Exercise 6a   Date: 29/01/2025

Gausian Mixture Model - Two-dimensional dataset

Aim: To apply the Gaussian Mixture Model (GMM) clustering algorithm to a two-dimensional dataset, to identify clusters within the data.

Algorithm:

The Gaussian Mixture Model (GMM) is a probabilistic clustering algorithm that assumes the data is generated from a mixture of several Gaussian distributions with unknown parameters. Unlike K-Means, which assigns points to the nearest cluster centroid, GMM assigns probabilities to each data point belonging to a particular cluster.

The probability of a data point x_i belonging to cluster k is given by:

$$P(z_i = k|x_i) = \frac{\pi_k \mathcal{N}(x_i|\mu_k, \Sigma_k)}{\sum_{j=1}^{K} \pi_j \mathcal{N}(x_i|\mu_j, \Sigma_j)}$$

Where:

- $\mathcal{N}(x_i|\mu_k, \Sigma_k)$ is the Gaussian distribution with mean μ_k and covariance Σ_k.
- π_k is the mixing coefficient for cluster k.
- K is the total number of clusters.

Step 1: Import Libraries

- Import necessary Python libraries such as NumPy, Matplotlib, and Scikit-learn.

Step 2: Load the Dataset

- Load the Iris dataset and extract only the sepal length and sepal width features (bivariate dataset).

Step 3: Scale the Data

- Normalize the dataset using StandardScaler to ensure that features contribute equally to clustering.

Step 4: Apply GMM

- Initialize the Gaussian Mixture Model with n_components=3 to find three clusters.
- Fit the model to the scaled data and predict cluster labels.

Step 5: Visualize the Clusters

- Plot the clustered data points with distinct colors.
- Display a color bar to indicate the assigned clusters.
- Label the axes and provide a title for the graph.

Import the libraries

In [19]:
```python
import numpy as np
import matplotlib.pyplot as plt
from sklearn import datasets
import warnings
from sklearn.mixture import GaussianMixture
from sklearn.preprocessing import StandardScaler
warnings.filterwarnings("ignore")
```

Load the Dataset

In [20]:
```python
iris = datasets.load_iris()
X = iris.data[:, :2]
```

Scale the data

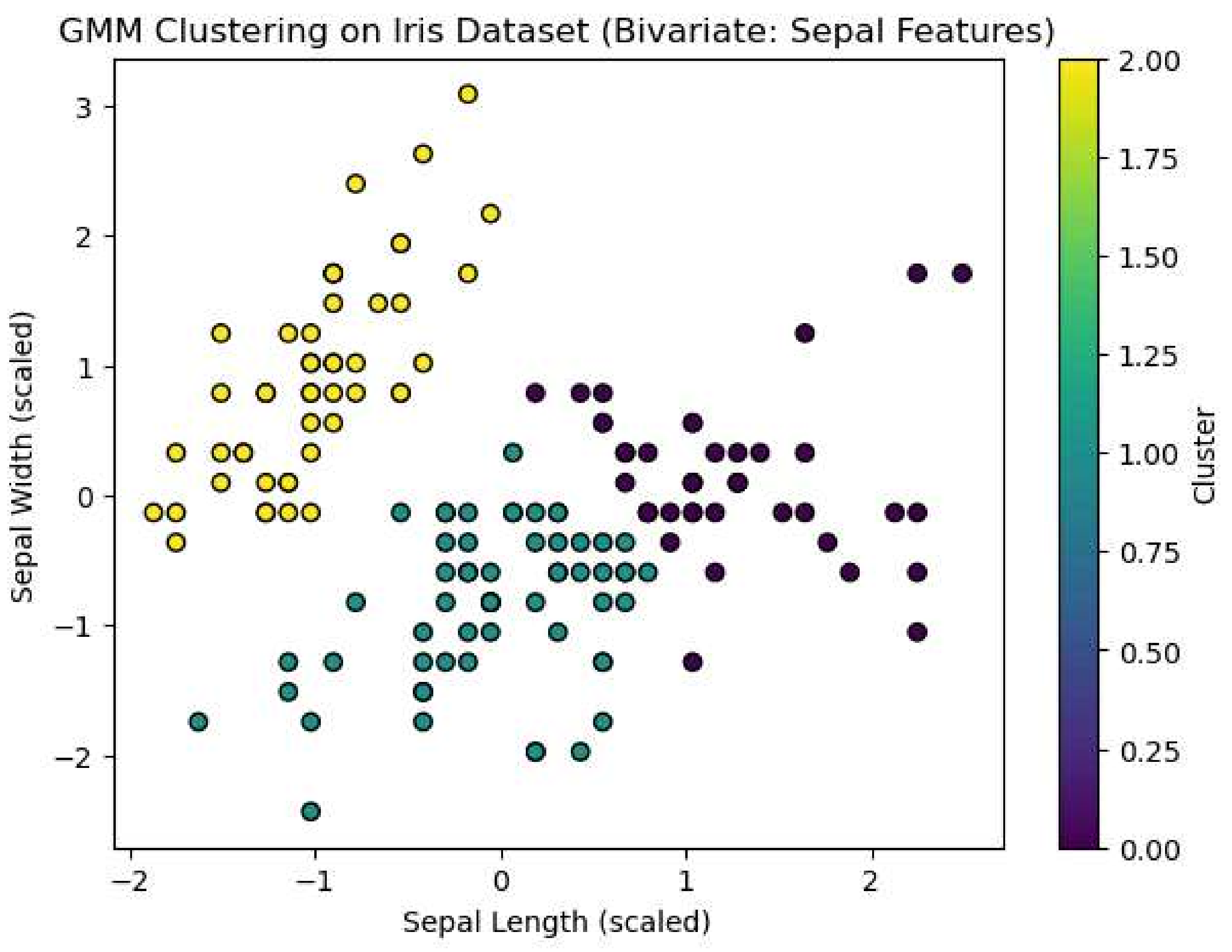
In [21]:
```python
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)
```

Apply the Gausian Mixture Model

```
In [22]: gmm = GaussianMixture(n_components=3, random_state=42)
         gmm_labels = gmm.fit_predict(X_scaled)
```

Plot the graph

```
In [23]: plt.figure(figsize=(7,5))
         plt.scatter(X_scaled[:, 0], X_scaled[:, 1], c=gmm_labels, cmap='viridis', edgecolors='k')
         plt.colorbar(label='Cluster')
         plt.xlabel('Sepal Length (scaled)')
         plt.ylabel('Sepal Width (scaled)')
         plt.title('GMM Clustering on Iris Dataset (Bivariate: Sepal Features)')
         plt.show()
```



Result

The Gaussian Mixture Model successfully clustered the two-dimensional dataset into three clusters based on probability distributions.

Exercise 6b Date: 29/01/2025

Gausian Mixture Model - Multidimensional dataset

<u>Aim:</u> To apply the Gaussian Mixture Model (GMM) clustering algorithm to a multidimensional dataset, to identify clusters within the data.

<u>Algorithm:</u>

The Gaussian Mixture Model (GMM) is a probabilistic clustering algorithm that assumes the data is generated from a mixture of several Gaussian distributions with unknown parameters. Unlike K-Means, which assigns points to the nearest cluster centroid, GMM assigns probabilities to each data point belonging to a particular cluster.

The probability of a data point x_i belonging to cluster k is given by:

$$P(z_i = k|x_i) = \frac{\pi_k \mathcal{N}(x_i|\mu_k, \Sigma_k)}{\sum_{j=1}^{K} \pi_j \mathcal{N}(x_i|\mu_j, \Sigma_j)}$$

Where:

- $\mathcal{N}(x_i|\mu_k, \Sigma_k)$ is the Gaussian distribution with mean μ_k and covariance Σ_k.
- π_k is the mixing coefficient for cluster k.
- K is the total number of clusters.

Step 1: Import Libraries

- Import necessary Python libraries such as NumPy, Matplotlib, and Scikit-learn.

Step 2: Load the Dataset

- Load the Iris dataset and extract all four features (multidimensional dataset).

Step 3: Scale the Data

- Normalize the dataset using StandardScaler to ensure that features contribute equally to clustering.

Step 4: Dimensionality Reduction using PCA

- Apply Principal Component Analysis (PCA) to reduce the dimensionality to 2D for visualization purposes.

Step 5: Apply GMM

- Initialize the Gaussian Mixture Model with n_components=3 to find three clusters.
- Fit the model to the scaled data and predict cluster labels.

Step 6: Visualize the Clusters

- Plot the clustered data points in 2D using PCA-reduced features.
- Display a 3D scatter plot using three original scaled features.
- Label the axes and provide a title for the graphs.

Import the libraries

```
In [46]: import numpy as np
         import matplotlib.pyplot as plt
         from sklearn import datasets
         import warnings
         from sklearn.mixture import GaussianMixture
         from sklearn.preprocessing import StandardScaler
         from sklearn.decomposition import PCA
         warnings.filterwarnings("ignore")
```

Load the Dataset

```
In [47]: iris = datasets.load_iris()
         X = iris.data
```

Scale the data

```
In [48]: scaler = StandardScaler()
         X_scaled = scaler.fit_transform(X)
```

Dimensionality Reduction using PCA

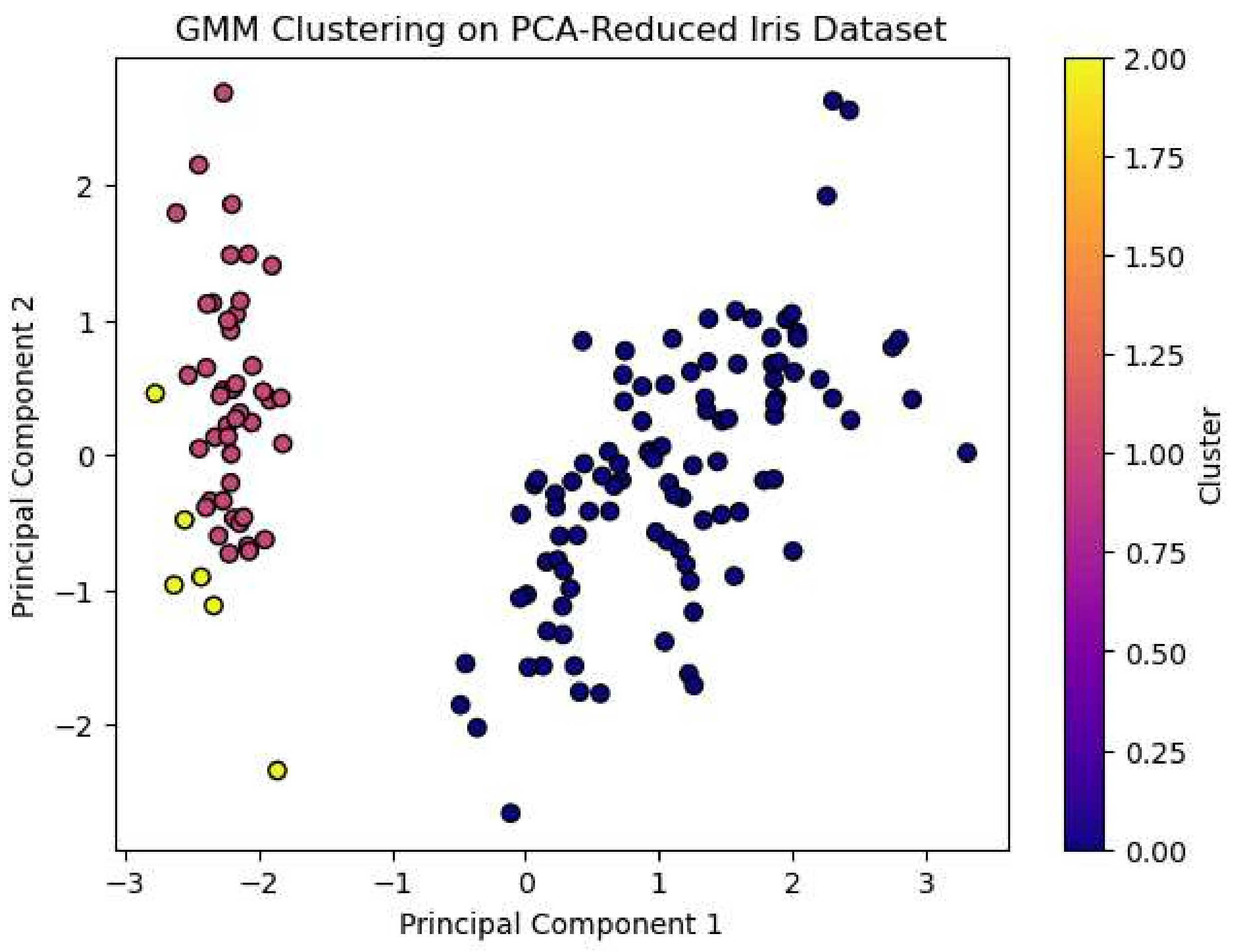
In [42]:
```
pca = PCA(n_components=2)
X_pca = pca.fit_transform(X_scaled)
```

Apply Gaussian Mixture Model

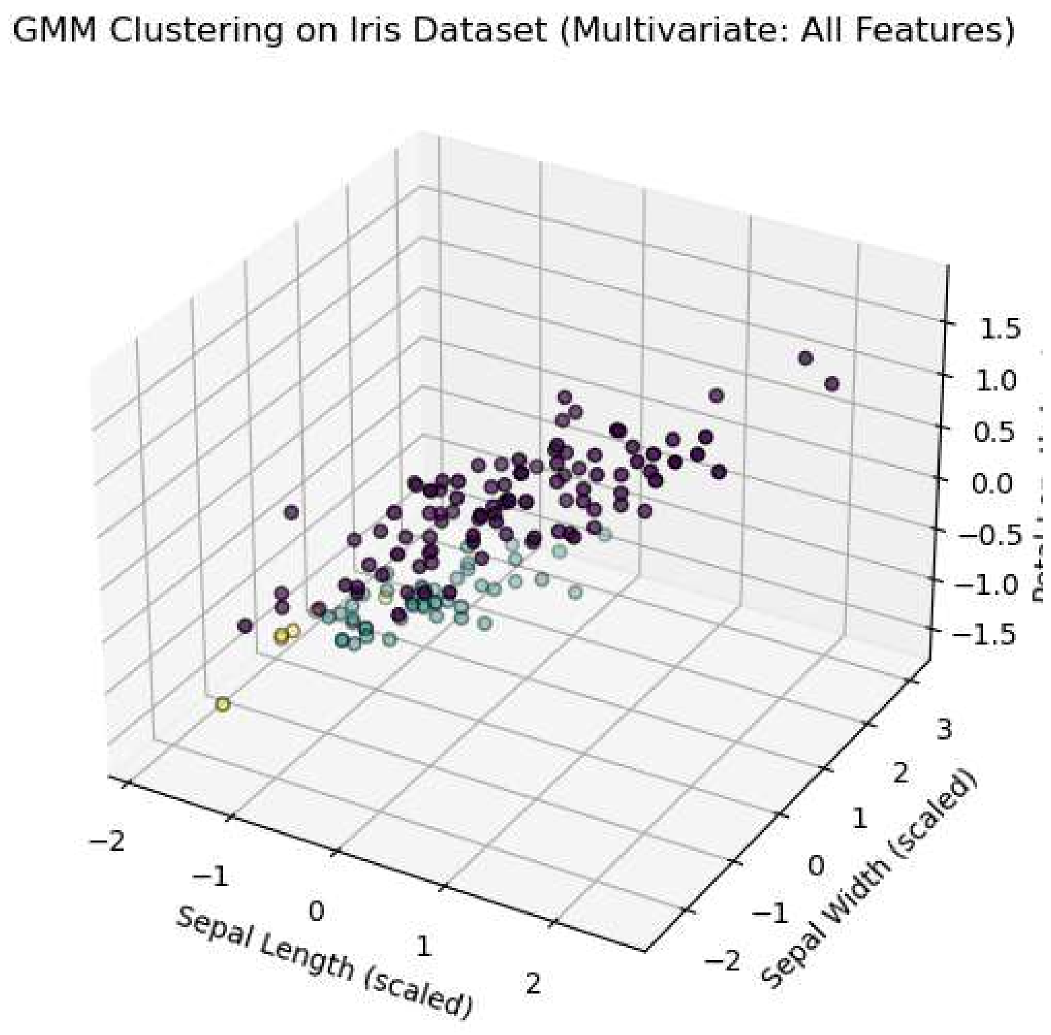
In [49]:
```
gmm = GaussianMixture(n_components=3, random_state=42)
gmm_labels = gmm.fit_predict(X_pca)
```

Plot the graphs

In [50]:
```
plt.figure(figsize=(7,5))
plt.scatter(X_pca[:, 0], X_pca[:, 1], c=gmm_labels, cmap='plasma', edgecolors='k')
plt.colorbar(label='Cluster')
plt.xlabel('Principal Component 1')
plt.ylabel('Principal Component 2')
plt.title('GMM Clustering on PCA-Reduced Iris Dataset')
plt.show()
```



In [51]:
```
fig = plt.figure(figsize=(8,6))
ax = fig.add_subplot(111, projection='3d')
ax.scatter(X_scaled[:, 0], X_scaled[:, 1], X_scaled[:, 2], c=gmm_labels, cmap='viridis', edgecolors='k')
ax.set_xlabel('Sepal Length (scaled)')
ax.set_ylabel('Sepal Width (scaled)')
ax.set_zlabel('Petal Length (scaled)')
ax.set_title('GMM Clustering on Iris Dataset (Multivariate: All Features)')
plt.show()
```



Result

The Gaussian Mixture Model successfully clustered the multidimensional dataset into three clusters based on probability distributions.

Comparison with K-Means

- Soft Clustering: Unlike K-Means, which assigns each point to a single cluster, GMM assigns probabilities, allowing for overlapping clusters.
- Elliptical Clusters: K-Means assumes clusters are spherical, whereas GMM models them as ellipses using covariance matrices.
- More Flexible Boundaries: GMM results in more flexible decision boundaries, making it more suitable for complex datasets where clusters are not well separated.
- Computation Time: GMM is generally more computationally expensive due to iterative probability updates, unlike K-Means which uses direct centroid assignments.