Wine Quality Prediction

The focus is on predicting the quality of wine based on its chemical characteristics, offering a real-world application of machine learning in the context of viticulture. The dataset encompasses diverse chemical attributes, including density and acidity, which serve as the features for three distinct classifier models.

Import Library

```
import pandas as pd
```

```
import numpy as np
```

```
import matplotlib.pyplot as plt
```

```
import seaborn as sns
```

Import Dataset

```
wq=pd.read_csv('/WineQT.csv')
```

Displaying first 10 rows

```
wq.head(10)
```

	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	pH	sulphates	alcohol	quality	Id
0	7.4	0.70	0.00	1.9	0.076	11.0	34.0	0.9978	3.51	0.56	9.4	5	0
1	7.8	0.88	0.00	2.6	0.098	25.0	67.0	0.9968	3.20	0.68	9.8	5	1
2	7.8	0.76	0.04	2.3	0.092	15.0	54.0	0.9970	3.26	0.65	9.8	5	2
3	11.2	0.28	0.56	1.9	0.075	17.0	60.0	0.9980	3.16	0.58	9.8	6	3
4	7.4	0.70	0.00	1.9	0.076	11.0	34.0	0.9978	3.51	0.56	9.4	5	4
5	7.4	0.66	0.00	1.8	0.075	13.0	40.0	0.9978	3.51	0.56	9.4	5	5
6	7.9	0.60	0.06	1.6	0.069	15.0	59.0	0.9964	3.30	0.46	9.4	5	6
7	7.3	0.65	0.00	1.2	0.065	15.0	21.0	0.9946	3.39	0.47	10.0	7	7
8	7.8	0.58	0.02	2.0	0.073	9.0	18.0	0.9968	3.36	0.57	9.5	7	8
9	6.7	0.58	0.08	1.8	0.097	15.0	65.0	0.9959	3.28	0.54	9.2	5	10

Next steps: View recommended plots

Data Preparation

Cleaning and Exploration

Checking for any missing values

```
wq.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
```

Now handling the missing values

```
wq.fillna(wq.median(),inplace=True)
```

Removing duplicates if any

```
wq.drop_duplicates(inplace=True)
```

Ensuring correct data types

```
wq.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
```

Feature Selection

Defining feature value x and targeted value y

```
features=['fixed acidity','volatile acidity','citric acid','residual sugar','chlorides','free sulfur dioxide','total sulfur dioxide','de
```

```
x=wq[features]
```

```
y=wq['quality']
```

Model Training

```
from sklearn.model_selection import train_test_split
```

Splitting the data into training and testing sets

```
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2,random_state=42)
```

```
from sklearn.ensemble import RandomForestClassifier
```

Train the RandomForestClassifier

```
rf_model=RandomForestClassifier(n_estimators=100,random_state=42)
```

```
rf_model.fit(x_train,y_train)
```

```
▾       RandomForestClassifier
RandomForestClassifier(random_state=42)
```

Now Making predictions

```
y_pred=rf_model.predict(x_test)
```

```
from sklearn.metrics import classification_report,confusion_matrix
```

Evaluating the Model

```
confusion_matrix(y_test,y_pred)
```

```
array([[ 0,  3,  3,  0,  0],
       [ 0, 77, 18,  1,  0],
       [ 0, 27, 66,  6,  0],
       [ 0,  0,  8, 18,  0],
       [ 0,  0,  2,  0,  0]])
```

```
classification_report(y_test,y_pred)
```

```
/usr/local/lib/python3.10/dist-packages/sklearn/metrics/_classification.py:1344: UndefinedMetricWarning: Precision and F-score are
  _warn_prf(average, modifier, msg_start, len(result))
/usr/local/lib/python3.10/dist-packages/sklearn/metrics/_classification.py:1344: UndefinedMetricWarning: Precision and F-score are
  _warn_prf(average, modifier, msg_start, len(result))
/usr/local/lib/python3.10/dist-packages/sklearn/metrics/_classification.py:1344: UndefinedMetricWarning: Precision and F-score are
  _warn_prf(average, modifier, msg_start, len(result))
'              precision    recall  f1-score   support\n\n           4       0.00      0.00      0.00         6\n           5
0.72      0.80      0.76        96\n           6       0.68      0.67      0.67        99\n           7       0.72      0.69
0.71        26\n           8       0.00      0.00      0.00         2\n\n    accuracy                           0.70       229\n
macro avg       0.42      0.43      0.43       229\nweighted avg       0.68      0.70      0.69       229\n'
```

Support Vector Classifier

```
from sklearn.svm import SVC
```

Training the SVC

```
svc_model=SVC(random_state=42)
```

```
svc_model.fit(x_train,y_train)
```

```
▾       SVC
SVC(random_state=42)
```

Making predictions

```
y_pred_svc=svc_model.predict(x_test)
```

Evaluating the model

```
print("Support vector Classifier:")
print(confusion_matrix(y_test,y_pred_svc))
print(classification_report(y_test,y_pred_svc))
```

```
Support vector Classifier:
[[ 0  2  4  0  0]
 [ 0 41 55  0  0]
 [ 0 12 87  0  0]
 [ 0  1 24  1  0]
 [ 0  0  2  0  0]]
              precision    recall  f1-score   support

           4       0.00      0.00      0.00         6
           5       0.73      0.43      0.54        96
           6       0.51      0.88      0.64        99
           7       1.00      0.04      0.07        26
           8       0.00      0.00      0.00         2

    accuracy                           0.56       229
   macro avg       0.45      0.27      0.25       229
weighted avg       0.64      0.56      0.51       229

/usr/local/lib/python3.10/dist-packages/sklearn/metrics/_classification.py:1344: UndefinedMetricWarning: Precision and F-score are
  _warn_prf(average, modifier, msg_start, len(result))
/usr/local/lib/python3.10/dist-packages/sklearn/metrics/_classification.py:1344: UndefinedMetricWarning: Precision and F-score are
  _warn_prf(average, modifier, msg_start, len(result))
```

```
/usr/local/lib/python3.10/dist-packages/sklearn/metrics/_classification.py:1344: UndefinedMetricWarning: Precision and F-score are i
  _warn_prf(average, modifier, msg_start, len(result))
```

Stochastic Gradient Descent Classifier

```
from sklearn.linear_model import SGDClassifier
```

Training the SGD Classifier

```
sgd_model= SGDClassifier(random_state=42)
```

```
sgd_model.fit(x_train,y_train)
```

```
▾        SGDClassifier
SGDClassifier(random_state=42)
```

Making predictions

```
y_pred_sgd=sgd_model.predict(x_test)
```

Evaluating the model

```
print("Stochastic Gradient Descent Classifier:")
print(confusion_matrix(y_test,y_pred_sgd))
print(classification_report(y_test,y_pred_sgd))
```

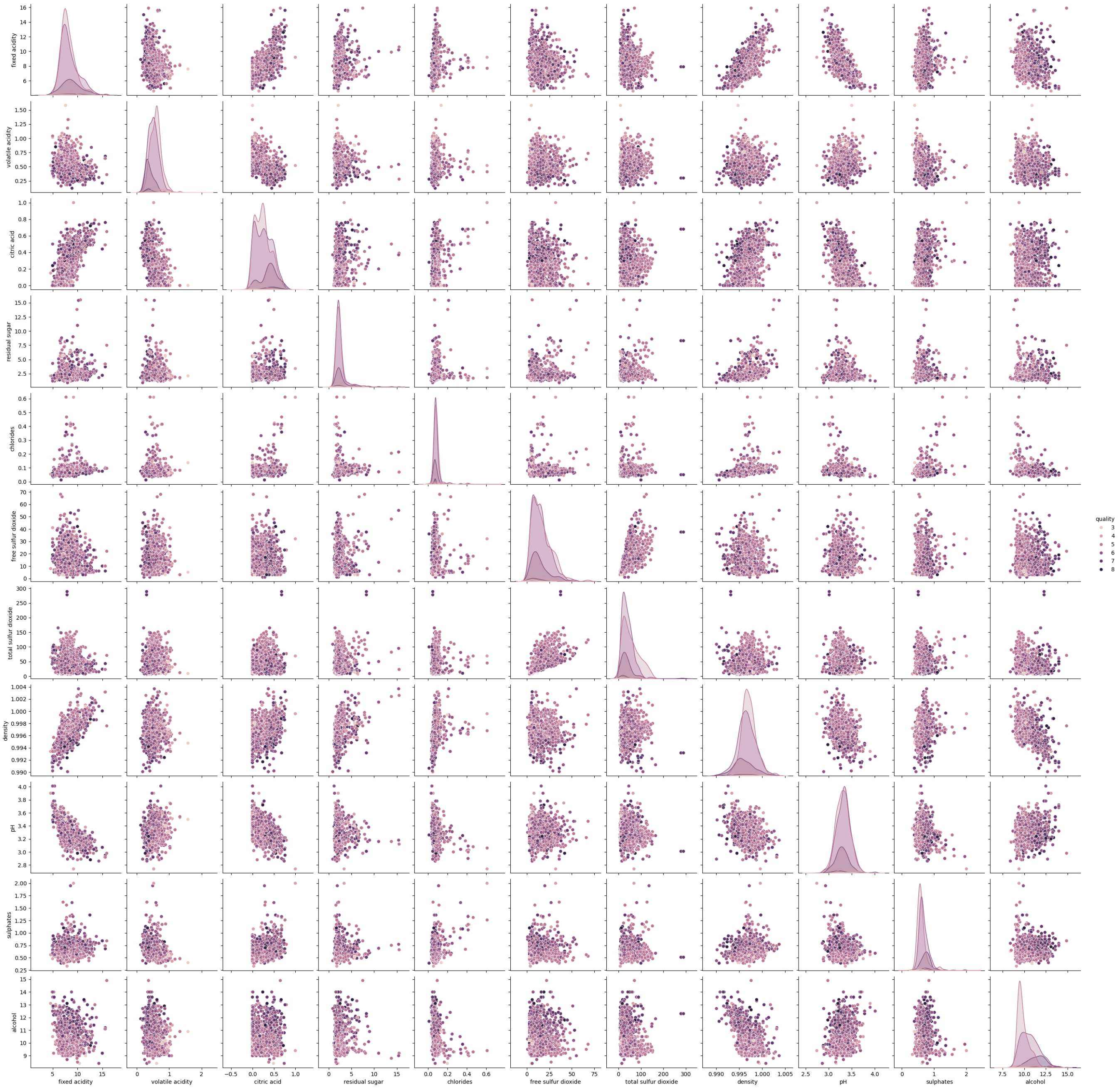
```
Stochastic Gradient Descent Classifier:
[[ 0  5  1  0  0]
 [ 0 94  2  0  0]
 [ 0 86  5  8  0]
 [ 0 19  1  6  0]
 [ 0  0  0  2  0]]
              precision    recall  f1-score   support

           4       0.00      0.00      0.00         6
           5       0.46      0.98      0.63        96
           6       0.56      0.05      0.09        99
           7       0.38      0.23      0.29        26
           8       0.00      0.00      0.00         2

    accuracy                           0.46       229
   macro avg       0.28      0.25      0.20       229
weighted avg       0.48      0.46      0.34       229

/usr/local/lib/python3.10/dist-packages/sklearn/metrics/_classification.py:1344: UndefinedMetricWarning: Precision and F-score are i
  _warn_prf(average, modifier, msg_start, len(result))
/usr/local/lib/python3.10/dist-packages/sklearn/metrics/_classification.py:1344: UndefinedMetricWarning: Precision and F-score are i
  _warn_prf(average, modifier, msg_start, len(result))
/usr/local/lib/python3.10/dist-packages/sklearn/metrics/_classification.py:1344: UndefinedMetricWarning: Precision and F-score are i
  _warn_prf(average, modifier, msg_start, len(result))
```

Data Visualization

Pairplot of features colored by quality

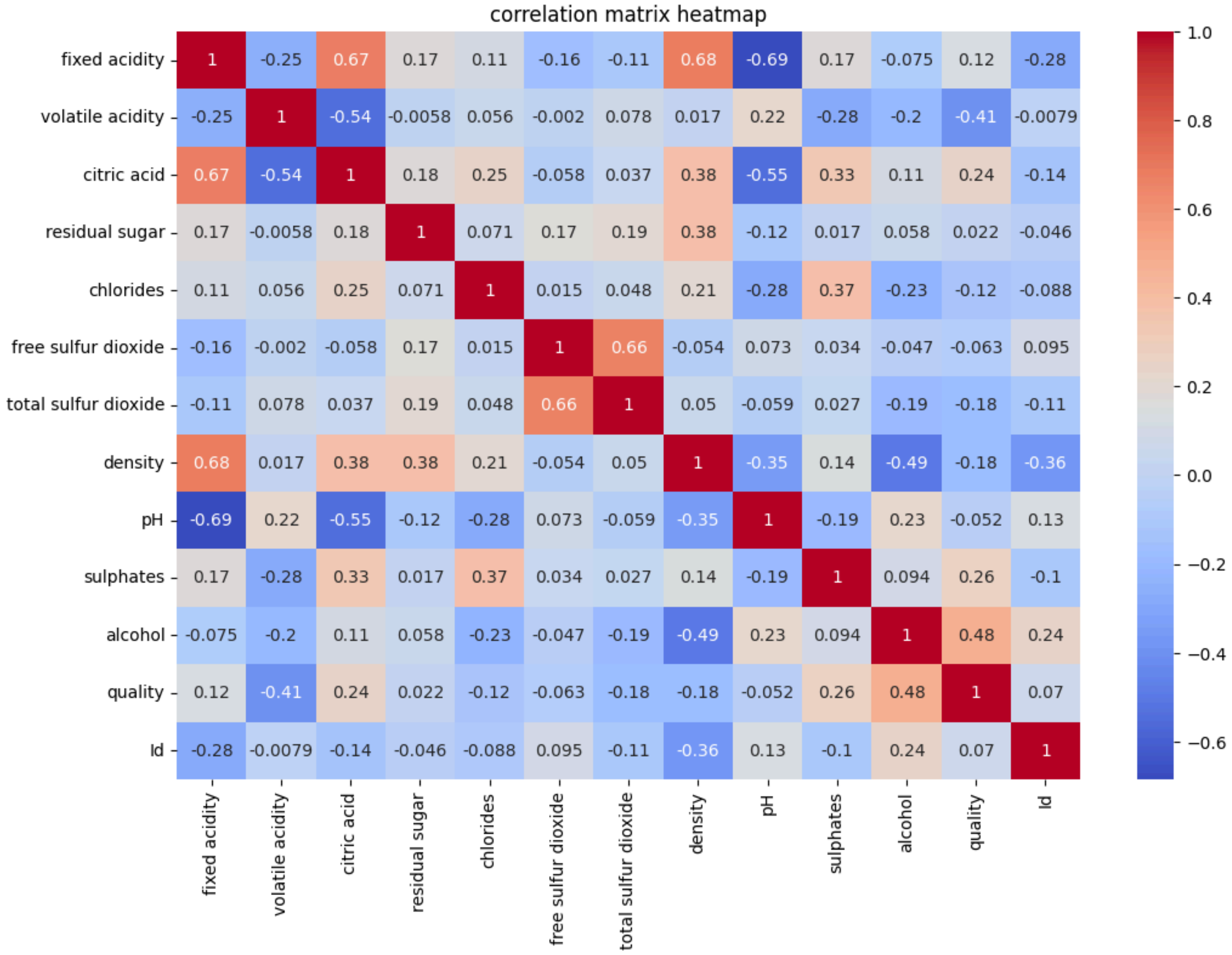
```
plt.figure(figsize=(12,8))
sns.pairplot(wq[features+['quality']],hue='quality')
plt.show()
```

```
<Figure size 1200x800 with 0 Axes>
```



Correlation Matrix heatmap

```
plt.figure(figsize=(12,8))
corr_matrix=wq.corr()
sns.heatmap(corr_matrix,annot=True,cmap='coolwarm')
plt.title('correlation matrix heatmap')
plt.show()
```



Distribution plot of quality

```
plt.figure(figsize=(10,6))
sns.countplot(x='quality',data=wq)
plt.title('Distribution of wine quality')
plt.show()
```