Lab Assignment 4: Feature Selection and Engineering

Aim: To apply feature selection and feature engineering techniques on the Wine Quality dataset,improving data representation and model performance.

Task 1: Load and Explore the Dataset

1. Load the Wine Quality dataset using pandas.

```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
%matplotlib inline
import warnings
warnings.filterwarnings('ignore')

df = pd.read_csv('WineQT.csv')
df.head()
```

```
   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  Id
0      9.4        5   0
1      9.8        5   1
2      9.8        5   2
```

```
3       9.8        6   3
4       9.4        5   4
```

2. Display dataset characteristics:

Number of records and features

```
df.shape

(1143, 13)
```

Data types of features (numerical, categorical)

```
df.dtypes

fixed acidity           float64
volatile acidity        float64
citric acid             float64
residual sugar          float64
chlorides               float64
free sulfur dioxide     float64
total sulfur dioxide    float64
density                 float64
pH                      float64
sulphates               float64
alcohol                 float64
quality                   int64
Id                        int64
dtype: object

df.info()

<class 'pandas.core.frame.DataFrame'>
Index: 1041 entries, 0 to 1142
Data columns (total 15 columns):
 #   Column                Non-Null Count  Dtype
---  ------                --------------  -----
 0   fixed acidity         1041 non-null   float64
 1   volatile acidity      1041 non-null   float64
 2   citric acid           1041 non-null   float64
 3   residual sugar        1041 non-null   float64
 4   chlorides             1041 non-null   float64
 5   free sulfur dioxide   1041 non-null   float64
 6   total sulfur dioxide  1041 non-null   float64
 7   density               1041 non-null   float64
 8   pH                    1041 non-null   float64
 9   sulphates             1041 non-null   float64
 10  alcohol               1041 non-null   float64
 11  quality               1041 non-null   int64
 12  Id                    1041 non-null   int64
```

```
 13  acidity_ratio         1041 non-null   float64
 14  alcohol_sugar_ratio   1041 non-null   float64
dtypes: float64(13), int64(2)
memory usage: 130.1 KB
```

Summary statistics (mean, median, standard deviation, etc.)

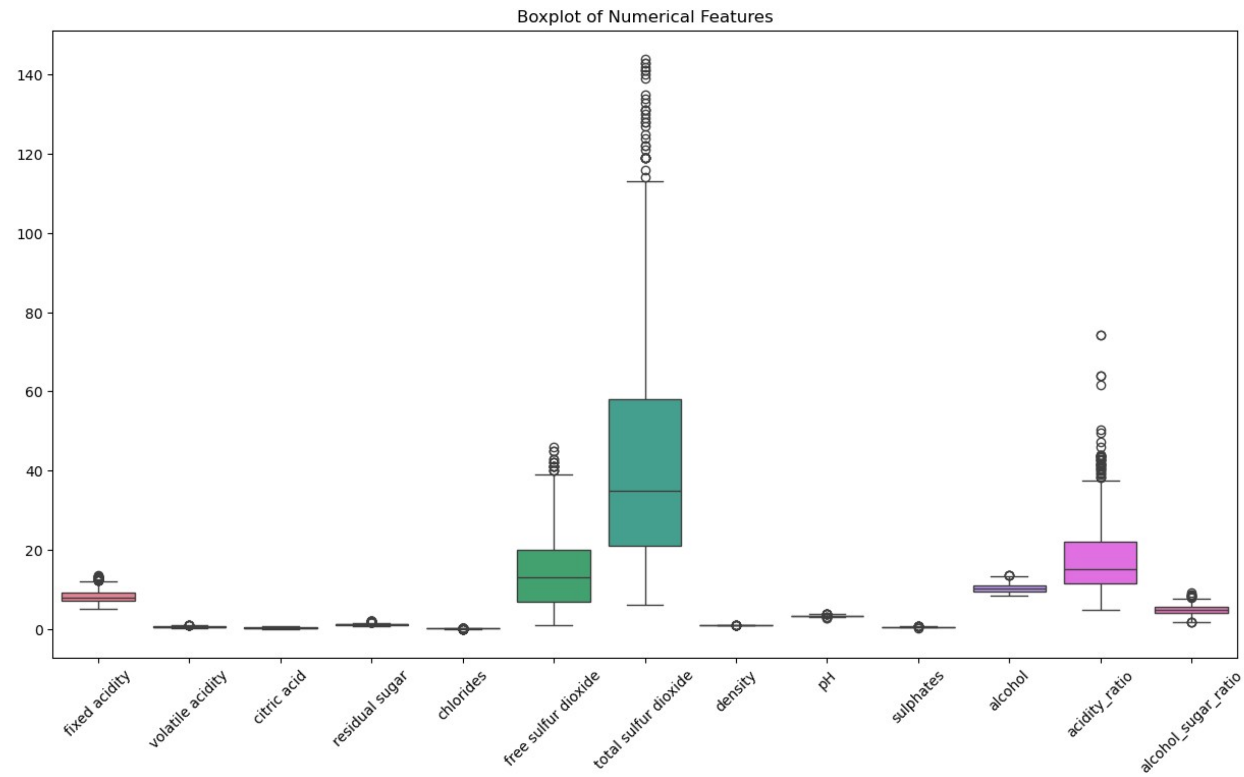
```
df.describe()

       fixed acidity  volatile acidity  citric acid  residual sugar  \
count    1143.000000       1143.000000  1143.000000     1143.000000   
mean        8.311111          0.531339     0.268364        2.532152   
std         1.747595          0.179633     0.196686        1.355917   
min         4.600000          0.120000     0.000000        0.900000   
25%         7.100000          0.392500     0.090000        1.900000   
50%         7.900000          0.520000     0.250000        2.200000   
75%         9.100000          0.640000     0.420000        2.600000   
max        15.900000          1.580000     1.000000       15.500000   

         chlorides  free sulfur dioxide  total sulfur dioxide
density  \
count  1143.000000          1143.000000           1143.000000
1143.000000   
mean      0.086933            15.615486             45.914698
0.996730   
std       0.047267            10.250486             32.782130
0.001925   
min       0.012000             1.000000              6.000000
0.990070   
25%       0.070000             7.000000             21.000000
0.995570   
50%       0.079000            13.000000             37.000000
0.996680   
75%       0.090000            21.000000             61.000000
0.997845   
max       0.611000            68.000000            289.000000
1.003690   

                pH    sulphates      alcohol      quality           Id

count  1143.000000  1143.000000  1143.000000  1143.000000  1143.000000

mean      3.311015     0.657708    10.442111     5.657043   804.969379

std       0.156664     0.170399     1.082196     0.805824   463.997116

min       2.740000     0.330000     8.400000     3.000000     0.000000

25%       3.205000     0.550000     9.500000     5.000000   411.000000
```

```
50%        3.310000      0.620000     10.200000      6.000000   794.000000

75%        3.400000      0.730000     11.100000      6.000000  1209.500000

max        4.010000      2.000000     14.900000      8.000000  1597.000000
```

3. Check for missing values and outliers using visualization techniques.

```
df.isnull().sum()

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
Id                      0
dtype: int64

df.duplicated()

0       False
1       False
2       False
3       False
4       False
        ...
1138    False
1139    False
1140    False
1141    False
1142    False
Length: 1143, dtype: bool

df.columns

Index(['fixed acidity', 'volatile acidity', 'citric acid', 'residual
sugar',
       'chlorides', 'free sulfur dioxide', 'total sulfur dioxide',
'density',
       'pH', 'sulphates', 'alcohol', 'quality', 'Id'],
      dtype='object')
```

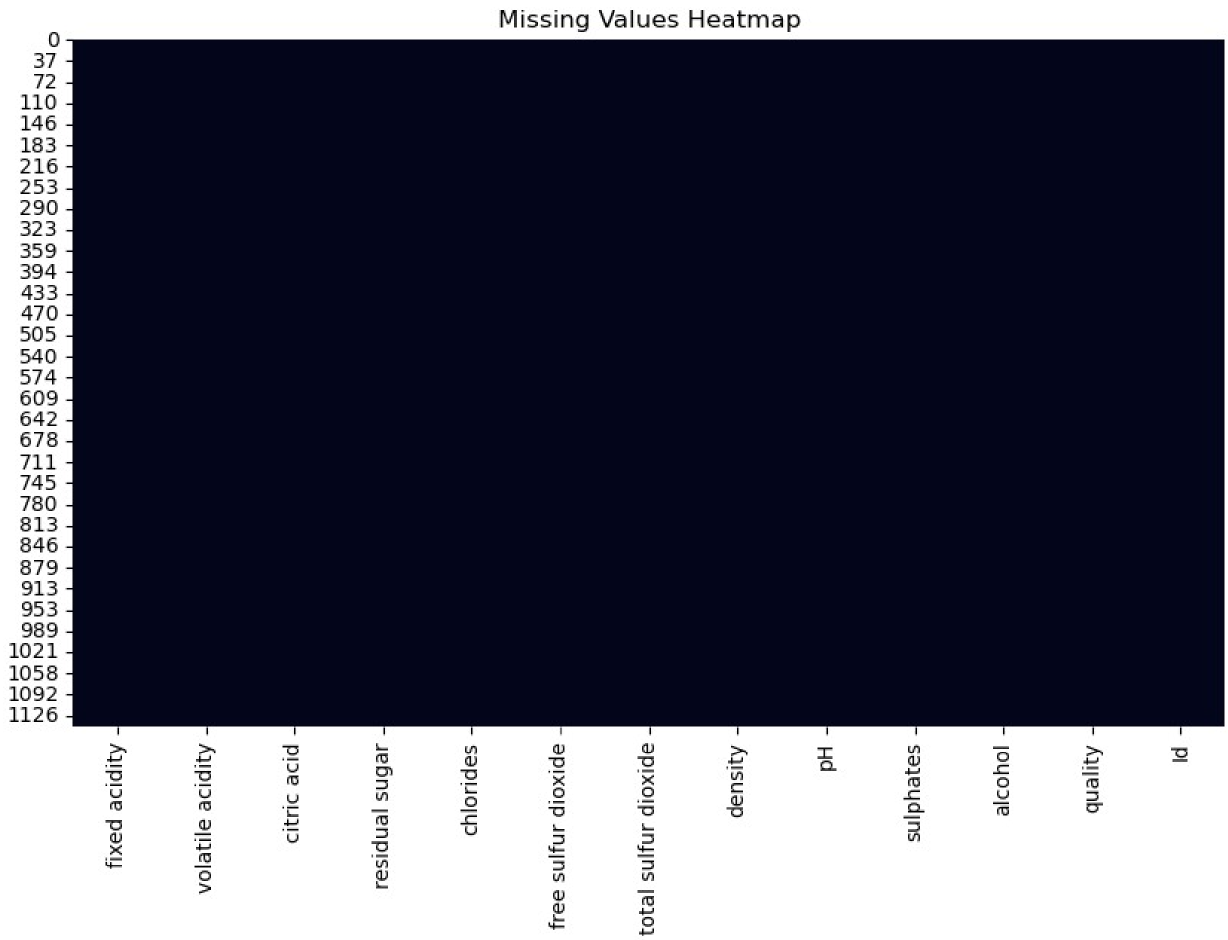
```
plt.figure(figsize=(15, 8))
sns.boxplot(data=df.drop(columns=["quality", "Id"]))
plt.xticks(rotation=45)
plt.title("Boxplot of Numerical Features")
plt.show()
```




```
from scipy import stats
import numpy as np
z_scores = np.abs(stats.zscore(df.drop(columns=["quality", "Id"])))

threshold = 3
df= df[(z_scores < threshold).all(axis=1)]
df.shape[0], df.shape[0]
```

```
(939, 939)
```

```
plt.figure(figsize=(10,6))
sns.heatmap(df.isnull(), cbar=False)
plt.title("Missing Values Heatmap")
plt.show()
```



Task 2: Feature Engineering

1. Create new features using existing attributes (e.g., acidity ratio, alcohol-to-sugar ratio).

```
df["acidity_ratio"] = df["fixed acidity"] / (df["volatile acidity"] +
1e-6)
df["alcohol_sugar_ratio"] = df["alcohol"] / (df["residual sugar"] +
1e-6)
df[["acidity_ratio", "alcohol_sugar_ratio"]].head()
```

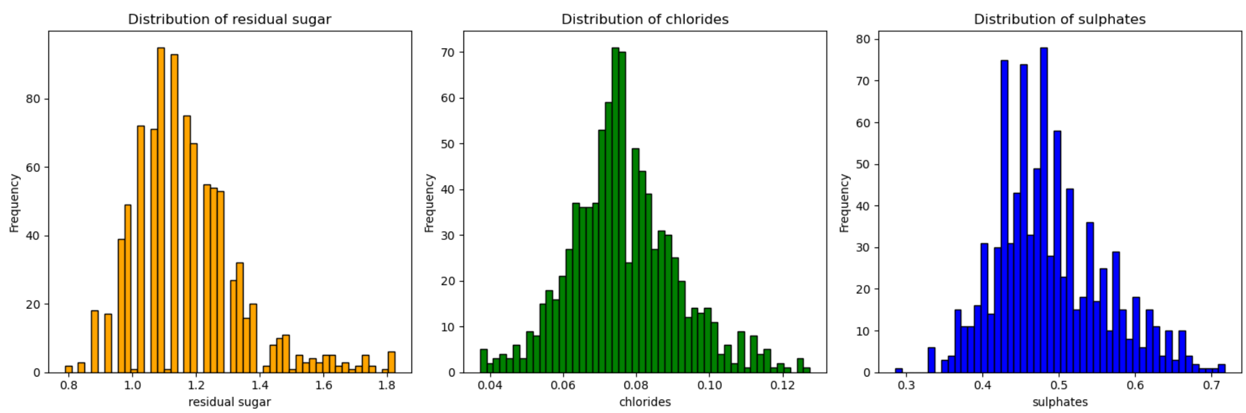
	acidity_ratio	alcohol_sugar_ratio
0	10.571413	8.828681
1	8.863626	7.650662
2	10.263144	8.208231
3	39.999857	9.204369
4	10.571413	8.828681

2. Transform variables if necessary (log transformation, polynomial features).

```
from scipy.stats import skew
skew = skew(df[['residual sugar','chlorides','sulphates']])
print("Skewness before log transformation:", skew)
```

```
Skewness before log transformation: [1.02627167 0.36827539 0.55780173]
```

```
plt.figure(figsize=(15, 5))
features_to_check = ['residual sugar', 'chlorides', 'sulphates']
for i, feature in enumerate(features_to_check):
    plt.subplot(1, 3, i + 1)
    plt.hist(df[feature], bins=50,edgecolor= 'black', color=['orange',
'green', 'blue'][i])
    plt.title(f'Distribution of {feature}')
    plt.xlabel(feature)
    plt.ylabel('Frequency')

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




```
from scipy.stats import skew
df['residual sugar'] = np.log1p(df['residual sugar'])
df['chlorides'] = np.log1p(df['chlorides'])
df['sulphates'] = np.log1p(df['sulphates'])
skew = skew(df[['residual sugar', 'chlorides', 'sulphates']], axis=0)
print("Skewness after log transformation:", skew)
```

```
Skewness after log transformation: [0.73258641 0.31735152 0.43149878]
```

4. Scale numerical features using Min-Max Scaling or Standardization.

```
from sklearn.preprocessing import MinMaxScaler,StandardScaler

scaler = StandardScaler()
scaled_features = scaler.fit_transform(df.drop(['quality'], axis=1))
df_scaled = pd.DataFrame(scaled_features, columns=df.columns[:-1])
```

Task 3: Feature Selection Techniques

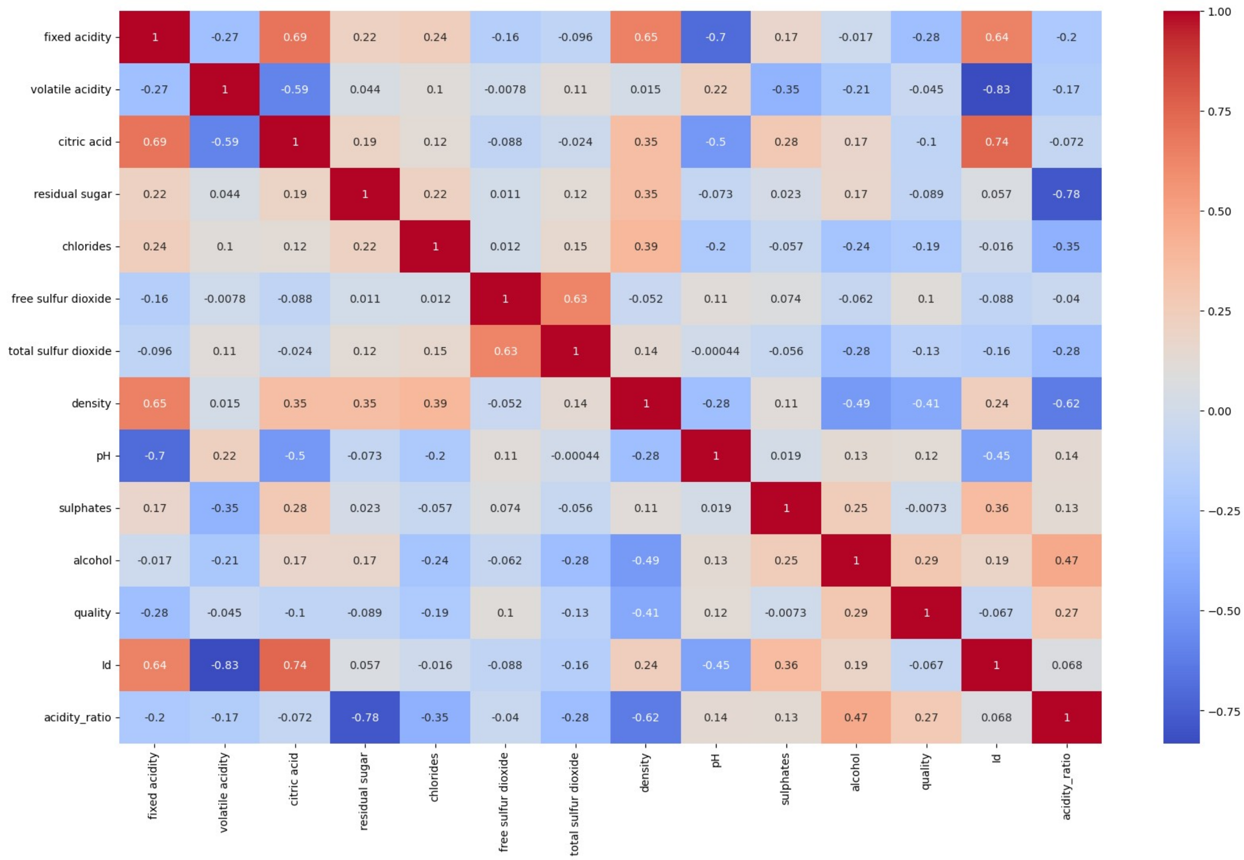
1. Perform correlation analysis to remove redundant features.

```
corr = df_scaled.corr()
corr
```

```
                      fixed acidity  volatile acidity  citric acid  \
fixed acidity              1.000000         -0.268733     0.693000   
volatile acidity          -0.268733          1.000000    -0.591148   
citric acid                0.693000         -0.591148     1.000000   
residual sugar             0.217626          0.044375     0.193224   
chlorides                  0.240073          0.103817     0.124499   
free sulfur dioxide       -0.158742         -0.007763    -0.088261   
total sulfur dioxide      -0.095544          0.106324    -0.023758   
density                    0.648581          0.015466     0.353180   
pH                        -0.700920          0.220999    -0.501057   
sulphates                  0.174711         -0.350996     0.279104   
alcohol                   -0.017115         -0.210359     0.171431   
quality                   -0.277629         -0.045194    -0.101821   
Id                         0.637874         -0.832257     0.744133   
acidity_ratio             -0.204526         -0.168920    -0.071863   

                      residual sugar  chlorides  free sulfur 
dioxide  \
fixed acidity               0.217626   0.240073             -0.158742 

volatile acidity            0.044375   0.103817             -0.007763 

citric acid                 0.193224   0.124499             -0.088261 

residual sugar              1.000000   0.218746              0.011086 

chlorides                   0.218746   1.000000              0.012038 

free sulfur dioxide         0.011086   0.012038              1.000000 

total sulfur dioxide        0.120996   0.150551              0.631017 

density                     0.352445   0.389546             -0.051641 

pH                         -0.072900  -0.201475              0.108283 

sulphates                   0.023324  -0.057346              0.074160 

alcohol                     0.169405  -0.240390             -0.062205 

quality                    -0.088844  -0.191289              0.104733 

Id                          0.057167  -0.015818             -0.088408 
```

```
acidity_ratio              -0.783283  -0.354633            -0.040373


                      total sulfur dioxide   density        pH
sulphates  \
fixed acidity                    -0.095544  0.648581 -0.700920
0.174711
volatile acidity                  0.106324  0.015466  0.220999  -
0.350996
citric acid                      -0.023758  0.353180 -0.501057
0.279104
residual sugar                    0.120996  0.352445 -0.072900
0.023324
chlorides                         0.150551  0.389546 -0.201475  -
0.057346
free sulfur dioxide               0.631017 -0.051641  0.108283
0.074160
total sulfur dioxide              1.000000  0.135190 -0.000439  -
0.056239
density                           0.135190  1.000000 -0.275158
0.106915
pH                               -0.000439 -0.275158  1.000000
0.019202
sulphates                        -0.056239  0.106915  0.019202
1.000000
alcohol                          -0.284549 -0.486574  0.127953
0.247504
quality                          -0.130400 -0.409586  0.123152  -
0.007304
Id                               -0.156927  0.235402 -0.445107
0.363938
acidity_ratio                    -0.283896 -0.624823  0.137652
0.129481

                       alcohol   quality         Id  acidity_ratio
fixed acidity        -0.017115 -0.277629  0.637874      -0.204526
volatile acidity     -0.210359 -0.045194 -0.832257      -0.168920
citric acid           0.171431 -0.101821  0.744133      -0.071863
residual sugar        0.169405 -0.088844  0.057167      -0.783283
chlorides            -0.240390 -0.191289 -0.015818      -0.354633
free sulfur dioxide  -0.062205  0.104733 -0.088408      -0.040373
total sulfur dioxide -0.284549 -0.130400 -0.156927      -0.283896
density              -0.486574 -0.409586  0.235402      -0.624823
pH                    0.127953  0.123152 -0.445107       0.137652
sulphates             0.247504 -0.007304  0.363938       0.129481
alcohol               1.000000  0.287465  0.187341       0.465080
quality               0.287465  1.000000 -0.066970       0.271821
Id                    0.187341 -0.066970  1.000000       0.068248
acidity_ratio         0.465080  0.271821  0.068248       1.000000
```

```
plt.figure(figsize=(20,12))
sns.heatmap(corr,annot=True,cmap = 'coolwarm')
plt.show()
```



2. Apply Recursive Feature Elimination (RFE) to find the most significant features.

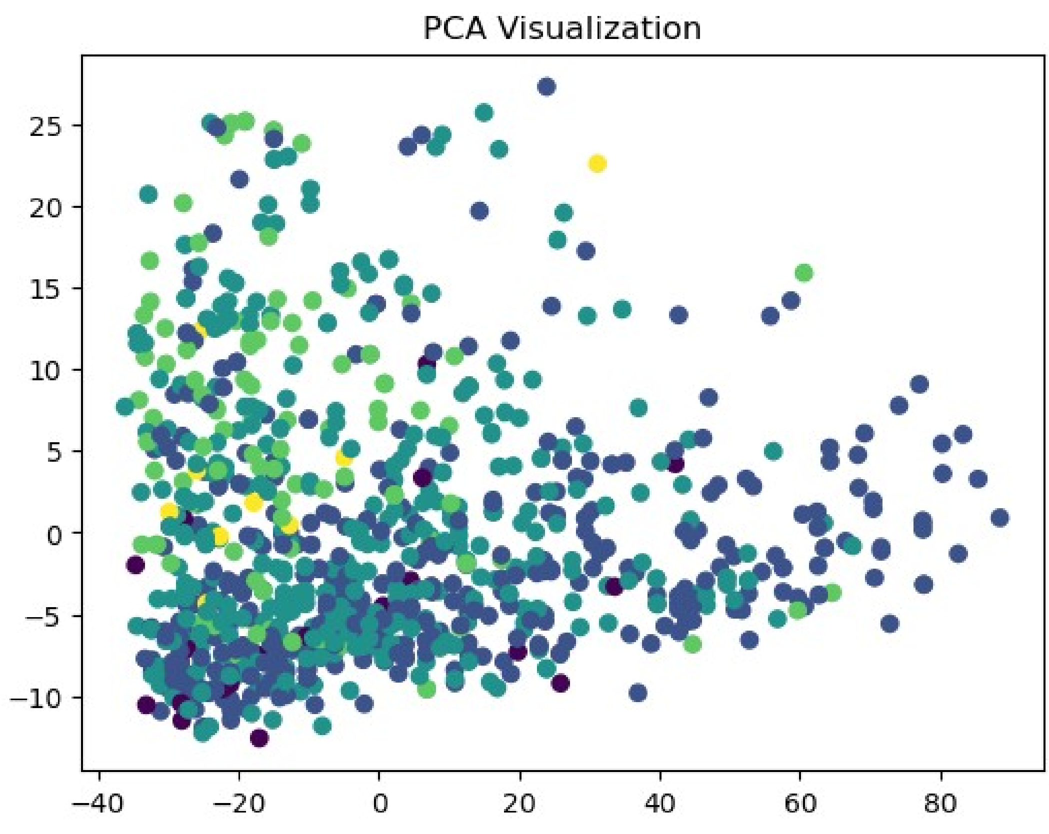
```
from sklearn.feature_selection import RFE
from sklearn.tree import DecisionTreeClassifier

df_reduced = df.drop(columns=["free sulfur dioxide", "density",
"citric acid", "Id"])
df_reduced.columns

Index(['fixed acidity', 'volatile acidity', 'residual sugar',
'chlorides',
       'total sulfur dioxide', 'pH', 'sulphates', 'alcohol',
'quality',
       'acidity_ratio', 'alcohol_sugar_ratio'],
      dtype='object')

X = df_reduced.drop(columns=["quality"])
y = df_reduced["quality"]
```

```
model = DecisionTreeClassifier()
rfe = RFE(model, n_features_to_select=5)
fit = rfe.fit(X, y)

selected_features = X.columns[fit.support_]
print(f"Selected Features: {selected_features}")
```

```
Selected Features: Index(['pH', 'sulphates', 'alcohol',
'acidity_ratio', 'alcohol_sugar_ratio'], dtype='object')
```

3. Use Principal Component Analysis (PCA) to reduce dimensionality and visualize feature space.

```
from sklearn.decomposition import PCA
pca = PCA(n_components=2)
x = pca.fit_transform(X)
plt.scatter(x[:, 0], x[:, 1], c=y, cmap='viridis')
plt.title("PCA Visualization")
plt.show()
```



Task 4: Model Evaluation with Selected Features

1. Train a classification model (e.g., Decision Tree or kNN) using all features.

```
from sklearn.model_selection import train_test_split
from sklearn.metrics import classification_report

X_train,X_test,y_train,y_test = train_test_split(X,y,test_size = 0.3,
                                                  random_state=42)

dtc = DecisionTreeClassifier()
dtc.fit(X_train,y_train)
```

```
DecisionTreeClassifier()
```

```
y_pred = dtc.predict(X_test)
```

```
print("Model Evaluation:")
print(classification_report(y_test, y_pred))
```

```
Model Evaluation:
              precision    recall  f1-score   support

           4       0.00      0.00      0.00         5
           5       0.63      0.60      0.61       123
           6       0.55      0.60      0.58       120
           7       0.62      0.52      0.56        31
           8       1.00      0.33      0.50         3

    accuracy                           0.58       282
   macro avg       0.56      0.41      0.45       282
weighted avg       0.59      0.58      0.58       282
```

2. Train the same model using selected features and compare performance.

```
X_train_selected = X_train[selected_features]
X_test_selected = X_test[selected_features]

dtc.fit(X_train_selected, y_train)
y_pred_selected = dtc.predict(X_test_selected)
```

3. Evaluate models using:

Accuracy, Precision, Recall, and F1-score Feature importance analysis

```
print("Model Evaluation (Selected Features):")
print(classification_report(y_test, y_pred_selected))
```

```
Model Evaluation (Selected Features):
              precision    recall  f1-score   support
```

```
           4       0.00      0.00      0.00         5
           5       0.71      0.64      0.67       123
           6       0.61      0.62      0.62       120
           7       0.43      0.52      0.47        31
           8       0.25      0.33      0.29         3

    accuracy                           0.61       282
   macro avg       0.40      0.42      0.41       282
weighted avg       0.62      0.61      0.61       282
```