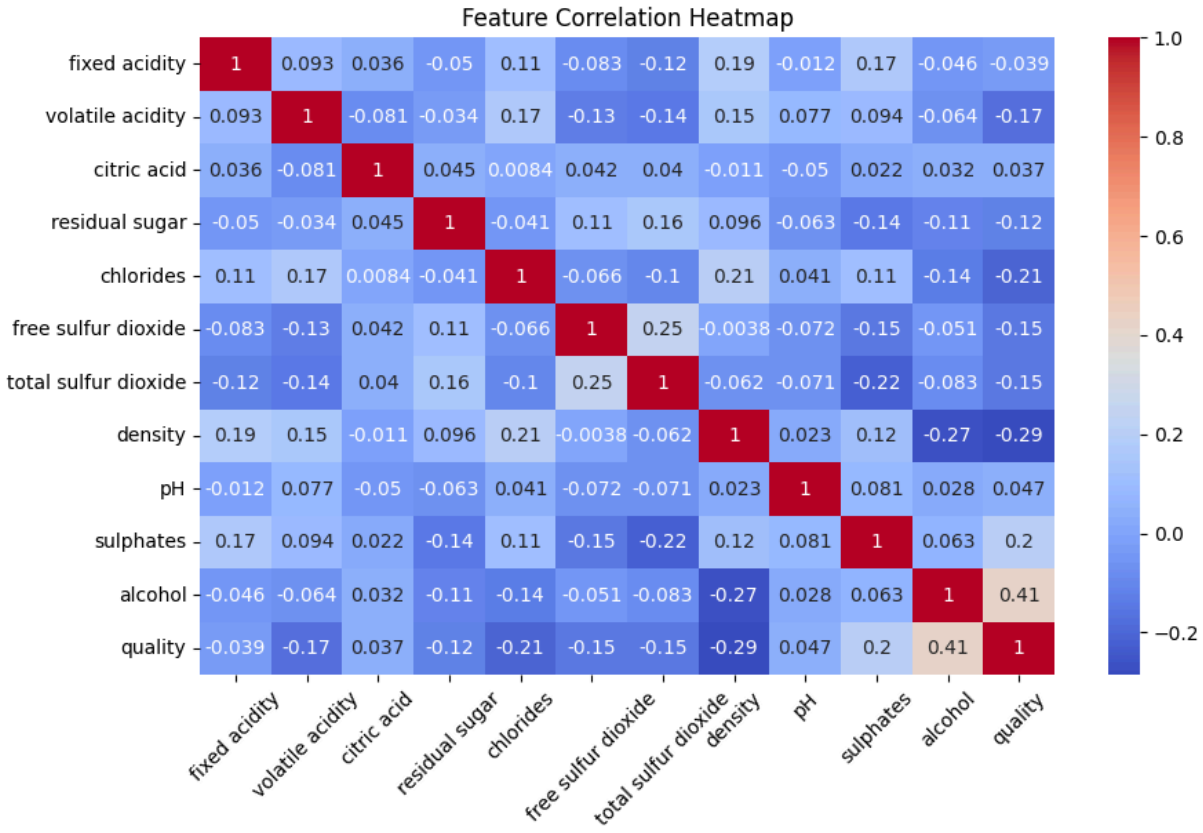
Connected to Python 3.13.5

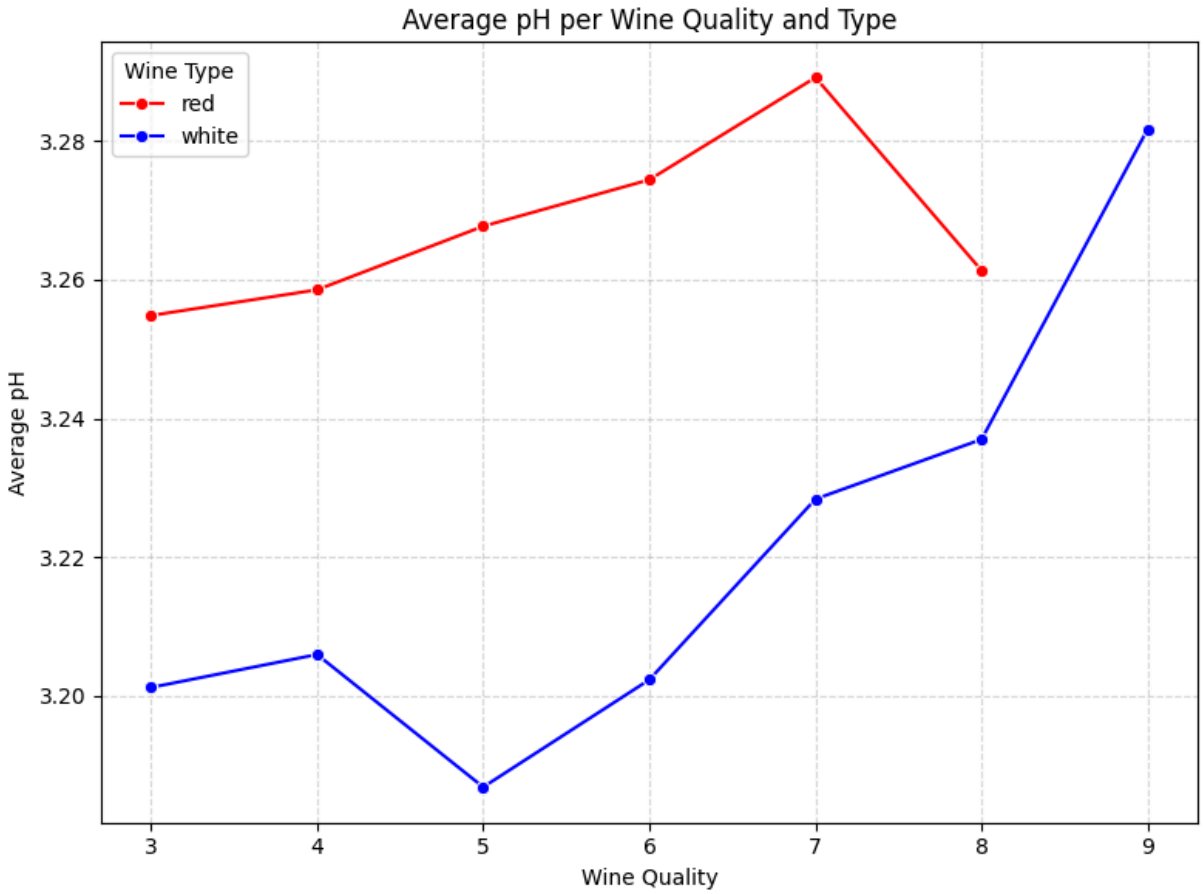
In [ ]:
```
# LIBRARIES

import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from rich import print
from sklearn.preprocessing import PowerTransformer, StandardScaler
from sklearn.model_selection import train_test_split, cross_val_score, StratifiedKF
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import classification_report, accuracy_score, confusion_matrix
from sklearn.svm import SVC
from imblearn.over_sampling import SMOTE
from collections import Counter
from sklearn.linear_model import LogisticRegression
import pickle
from IPython.display import display
from sklearn.ensemble import GradientBoostingClassifier, RandomForestClassifier, St
import xgboost as xgb
```

In [ ]:
```
# IMPORT CLEANED WINES DATASET
df = pd.read_csv(r'wine_cleaned.csv')
```

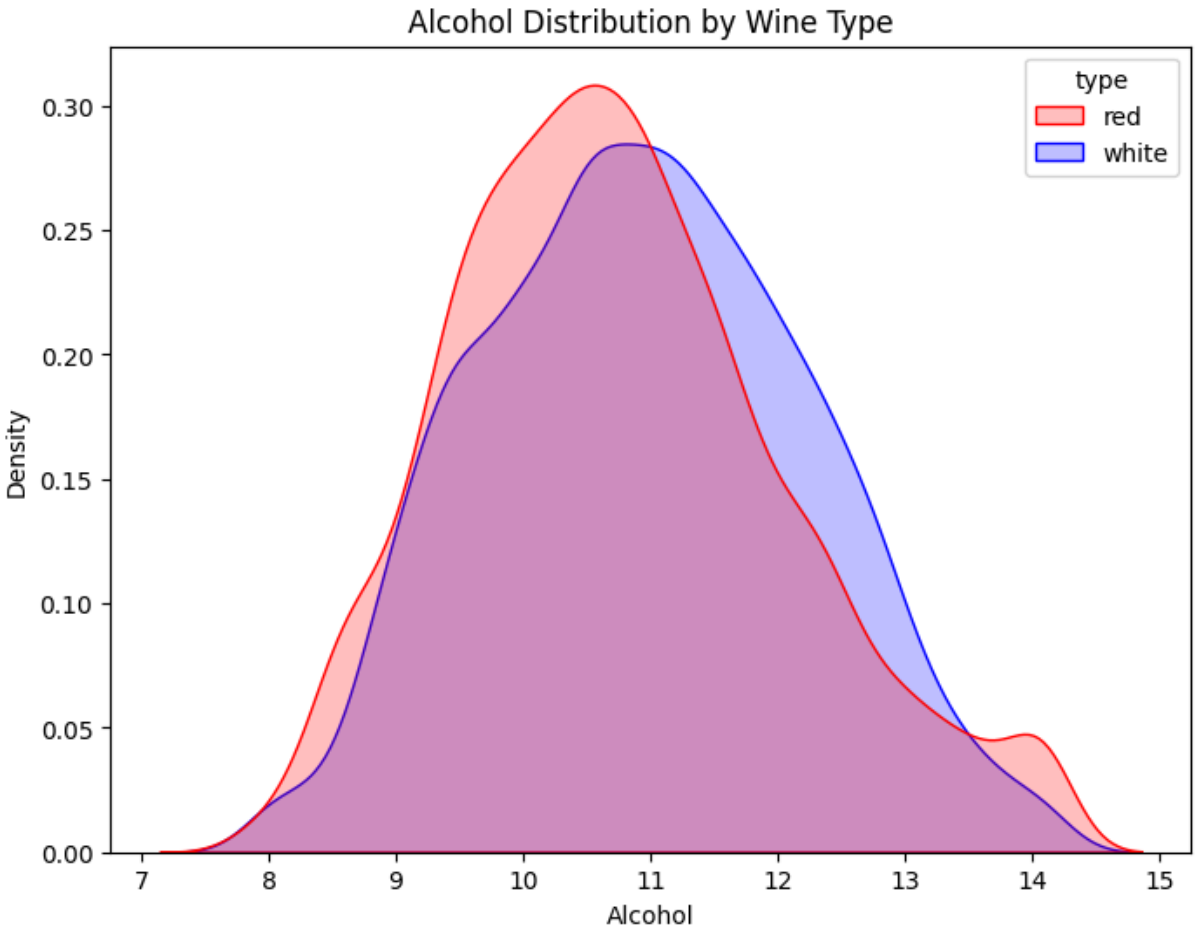
In [ ]:
```
# HEATMAP

corr_matrix = df.drop(columns=['type']).corr(method='pearson')

plt.figure(figsize=(10,6))
sns.heatmap(
    corr_matrix,
    annot=True,
    cmap="coolwarm")

plt.title("Feature Correlation Heatmap")
plt.xticks(rotation=45)
plt.show()
```



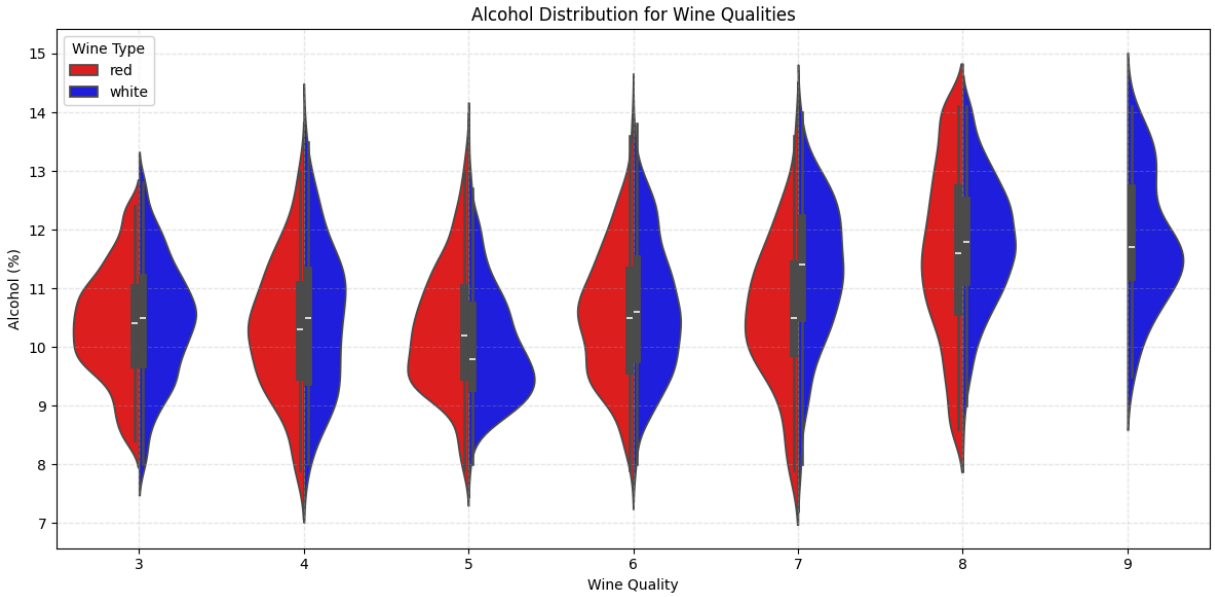
In [ ]:
```
# LINE GRAPH

avg_ph = df.groupby(['quality','type'])["pH"].mean().reset_index()

plt.figure(figsize=(8,6))
sns.lineplot(
    data=avg_ph,
    x="quality",
    y="pH",
    hue="type",
    marker='o',
    palette= {"red":"red","white":"blue"})

plt.title("Average pH per Wine Quality and Type")
plt.xlabel("Wine Quality")
plt.ylabel("Average pH")
plt.legend(title="Wine Type")
plt.grid(True, linestyle='--', alpha=0.5)
plt.tight_layout()
plt.show()
```



In [ ]:
```
# KDE PLOT

plt.figure(figsize=(8,6))
sns.kdeplot(
    data=df,
    x="alcohol",
    hue="type",
    fill=True,
    common_norm=False,
    # palette = "Set2"
    palette={"red": "red", "white": "blue"}
)
plt.title("Alcohol Distribution by Wine Type")
plt.xlabel("Alcohol")
plt.ylabel("Density")
plt.show()
```



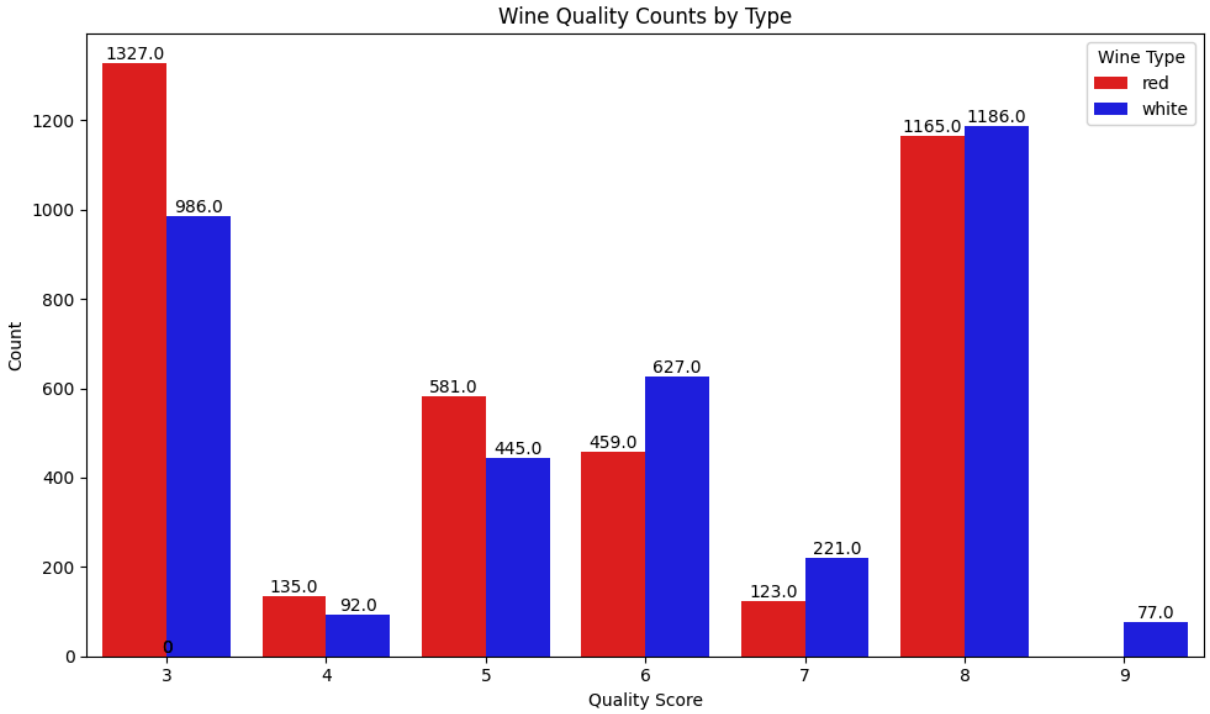
In [ ]:
```
# VIOLIN PLOT

plt.figure(figsize=(12,6))
sns.violinplot(
    x='quality',
    y='alcohol',
    hue='type',
    data=df,
    palette={"red":"red","white":"blue"},
    split=True)

plt.title(f"Alcohol Distribution for Wine Qualities")
plt.xlabel("Wine Quality")
plt.ylabel("Alcohol (%)")
plt.legend(title='Wine Type')
plt.grid(True, linestyle='--', alpha=0.3)
plt.tight_layout()
plt.show()
```



In [ ]:
```
# BAR GRAPH

quality_counts = df['quality'].value_counts().sort_index()
print("[bold]Wine Quality Counts:[/bold]")
print(quality_counts)


plt.figure(figsize=(10, 6))

# Plot countplot with 'quality' on x-axis and 'type' as hue
ax = sns.countplot(
    x='quality',
    hue='type',
    data=df,
    palette={"red":"red","white":"blue"}
    )

plt.title('Wine Quality Counts by Type')
plt.xlabel('Quality Score')
plt.ylabel('Count')

# Add count labels on top of each bar
for p in ax.patches:
    height = p.get_height()
    ax.annotate(f'{height}',
                (p.get_x() + p.get_width() / 2, height),
                ha='center', va='bottom', fontsize=10)

plt.legend(title='Wine Type')
plt.tight_layout()
plt.show()
```

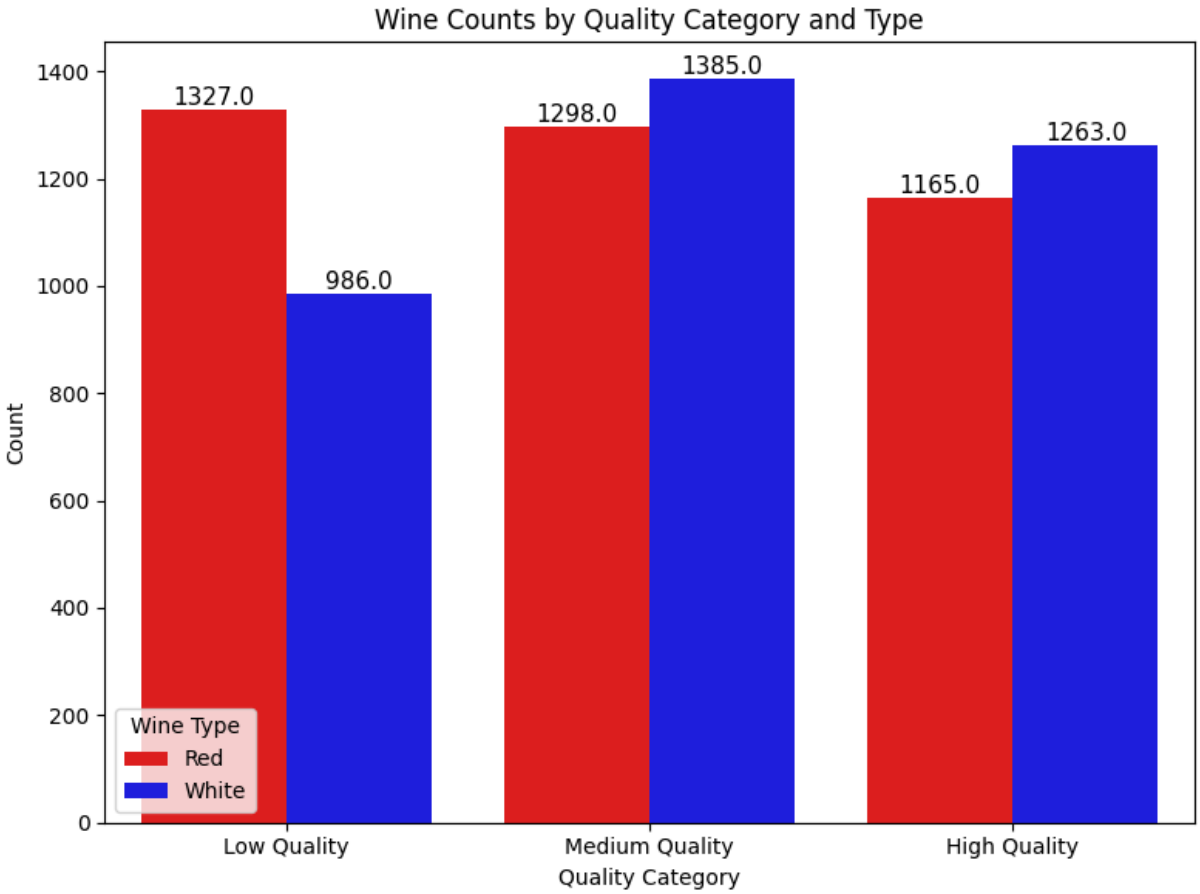
Wine Quality Counts:

```
quality
3    2313
4     227
5    1026
6    1086
7     344
8    2351
9      77
Name: count, dtype: int64
```



In [ ]:
```python
# BIN 'QUALITY' COLUMN AND CHECK THE COUNT

# 0-3: Low Quality, 4-7: Medium Quality, 8-10: High Quality
def bin_quality(val):
    if val <= 3:
        return 'Low Quality'
    elif val <= 7:
        return 'Medium Quality'
    else:
        return 'High Quality'

# Creating binned quality column
df['quality_binned'] = df['quality'].apply(bin_quality)

# Encode type and quality binned
type_map = {'red': 0, 'white': 1}
quality_map = {'Low Quality': 0, 'Medium Quality': 1, 'High Quality': 2}

df['type'] = df['type'].map(type_map)
df['quality_binned'] = df['quality_binned'].map(quality_map)

# Reverse maps for decoding later
type_map_rev = {v: k.title() for k, v in type_map.items()}         # {0: 'Red', 1:
quality_map_rev = {v: k for k, v in quality_map.items()}
```

```
# <-------PLOT------->

plt.figure(figsize=(8, 6))
ax = sns.countplot(
    data=df,
    x='quality_binned',
    hue='type',
    order=[0, 1, 2],
    palette={0: 'red', 1: 'blue'}
)

# Set readable x-axis labels
ax.set_xticks([0, 1, 2])
ax.set_xticklabels([quality_map_rev[i] for i in [0, 1, 2]])

plt.title('Wine Counts by Quality Category and Type')
plt.xlabel('Quality Category')
plt.ylabel('Count')

# Add count labels on top of bars
for p in ax.patches:
    height = p.get_height()
    if height > 0:
        ax.annotate(
            f'{height}',
            (p.get_x() + p.get_width() / 2, height),
            ha='center',
            va='bottom',
            fontsize=11
        )

# legend labels
handles, labels = ax.get_legend_handles_labels()
ax.legend(handles=handles, labels=['Red', 'White'], title='Wine Type')

plt.tight_layout()
plt.show()
```


Wine Counts by Quality Category and Type
Count
Quality Category
Low Quality
Medium Quality
High Quality
1327.0
986.0
1298.0
1385.0
1165.0
1263.0
Wine Type
Red
White
0
200
400
600
800
1000
1200
1400