Assignment: K-Means Clustering

Aim

To perform K-Means clustering on a given set of 2D data points, identify clusters, compute cluster centroids, and visualize the clustered data.

Lab Setup

- Software: Python 3.x (Google Colab, Jupyter Notebook, or any Python IDE)
- Required libraries: **numpy**, **matplotlib**, **scikit-learn**
- Installation command (if needed):

bash
pip install numpy matplotlib scikit-learn

- Dataset: Sample 2D points defined within the code.

Input

- A predefined set of 2D data points:

python
X = np.array([[1, 2], [1, 4], [1, 0],
[4, 2], [4, 4], [4, 0]])

- Number of clusters, k
  =
  2

  k=2.

Expected Output

- Coordinates of the cluster centroids.
- Cluster labels for each data point.
- A scatter plot displaying the clusters with different colors and centroids marked distinctly.

Example textual output:

text
Cluster Centers:
[[1. 2.]
[4. 2.]]
Labels for each point:
[0 0 0 1 1 1]

Theory / Algorithm

K-Means clustering is an unsupervised algorithm that partitions data points into
k

k clusters by minimizing the within-cluster variance. The algorithm operates as follows:

1. Initialize k

k centroids randomly.
2. Assign each data point to the nearest centroid based on Euclidean distance.
3. Recalculate centroids by computing the mean of all points assigned to each cluster.
4. Repeat steps 2 and 3 until the cluster assignments no longer change or maximum iterations are reached.

This iterative process converges to clusters that minimize intra-cluster distances and maximize separation between clusters.

Code

```
python
import numpy as np
import matplotlib.pyplot as plt
from sklearn.cluster import KMeans
# Sample 2D data points
X = np.array([[1, 2], [1, 4], [1, 0],
              [4, 2], [4, 4], [4, 0]])
# Number of clusters
k = 2
kmeans = KMeans(n_clusters=k, random_state=42)
# Fit the model to the data
kmeans.fit(X)
# Get cluster centers and labels
centers = kmeans.cluster_centers_
labels = kmeans.labels_
print("Cluster Centers:")
print(centers)
print("Labels for each point:")
print(labels)
# Plot data points colored by cluster assignment
plt.scatter(X[:, 0], X[:, 1], c=labels, s=50, cmap='viridis')
plt.scatter(centers[:, 0], centers[:, 1], c='red', s=200, alpha=0.75, marker='X', label='Centers')
plt.title("K-Means Clustering")
plt.xlabel("Feature 1")
plt.ylabel("Feature 2")
plt.legend()
plt.show()
```

Conclusion

The K-Means algorithm successfully grouped the data points into two clusters based on their proximity. The centroids represent the mean position of points in each cluster, and the scatter plot clearly visualizes these clusters with distinct colors and marked centers. This lab illustrates the effectiveness of K-Means clustering in classifying unlabeled data based on feature similarity.