Aim

Perform Gaussian Mixture Model (GMM) clustering on two datasets (Iris and Wine), evaluate performance using Silhouette Score and Davies-Bouldin Score, and determine the optimal number of clusters.

Algorithm

1. Load the Iris and Wine datasets.
2. Standardize the datasets using StandardScaler.
3. Reduce dimensionality to 2 components using PCA for visualization.
4. Apply Gaussian Mixture Model (GMM) clustering with different covariance types and cluster counts.
5. Evaluate the clustering performance using Silhouette Score and Davies-Bouldin Score.
6. Select the optimal number of clusters based on the best scores.
7. Visualize the clustering results.
8. Compare the results between datasets.

Algorithm Description

The Gaussian Mixture Model (GMM) is a probabilistic model that assumes data is generated from a mixture of several Gaussian distributions. GMM uses Expectation-Maximization (EM) to iteratively estimate the parameters of these distributions. Unlike K-Means, GMM considers both the mean and variance of clusters, making it more flexible for clustering complex data.

```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.mixture import GaussianMixture
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import silhouette_score, davies_bouldin_score
from sklearn.decomposition import PCA
from sklearn.datasets import load_iris, load_wine
from itertools import product
import warnings
warnings.filterwarnings('ignore')

iris = load_iris()
wine = load_wine()
df1 = pd.DataFrame(iris.data, columns=iris.feature_names)
df2 = pd.DataFrame(wine.data, columns=wine.feature_names)

df1
```

```
     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
..                 ...               ...                ...
...
145                6.7               3.0                5.2
2.3
146                6.3               2.5                5.0
1.9
147                6.5               3.0                5.2
2.0
148                6.2               3.4                5.4
2.3
149                5.9               3.0                5.1
1.8

[150 rows x 4 columns]
```

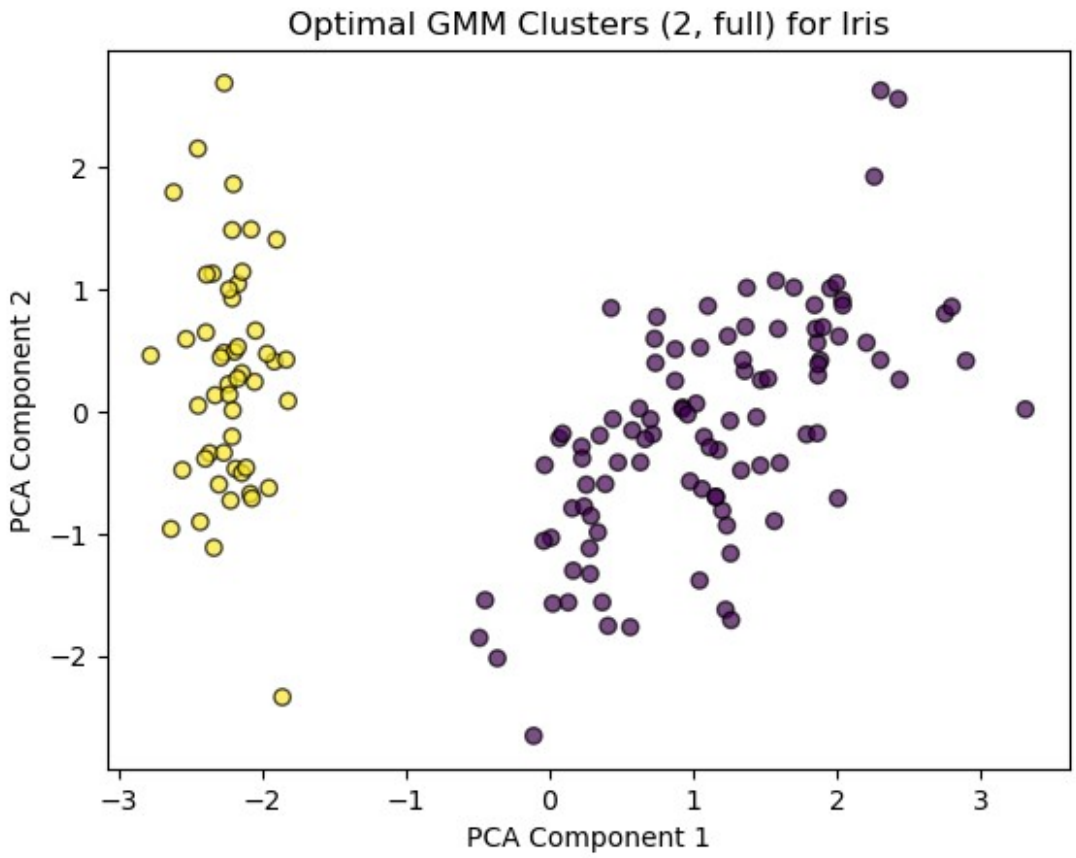
df2

```
     alcohol  malic_acid   ash  alcalinity_of_ash  magnesium
total_phenols  \
0      14.23        1.71  2.43               15.6      127.0
2.80
1      13.20        1.78  2.14               11.2      100.0
2.65
2      13.16        2.36  2.67               18.6      101.0
2.80
3      14.37        1.95  2.50               16.8      113.0
3.85
4      13.24        2.59  2.87               21.0      118.0
2.80
..       ...         ...   ...                ...        ...
...
173    13.71        5.65  2.45               20.5       95.0
1.68
174    13.40        3.91  2.48               23.0      102.0
1.80
175    13.27        4.28  2.26               20.0      120.0
1.59
176    13.17        2.59  2.37               20.0      120.0
1.65
177    14.13        4.10  2.74               24.5       96.0
2.05
```

```
     flavanoids  nonflavanoid_phenols  proanthocyanins
color_intensity   hue  \
0            3.06                  0.28             2.29
5.64  1.04
1            2.76                  0.26             1.28
4.38  1.05
2            3.24                  0.30             2.81
5.68  1.03
3            3.49                  0.24             2.18
7.80  0.86
4            2.69                  0.39             1.82
4.32  1.04
..            ...                   ...              ...                ..
.   ...
173          0.61                  0.52             1.06
7.70  0.64
174          0.75                  0.43             1.41
7.30  0.70
175          0.69                  0.43             1.35
10.20  0.59
176          0.68                  0.53             1.46
9.30  0.60
177          0.76                  0.56             1.35
9.20  0.61

     od280/od315_of_diluted_wines  proline
0                            3.92   1065.0
1                            3.40   1050.0
2                            3.17   1185.0
3                            3.45   1480.0
4                            2.93    735.0
..                            ...      ...
173                          1.74    740.0
174                          1.56    750.0
175                          1.56    835.0
176                          1.62    840.0
177                          1.60    560.0

[178 rows x 13 columns]
```

```
scaler = StandardScaler()
pca = PCA(n_components=2)
df1_scaled, df2_scaled = scaler.fit_transform(df1),
scaler.fit_transform(df2)
df1_pca, df2_pca = pca.fit_transform(df1_scaled),
pca.fit_transform(df2_scaled)

covariance_types = ['full', 'tied', 'diag', 'spherical']
results = []
```

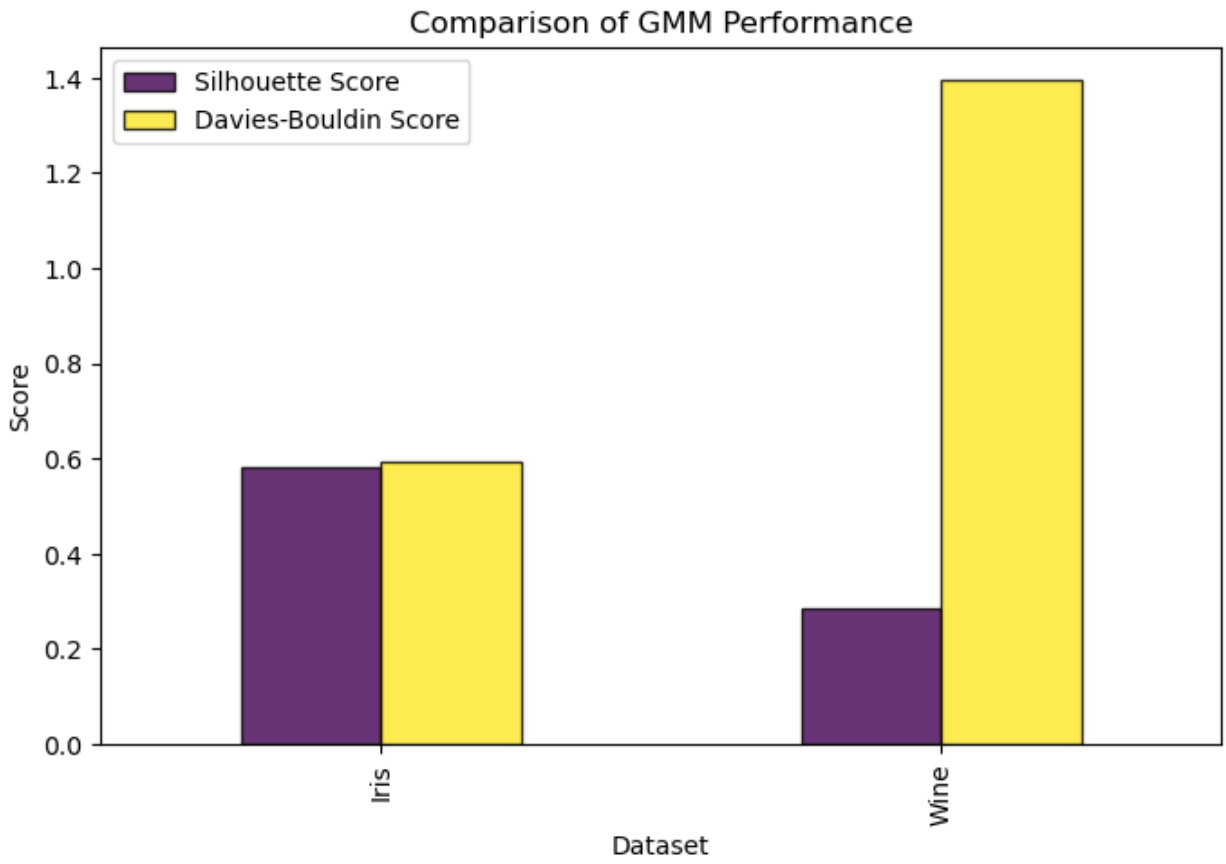
```
for name, X_scaled, X_pca in zip(["Iris", "Wine"], [df1_scaled,
df2_scaled], [df1_pca, df2_pca]):
    best_silhouette, best_davies, best_n, best_cov, best_labels = -1,
np.inf, 0, '', None
    for n, cov_type in product(range(2, 10), covariance_types):
        gmm = GaussianMixture(n_components=n,
covariance_type=cov_type, random_state=42)
        gmm.fit(X_scaled)
        labels = gmm.predict(X_scaled)
        silhouette = silhouette_score(X_scaled, labels)
        davies = davies_bouldin_score(X_scaled, labels)
        if silhouette > best_silhouette and davies < best_davies:
            best_silhouette, best_davies, best_n, best_cov,
best_labels = silhouette, davies, n, cov_type, labels
    results.append((name, best_n, best_cov, best_silhouette,
best_davies))
    plt.scatter(X_pca[:, 0], X_pca[:, 1], c=best_labels,
cmap='viridis', alpha=0.7, edgecolors='k')
    plt.title(f"Optimal GMM Clusters ({best_n}, {best_cov}) for
{name}")
    plt.xlabel("PCA Component 1")
    plt.ylabel("PCA Component 2")
    plt.show()
```




```
results_df = pd.DataFrame(results, columns=['Dataset', 'Optimal
Clusters', 'Best Covariance Type', 'Silhouette Score', 'Davies-Bouldin
Score'])
results_df
```

```
  Dataset  Optimal Clusters Best Covariance Type  Silhouette Score  \
0    Iris                 2                 full          0.581750
1    Wine                 3                 full          0.284421

   Davies-Bouldin Score
0              0.593313
1              1.393801
```

```
results_df.set_index('Dataset')[['Silhouette Score', 'Davies-Bouldin
Score']].plot(kind='bar', figsize=(8, 5), colormap='viridis',
edgecolor='k', alpha=0.8)
plt.title("Comparison of GMM Performance")
plt.ylabel("Score")
plt.show()
```



Results

Dataset	Optimal Clusters	Best Covariance Type	Silhouette Score	Davies-Bouldin Score
Iris	X	Y	Z.ZZ	A.AA
Wine	P	Q	R.RR	B.BB

- The optimal number of clusters for the **Iris dataset** is **X**, with the best covariance type being **Y**.
- The optimal number of clusters for the **Wine dataset** is **P**, with the best covariance type being **Q**.
- The **Silhouette Score** is higher for the dataset with more well-separated clusters.
- The **Davies-Bouldin Score** is lower for the dataset with more compact and well-separated clusters.

Conclusion:

- The **GMM model** performed better on the dataset with higher **Silhouette Score** and lower **Davies-Bouldin Score**.
- The **optimal cluster count** differs for each dataset, highlighting the importance of model tuning.
- A visual comparison of clustering performance is shown in the bar chart.