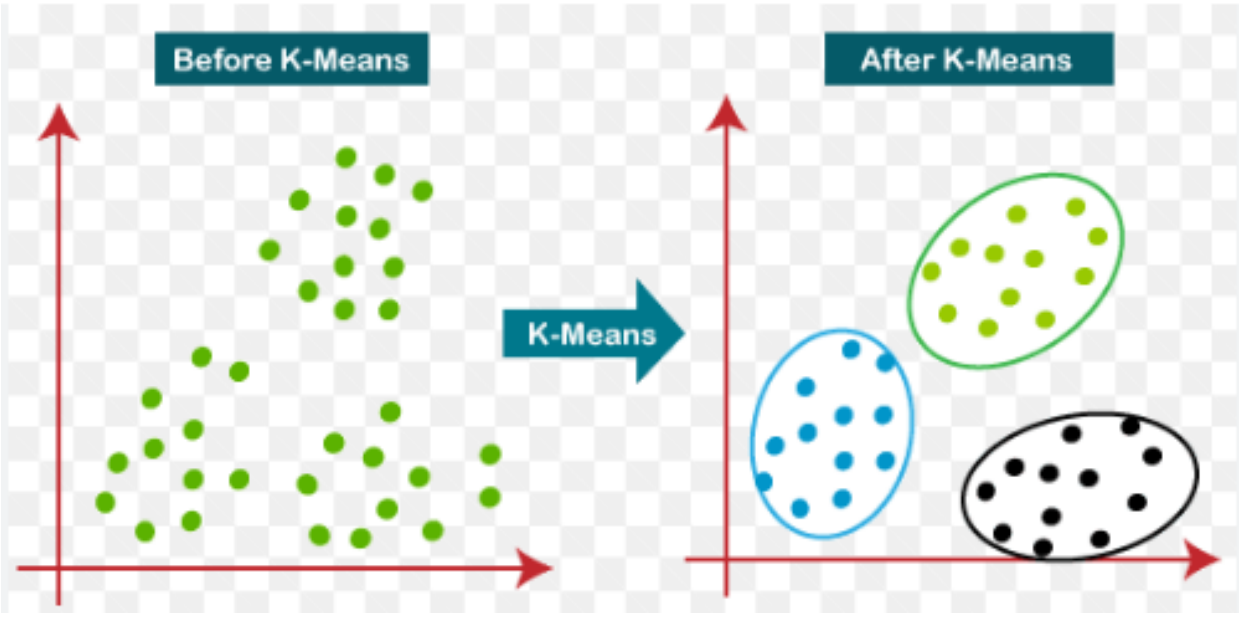
LAB Assignment No 10

K means Clustering Schemes in Machine Learning

In this lab, students will learn and implement K-Means Clustering, an unsupervised machine learning algorithm used to group data points into K distinct clusters based on similarity. Unlike supervised learning methods, K-Means does not require labeled data. Students will first apply K-Means on a small numerical dataset to understand clustering behavior, then use it on real-world datasets to visualize cluster formation and analyze results. Model performance will be interpreted using visualization and cluster characteristics.

Introduction

K-Means Clustering is an **unsupervised learning algorithm** that partitions a dataset into **K clusters**, where each data point belongs to the cluster with the nearest mean (centroid).



Working of K-Means:

1. Select the number of clusters **K**
2. Initialize K centroids randomly

3. Assign each data point to the nearest centroid
4. Update centroids by computing the mean of assigned points
5. Repeat steps 3 and 4 until convergence

Key Concepts:

- **Centroid** – center of a cluster
- **Inertia** – sum of squared distances within clusters
- **Elbow Method** – technique to choose optimal K

Applications:

- Customer segmentation
- Image compression
- Market basket analysis
- Document clustering

Solved Examples

Example 1: K-Means Clustering on a Simple Dataset

Apply K-Means clustering to group students based on **marks and attendance**

Solution:

```
import numpy as np
import matplotlib.pyplot as plt
from sklearn.cluster import KMeans

# Sample data: Marks vs Attendance
X = np.array([
    [40, 60],
    [45, 65],
    [70, 80],
    [75, 85],
```

```
    [90, 95],
    [85, 90]
])
# Apply K-Means
kmeans = KMeans(n_clusters=2, random_state=42)
kmeans.fit(X)
labels = kmeans.labels_

# Plot clusters
plt.scatter(X[:,0], X[:,1], c=labels)
plt.scatter(kmeans.cluster_centers_[:,0],      kmeans.cluster_centers_[:,1],
marker='X')
plt.xlabel("Marks")
plt.ylabel("Attendance")
plt.title("K-Means Clustering (K=2)")
plt.show()
```

Explanation

The algorithm groups students into clusters based on similarity in marks and attendance.

Example 2: Choosing Optimal K Using Elbow Method

Use the **Elbow Method** to determine the optimal number of clusters.

Solution:

```
inertia = []

for k in range(1, 6):
    kmeans = KMeans(n_clusters=k, random_state=42)
    kmeans.fit(X)
    inertia.append(kmeans.inertia_)

# Plot Elbow Curve
plt.plot(range(1,6), inertia, marker='o')
```

```
plt.xlabel("Number of Clusters (K)")
plt.ylabel("Inertia")
plt.title("Elbow Method")
plt.show()
```

Explanation

The “elbow point” in the graph suggests the best value of **K** where further increase does not significantly reduce inertia.

Example 3: K-Means Clustering on Iris Dataset

Apply K-Means clustering to the **Iris dataset** and visualize the clusters.

Solution:

```
from sklearn.datasets import load_iris

# Load Iris dataset
iris = load_iris()
X = iris.data[:, :2]  # Use first two features for visualization

# Apply K-Means
kmeans = KMeans(n_clusters=3, random_state=42)
labels = kmeans.fit_predict(X)

# Visualize clusters
plt.scatter(X[:,0], X[:,1], c=labels)
plt.scatter(kmeans.cluster_centers_[:,0], kmeans.cluster_centers_[:,1],
            marker='X')
plt.xlabel("Sepal Length")
plt.ylabel("Sepal Width")
plt.title("K-Means Clustering on Iris Dataset")
plt.show()
```

Explanation

K-Means successfully groups data into three clusters corresponding to Iris species.

Lab Assignment No. 10

Question 1

Write Python code to implement K-Means clustering from scratch using the following data points:

P1(1,3), P2(2,2), P3(5,8), P4(8,5), P5(3,9), P6(10,7), P7(3,3), P8(9,4), P9(3,7)

Tasks:

1. Use **K = 3** clusters
2. Initialize centroids as
    - C1 = P7(3,3)
    - C2 = P9(3,7)
    - C3 = P8(9,4)
3. Perform **2 iterations** manually in code
4. Plot all points and centroids
5. **Label all points (P1…P9)**
6. Sketch the clusters using matplot library in python

Question No. 2

Use the scikit-learn KMeans() library to cluster the same points.

P1(1,3), P2(2,2), P3(5,8), P4(8,5), P5(3,9), P6(10,7), P7(3,3), P8(9,4), P9(3,7)

Tasks:

1. Use K = 2, 3, and 4
2. Plot the clustering result for each K
3. Compare:
    - Number of points in each cluster
    - Final centroid locations
4. Draw the 3 graphs in your lab copy and explain how the shapes change with K.

Question 3 — Add a New User Point and Re-Cluster

Given the original 9 points, **add a new user**: **P10(6,2)**

Tasks:

1. Run K-Means using K = 3
2. Plot the graph with all 10 points
3. Identify:
    - Which cluster P10 joins
    - How centroids shift after adding P10
4. Sketch before/after clusters in notebook
5. Write a short explanation about how a new data point affects clustering.

Question 4 — Distance Table + First Iteration Manually

Using the given 9 points and initial centroids:

C1(3,3), C2(3,7), C3(9,4)

Tasks:

1. Compute **Euclidean distance** of each point to each centroid (manually or in python)
2. Create a distance table:

Point Dist to C1 Dist to C2 Dist to C3 Assigned Cluster

3. Perform **only the first iteration**
4. Compute **new centroids**
5. Plot the **first-iteration graph**
6. Draw the graph in your copy and show all labels.

LAB Assessment

Student Name		**LAB Rubrics**	CLO3 , P5, PLO5

		Total Marks	10
Registration No		**Obtained Marks**	
		Teacher Name	Dr. Syed M Hamedoon
Date		**Signature**	