Practical No. 9

Submitted By:		
Name: Mahak Mor	**Roll No.:** 20	**Section:** A

Aim: To implement K-Means clustering.

Theory: K-means is one of the simplest and most popular unsupervised machine learning algorithms used for clustering. It is a partitioning method where the goal is to divide the dataset into K distinct, non-overlapping subsets (clusters) based on the similarity of data points. It aims to minimize the variance within each cluster and maximize the variance between different clusters.

Steps in K-Means Clustering:

1. Initialize K centroids randomly.
2. Assign each data point to the nearest centroid to form K clusters.
3. Recalculate the centroids by taking the mean of all data points in each cluster.
4. Repeat the process of assigning and recalculating until the centroids stabilize or reach a predefined number of iterations.

Key Features:

- Centroid: The center of a cluster.
- Intra-cluster distance: Distance between data points within the same cluster.
- Inter-cluster distance: Distance between different clusters.

Advantages:

- Simple and fast for small datasets.
- Efficient for clustering when the number of clusters is known.

Disadvantages:

- Sensitive to the initial placement of centroids.
- The number of clusters (K) must be predefined.
- May not perform well for non-spherical clusters.

Code and Output:

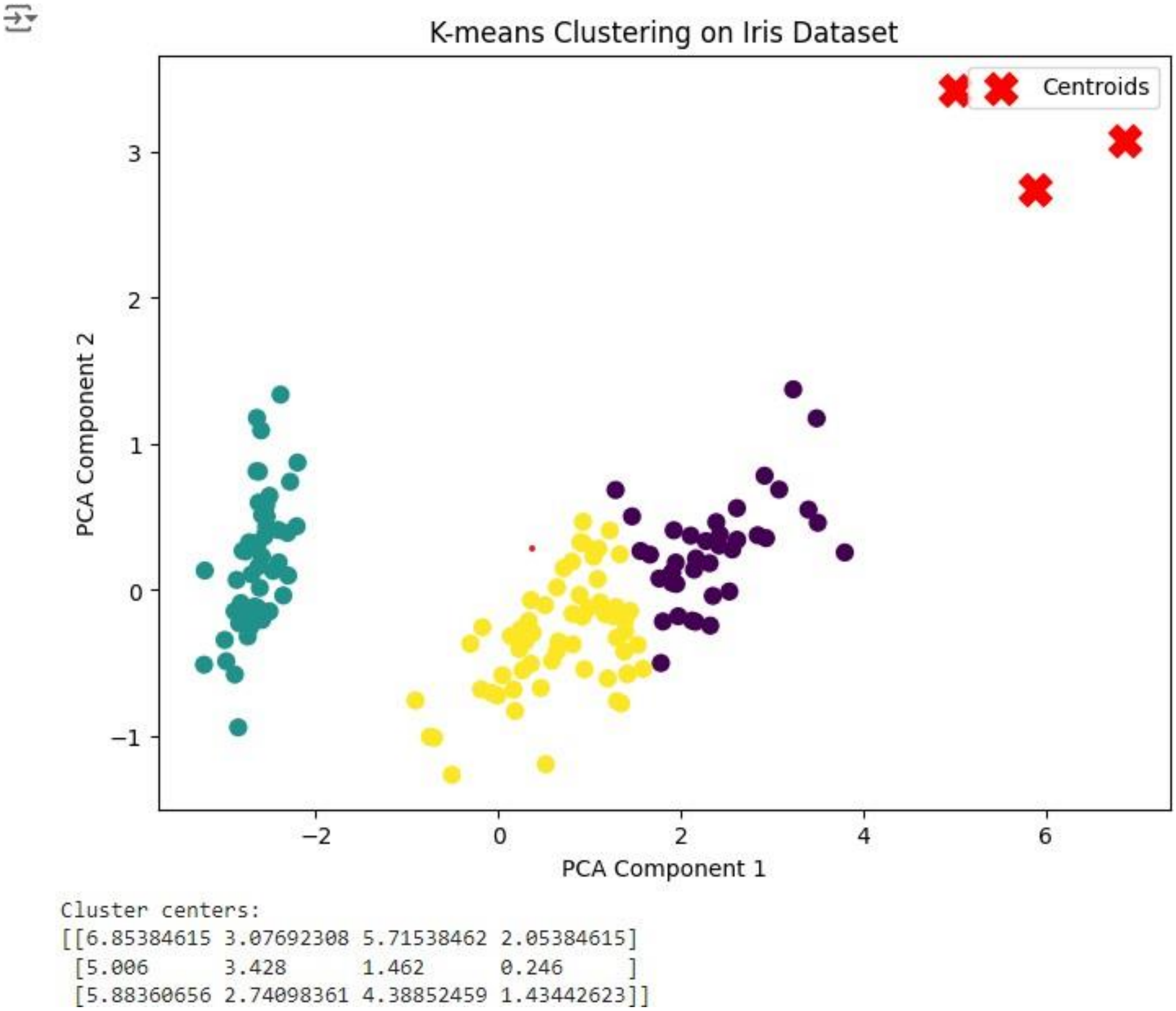
```
# Importing necessary libraries
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.cluster import KMeans
from sklearn.datasets import load_iris
from sklearn.decomposition import PCA
# Load the Iris dataset
iris = load_iris()
X = iris.data  # Features
# Applying K-means clustering
kmeans = KMeans(n_clusters=3, random_state=42)
y_kmeans = kmeans.fit_predict(X)
# Reducing dimensions for visualization using PCA (Principal Component
Analysis)
pca = PCA(2)
X_pca = pca.fit_transform(X)
# Plotting the clustered data points
plt.figure(figsize=(8,6))
plt.scatter(X_pca[:, 0], X_pca[:, 1], c=y_kmeans, cmap='viridis', s=50)
plt.scatter(kmeans.cluster_centers_[:, 0], kmeans.cluster_centers_[:, 1],
s=200, c='red', marker='X', label='Centroids')
plt.title('K-means Clustering on Iris Dataset')
plt.xlabel('PCA Component 1')
plt.ylabel('PCA Component 2')
plt.legend()

plt.show()

# Output the cluster centers

print("Cluster centers:")

print(kmeans.cluster_centers_)
```



```
Cluster centers:
[[6.85384615 3.07692308 5.71538462 2.05384615]
 [5.006      3.428      1.462      0.246     ]
 [5.88360656 2.74098361 4.38852459 1.43442623]]
```

Conclusion: In this practical, I implemented the K-means clustering algorithm on the Iris dataset. The algorithm successfully grouped the data points into 3 clusters based on the similarity of their features. The cluster centers and the visual representation of the clusters helped us understand how the data points were grouped. K-means is an efficient algorithm for clustering tasks, but it requires the number of clusters (K) to be predefined and is sensitive to the initialization of centroids.