UNSUPERVISED LEARNING

Dr. Isaac Osei Nyantakkyi

Unsupervised learning

- Unsupervised learning is a machine learning concept where the **unlabelled and unclassified** information is **analysed to discover hidden knowledge**.
- The **algorithms work on the data without any prior training.**

Example:

- movie promotions to the correct group of people.
- Earlier times: same set of movie to all the visitors of the page.
- Now: based on their interest, understand what type of movie is liked by what segment of the people.

Clustering and Association Analysis

- **Cluster analysis finds the commonalities between the data objects and categorizes them as per the presence and absence of those commonalities**. Clustering which helps in segmentation of the set of objects into **groups of similar objects.**

- Association: An association rule is an unsupervised learning method which is used for **finding the relationships between variables/objects** in the large database (dataset).

Clustering

- clustering is defined as an unsupervised machine learning task that automatically divides the data into clusters or groups of similar items.

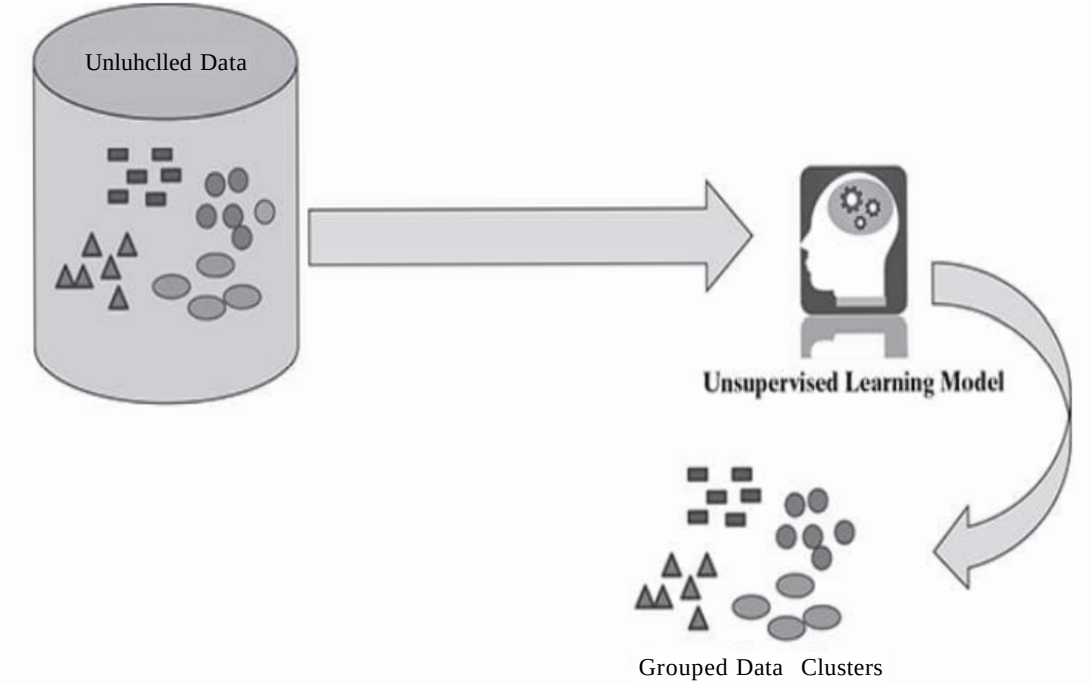


FIG. 9.1 Unsupervised learning - clustering

Different types of clustering techniques

The major clustering techniques are

- Partitioning methods,
- Hierarchical methods, and
- Density-based methods.

Different Clustering Methods

Method	Characteristics
Partitioning methods	• Uses mean or medoid (etc.) to represent cluster centre • Adopts distance-based approach to refine clusters • Finds mutually exclusive clusters of spherical or nearly spherical shape • Effective for data sets of small to medium size
Hierarchical methods	• Creates hierarchical or tree-like structure through decomposition or merger • Uses distance between the nearest or furthest points in neighboring clusters as a guideline for refinement • Erroneous merges or splits cannot be corrected at subsequent levels
Density based methods	• Useful for identifying arbitrarily shaped clusters • Guiding principle of cluster creation is the identification of dense regions of objects in space which are separated by low-density regions • May filter out outliers

Partitioning methods

- **Partitional clustering** **divides data objects into nonoverlapping groups**. In other words, **no object can be a member of more than one cluster**, and every cluster must have at least one object.

Two of the most important algorithms for partitioning-based clustering are **k-means and k-medoid**

- In the k-means algorithm, the centroid of the prototype is identified for clustering, which is **normally the mean of a group of points**.

- Similarly, the k-medoid algorithm **identifies the medoid** which is the **most representative point for a group of points**.

- These algorithms are both **nondeterministic**, meaning they could produce **different results from two separate runs** even if the runs were based on the same input.

Hierarchical Clustering

Hierarchical clustering determines cluster assignments by building a hierarchy. This is implemented by either a **bottom-up or a top-down approach**:

- **Agglomerative clustering** is the **bottom-up approach**. It merges the two points that are the most similar until **all points have been merged into a single cluster**.
- **Divisive clustering** is the **top-down approach**. It starts with all points as one cluster and splits the least similar clusters at each step until only single data points remain.

- These methods produce a tree-based hierarchy of points called a **dendrogram**.
- hierarchical clustering is a **deterministic** process, meaning cluster assignments won't change when you run an algorithm twice on the same input data.

- we have seen in the K-means clustering that there are some challenges with this algorithm, which are a **predetermined number of clusters**, and it always **tries to create the clusters of the same size**. To solve these two challenges, we can opt for the hierarchical clustering algorithm because, in this algorithm, we **don't need to have knowledge about the predefined number of clusters.**

Density-Based Clustering

- **Density-based clustering** determines cluster assignments based on the **density of data points in a region**.
- Clusters are assigned where there are high densities of data points separated by low-density regions.

Partitioning (**K-means** - A centroid-based technique)

- ***K Means segregates the unlabeled data into various groups, called clusters, based on having similar features, common patterns***.
- The principle of the k-means algorithm is to assign each of the 'n' data points to one of the K clusters where 'K' is a userdefined parameter as the number of clusters desired.
- The objective is to **maximize the homogeneity within the clusters** and also to **maximize the differences between the clusters**.
- The homogeneity and differences are measured in terms of the **distance between the objects or points** in the data set.

- ***Kmeans Algorithm is an Iterative algorithm that divides a group of n datasets into k subgroups /clusters based on the similarity and their mean distance from the centroid of that particular subgroup/ formed***.
- **K, here is the pre-defined number of clusters** to be formed by the Algorithm.
- If K=3, It means the number of clusters to be formed from the dataset is 3

Step-1: Select the value of K, to decide the number of clusters to be formed.

Step-2: Select random K points which will act as centroids.

Step-3: Assign each data point, based on their distance from the randomly selected points (Centroid), to the nearest/closest centroid which will form the predefined clusters.

Step-4: place a new centroid of each cluster.

Step-5: Repeat step no.3, which reassign each datapoint to the new closest centroid of each cluster.

Step-6: If any reassignment occurs, then go to step-4 else go to Step 7.

Step-7: FINISH

Algorithm 1 shows the simple algorithm of K-means

Step 1: Select K points in the data space and mark them as initial centroids

loop

Step 2: Assign each point in the data space to the nearest centroid to form **K** clusters

Step 3: Measure the distance of each point in the cluster from the centroid

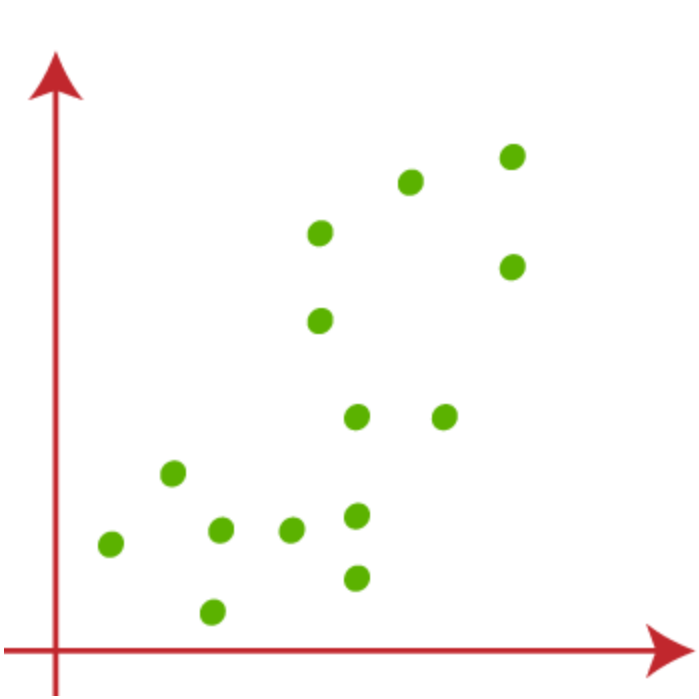
Step 4: Calculate the Sum of Squared Error (SSE) to measure the quality of the clusters *(described later in this chapter)*

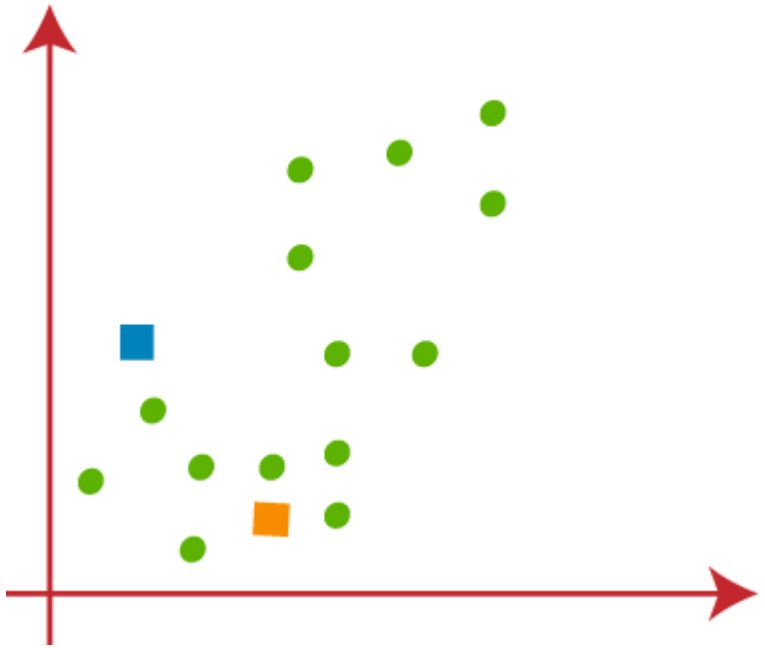
Step 5: Identify the new centroid of each cluster on the basis of distance between points

Step 6: Repeat Steps 2 to 5 to refine until centroids do not change

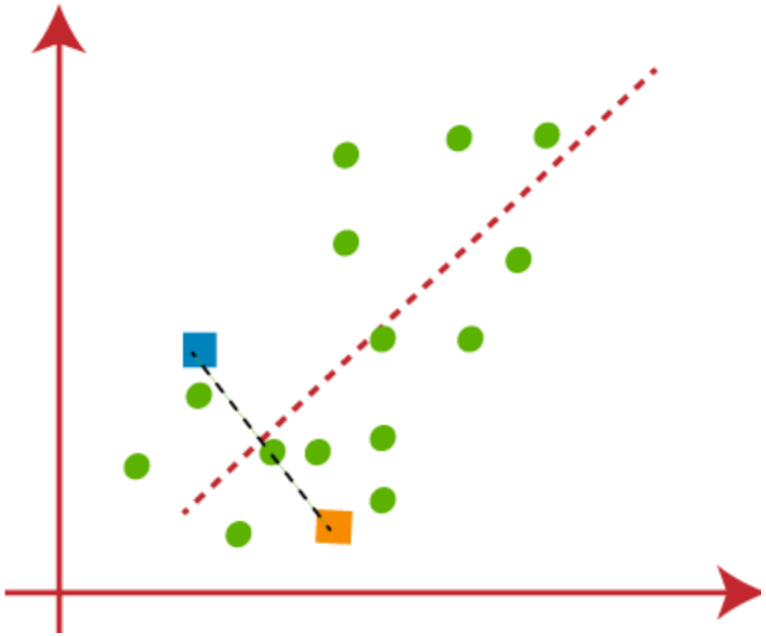
end loop

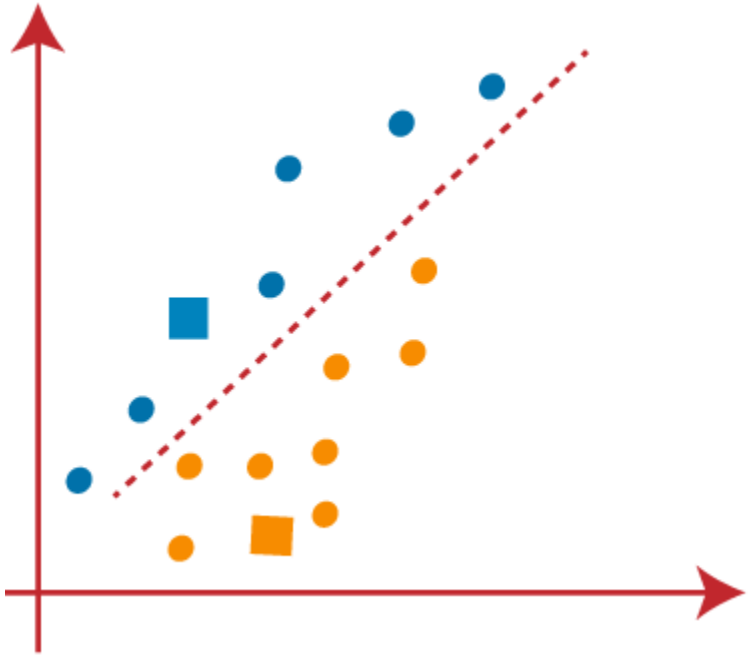
- Let's take number k of clusters, i.e., **K=2**, to identify the dataset and to put them into different clusters. It means here we will try to group these **datasets into two different clusters**.
- We need to choose some random k points or centroid to form the cluster. These points can be either the **points from the dataset or any other point**. So, here we are selecting the below two points as k points, which are not the part of our dataset.



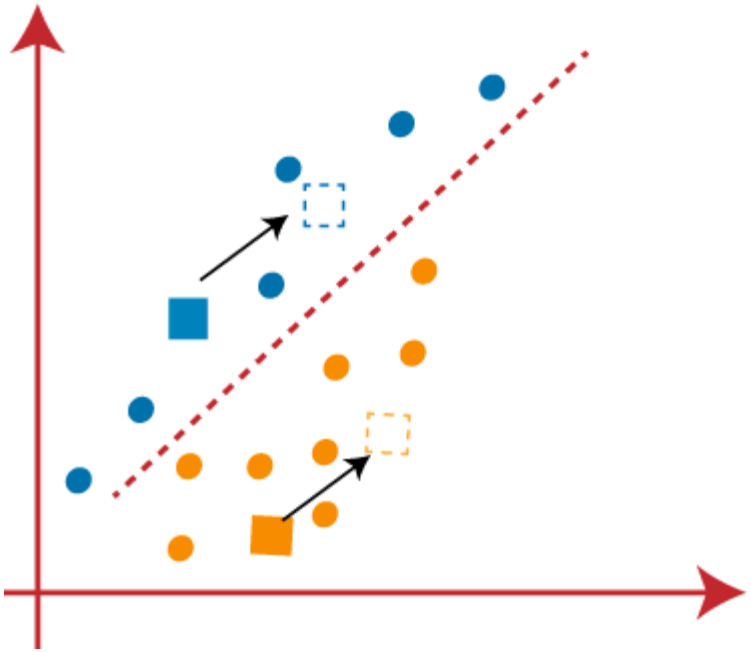


- Now we will assign each data point of the scatter plot to its closest K-point or centroid. We will compute it by applying some mathematics that we have studied to calculate the distance between two points. So, we will draw a median between both the centroids.
- From the image, it is clear that points left side of the line is near to the K1 or blue centroid, and points to the right of the line are close to the yellow centroid. Let's color them as blue and yellow for clear visualization.





- As we need to find the closest cluster, so we will repeat the process by choosing **a new centroid**. To choose the new centroids.



- Consider the below data set which has the values of the data points
- We can **randomly choose two initial points as the centroids** and from there we can start calculating distance of each point.
- For now we will consider that **D2 and D4** are the centroids.
- To start with we should calculate the distance with the help of **Euclidean Distance** which is $\sqrt{((x1-y1)^2 + (x2-y2)^2}$

Documents (Data Points}	W1 (x-axis)	W2 (y-axis)
DI	2	0
D2	1	3
D3	3	5
D4	2	2
D5	4	6

Iteration 1:

- ***Step 1:*** We need to calculate the **distance between the initial centroid points with other data points**. Below have shown the calculation of distance from initial centroids D2 and D4 from data point D1.
- After calculating the distance of all data points, we get the values as below.

Distance between DI and D2	Distance between DI and D4
$7(2- \ I)^2 + (0 - \ 3)^2$	$7(2 — 2)^2 + (0 - \ 2)^2$
$= \ 7(D^2 + (3)^2$	$= \ 7(0)^2 + (-2)^2$
$= \ \sqrt{1+9}$	$= \ V0 + \ 4$
$= \ \sqrt{10} = 3.17$	$= \ \sqrt{4} = 2$

Documents (Data Points)	Distance between D2 and other data points	Distance between D4 and other data points
DI	3.17	2.0
D3	*2* 83	117
D5	4.25	4.48

Step 2: Next, we need to group the data points which are closer to centriods. Observe the above table, we can notice that D1 is closer to D4 as the distance is less. Hence we can say that D1 belongs to D4 Similarly, D3 and D5 belongs to D2. After grouping, we need to calculate the mean of grouped values from Table 1.

Cluster 1: (D1, D4) Cluster 2: (D2, D3, D5)

Step 3: Now, we calculate the mean values of the clusters created and the new centriod values will these mean values and centroid is moved along the graph.

Clusters	Mean value of data points along x-axis	Distance between 04 and other data points
DI, D4	2.0	1.0
D2, D3, D5	2.67	4.67

From the above table, we can say the new centroid for cluster 1 is (2.0, 1.0) and for cluster 2 is (2.67, 4.67)

Iteration 2:

Step 4: Again the values of euclidean distance is calculated from the new centriods. Below is the table of distance between data points and new centroids.

- We can notice now that clusters have changed the data points. Now the cluster 1 has D1, D2 and D4 data objects. Similarly, cluster 2 has D3 and D5

Step 5: Calculate the mean values of new clustered groups from Table 1 which we followed in step 3. The below table will show the mean values

Documents (Data Points)	Distance between centroid of cluster 1 and data points	Distance between centroid of duster 2 and data points
01	i.o	4.72
02	2.24	2.37
D3	4,13	0,47
04	1	2.76
D5	5,39	1,89

Clusters	Mean value of data points along x -axis	Distance between D4 and other data points
DI, D2, D4	1.67	1.67
D3, D5	3.5	5.5

Now we have the new centroid value as following:

cluster 1 (D1, D2, D4) – (1.67, 1.67) and cluster 2 (D3, D5) – (3.5, 5.5)

This process has to be repeated until we find a constant value for centroids and the latest cluster will be considered as the final cluster solution.

Choosing the value of K :

- For a small data set, sometimes a rule of thumb that is followed is $K = \sqrt{\frac{n}{2}}$
- But unfortunately, this thumb rule does **not work well for large data sets**. There are several statistical methods to arrive at the suitable number of clusters.

- To find the number of clusters in the data, we need to **run the K-Means clustering algorithm for different values of K** and **compare the results**.
- We should choose the **optimal value of K** that gives us best performance.

There are different techniques available to find the optimal value of K.

- The most common technique is the **elbow method** which is described below.

- one effective approach is to employ the **hierarchical clustering technique** on sample points from the data set and then arrive at sample K clusters.

How to choose the value of "K number of clusters" in K-means Clustering?

- The performance of the K-means clustering algorithm depends upon highly efficient clusters that it forms.
- To choose the **optimal number of clusters is a big task**.

Elbow Method:

- The Elbow method is one of the most popular ways to find the optimal number of clusters.
- This method **uses the concept of WCSS value**.
- **WCSS** stands for **Within Cluster Sum of Squares**, which defines the total variations within a cluster.

The formula to calculate the value of WCSS (for 3 clusters) is given below:

$$WCSS= \sum_{Pi \text{ in Cluster1}} distance(P_i\ C_1)^2 + \sum_{Pi \text{ in Cluster2}} distance(P_i\ C_2)^2 + \sum_{Pi \text{ in CLuster3}} distance(P_i\ C_3)^2$$

In the above formula of WCSS,

- $\sum_{Pi \text{ in Cluster1}}$ distance(P_i C_1)^2: It is the **sum of the square of the distances between each data point and its centroid within a cluster1** and the same for the other two terms.
- To measure the distance between data points and centroid, we can use any method such as **Euclidean distance or Manhattan distance**.

To find the optimal value of clusters, the elbow method follows the below steps:

- It executes the K-means clustering on a given dataset for **different K values** (ranges from 1-10).
- For each value of K, **calculates the WCSS value**.
- Plots a curve between calculated WCSS values and the number of clusters K.
- The **sharp point of bend** or a point of the plot looks like an arm, then that point is considered as the **best value of K.**

- *Between Clusters Sum of Squares* (BCSS), which measures the squared average distance between all centroids. To calculate BCSS, you find the Euclidean distance from a given cluster centroid to all other cluster centroids. You then iterate this process for all of the clusters, and sum all of the values together. This value is the BCSS. You can divide by the number of clusters to calculate the average BCSS.
- Essentially, BCSS measures the variation between all clusters. A large value can indicate clusters that are spread out, while a small value can indicate clusters that are close to each other.
- We iterate the values of k from 1 to 9 and calculate the values of distortions for each value of k and calculate the distortion and inertia for each value of k in the given range.

- **Distortion is the average of the euclidean squared distance from the centroid of the respective clusters.** Typically, the Euclidean distance metric is used.

- **Inertia is the sum of squared distances of samples to their closest cluster centre**.
- The measure of quality of clustering uses the SSE technique

$$\text{dist}(x, y) = \sqrt{\sum_{1}^{n} (x_i - y_i)^2}$$

$$\text{SSE} = \sum_{k=1}^{K} \sum_{x_i \in C_k} \|x_i - \mu_k\|^2$$

\Jo\ve	De\iuM\on	DexA orx^2-
99.0		
<18.6		
<18.5		
101.1		
<18.3		
<18.6		
<17.9		
98.4		
<PI2		
991		

- $M = \frac{\Sigma z}{n}$
- For this data, the mean is calculated as:
  - $\mu = \frac{99.0 + 98.6 + 98.5 + 101.1 + 98.3 + 98.6 + 97.9 + 98.4 + 99.2 + 99.1}{10}$
  - $\mu = \frac{988.7}{10}$
  - $n = 98.87$

\Vo\iAe
99.0
+ 98.G
+ <185
+ 101. <
+ <18.3
+ "18.6
+ 97.9
+38.4
+ 342
+ 43.1

X =9887

$$\bar{x} = \frac{\Sigma value}{N}$$

$$\mu = \frac{988.7}{10}$$

$$\mu = 98.87$$

CALCULATING SSE

μ = 98.87

\Zo\ue	Dev«M\on	DeMkM\ov\$^2$
99.0	0.13	
98.G	-0.27	
385	-0.37	
101.1	223	
98.3	-0.57	
98.G	-0.27	
97.9	-0.37	
98.1	-0.4-7	
99.2	0.33	
331	0.23	

- 99.0 - 98.87 = 0.13
- 98.6 - 98.87 = -0.27
- 98.5 - 98.87 = -0.37
- 101.1 - 98.87 = 2.23
- 98.3 - 98.87 = -0.57
- 98.6 - 98.87 = -0.27
- 97.9 - 98.87 = -0.97
- 98.4 - 98.87 = -0.47
- 99.2 - 98.87 = 0.33
- 99.1 - 98.87 = 0.23

\Jo\ue	DewMxon	De\toA\ov\$^{2,}$
99.0	0.13	0.01G9
98.G	-0.27	0.0729
385	-0.37	0.13G9
101.1	2.23	4.3723
98.3	-0.57	03243
<18.G	-0.27	0.0723
°I7°I	-0.97	0.9409
<18.-4	-0.4-7	0.2209
<m	0.33	0.1083
99.1	0.23	0.0529

- $0.13^2 = 0.0169$
- $(-0.27)^2 = 0.0729$
- $(-0.37)^2 = 0.1369$
- $2.23^2 = 4.9729$
- $(-0.57)^2 = 0.3249$
- $(-0.27)^2 = 0.0729$
- $(-0.97)^2 = 0.9409$
- $(-0.47)^2 = 0.2209$
- $0.33^2 = 0.1089$
- $0.23^2 = 0.0529$

\lo\ue	Deviation	OevuMion ^{2}-
99.0	0.13	0.01&3
98.G	-0.27	0.0729
985	-0.37	0.13G9
101.1	2.23	4.3723
98.3	-0.57	0.3243
98.G	-0.27	0.0723
m	-0.37	0.9409
981	-0.17	0.2209
99.2	0.23	0.1083
99.1	0.23	0.0523

SSE=G.21

$$SSE = 6.921$$

K-Medoids: a representative object-based technique

- *k-means algorithm is sensitive to* outliers in the data set.
- Consider the values 1, 2, 3, 5, 9, 10, 11, and 25.
- Point 25 is the outlier, and it affects the cluster formation negatively when the mean of the points is considered as centroids.

With K = 2, the initial clusters we arrived at are {1, 2, 3, 6} and {9, 10, 11, 25}.

The mean of the cluster {1,2, 3. 6} $= \frac{12}{4} = 3,$

and the mean of the cluster (9, |0. 12.25} = $\frac{56}{4} = 14.$

So, the SSE within the clusters is

$(1 - 3)^2 + (2 - 3)^2 + (3 - 3)^2 + (6 - 3)^2$ 4- $(9 - 14)^2$

$+ (10 - 14)^2 + (12 - J4)^2 + (25 - 14)^2 = 179$

If we compare this with the cluster {1, 2, 3, 6, 9} and {10, 11,25},

the mean of the cluster $\{1,2,3,6,9\} = \frac{21}{5} = 4.2,$

and the mean of the cluster $\{[().12,25\} = \frac{47}{3} = 15.67.$

So, the SSE within the clusters is

$$(1 - 4.2)^2 + (2 - 4.2)^2 + (3 - 4.2)^2 + (6 - 4.2)^2 + (9 - 4.2)^2$$
$$+ (10 - 15.67)^2 + (12 - 15.67)^2 + (25 - 15.67)^2 = 113.84$$

- Because the SSE of the second clustering is lower, *k-means* tend to put point 9 in the same cluster with 1, 2, 3, and 6 though the point is logically nearer to points 10 and 11.
- This skewedness is introduced due to the outlier point 25, which shifts the mean away from the centre of the cluster.

k-medoids provides a solution to this problem.

- *Instead of* considering the mean of the data points in the cluster, *kmedoids* considers *k representative data points from the* existing points in the data set as the centre of the clusters.
- Note that the medoids in this case are actual data points or objects from the data set and not an imaginary point as in the case when the mean of the data sets within cluster is used as the centroid in the *k-means* technique.

- One of the practical implementation of the *k-medoids principle is the Partitioning a*round Medoids (PAM) algorithm.

Step 1: Randomly choose k points in the data set as the initial representative points

loop

Step 2: Assign each of the remaining points to the cluster which has the nearest representative point

Step 3: Randomly select a non-representative point o_r in each cluster

Step 4: Swap the representative point o, with o_r and compute the new SSE after swapping

Step 5: If SSEnew < SSEold, then swap *Oj* with o_r to form the new set of *k* representative objects;

Step 6: Refine the k clusters on the basis of the nearest representative point. Logic continues until there is no change

end loop