Libraries Checkpoint

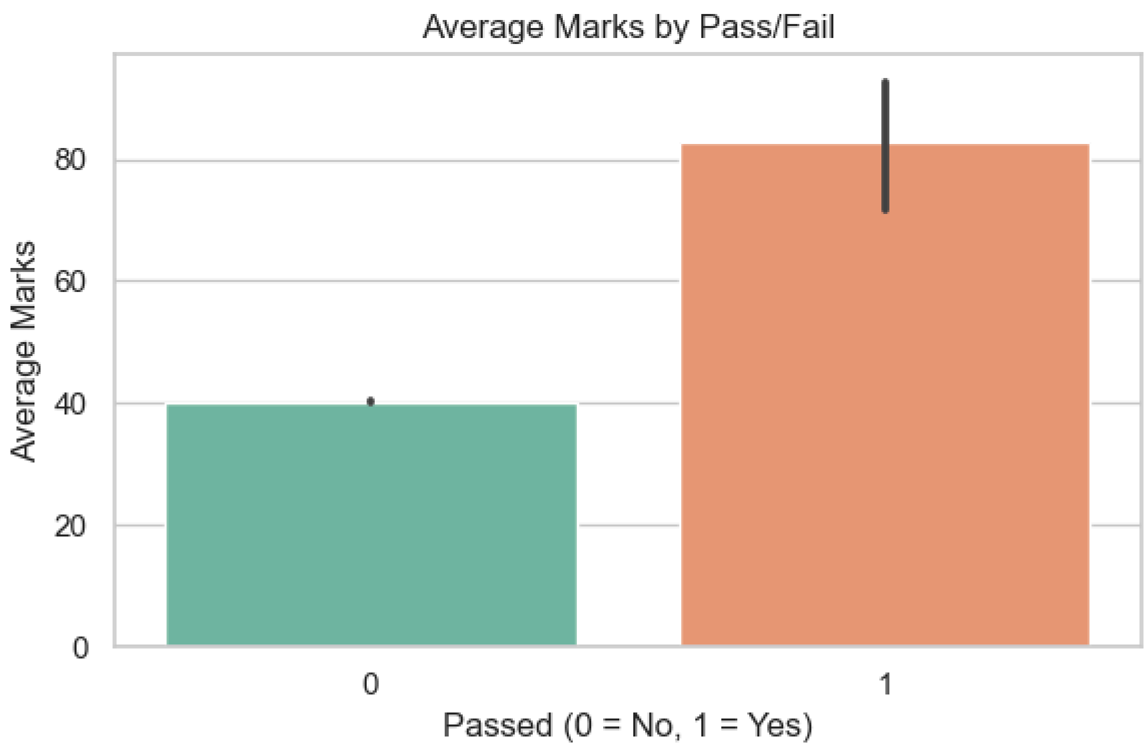
In [1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.datasets import load_iris
from sklearn.ensemble import RandomForestClassifier
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
print("All essential libraries are working fine!")
```

```
All essential libraries are working fine!
```

Level 1

In [2]:
```
# STEP 1: Create a simple DataFrame to simulate student marks in two subjects
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt

sns.set(style="whitegrid")

# Create a basic DataFrame
df_easy = pd.DataFrame({
'Maths': [35, 67, 80, 95, 42],
'Science': [45, 76, 88, 90, 39],
'Passed': [0, 1, 1, 1, 0] # Target: 1 = Pass, 0 = Fail
})

# STEP 2: Add a new feature - Average Marks
df_easy['Average'] = df_easy[['Maths', 'Science']].mean(axis=1)

# View the new DataFrame
print("Basic Marks Dataset with Engineered 'Average' Feature:")
print(df_easy)

# STEP 3: Visualize Average Marks vs Passed
plt.figure(figsize=(6, 4))
sns.barplot(x='Passed', y='Average', data=df_easy, palette='Set2')
plt.title("Average Marks by Pass/Fail")
plt.xlabel("Passed (0 = No, 1 = Yes)")
plt.ylabel("Average Marks")
plt.tight_layout()
plt.show()

# Summary:
print("✅ We added a new feature 'Average' and found that passing students have
```

```
Basic Marks Dataset with Engineered 'Average' Feature:
   Maths  Science  Passed  Average
0     35       45       0     40.0
1     67       76       1     71.5
2     80       88       1     84.0
3     95       90       1     92.5
4     42       39       0     40.5
```



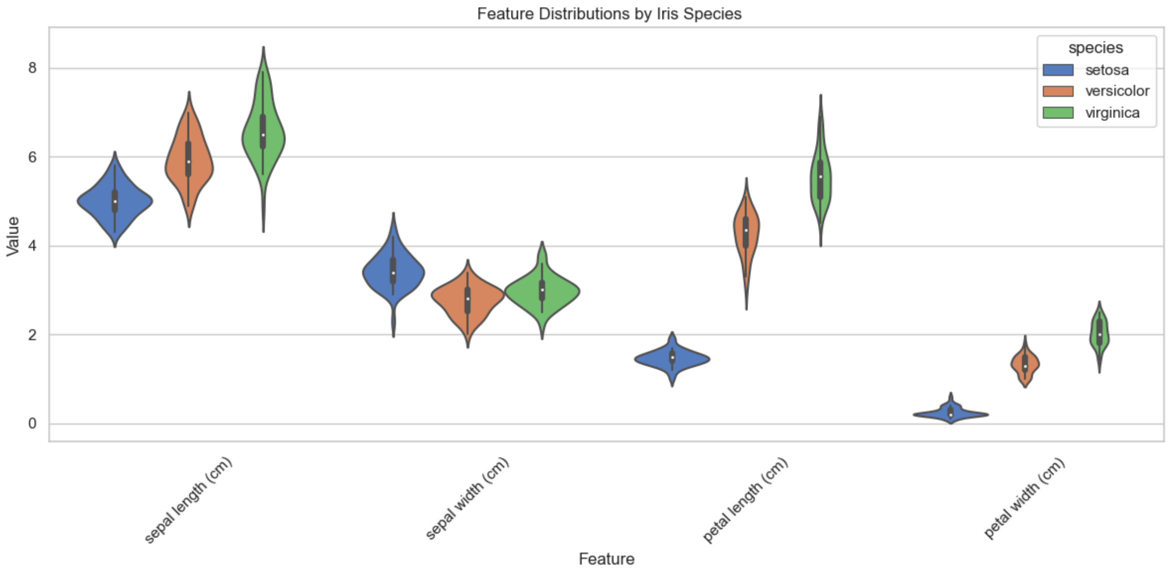
✅ We added a new feature 'Average' and found that passing students have clearly higher average scores.

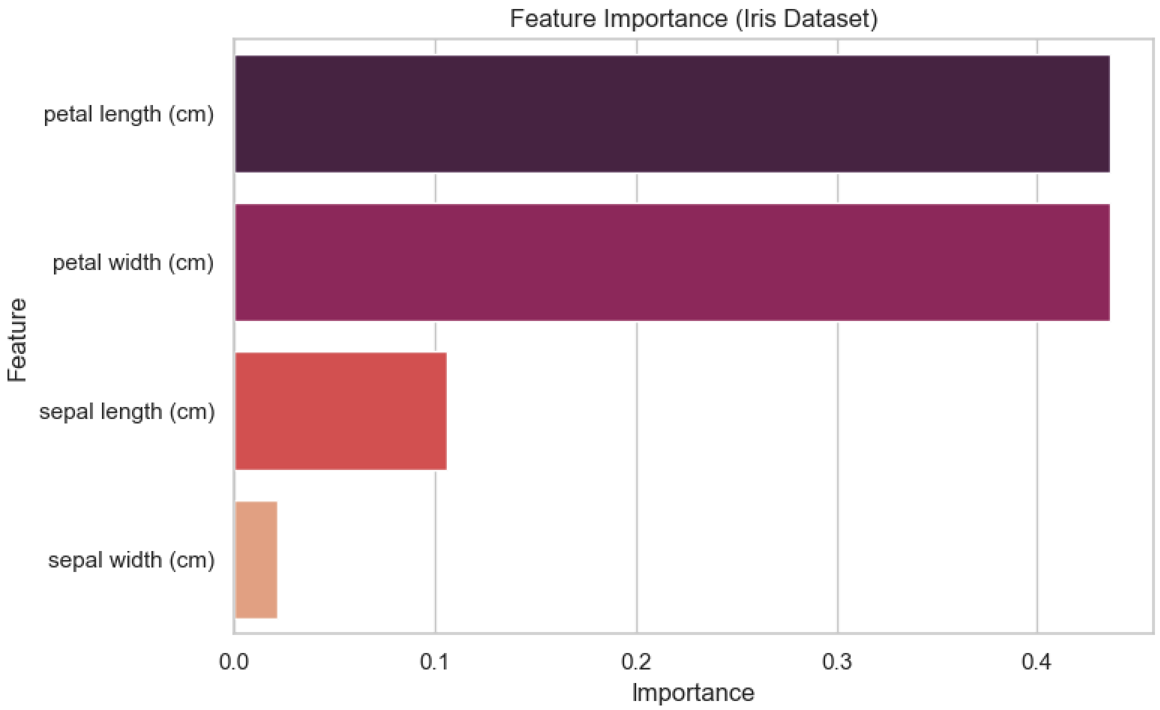
Level 2 IRIS Dataset

In [3]:
```
#  Load and Explore the Iris Dataset
from sklearn.datasets import load_iris
import numpy as np
from sklearn.ensemble import RandomForestClassifier

# Load the dataset
iris = load_iris()
df_iris = pd.DataFrame(iris.data, columns=iris.feature_names)
df_iris['species'] = pd.Categorical.from_codes(iris.target, iris.target_names)

print("Iris Dataset (First 5 Rows):")
print(df_iris.head())

#  Melt data for better multi-feature plotting
df_melted = df_iris.melt(id_vars='species', var_name='Feature', value_name='Valu

#  Violin Plot for feature distributions
plt.figure(figsize=(12, 6))
sns.violinplot(x="Feature", y="Value", hue="species", data=df_melted, palette='m
plt.title(" Feature Distributions by Iris Species")
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()

#  Random Forest for Feature Importance
X = df_iris[iris.feature_names]
y = df_iris['species']

rf_model = RandomForestClassifier(n_estimators=100, random_state=42)
rf_model.fit(X, y)
```

```
#  Extract and plot feature importance
importances = pd.DataFrame({
'Feature': X.columns,
'Importance': rf_model.feature_importances_
}).sort_values(by='Importance', ascending=False)

plt.figure(figsize=(8, 5))
sns.barplot(x='Importance', y='Feature', data=importances, palette='rocket')
plt.title("Feature Importance (Iris Dataset)")
plt.tight_layout()
plt.show()

# Summary:
print("From the violin plots and Random Forest, petal length and width are the m
```

```
Iris Dataset (First 5 Rows):
   sepal length (cm)  sepal width (cm)  petal length (cm)  petal width (cm)  \
0                5.1               3.5                1.4               0.2
1                4.9               3.0                1.4               0.2
2                4.7               3.2                1.3               0.2
3                4.6               3.1                1.5               0.2
4                5.0               3.6                1.4               0.2

  species
0  setosa
1  setosa
2  setosa
3  setosa
4  setosa
```





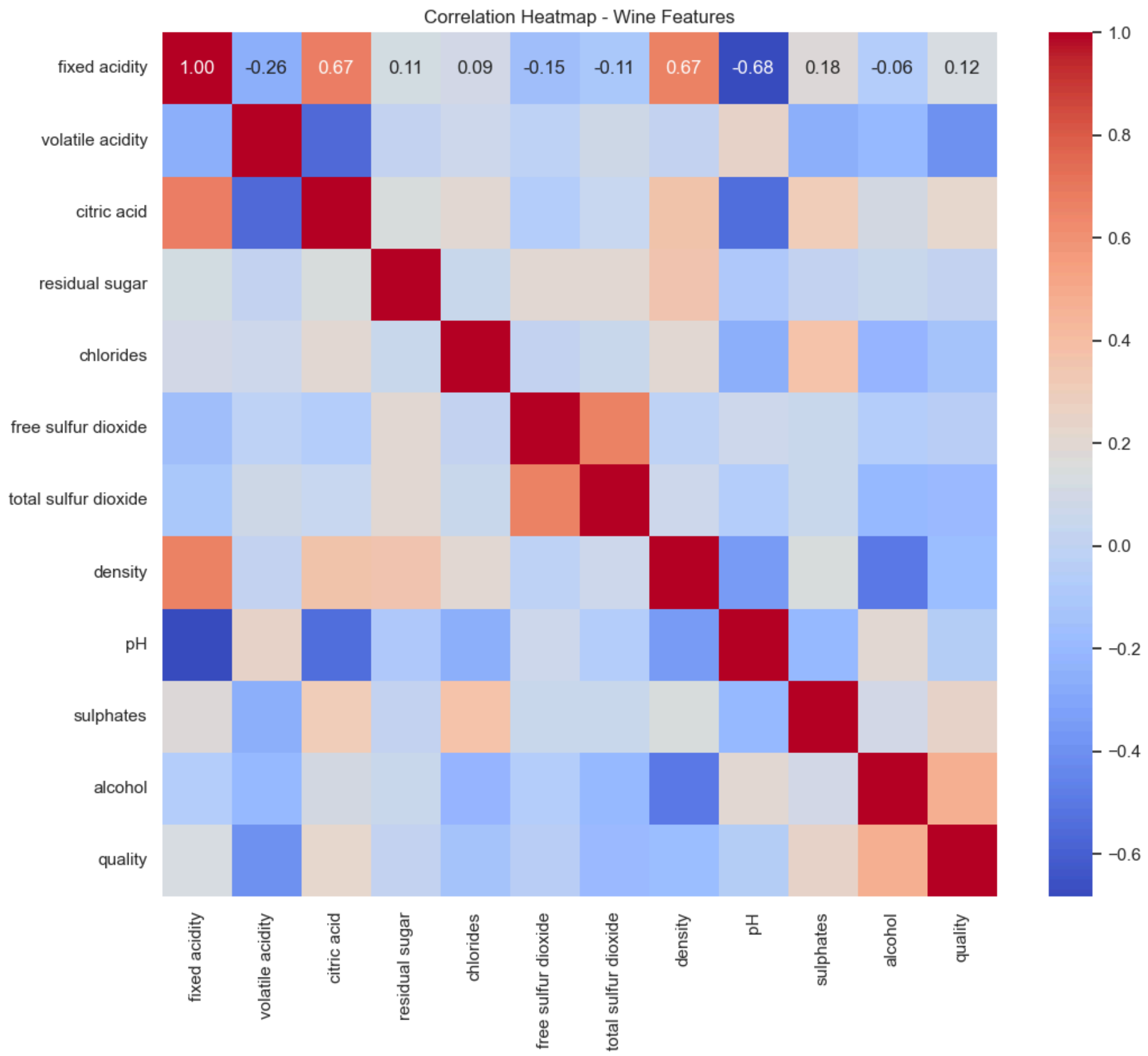

```
From the violin plots and Random Forest, petal length and width are the most impo
rtant features for predicting species.
```

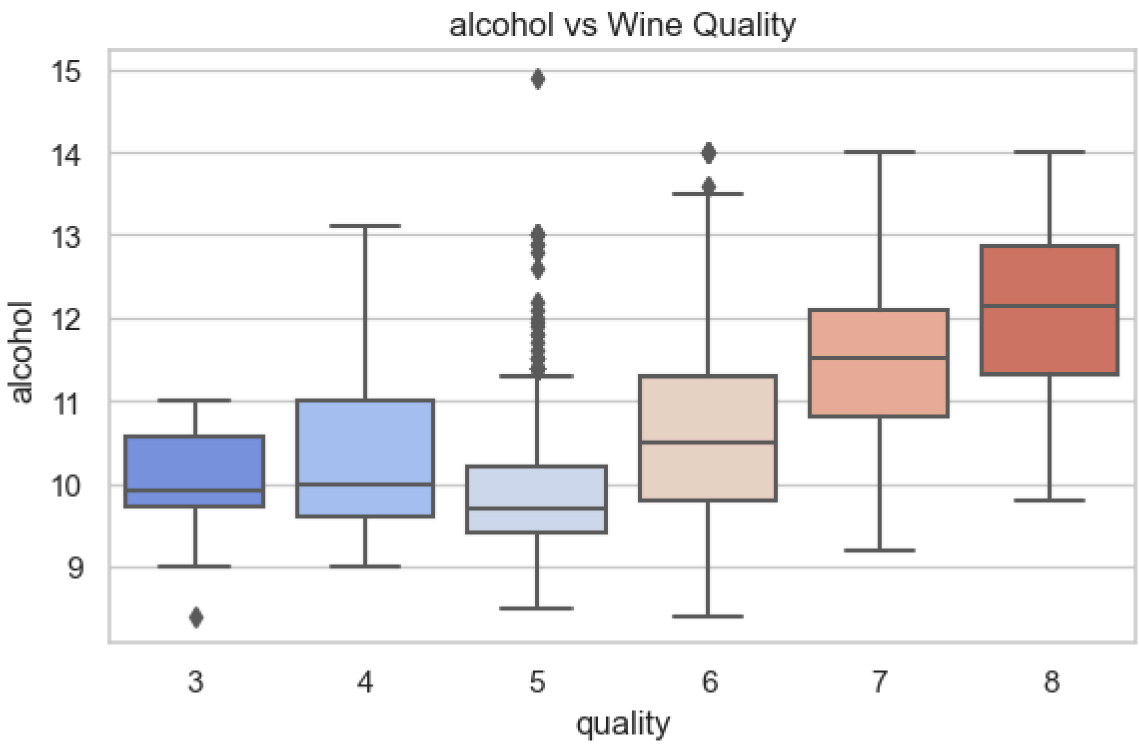
Level 3 Wine Quality Dataset

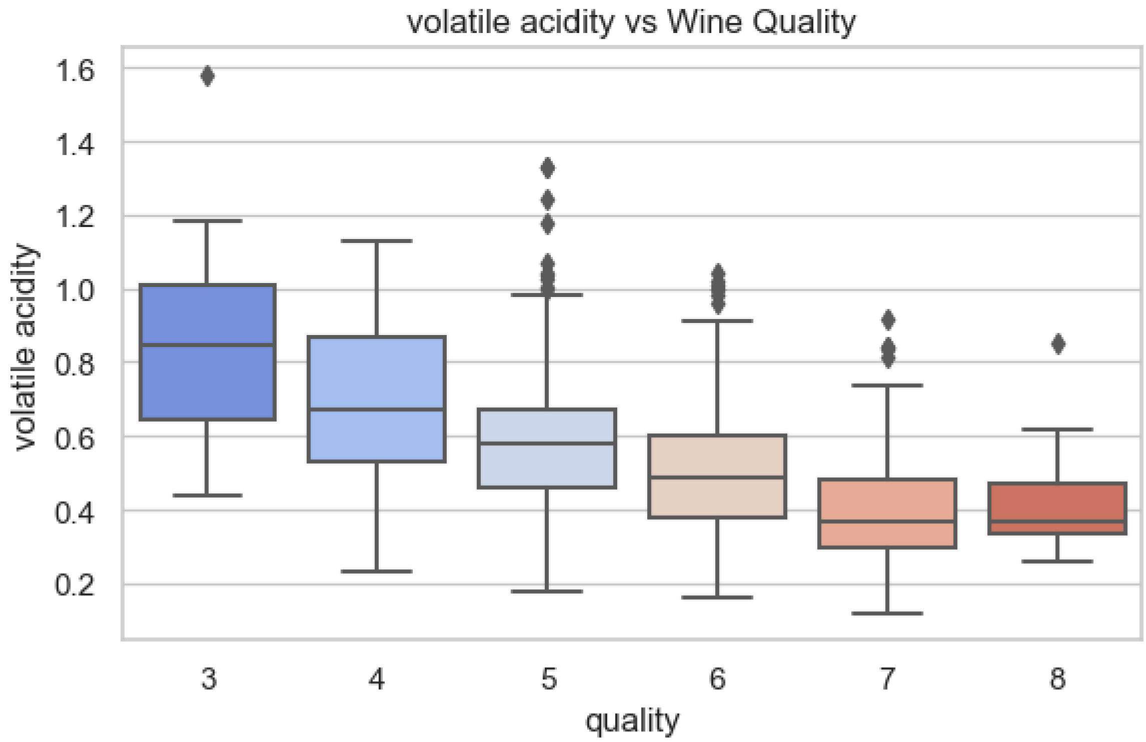
In [4]:
```
# Load Wine Dataset
import pandas as pd
wine = pd.read_csv("C:/Users/mahip/Downloads/winequality-red.csv")  # ← Ensure f

# Explore structure
print("Wine Dataset Shape:", wine.shape)
print(wine.head())

# Null values check
print("Null Values:\n", wine.isnull().sum())

# Correlation Heatmap
import seaborn as sns
import matplotlib.pyplot as plt

plt.figure(figsize=(12, 10))
sns.heatmap(wine.corr(), annot=True, cmap='coolwarm', fmt=".2f")
plt.title("Correlation Heatmap - Wine Features")
plt.show()

# Boxplots for important features vs quality
key_features = ['alcohol', 'volatile acidity', 'citric acid']
for feature in key_features:
    plt.figure(figsize=(6, 4))  # ← INDENTED block inside for loop
    sns.boxplot(x='quality', y=feature, data=wine, palette='coolwarm')
    plt.title(f"{feature} vs Wine Quality")
    plt.tight_layout()
    plt.show()

# Feature Importance with Random Forest
from sklearn.ensemble import RandomForestClassifier
from sklearn.model_selection import train_test_split
```

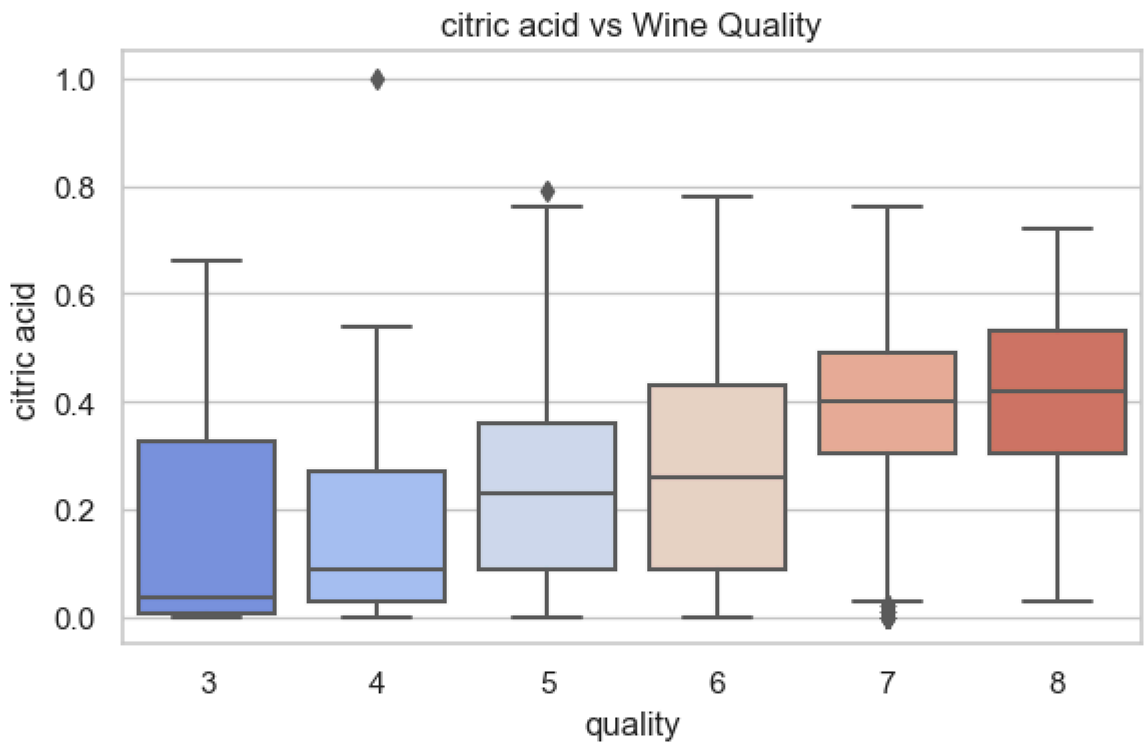
```python
from sklearn.preprocessing import StandardScaler

# Prepare inputs
X = wine.drop('quality', axis=1)
y = wine['quality']

# Scale and split
X_scaled = StandardScaler().fit_transform(X)
X_train, X_test, y_train, y_test = train_test_split(X_scaled, y, test_size=0.2,

# Train model
rf = RandomForestClassifier(n_estimators=100, random_state=42)
rf.fit(X_train, y_train)

# Feature Importance
importances = pd.DataFrame({
    'Feature': X.columns,
    'Importance': rf.feature_importances_
}).sort_values(by='Importance', ascending=False)

plt.figure(figsize=(10, 6))
sns.barplot(x='Importance', y='Feature', data=importances, palette='mako')
plt.title("Feature Importance - Wine Dataset")
plt.tight_layout()
plt.show()

# Summary:
print("Alcohol and sulphates are highly predictive of wine quality. Feature engi
```

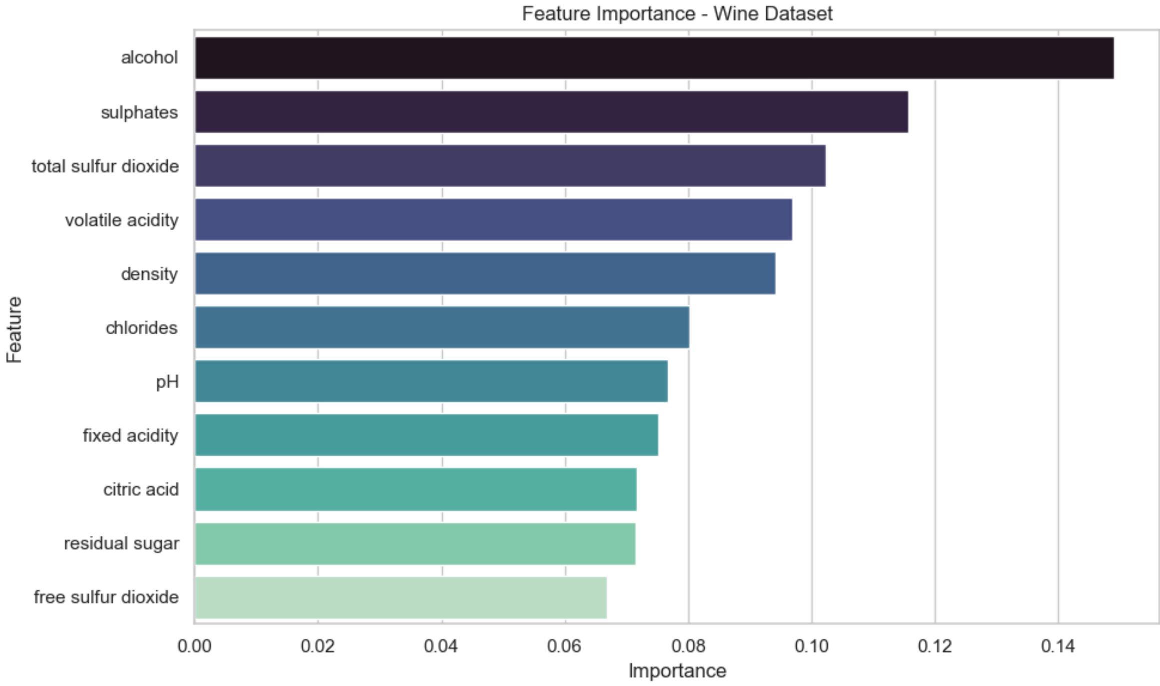
```
Wine Dataset Shape: (1599, 12)
   fixed acidity  volatile acidity  citric acid  residual sugar  chlorides  \
0            7.4              0.70         0.00             1.9      0.076   
1            7.8              0.88         0.00             2.6      0.098   
2            7.8              0.76         0.04             2.3      0.092   
3           11.2              0.28         0.56             1.9      0.075   
4            7.4              0.70         0.00             1.9      0.076   

   free sulfur dioxide  total sulfur dioxide  density    pH  sulphates  \
0                 11.0                  34.0   0.9978  3.51       0.56   
1                 25.0                  67.0   0.9968  3.20       0.68   
2                 15.0                  54.0   0.9970  3.26       0.65   
3                 17.0                  60.0   0.9980  3.16       0.58   
4                 11.0                  34.0   0.9978  3.51       0.56   

   alcohol  quality  
0      9.4        5  
1      9.8        5  
2      9.8        5  
3      9.8        6  
4      9.4        5  
Null Values:
 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
```


Correlation Heatmap - Wine Features
fixed acidity
volatile acidity
citric acid
residual sugar
chlorides
free sulfur dioxide
total sulfur dioxide
density
pH
sulphates
alcohol
quality
1.00
-0.26
0.67
0.11
0.09
-0.15
-0.11
0.67
-0.68
0.18
-0.06
0.12
1.0
0.8
0.6
0.4
0.2
0.0
−0.2
−0.4
−0.6


alcohol vs Wine Quality
alcohol
quality
15
14
13
12
11
10
9
3
4
5
6
7
8


volatile acidity vs Wine Quality
volatile acidity
1.6
1.4
1.2
1.0
0.8
0.6
0.4
0.2
3
4
5
6
7
8
quality


citric acid vs Wine Quality
citric acid
1.0
0.8
0.6
0.4
0.2
0.0
3
4
5
6
7
8
quality



Alcohol and sulphates are highly predictive of wine quality. Feature engineering here is crucial for improving model performance.