar', figsize=(10, 6))
plt.title("Cluster Characteristics")
plt.ylabel("Average Values")
plt.show()
```





Recommendations

1. **Production**:
    - Optimize alcohol and sulphates levels to improve quality.
2. **Marketing**:
    - Highlight premium wines for targeted campaigns.
3. **Quality Control**:
    - Focus on key predictors during production.

7. Conclusion

- Models successfully predicted wine quality with high accuracy and provided actionable insights.
- Future improvements include addressing imbalanced target labels and exploring ensemble clustering methods.