TITLE : Implement Agglomerative hierarchical clustering algorithm to predict the quality of wine. Use Wine Quality dataset from UCI Machine Learning repository.

NAME : Shinde Shubham Dnyandev,

ROLL NO : 23107121,

BATCH : B.

In [1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.cluster import AgglomerativeClustering
from scipy.cluster.hierarchy import linkage, dendrogram
```

In [3]:
```
white  = pd.read_csv("/home/admin1/winequality-white.csv", sep=';')
red  = pd.read_csv("/home/admin1/winequality-red.csv", sep=';')
```

In [5]:
```
white
```

Out[5]:

	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	pH	sulph
0	7.0	0.27	0.36	20.7	0.045	45.0	170.0	1.00100	3.00	
1	6.3	0.30	0.34	1.6	0.049	14.0	132.0	0.99400	3.30	
2	8.1	0.28	0.40	6.9	0.050	30.0	97.0	0.99510	3.26	
3	7.2	0.23	0.32	8.5	0.058	47.0	186.0	0.99560	3.19	
4	7.2	0.23	0.32	8.5	0.058	47.0	186.0	0.99560	3.19	
...	...	...	...	...	...	...	...	...	...	
4893	6.2	0.21	0.29	1.6	0.039	24.0	92.0	0.99114	3.27	
4894	6.6	0.32	0.36	8.0	0.047	57.0	168.0	0.99490	3.15	
4895	6.5	0.24	0.19	1.2	0.041	30.0	111.0	0.99254	2.99	
4896	5.5	0.29	0.30	1.1	0.022	20.0	110.0	0.98869	3.34	
4897	6.0	0.21	0.38	0.8	0.020	22.0	98.0	0.98941	3.26	

4898 rows × 12 columns

In [7]:
```
red
```

Out[7]:

	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	pH	sulph
0	7.4	0.700	0.00	1.9	0.076	11.0	34.0	0.99780	3.51	
1	7.8	0.880	0.00	2.6	0.098	25.0	67.0	0.99680	3.20	
2	7.8	0.760	0.04	2.3	0.092	15.0	54.0	0.99700	3.26	
3	11.2	0.280	0.56	1.9	0.075	17.0	60.0	0.99800	3.16	
4	7.4	0.700	0.00	1.9	0.076	11.0	34.0	0.99780	3.51	
...	...	...	...	...	...	...	...	...	...	
1594	6.2	0.600	0.08	2.0	0.090	32.0	44.0	0.99490	3.45	
1595	5.9	0.550	0.10	2.2	0.062	39.0	51.0	0.99512	3.52	
1596	6.3	0.510	0.13	2.3	0.076	29.0	40.0	0.99574	3.42	
1597	5.9	0.645	0.12	2.0	0.075	32.0	44.0	0.99547	3.57	
1598	6.0	0.310	0.47	3.6	0.067	18.0	42.0	0.99549	3.39	

1599 rows × 12 columns

In [9]:
```
df = pd.concat([white,red], axis=0)
df
```

Out[9]:

	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	pH	sulph
0	7.0	0.270	0.36	20.7	0.045	45.0	170.0	1.00100	3.00	
1	6.3	0.300	0.34	1.6	0.049	14.0	132.0	0.99400	3.30	
2	8.1	0.280	0.40	6.9	0.050	30.0	97.0	0.99510	3.26	
3	7.2	0.230	0.32	8.5	0.058	47.0	186.0	0.99560	3.19	
4	7.2	0.230	0.32	8.5	0.058	47.0	186.0	0.99560	3.19	
...	...	...	...	...	...	...	...	...	...	
1594	6.2	0.600	0.08	2.0	0.090	32.0	44.0	0.99490	3.45	
1595	5.9	0.550	0.10	2.2	0.062	39.0	51.0	0.99512	3.52	
1596	6.3	0.510	0.13	2.3	0.076	29.0	40.0	0.99574	3.42	
1597	5.9	0.645	0.12	2.0	0.075	32.0	44.0	0.99547	3.57	
1598	6.0	0.310	0.47	3.6	0.067	18.0	42.0	0.99549	3.39	

6497 rows × 12 columns

In [11]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 6497 entries, 0 to 1598
Data columns (total 12 columns):
 #   Column                Non-Null Count  Dtype  
---  ------                --------------  -----  
 0   fixed acidity         6497 non-null   float64
 1   volatile acidity      6497 non-null   float64
 2   citric acid           6497 non-null   float64
 3   residual sugar        6497 non-null   float64
 4   chlorides             6497 non-null   float64
 5   free sulfur dioxide   6497 non-null   float64
 6   total sulfur dioxide  6497 non-null   float64
 7   density               6497 non-null   float64
 8   pH                    6497 non-null   float64
 9   sulphates             6497 non-null   float64
 10  alcohol               6497 non-null   float64
 11  quality               6497 non-null   int64  
dtypes: float64(11), int64(1)
memory usage: 659.9 KB
```

In [13]:
```
df.describe()
```

Out[13]:

	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	t
count	6497.000000	6497.000000	6497.000000	6497.000000	6497.000000	6497.000000	6
mean	7.215307	0.339666	0.318633	5.443235	0.056034	30.525319	
std	1.296434	0.164636	0.145318	4.757804	0.035034	17.749400	
min	3.800000	0.080000	0.000000	0.600000	0.009000	1.000000	
25%	6.400000	0.230000	0.250000	1.800000	0.038000	17.000000	
50%	7.000000	0.290000	0.310000	3.000000	0.047000	29.000000	
75%	7.700000	0.400000	0.390000	8.100000	0.065000	41.000000	
max	15.900000	1.580000	1.660000	65.800000	0.611000	289.000000	

In [15]:
```
df.isnull().sum()
```

Out[15]:
```
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
```

In [17]:
```
X = df.drop('quality', axis=1)
Y = df['quality']
```

In [19]:
```
X
```

Out[19]:

	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	pH	sulph
0	7.0	0.270	0.36	20.7	0.045	45.0	170.0	1.00100	3.00	
1	6.3	0.300	0.34	1.6	0.049	14.0	132.0	0.99400	3.30	
2	8.1	0.280	0.40	6.9	0.050	30.0	97.0	0.99510	3.26	
3	7.2	0.230	0.32	8.5	0.058	47.0	186.0	0.99560	3.19	
4	7.2	0.230	0.32	8.5	0.058	47.0	186.0	0.99560	3.19	
...	...	...	...	...	...	...	...	...	...	
1594	6.2	0.600	0.08	2.0	0.090	32.0	44.0	0.99490	3.45	
1595	5.9	0.550	0.10	2.2	0.062	39.0	51.0	0.99512	3.52	
1596	6.3	0.510	0.13	2.3	0.076	29.0	40.0	0.99574	3.42	
1597	5.9	0.645	0.12	2.0	0.075	32.0	44.0	0.99547	3.57	
1598	6.0	0.310	0.47	3.6	0.067	18.0	42.0	0.99549	3.39	

6497 rows × 11 columns

In [21]:
```
Y
```

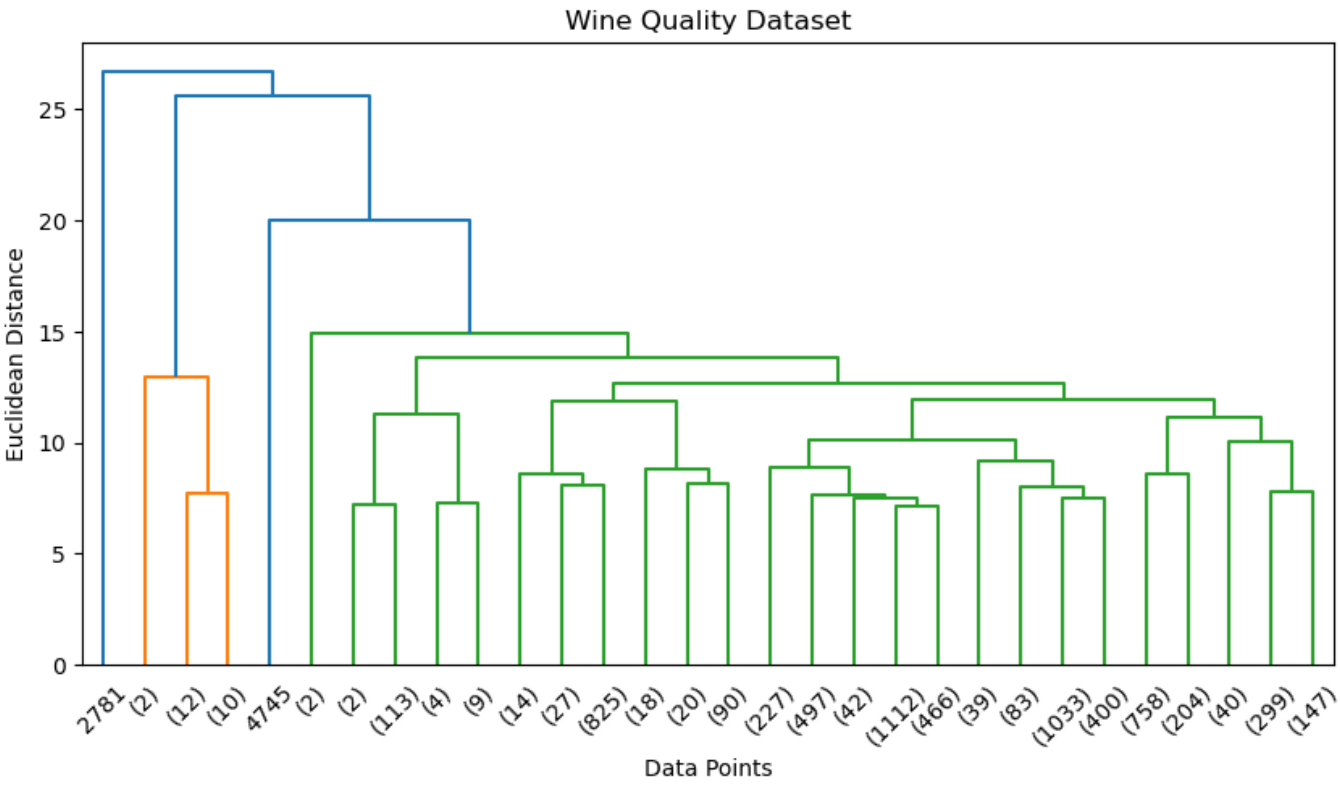
Out[21]:
```
0       6
1       6
2       6
3       6
4       6
       ..
1594    5
1595    6
1596    6
1597    5
1598    6
Name: quality, Length: 6497, dtype: int64
```

In [23]:
```
from sklearn.preprocessing import StandardScaler
SS = StandardScaler()

X_Scaled = SS.fit_transform(X)
```

In [25]:
```
linked = linkage(X_Scaled, method='complete')
```

In [27]:
```
plt.figure(figsize=(10, 5))
dendrogram(linked, truncate_mode='lastp', p=30)
plt.title("Wine Quality Dataset")
plt.xlabel("Data Points")
plt.ylabel("Euclidean Distance")
plt.show()
```



In [29]:
```
HC = AgglomerativeClustering(
    n_clusters = 3,
    metric = 'euclidean',
    linkage = 'complete'
)
```

In [31]:
```
df['cluster'] = HC.fit_predict(X_Scaled)
```

In [33]:
```
print("\nAverage Quality per Cluster")
print(df.groupby('cluster')['quality'].mean())
```

```
Average Quality per Cluster
cluster
0    5.820303
1    6.000000
2    5.291667
Name: quality, dtype: float64
```