Exp No: Implementation of clustering techniques - K Means.

Date.

Aim: To implement a K-means clustering techniques using Python language.

Explanation:

* import KMeans from sklearn.cluster.
* Assign x & y.
* call the function KMeans().
* perform scatter operation and display the output.

Algorithm:

* Initialize.
  -> choose the number of clusters K.
  -> Randomly initialize K centroids
* Assign Data points to clusters.
  => for each data set.
  -> calculate the distance between the data point and each centroid.
  -> Assign the data point to the cluster whose centroid is the closest.
* Recalculate centroids:
  -> for each cluster, compute the new centroid by calculating the mean of all data points assigned to that cluster.
* Repeat
  -> Repeat step 2 & 3 until the cluster assignment do not change. This is called convergence.

Scanned with CamScanner

* Stopping criteria:

→ algorithm stops when one of the following occurs.

→ the centroids do not change significantly between iterations.

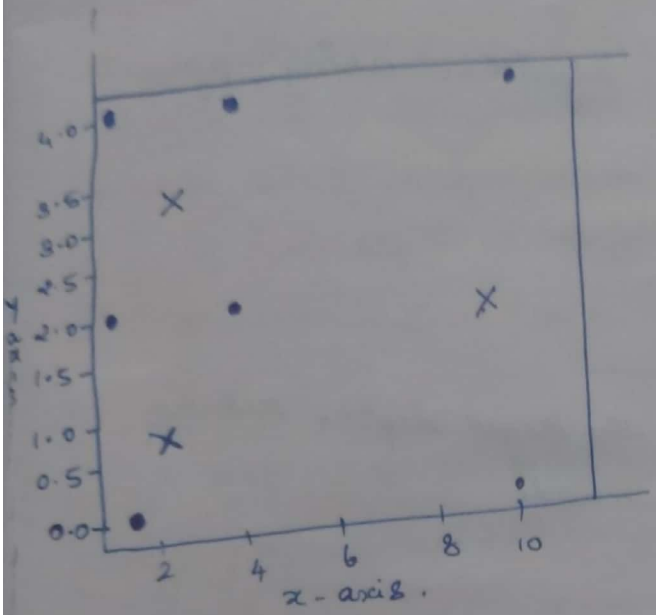
→ A maximum number of iterations is reached.

code:

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

x = np.array([[1,2], [1,4], [1,0],
              [4,2], [4,4], [4,0],
              [10,2], [10,4], [10,0]])

kmeans = KMeans(n_clusters=3, random_state=0)

kmeans.fit(x)
y_kmeans = kmeans.predict(x)
plt.scatter(x[:,0], x[:,1], c=y_kmeans
s=50, cmap='viridis')

centroids = kmeans.cluster_centers_
plt.scatter(centroids[:,0], centroids[:,1],
            c='red', s=200, alpha=0.75,
            marker='x')

plt.xlabel("x-axis")
plt.ylabel("y-axis")
plt.title("K-Means clustering")
plt.show()
```

Scanned with CamScanner

output:




Result:

Thus the program is successfully executed and output is verified.

Scanned with CamScanner