Unit-4

Cluster Analysis

Cluster analysis, also known as clustering, is a method of data mining that groups similar data points together. The goal of cluster analysis is to divide a dataset into groups (or clusters) such that the data points within each group are more like each other than to data points in other groups. This process is often used for exploratory data analysis and can help identify patterns or relationships within the data that may not be immediately obvious. There are many different algorithms used for cluster analysis, such as k-means, hierarchical clustering, and density-based clustering. The choice of algorithm will depend on the specific requirements of the analysis and the nature of the data being analyzed.

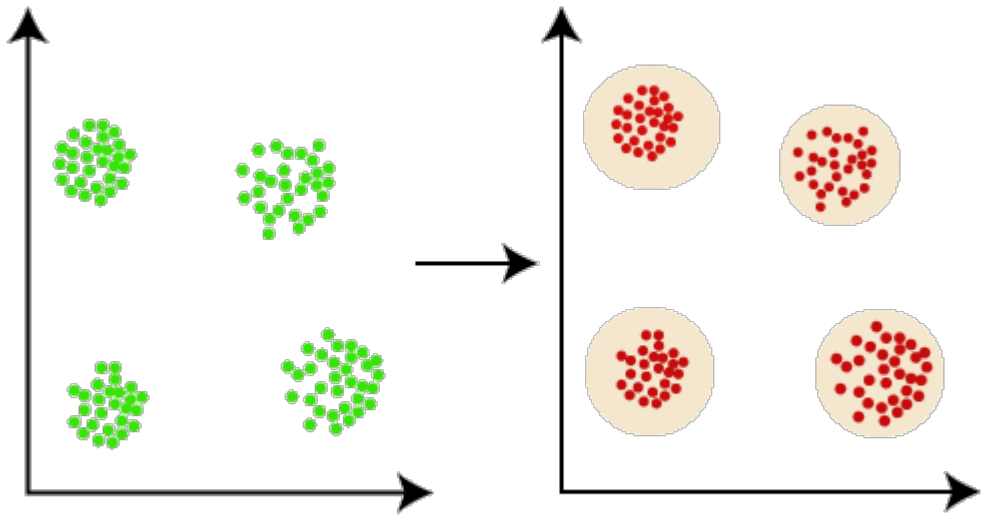
Cluster Analysis is the process to find similar groups of objects in order to form clusters. It is an unsupervised machine learning-based algorithm that acts on unlabelled data. A group of data points would comprise together to form a cluster in which all the objects would belong to the same group.

The given data is divided into different groups by combining similar objects into a group. This group is nothing but a cluster. A cluster is nothing but a collection of similar data which is grouped together. For example, consider a dataset of vehicles given in which it contains information about different vehicles like cars, buses, bicycles, etc. As it is unsupervised learning there are no class labels like Cars, Bikes, etc for all the vehicles, all the data is combined and is not in a structured manner.

Clustering is an unsupervised Machine Learning-based Algorithm that comprises a group of data points into clusters so that the objects belong to the same group.

Clustering helps to splits data into several subsets. Each of these subsets contains data similar to each other, and these subsets are called clusters. Now that the data from our customer base is divided into clusters, we can make an informed decision about who we think is best suited for this product.

Let's understand this with an example, suppose we are a market manager, and we have a new tempting product to sell. We are sure that the product would bring enormous profit, as long as it is sold to the right people. So, how can we tell who is best suited for the product from our company's huge customer base?



Showing four clusters formed from the set of unlabeled data

Clustering, falling under the category of **unsupervised machine learning**, is one of the problems that machine learning algorithms solve.

- A cluster is a subset of similar objects
- A subset of objects such that the distance between any of the two objects in the cluster is less than the distance between any object in the cluster and any object that is not located inside it.
- A connected region of a multidimensional space with a comparatively high density of objects.

Applications of Cluster Analysis

- Clustering analysis is broadly used in many applications such as market research, pattern recognition, data analysis, and image processing.
- Clustering can also help marketers discover distinct groups in their customer base. And they can characterize their customer groups based on the purchasing patterns.
- In the field of biology, it can be used to derive plant and animal taxonomies, categorize genes with similar functionalities and gain insight into structures inherent to populations.
- Clustering also helps in identification of areas of similar land use in an earth observation database. It also helps in the identification of groups of houses in a city according to house type, value, and geographic location.
- Clustering also helps in classifying documents on the web for information discovery.
- Clustering is also used in outlier detection applications such as detection of credit card fraud.
- As a data mining function, cluster analysis serves as a tool to gain insight into the distribution of data to observe characteristics of each cluster.

Properties of Clustering :

1. Clustering Scalability: Nowadays there is a vast amount of data and should be dealing with huge databases. In order to handle extensive databases, the clustering algorithm should be scalable. Data should be scalable, if it is not scalable, then we can't get the appropriate result which would lead to wrong results.

2. High Dimensionality: The algorithm should be able to handle high dimensional space along with the data of small size.

3. Algorithm Usability with multiple data kinds: Different kinds of data can be used with algorithms of clustering. It should be capable of dealing with different types of data like discrete, categorical and interval-based data, binary data etc.

4. Dealing with unstructured data: There would be some databases that contain missing values, and noisy or erroneous data. If the algorithms are sensitive to such data then it may lead to poor quality clusters. So it should be able to handle unstructured data and give some structure to the data by organising it into groups of similar data objects. This makes the job of the data expert easier in order to process the data and discover new patterns.

5. Interpretability: The clustering outcomes should be interpretable, comprehensible, and usable. The interpretability reflects how easily the data is understood.

Requirements of Clustering in Data Mining

The following points throw light on why clustering is required in data mining −

- **Scalability** − We need highly scalable clustering algorithms to deal with large databases.
- **Ability to deal with different kinds of attributes** − Algorithms should be capable to be applied on any kind of data such as interval-based (numerical) data, categorical, and binary data.
- **Discovery of clusters with attribute shape** − The clustering algorithm should be capable of detecting clusters of arbitrary shape. They should not be bounded to only distance measures that tend to find spherical cluster of small sizes.
- **High dimensionality** − The clustering algorithm should not only be able to handle low-dimensional data but also the high dimensional space.
- **Ability to deal with noisy data** − Databases contain noisy, missing or erroneous data. Some algorithms are sensitive to such data and may lead to poor quality clusters.
- **Interpretability** − The clustering results should be interpretable, comprehensible, and usable.

Clustering Methods:

The clustering methods can be classified into the following categories:

- Partitioning Method
- Hierarchical Method
- Density-based Method

Partitioning Method

It is used to make partitions on the data in order to form clusters. If "n" partitions are done on "p" objects of the database then each partition is represented by a cluster and $n < p$. The two conditions which need to be satisfied with this Partitioning Clustering Method are:

- One objective should only belong to only one group.
- There should be no group without even a single purpose.

In the partitioning method, there is one technique called iterative relocation, which means the object will be moved from one group to another to improve the partitioning.

This clustering method classifies the information into multiple groups based on the characteristics and similarity of the data. It's the data analysts to specify the number of clusters that has to be generated for the clustering methods. In the partitioning method when database(D) that contains multiple(N) objects then the partitioning method constructs user-specified(K) partitions of the data in which each partition represents a cluster and a particular region. There are many algorithms that come under partitioning method some of the popular ones are K-Mean, PAM(K-Medoids), CLARA algorithm (Clustering Large Applications) etc.

K-Mean (A centroid based Technique): The K means algorithm takes the input parameter K from the user and partitions the dataset containing N objects into K clusters so that resulting similarity among the data objects inside the group (intracluster) is high but the similarity of data objects with the data objects from outside the cluster is low (intercluster). The similarity of the cluster is determined with respect to the mean value of the cluster. It is a type of square error algorithm. At the start randomly k objects from the dataset are chosen in which each of the objects represents a cluster mean(centre). For the rest of the data objects, they are assigned to the nearest cluster based on their distance from the cluster mean. The new mean of each of the cluster is then calculated with the added data objects.

Algorithm: K mean:

Input:

K: The number of clusters in which the dataset has to be divided

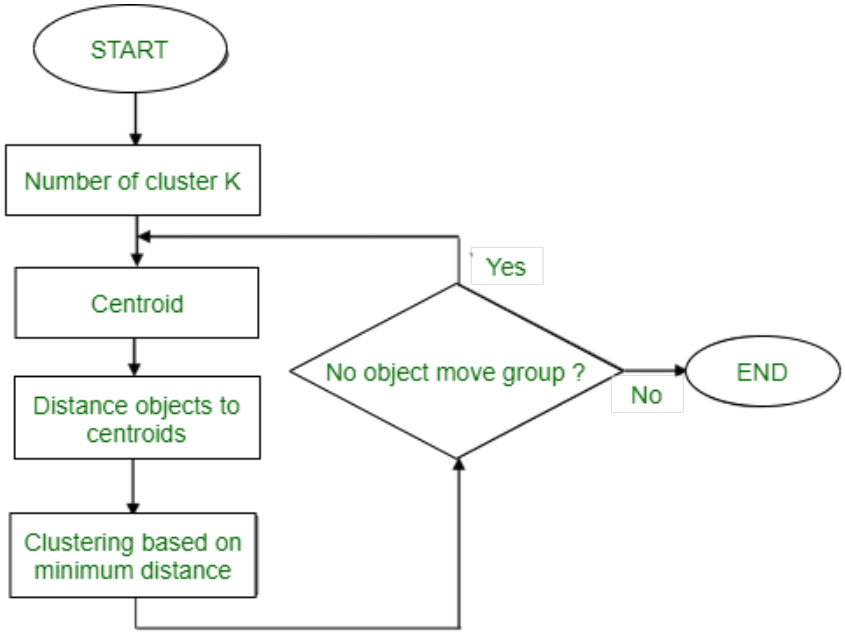
D: A dataset containing N number of objects

Output:

A dataset of K clusters

Method:

1. Randomly assign K objects from the dataset(D) as cluster centres(C)
2. (Re) Assign each object to which object is most similar based upon mean values.
3. Update Cluster means, i.e., Recalculate the mean of each cluster with the updated values.
4. Repeat Step 2 until no change occurs.



Advantages of K-Means

Here are some advantages of the K-means clustering algorithm -

- **Scalability** - K-means is a scalable algorithm that can handle large datasets with high dimensionality. This is because it only requires calculating the distances between data points and their assigned cluster centroids.
- **Speed** - K-means is a relatively fast algorithm, making it suitable for real-time or near-real-time applications. It can handle datasets with millions of data points and converge to a solution in a few iterations.
- **Simplicity** - K-means is a simple algorithm to implement and understand. It only requires specifying the number of clusters and the initial centroids, and it iteratively refines the clusters' centroids until convergence.
- **Interpretability** - K-means provide interpretable results, as the clusters' centroids represent the centre points of the clusters. This makes it easy to interpret and understand the clustering results.

K-means

Q $n = 19$

15, 15, 16, 19, 19, 20, 20, 21, 22, 28, 35, 40, 41, 42, 43, 44, 60, 61, 65

Initial clusters

$k = 2$

$C_1 = 16$

$C_2 = 22$

$D1 = \sqrt{(x_i - C_1)^2}$ $D_2 = \sqrt{(x_i - C_2)^2}$

x_i	C_1	C_2	D1	D2	Nearest cluster	New centroid
15	16	22	1	7	1	15.33
15	16	22	1	7	1	
16	16	22	0	6	1	
19	16	22	3	3	2	
19	16	22	3	3	2	
20	16	22	4	2	2	
20	16	22	4	2	2	
21	16	22	5	1	2	
22	16	22	5	0	2	
28	16	22	12	6	2	36.25
35	16	22	19	13	2	
40	16	22	24	18	2	
41	16	22	25	19	2	
42	16	22	26	20	2	
43	16	22	27	21	2	
44	16	22	28	22	2	
60	16	22	44	38	2	
61	16	22	45	39	2	
65	16	22	49	43	2	

Iteration 2

x_i	C_1	C_2	D1	D2	Nearest cluster	New centroid
15	15.33	36.25	0.33	21.25	1	
15	15.33	36.25	0.33	21.25	1	
16	"	"	0.67	20.25	1	
19			3.67	17.25	1	18.56
19			3.67	17.25	1	
20	"	"	4.67	16.25	1	
20			4.67	16.25	1	
21			5.67	15.25	1	
22	"	"	6.67	14.25	1	
28			12.67	8.25	2	
35	"	"	19.67	1.25	2	
40			24.67	3.75	2	
41	"	"	25.67	4.75	2	
42			26.67	5.75	2	45.9
43			27.67	6.75	2	
44	"	"	28.67	7.75	2	
60			44.67	23.75	2	
61		"	45.67	24.75	2	
65	"		49.67	28.75	2	

Iteration 3

x_i	C_1	C_2	D1	D2	Nearest cluster	New centroid
15	18.56	45.9	3.56	30.9	1	
15	18.56	45.9	3.56	30.9	1	
16	"	"	2.56	29.9	1	
19			0.44	26.9	1	
19			0.44	26.9	1	19.5
19	"	"	1.44	25.9	1	
20			1.44	25.9	1	
20	"	"	2.44	24.9	1	
21			3.44	23.9	1	
22	"	"	9.44	17.9	1	
28			16.44	10.9	2	
35			21.44	5.9	2	
40	"	"	22.44	4.9	2	
41			23.44	3.9	2	47.89
42			24.44	2.9	2	
43	"	"	25.44	1.9	2	
44			41.44	14.1	2	
60			42.44	15.1	2	
61	"	"	43.44	19.1	2	
65						

Iteration 4

x_i	C_1	C_2	D1	D2	Nearest cluster	New centroid
15	19.5	47.89	4.5	32.89	1	
15	19.5	47.89	4.5	32.89	1	
16	19.5	47.89	3.5	31.89	1	
19	19.5	47.89	0.5	28.89	1	

x_i	C_1	C_2	D1	D2	Nearest cluster	New centroid
19	19.5	47.89	0.5	28.89	1	
20	19.5	47.89	0.5	27.89	1	
20	"	"	0.5	27.89	1	
21			1.5	26.89	1	
22	"	"	2.5	25.89	1	
28			8.5	19.89	1	
35	"	"	15.5	12.89	2	
40			20.5	7.89	2	
41	"	"	21.5	6.89	2	
42			22.5	5.89	2	
43	"	"	23.5	4.89	2	
44			24.5	3.89	2	
60	"	"	40.5	12.11	2	
61			41.5	13.11	2	
65	"	"	45.5	17.11	2	

→ No change between iterations 3 and 4 has been noted. By using clustering 2 groups have been identified (15-28) and (36-65).

K-medoids is a clustering algorithm that is similar to K-means but uses medoids instead of centroids. Medoids are representative data points within a cluster that are most centrally located concerning all other data points in the cluster. In contrast, centroids are the arithmetic mean of all the data points in a cluster. In the K-medoids algorithm, the initial medoids are randomly selected from the dataset, and the algorithm iteratively updates the medoids until convergence. The algorithm assigns each data point to the nearest medoid and then computes the total dissimilarity between each medoid and its assigned data points. It then selects the data point with the lowest dissimilarity as the new medoid for each cluster.

Difference Between K-Means & K-Medoids Clustering

Factor	K-Means	K-Medoids
Objective	Minimizing the sum of squared distances between data points and their assigned cluster centroids.	Minimizing the sum of dissimilarities between data points and their assigned cluster medoids.
Cluster Center Metric	Use centroids, which are the arithmetic means of all data points in a cluster.	Use medoids, which are representative data points within each cluster that are most centrally located concerning all other data points in the cluster.
Robustness	Less robust to noise and outliers.	More robust to noise and outliers.
Computational Complexity	Faster and more efficient for large datasets.	Slower and less efficient for large datasets.
Cluster Shape	Assumes spherical clusters and is not suitable for non-convex clusters.	Can handle non-convex clusters.
Initialization	Requires initial centroids to be randomly selected.	Requires initial medoids to be randomly selected.
Applications	Suitable for applications such as customer segmentation, image segmentation, and anomaly detection.	Suitable for applications where robustness to noise and outliers is important, such as clustering DNA sequences or gene expression data.

Algorithm:

Given the value of k and unlabelled data:

1. Choose k number of random points from the data and assign these k points to k number of clusters. These are the initial medoids.
2. For all the remaining data points, calculate the distance from each medoid and assign it to the cluster with the nearest medoid.
3. Calculate the total cost (Sum of all the distances from all the data points to the medoids)
4. Select a random point as the new medoid and swap it with the previous medoid. Repeat 2 and 3 steps.
5. If the total cost of the new medoid is less than that of the previous medoid, make the new medoid permanent and repeat step 4.
6. If the total cost of the new medoid is greater than the cost of the previous medoid, undo the swap and repeat step 4.The Repetitions have to continue until no change is encountered with new medoids to classify data points.

K-Medoids - A medoid can be defined as a point in the cluster whose dissimilarities with all the other points in the cluster are minimum. The dissimilarity of the medoid (C_i) and object (P_i) is calculated by using $E = |P_i - C_i|$

Manhattan Dist

Algorithm -

① Initialize : Select k random points out of the n data points as the medoids.

② Associate each data point to the closest medoid by using any common distance metric methods.

③ While the cost decreases. For each medoid m, for each data o point which is not a medoid :

→ Swap m & o, associate each data point to the closest medoid and recompute the cost.

→ If the total cost is more than that in the previous step undo the swap.

Eg -

	X	Y
0	8	7
1	3	7
2	4	9
3	9	6
4	8	5
5	5	8
6	7	3
7	8	4
8	7	5
9	4	5

Step 1: Let the randomly selected 2 medoids, so select k = 2 & let C1 - (4,5) and C2 - (8,5) are two medoids.

Step 2: Calculating cost :- The dissimilarity of each non-medoid point with the medoids is calculated

Distance $= |x_1 - x_2| + |y_1 - y_2|$

	X	Y	Dissimilarity from C1	Diss. from C2
0	8	7	6	2
1	3	7	3	7
2	4	9	4	8
3	9	6	6	2
4	8	5	—	—
5	5	8	4	6
6	7	3	5	3
7	8	4	5	1
8	7	5	3	1
9	4	5	—	—

~~|8-4|~~ $|4-8| + |5-7| = |-4| + |2| = 6$

$|4-3| + |5-7| = |1| + |2| = 3$

Each point is assigned to the cluster of that medoid whose dissimilarity is less. Points 1, 2 & 5 go to cluster C1 & 0, 3, 6, 7, 8 go to cluster C2

Cost = $(3 + 4 + 4) + (2 + 2 + 3 + 1 + 1) = 20$

C1

Step 3: Randomly select one non-medoid point & recalc. the cost. Let (8, 4). The dissimilarity with the medoids - C1(4, 5) & C2(8, 4) is

	X	Y	C1	C2
0	8	7	6	3
1	3	7	3	8
2	4	9	4	9
3	9	6	6	3
4	8	5	4	1
5	5	8	4	7
6	7	3	5	2
7	8	4	–	–
8	7	5	3	2
9	4	5	–	–

C1 – 1, 2, 5 C2 = 0, 3, 6, 4, 8)

Cost = (3 + 4 + 4) + (3 + 3 + 2 + 2 + 1)

= 22

Swap cost = New Cost - Prev. Cost = 22 - 20

2 > 0

As the swap cost is not less than zero, we undo the swap. Hence (4, 5) & (8, 5) are final medoids.

K-Medoids

Data set:

	x	y
0	5	4
1	7	7
2	1	3
3	8	6
4	4	9

If k is given as 2, we need to break down the data points into 2 clusters.

1. **Initial medoids: M1(1, 3) and M2(4, 9)**
2. Calculation of distances

$(5-1)+(4-3)=5$
$(7-1)+(7-3)=6+4=10$
$(8-1)+(6-3)=10$

Manhattan Distance: |x1 - x2| + |y1 - y2|

	x<	y	From M1(1, 3)	From M2(4, 9)
0	5	4	5	6
1	7	7	10	5
2	1	3	-	-
3	8	6	10	7
4	4	9	-	-

$(5-4)+(4-9)=6$
$(7-4)+(7-9)=5$
$(8-4)+(6-9)=7$

Cluster 1: 0

Cluster 2: 1, 3

1. Calculation of total cost:
   (5) + (5 + 7) = 17
2. Random medoid: (5, 4)

M1(5, 4) and M2(4, 9):

	x	y	From M1(5, 4)	From M2(4, 9)
0	5	4	-	-
1	7	7	5	5
2	1	3	5	9
3	8	6	5	7
4	4	9	-	-

Cluster 1: 2, 3

Cluster 2: 1

1. Calculation of total cost:
   (5 + 5) + 5 = 15
   Less than the previous cost
   New medoid: (5, 4).
2. Random medoid: (7, 7)

M1(5, 4) and M2(7, 7)

	x	y	From M1(5, 4)	From M2(7, 7)
0	5	4	-	-
1	7	7	-	-
2	1	3	5	10
3	8	6	5	2
4	4	9	6	5

Cluster 1: 2

Cluster 2: 3, 4

1. Calculation of total cost:
   (5) + (2 + 5) = 12
   Less than the previous cost
   New medoid: (7, 7).
2. Random medoid: (8, 6)

M1(7, 7) and M2(8, 6)

	x	y	From M1(7, 7)	From M2(8, 6)
0	5	4	5	5
1	7	7	-	-
2	1	3	10	10
3	8	6	-	-
4	4	9	5	7

Cluster 1: 4

Cluster 2: 0, 2

1. Calculation of total cost:
   (5) + (5 + 10) = 20
   Greater than the previous cost
   UNDO
   Hence, the final medoids: **M1(5, 4) and M2(7, 7)**
   Cluster 1: 2
   Cluster 2: 3, 4
   Total cost: 12
   Clusters:

A **Hierarchical clustering** method works via grouping data into a tree of clusters. Hierarchical clustering begins by treating every data point as a separate cluster. Then, it repeatedly executes the subsequent steps:

1. Identify the 2 clusters which can be closest together, and
2. Merge the 2 maximum comparable clusters. We need to continue these steps until all the clusters are merged together.

In Hierarchical Clustering, the aim is to produce a hierarchical series of nested clusters. A diagram called **Dendrogram** (A Dendrogram is a tree-like diagram that statistics the sequences of merges or splits) graphically represents this hierarchy and is an inverted tree that describes the order in which factors are merged (bottom-up view) or clusters are broken up (top-down view).

Hierarchical clustering is a method of cluster analysis in data mining that creates a hierarchical representation of the clusters in a dataset. The method starts by treating each data point as a separate cluster and then iteratively combines the closest clusters until a stopping criterion is reached. The result of hierarchical clustering is a tree-like structure, called a dendrogram, which illustrates the hierarchical relationships among the clusters.

Hierarchical clustering has a number of advantages over other clustering methods, including:

1. The ability to handle non-convex clusters and clusters of different sizes and densities.
2. The ability to handle missing data and noisy data.
3. The ability to reveal the hierarchical structure of the data, which can be useful for understanding the relationships among the clusters.
   However, it also has some drawbacks, such as:
4. The need for a criterion to stop the clustering process and determine the final number of clusters.
5. The computational cost and memory requirements of the method can be high, especially for large datasets.
6. The results can be sensitive to the initial conditions, linkage criterion, and distance metric used.
   In summary, Hierarchical clustering is a method of data mining that groups similar data points into clusters by creating a hierarchical structure of the clusters.
7. This method can handle different types of data and reveal the relationships among the clusters.
   However, it can have high computational cost and results can be sensitive to some conditions.

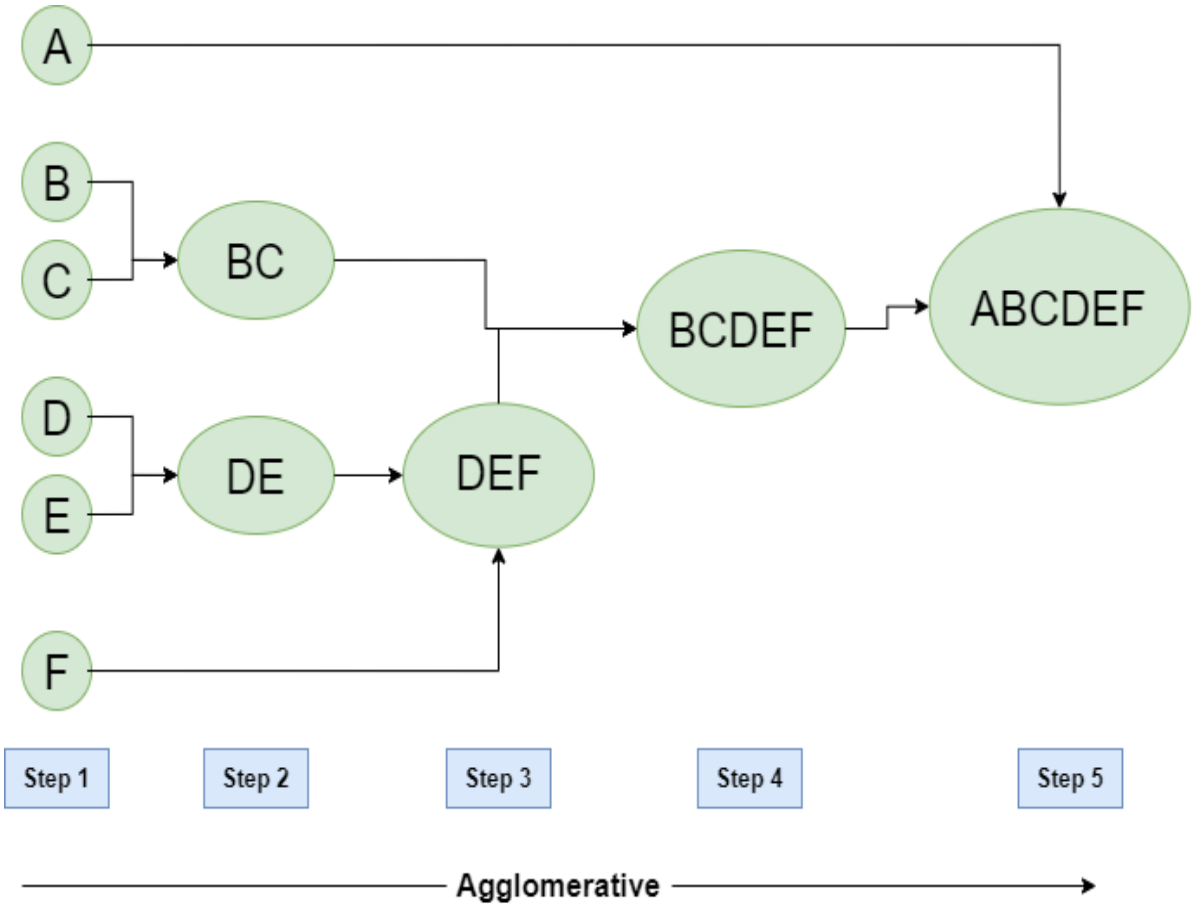
There are two types of hierarchical clustering

- Agglomerative Hierarchical Clustering
- Divisive Clustering

Agglomerative: Initially consider every data point as an **individual** Cluster and at every step, **merge** the nearest pairs of the cluster. (It is a bottom-up method). At first, every dataset is considered an individual entity or cluster. At every iteration, the clusters merge with different clusters until one cluster is formed.

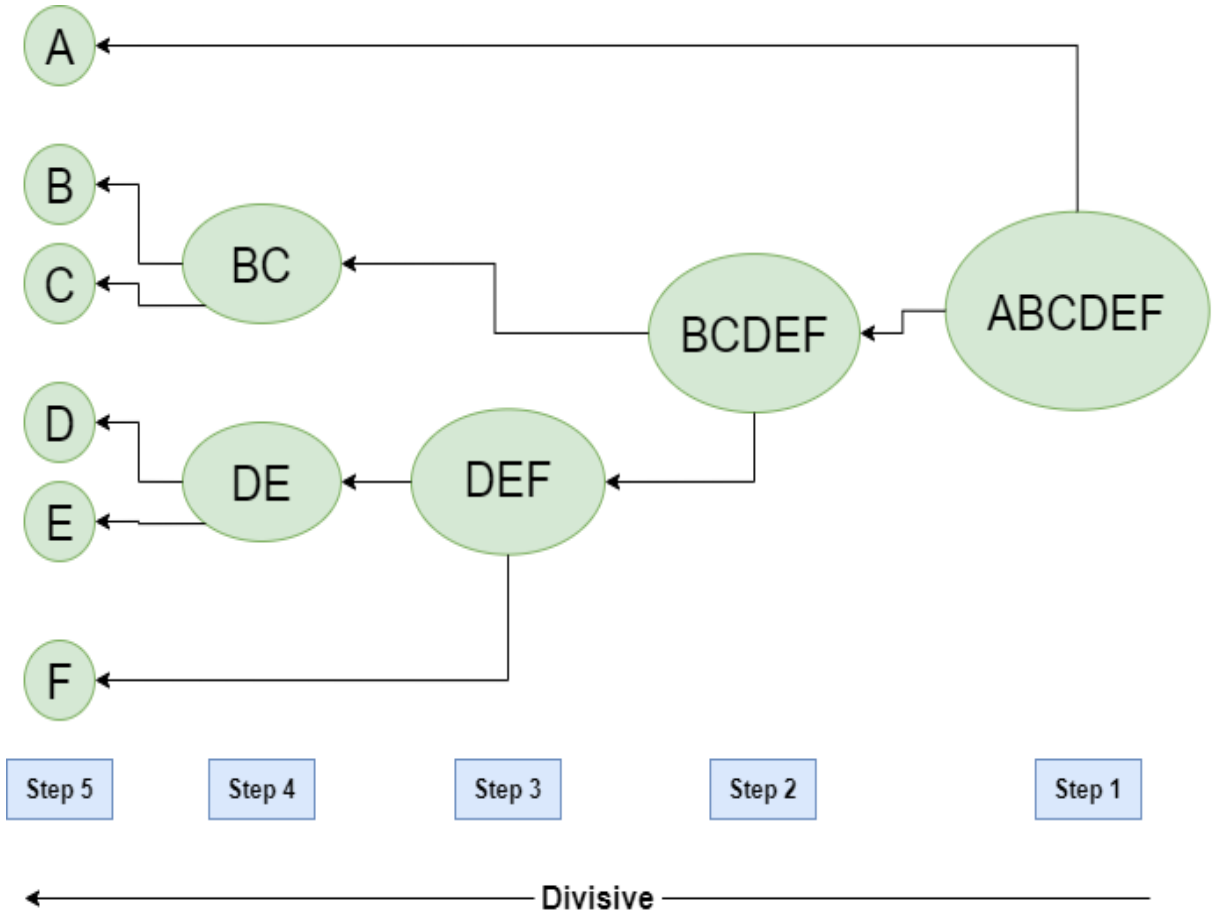
The algorithm for Agglomerative Hierarchical Clustering is:

- Calculate the similarity of one cluster with all the other clusters (calculate proximity matrix)
- Consider every data point as an individual cluster
- Merge the clusters which are highly similar or close to each other.
- Recalculate the proximity matrix for each cluster
- Repeat Steps 3 and 4 until only a single cluster remains.



- **Step-1:** Consider each alphabet as a single cluster and calculate the distance of one cluster from all the other clusters.
- **Step-2:** In the second step comparable clusters are merged together to form a single cluster. Let's say cluster (B) and cluster (C) are very similar to each other therefore we merge them in the second step similarly to cluster (D) and (E) and at last, we get the clusters [(A), (BC), (DE), (F)]
- **Step-3:** We recalculate the proximity according to the algorithm and merge the two nearest clusters([(DE), (F)]) together to form new clusters as [(A), (BC), (DEF)]
- **Step-4:** Repeating the same process; The clusters DEF and BC are comparable and merged together to form a new cluster. We're now left with clusters [(A), (BCDEF)].
- **Step-5:** At last the two remaining clusters are merged together to form a single cluster [(ABCDEF)].

Divisive: Divisive Hierarchical clustering is precisely the **opposite** of Agglomerative Hierarchical clustering. In Divisive Hierarchical clustering, we take into account all of the data points as a single cluster and in every iteration, we separate the data points from the clusters which aren't comparable. In the end, we are left with N clusters.

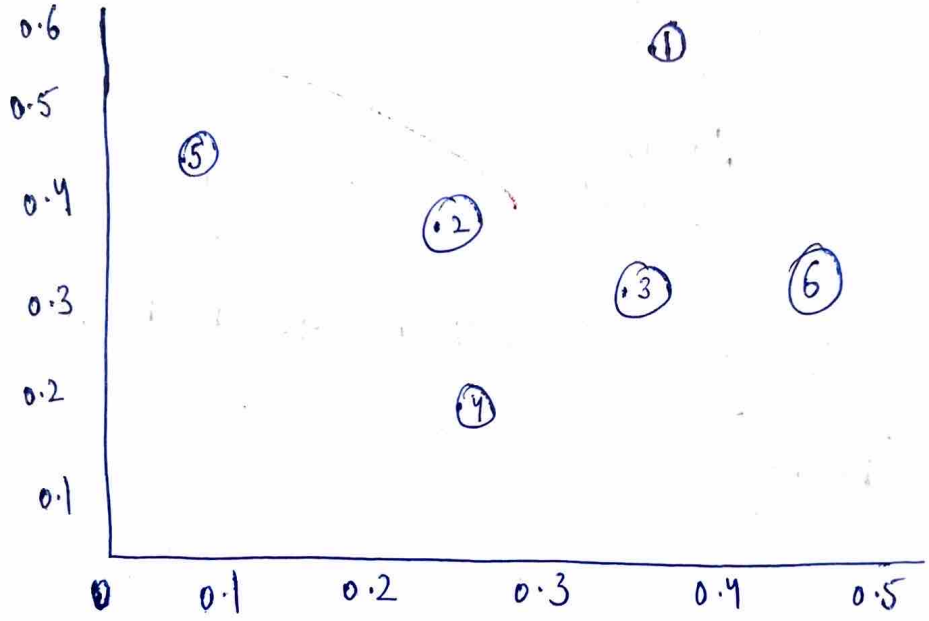


Hierarchical Clustering (Agglomerative)

Find the clusters using single link technique.

	X	Y
P1	0.40	0.53
P2	0.22	0.38
P3	0.35	0.32
P4	0.26	0.19
P5	0.08	0.41
P6	0.45	0.30

P1, P3
P1, P4
P1, P5
P1, P6
P2, P3
P2, P4
P2, P5
P2, P6



Calculate Euclidean distance, create distance matrix

$$\text{Distance}[(x,y),(a,b)] = \sqrt{(x-a)^2 + (x-b)^2}$$

$$\text{Distance}(P1,P2) = \sqrt{(0.40-0.22)^2 + (0.53-0.38)^2}$$

$$(0.40,0.53),(0.22,0.38) = \sqrt{(0.18)^2 + (0.15)^2}$$

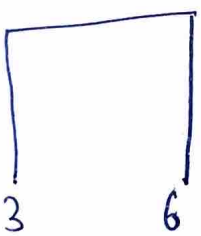
$$= \sqrt{0.0324 + 0.0225}$$

$$= \sqrt{0.0549} = 0.23$$

The distance matrix is

	P1	P2	P3	P4	P5	P6
P1	0					
P2	0.23	0				
P3	0.22	0.15	0			
P4	0.37	0.20	0.15	0		
P5	0.34	0.14	0.28	0.29	0	
P6	0.23	0.25	0.11	0.22	0.39	0

Dendrogram →



To update the distance matrix MIN [dist (P3, P6), P1)]

Min (dist (P3, P1), (P6, P1))

min [(0.22, 0.23)]

= 0.22

To, update the distance matrix MIN [dist (P3, P6), P2]

Min (dist (P3, P2), (P6, P2))

min [(0.15), (0.25)]

= 0.15

~~To, update the distance~~

To, update the distance matrix MIN[dist(P3,P6),P4]

MIN(dist(P3,P4),(P6,P4))

min[(0.15, 0.22)]

= 0.15

To, update the distance matrix MIN[dist(P3,P6),P5]

MIN(dist(P3,P5),(P6,P5))

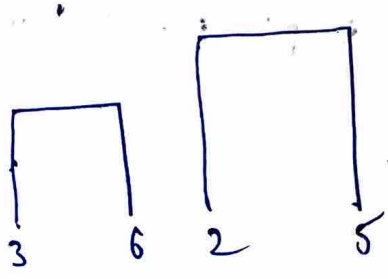
min[0.28, 0.39]

0.28

The updated distance matrix for cluster (P3, P6)

	P1	P2	P3, P6	P4	P5
P1	0				
P2	0.23	0			
P3, P6	0.22	0.15	0		
P4	0.37	0.20	0.15	0	
P5	0.34	0.14	0.28	0.29	0

Now, look for minimum value in updated distance matrix & min value is 0.14. 0.14 is a cluster between P2 & P5



To, update the distance matrix MIN [dist(P2, P5), P1)] (4)

MIN [dist (P2, P1) , (P5, P1)]

= min [0.23, 0.34]

= 0.23

To, update the distance matrix MIN [dist (P2, P5), (P3, P6)]

MIN [dist (P2, (P3, P6), (P5, (P3, P6))]

= min [(0.15, 0.28)]

= 0.15.

To, update the distance matrix MIN [dist (P2, P5), P4)]

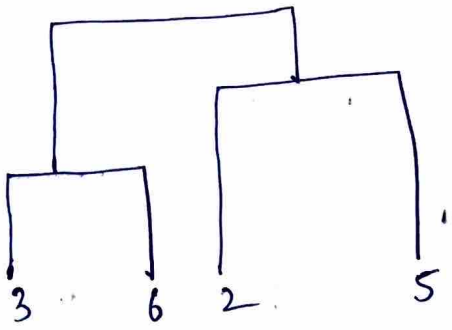
MIN [dist (P2, P4), (P5, P4)]

= min [0.20, 0.29]

= 0.20

Updated distance matrix for cluster P2, P5

	P1	P2, P5	P3, P6	P4
P1	0			
P2, P5	0.23	0		
P3, P6	0.22	0.15	0	
P4	0.37	0.20	0.15	0



To, update the distance matrix MIN[dist(P2, P5), (P3, P6), P1)]

MIN[dist(P2, P5), P1), ((P3, P6), P1)]

= min [(0.23, 0.22)]

= 0.22

To, update the distance matrix MIN[dist(P2, P5), (P3, P6), P4]

MIN[dist(P2, P5), P4), (P3, P6), P4]

= min [0.20, 0.15]

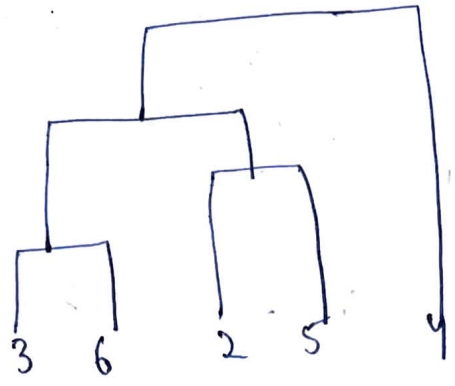
= 0.15

Updated matrix for cluster P2, P5, P3, P6

	P1	P2, P5, P3, P6	P4
P1	0		
P2, P5, P3, P6	0.22	0	
P4	0.37	0.15	0

min value.

$(x_i - x_j)^2$



To, update the distance matrix MIN[dist(P2, P5, P3, P6), P4]

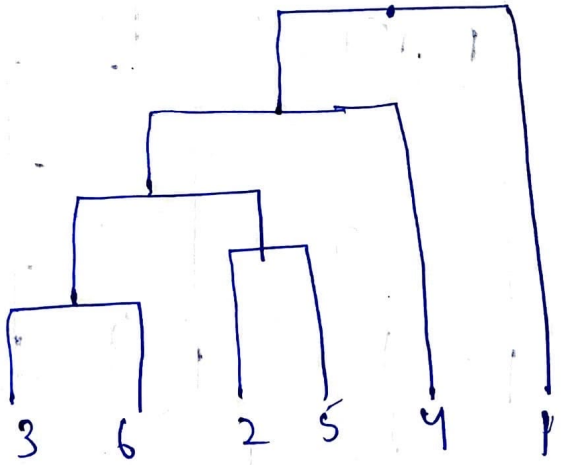
MIN[dist(P2, P5, P3, P6), P1), (P4, P1)]

= min (0.22, 0.37)

= 0.22

Updated distance matrix for cluster (P2, P5, P3, P6, P4)

	P1	P2, P5, P3, P6, P4
P1	0	
P2, P5, P3, P6, P4	0.22	0



Distance measures in algorithm methods

Clustering consists of grouping certain objects that are similar to each other, it can be used to decide if two items are similar or dissimilar in their properties.

In a Data Mining sense, the similarity measure is a distance with dimensions describing object features. That means if the distance among two data points is **small** then there is a **high** degree of similarity among the objects and vice versa. The similarity is **subjective** and depends heavily on the context and application. For example, similarity among vegetables can be determined from their taste, size, colour etc. Most clustering approaches use distance measures to assess the similarities or differences between a pair of objects, the most popular distance measures used are:

1. Euclidean Distance: Euclidean distance is considered the traditional metric for problems with geometry. It can be simply explained as the **ordinary distance** between two points. It is one of the most used algorithms in the cluster analysis. One of the algorithms that use this formula would be **K-mean**.

$$d = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

2. Manhattan Distance:
This determines the absolute difference among the pair of the coordinates.

Suppose we have two points P and Q to determine the distance between these points we simply have to calculate the perpendicular distance of the points from X-Axis and Y-Axis.
In a plane with P at coordinate (x1, y1) and Q at (x2, y2).

Manhattan distance between P and Q = |x1 – x2| + |y1 – y2|

3. Minkowski distance:
It is the **generalized** form of the Euclidean and Manhattan Distance Measure. In an **N-dimensional space**, a point is represented as,

(x1, x2, ..., xN)

Consider two points P1 and P2:

P1: (X1, X2, ..., XN)

P2: (Y1, Y2, ..., YN)

Then, the Minkowski distance between P1 and P2 is given as:

$$\sqrt[p]{(x1 - y1)^p + (x2 - y2)^p + \ldots + (xN - yN)^p}$$

- When **p = 2**, Minkowski distance is same as the **Euclidean** distance.
- When **p = 1**, Minkowski distance is same as the **Manhattan** distance.

BIRCH Technique

The Agglomerative approach is relatively easy to implement because we can simply go on merging the clusters (nodes) having the smallest distance (during each iteration) into larger clusters. The distance between the clusters can be based on single link, complete link or average link approach. This will satisfy the main goal of the clustering process — minimize intra-cluster distance and maximize inter-cluster distance. But, there two main issues — first, inability to undo what was done in previous step, that is, once two clusters are merged into a single single cluster in a particular iteration then this cannot be undone in any further iterations. And second, scalability, that is, Agglomerative approach doesn't scale well.

The Divisive approach is difficult to implement, mainly because of two challenges — first, how to decide splitting criterion, and second, having a set of clusters at particular level in tree, which cluster to split. For both these challenges, the answer is select a cluster to split and split it such that error is minimized. But, it is not trivial. The BIRCH algorithm solves these challenges and also overcomes the above mentioned limitations of agglomerative approach.

BIRCH (balanced iterative reducing and clustering using hierarchies) is an unsupervised data mining algorithm that performs hierarchical clustering over large data sets. With modifications, it can also be used to accelerate k-means clustering and Gaussian mixture modeling with the expectation-maximization algorithm. An advantage of BIRCH is its ability to incrementally and dynamically cluster incoming, multi-dimensional metric data points to produce the best quality clustering for a given set of resources (memory and time constraints). In most cases, BIRCH only requires a single scan of the database. BIRCH stands for Balanced Iterative Reducing & Clustering using Hierarchy. It is multi-phase hierarchical clustering based on Clustering Features (CFs). It is designed for clustering large amount of numeric data by integrating hierarchical clustering in initial phase, called micro-clustering and other clustering methods in later phase, called macro-clustering.

The Clustering Features (CFs)

The most important characteristic of this algorithm is that it uses CF to summarize a cluster and CF tree to represent the clustering hierarchy. So, let's first see what Clustering Feature (CF) is.

The Clustering Feature (CF) of a cluster is a 3-D vector summarizing information about clusters of objects. It is defines as,

CF = (n, LS, SS)

where n is the number of objects in the cluster, LS is the linear sum of the objects and SS is the squared sum of the objects.

$$LS = \sum_{i=1}^{n} x_i \quad \text{and} \quad SS = \sum_{i=1}^{n} x_i^2$$

For example, consider a cluster C1={9,12,10,8,11} then CF(C1)=(5,50,510) where n=5, LS=9+12+10+8+11=50 and SS=$9^2+12^2+10^2+8^2+11^2$=510

Another example with 2-D objects, C2={(1,1),(2,1),(3,2)} then CF(C2)=(3,(6,4),(14,6)) where n=3,LL=(1+2+3,1+1+2)=(6,4) and SS=($1^2+2^2+3^2$, $1^2+1^2+2^2$)=(14,6)

Using CF, we can derive many other useful statistics of a cluster as given below,

Cluster's centroid, $X_0 = \frac{LS}{n}$

Cluster's radius, $R = \sqrt{\frac{\sum_{i=1}^{n}(x_i - X_0)^2}{n}} = \sqrt{\frac{n(SS) - 2LS^2 - n(LS)}{n^2}}$

Cluster's diameter, $D = \sqrt{\frac{\sum_{i=1}^{n}\sum_{j=1}^{n}\left(x_i - x_j\right)^2}{n(n-1)}} = \sqrt{\frac{2n(SS) - 2(LS)^2}{n(n-1)}}$

The algorithm improves space efficiency as it needs to store only CFs of the clusters instead of detailed information about the individual objects within the clusters.

Another important property of the CFs is that they are additive. That is, two disjoint clusters C1 and C2 with CFs CF1=(n1,LS2,SS1) and CF2=(n2,LS2,SS2) respectively, the CF of the cluster formed by merging C1 and C2 is given as, CF1+CF2=(n1+n2,LS1+LS2,SS1+SS2)

For example, C1={(2,5),(3,2),(4,3)} and C2={(1,1),(2,1),(3,1)} then

CF1=(3,(2+3+4,5+2+3),(2^2+3^2+4^2,5^2+2^2+3^2))=(3,(9,10),(29,38)) and
CF2=(3,(1+2+3,1+1+1),(1^2+2^2+3^2,1^2+1^2+1^2))=(3,(6,3), (14,3)) now, if C3=C1UC2 then
CF3=CF1+CF2=(6,(15,13),(43,41))

Stages of BIRCH

BIRCH is often used to complement other clustering algorithms by creating a summary of the dataset that the other clustering algorithm can now use. However, BIRCH has one major drawback it can only process metric attributes. A metric attribute is an attribute whose values can be represented in Euclidean space, i.e., no categorical attributes should be present. The BIRCH clustering algorithm consists of two stages:

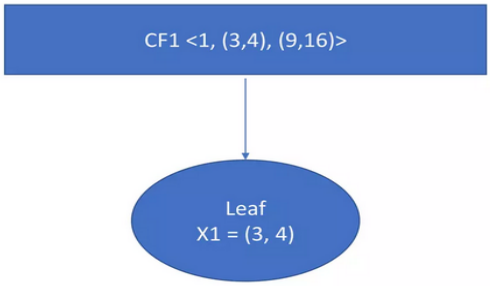
1. **Building the CF Tree:** BIRCH summarizes large datasets into smaller, dense regions called Clustering Feature (CF) entries. Formally, a Clustering Feature entry is defined as an ordered triple (N, LS, SS) where 'N' is the number of data points in the cluster, 'LS' is the linear sum of the data points, and 'SS' is the squared sum of the data points in the cluster. A CF entry can be composed of other CF entries. Optionally, we can condense this initial CF tree into a smaller CF.
2. **Global Clustering:** Applies an existing clustering algorithm on the leaves of the CF tree. A CF tree is a tree where each leaf node contains a sub-cluster. Every entry in a CF tree contains a pointer to a child node, and a CF entry made up of the sum of CF entries in the child nodes. Optionally, we can refine these clusters. Due to this two-step process, BIRCH is also called Two-Step Clustering.

Example:

Let Have Following Data
X1=(3,4), x2= (2,6), x3=(4,5), x4=(4,7), x5=(3,8), x6=(6,2), x7=(7,2), x8=(7,4), x9=(8,4), x10=(7,9)
Cluster the Above Data Using BIRCH Algorithm, , considering T<1.5, and Max Branch = 2

For each Data Point we need to evaluate Radius and Cluster Feature:

->Consider Data Pint (3,4):

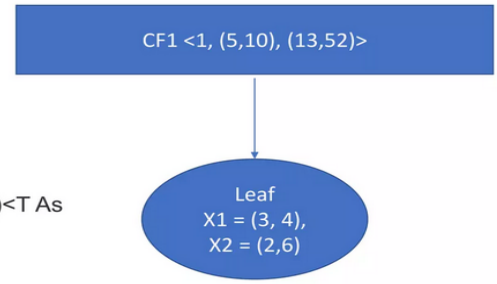


As it is alone in the Feature map, Hence
1. Radius = 0
2. Cluster Feature CF1 <N, LS, SS>
   N = 1 as there is now one data point under consideration.
   LS = Sum of Data Point under consideration = (3,4)
   SS = Square Sum of Data Point Under Consideration
   = (3^2, 4^2)=(9,16)
3. Now construct the Leaf with Data Point X1 and Branch as CF1.

Let Have Following Data
X1=(3,4), x2= (2,6), x3=(4,5), x4=(4,7), x5=(3,8), x6=(6,2), x7=(7,2), x8=(7,4), x9=(8,4), x10=(7,9)
Cluster the Above Data Using BIRCH Algorithm, considering T<1.5, and Max Branch = 2

For each Data Point we need to evaluate Radius and Cluster Feature:

->Consider Data Pint x2 = (2,6):



1. Linear Sum LS = (3,4) + (2,6) = (5,10)
2. Square Sum SS = (3^2+2^2 , 4^2+6^2) =(13, 52)

Now Evaluate Radius considering N=2

$$R = \sqrt{\frac{SS - LS^2/N}{N}} = \sqrt{\frac{(13,52) - (5,10)^2/2}{2}} = \sqrt{\frac{(13,52) - (25,100)/2}{2}} =$$

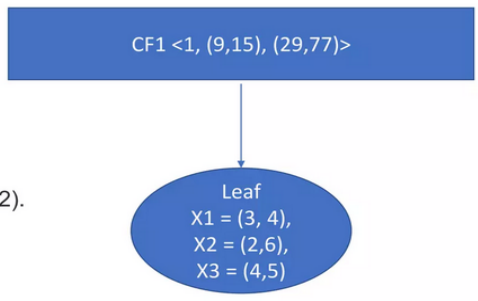
$$\sqrt{\frac{(13,52) - (12.5,50)}{2}} = \sqrt{(6.5,26) - (6.25,25)} = \sqrt{(0.25,1)} = (0.5, 1) < T$$ As
(0.25,1) < (T, T), hence X2 will cluster with Leaf X1.

2. Cluster Feature CF1 <N, LS, SS> = <2,(5,10),(13,52)>
   N = 2 as there is now two data point under CF1.
   LS = (3,4) + (2,6) = (5,10)
   SS = (3^2+2^2 , 4^2+6^2) =(13, 52)

Let Have Following Data
X1=(3,4), x2= (2,6), x3=(4,5), x4=(4,7), x5=(3,8), x6=(6,2), x7=(7,2), x8=(7,4), x9=(8,4), x10=(7,9)
Cluster the Above Data Using BIRCH Algorithm, considering T<1.5, and Max Branch = 2

For each Data Point we need to evaluate Radius and Cluster Feature:

->Consider Data Pint x3 = (4,5) on CF1:



1. Linear Sum LS = (4,5) + (5,10) = (9,15)
2. Square Sum SS = (4^2+13 , 5^2 + 52) =(29, 77)

Now Evaluate Radius considering N=3

$$R = \sqrt{\frac{SS - LS^2/N}{N}} = \sqrt{\frac{(29,77) - (9,15)^2/3}{3}} = (0.47, 0.4714) < T$$

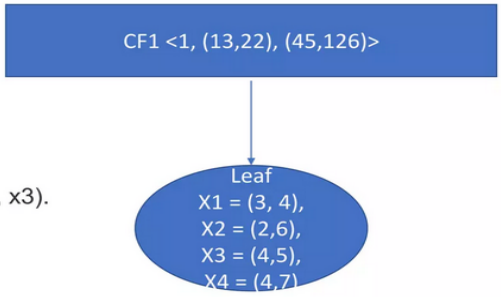
As (0.47, 0.471) < (T, T), hence X3 will cluster with Leaf (X1, x2).

2. Cluster Feature CF1 <N, LS, SS> = <3,(9,15),(29,77)>
   N = 3 as there is now Three data point under CF1.
   LS = (4,5) + (5,10) = (9,15)
   SS = (4^2+13 , 5^2 + 52) =(29, 77)

Let Have Following Data
X1=(3,4), x2= (2,6), x3=(4,5), x4=(4,7), x5=(3,8), x6=(6,2), x7=(7,2), x8=(7,4), x9=(8,4), x10=(7,9)
Cluster the Above Data Using BIRCH Algorithm, considering T<1.5, and Max Branch = 2

For each Data Point we need to evaluate Radius and Cluster Feature:

->Consider Data Pint x4 = (4,7) on CF1:



1. Linear Sum LS = (4,7) + (9,15) = (13,22)
2. Square Sum SS = (4^2+29 , 7^2 + 77) =(45, 126)

Now Evaluate Radius considering N=4

$$R = \sqrt{\frac{SS - LS^2/N}{N}} = \sqrt{\frac{(45,126) - (13,22)^2/4}{4}} = (0.41, 0.55)$$

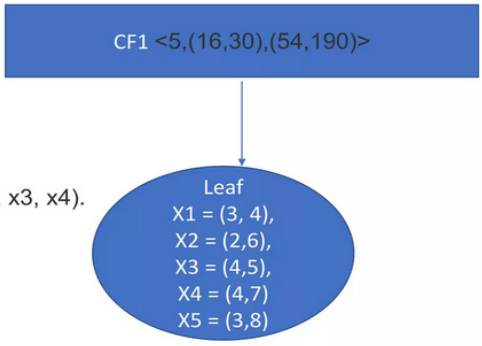
As (0.41, 0.55) < (T, T), hence X4 will cluster with Leaf (X1, x2, x3).

2. Cluster Feature CF1 <N, LS, SS> = <4,(13,22),(45,126)>
   N = 4 as there is now four data point under CF1.
   LS = (4,7) + (9,15) = (13,22)
   SS = (4^2+29 , 7^2 + 77) =(45, 126)

Let Have Following Data
X1=(3,4), x2= (2,6), x3=(4,5), x4=(4,7), x5=(3,8), x6=(6,2), x7=(7,2), x8=(7,4), x9=(8,4), x10=(7,9)
Cluster the Above Data Using BIRCH Algorithm, considering T<1.5, and Max Branch = 2

For each Data Point we need to evaluate Radius and Cluster Feature:

->Consider Data Pint x5 = (3,8) on CF1:



1. Linear Sum LS = (3,8) + (13,22) = (16,30)
2. Square Sum SS = (3^2+45 , 8^2 + 126) =(54, 190)

Now Evaluate Radius considering N=5

$$R = \sqrt{\frac{SS - LS^2/N}{N}} = \sqrt{\frac{(54,190) - (16,30)^2/5}{5}} = (0.33, 0.63)$$

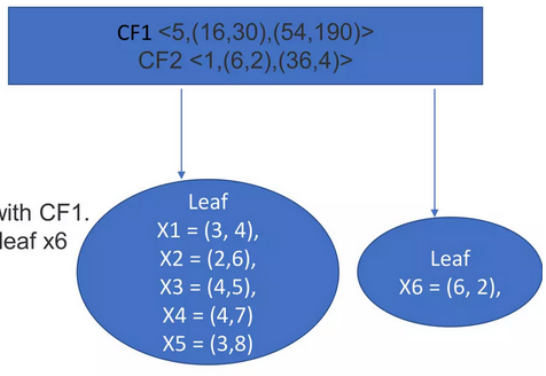
As (0.33, 0.63) < (T, T), hence X5 will cluster with Leaf (X1, x2, x3, x4).

2. Cluster Feature CF1 <N, LS, SS> = <5,(16,30),(54,190)>
   N = 5 as there is now four data point under CF1.

Let Have Following Data
X1=(3,4), x2= (2,6), x3=(4,5), x4=(4,7), x5=(3,8), x6=(6,2), x7=(7,2), x8=(7,4), x9=(8,4), x10=(7,9)
Cluster the Above Data Using BIRCH Algorithm, considering T<1.5, and Max Branch = 2

For each Data Point we need to evaluate Radius and Cluster Feature:

->Consider Data Pint x6 = (6,2) on CF1:



1. Linear Sum LS = (6,2) + (16,30) = (22,32)
2. Square Sum SS = (6^2+54 , 2^2 + 190) =(90, 194)

Now Evaluate Radius considering N=6

$$R = \sqrt{\frac{SS - LS^2/N}{N}} = \sqrt{\frac{(90,194) - (22,32)^2/6}{6}} = (1.24, 1.97)$$

As (1.24, 1.97) < (T, T), False. hence X6 will Not form cluster with CF1. CF1 will remain as it was in previous step. And New CF2 with leaf x6 will be created.

2. Cluster Feature CF2 <N, LS, SS> = <1,(6,2),(36,4)>
   N = 1 as there is now one data point under CF2.
   LS = (6,2)
   SS = (6^2, 2^2)= (36,4)

Let Have Following Data
X1=(3,4), x2= (2,6), x3=(4,5), x4=(4,7), x5=(3,8), x6=(6,2), x7=(7,2), x8=(7,4), x9=(8,4), x10=(7,9)
Cluster the Above Data Using BIRCH Algorithm, considering T<1.5, and Max Branch = 2

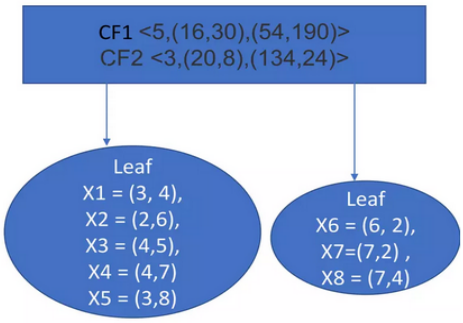
->Consider Data Pint x8 = (7,4). As There are Two Branch CF1 and CF2 hence we need to find with which branch X8 is nearer, then with that leaf, radius will be evaluated.
With CF1 = LS/N= (16,30)/5=(8,6) As there are N=5 Data Point
With CF2 = LS/N= (13,4)/2=(6.5,2) As there is N=2 Data Point
Now x8 is closer to (6.5,2) then (8,6). Hence X8 will calculate radius with CF2.

1. Linear Sum LS = (7,4) + (13,4) = (20,8)
2. Square Sum SS = (7^2+85 , 4^2 + 8) =(134, 24)

Now Evaluate Radius considering N=3

$$R = \sqrt{\frac{SS - LS^2/N}{N}} = \sqrt{\frac{(134,24) - (20,8)^2/3}{3}} = (0.47, 0.94)$$

As (0.47, 94) < (T, T), True. hence X8 will form cluster with CF2
2. Cluster Feature CF2 <N, LS, SS> = <3,(20,8),(134,24)>
N = 3 as there is now two data point under CF2.
LS (7,4) + (13,4) = (20,8)
SS = (134,24)



Let Have Following Data
X1=(3,4), x2= (2,6), x3=(4,5), x4=(4,7), x5=(3,8), x6=(6,2), x7=(7,2), x8=(7,4), x9=(8,4) , x10=(7,9)
Cluster the Above Data Using BIRCH Algorithm, considering T<1.5, and Max Branch = 2

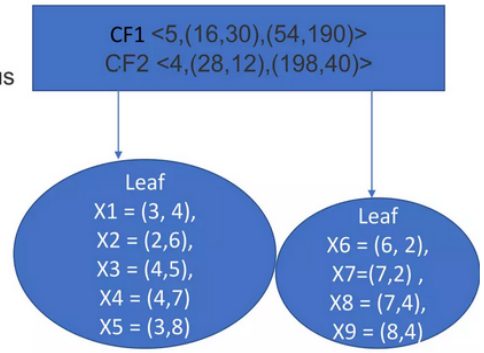
->Consider Data Pint x9 = (8,4). As There are Two Branch CF1 and CF2 hence we need to find with which branch X9 is nearer, then with that leaf, radius will be evaluated.
With CF1 = LS/N= (16,30)/5=(8,6) As there are N=5 Data Point
With CF2 = LS/N= (20,8)/3=(6.6,2.6) As there is N=3 Data Point
Now x9 is closer to (6.6,2.6) then (8,6). Hence X8 will calculate radius with CF2.

1. Linear Sum LS = (8,4) + (20,8) = (28,12)
2. Square Sum SS = (8^2+134 , 4^2 + 24) =(198, 40)

Now Evaluate Radius considering N=4

$$R = \sqrt{\frac{SS - LS^2/N}{N}} = \sqrt{\frac{(198,40) - (28,12)^2/4}{4}} = (0.70, 1)$$

As (0.7, 1) < (T, T), True. hence X9 will form cluster with CF2
2. Cluster Feature CF2 <N, LS, SS> = <4,(28,12),(198,40)>
N = 4 as there is now four data point under CF2.
LS = (28,12)
SS = (198,40)



Example

Let Have Following Data
X1=(3,4), x2= (2,6), x3=(4,5), x4=(4,7), x5=(3,8), x6=(6,2), x7=(7,2), x8=(7,4), x9=(8,4) , x10=(7,9)
Cluster the Above Data Using BIRCH Algorithm, considering T<1.5, and Max Branch = 2

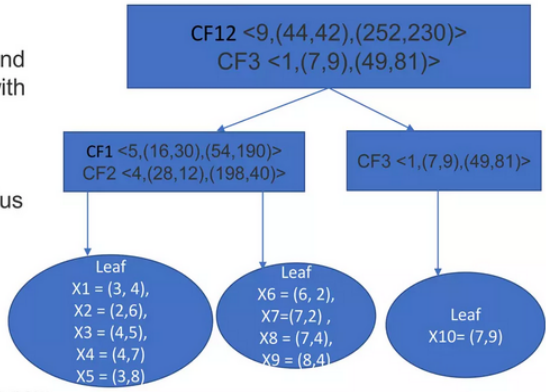
->Consider Data Pint x10 = (7,9). As There are Two Branch CF1 and CF2 hence we need to find with which branch X9 is nearer, then with that leaf, radius will be evaluated.
With CF1 = LS/N= (16,30)/5=(8,6) As there are N=5 Data Point
With CF2 = LS/N= (28,12)/4=(7,3) As there is N=4 Data Point
Now x10 is closer to (8,6) then (7,3). Hence X10 will calculate radius with CF1.

1. Linear Sum LS = (7,9) + (16,30) = (23,39)
2. Square Sum SS = (7^2+54 , 9^2 + 190) =(103, 271)

Now Evaluate Radius considering N=6

$$R = \sqrt{\frac{SS - LS^2/N}{N}} = \sqrt{\frac{(103,271) - (23,39)^2/6}{6}} = (1.57, 1.70)$$

As (1.57, 1.70) < (T, T), False. hence X10 will become new leaf and Create new cluster feature CF3. But in a Branch only two CF is allowed hence Branch will Split.
2. Cluster Feature CF3 <N, LS, SS> = <1,(7,9),(49,81)>



DBSCAN

Clustering analysis or simply Clustering is basically an Unsupervised learning method that divides the data points into a number of specific batches or groups, such that the data points in the same groups have similar properties and data points in different groups have different properties in some sense. It comprises many different methods based on differential evolution.
E.g. K-Means (distance between points), Affinity propagation (graph distance), Mean-shift (distance between points), DBSCAN (distance between nearest points), Gaussian mixtures (Mahalanobis distance to centers), Spectral clustering (graph distance), etc. Fundamentally, all clustering methods use the same approach i.e. first we calculate similarities and then we use it to cluster the data points into groups or batches.

Density-Based Spatial Clustering Of Applications With Noise (DBSCAN)

Clusters are dense regions in the data space, separated by regions of the lower density of points. The **DBSCAN algorithm** is based on this intuitive notion of "clusters" and "noise". The key idea is that for each point of a cluster, the neighborhood of a given radius has to contain at least a minimum number of points.

Density-Based **Clustering** refers to unsupervised learning methods that identify distinctive groups/clusters in the data, based on the idea that a cluster in data space is a contiguous region of high point density, separated from other such clusters by contiguous regions of low point density.

Density-Based Spatial Clustering of Applications with Noise (DBSCAN) is a base algorithm for density-based clustering. It can discover clusters of different shapes and sizes from a large amount of data, which is containing noise and outliers.

Why DBSCAN?

Partitioning methods (K-means, PAM clustering) and hierarchical clustering work for finding spherical-shaped clusters or convex clusters. In other words, they are suitable only for compact and well-separated clusters. Moreover, they are also severely affected by the presence of noise and outliers in the data.

Parameters Required For DBSCAN Algorithm

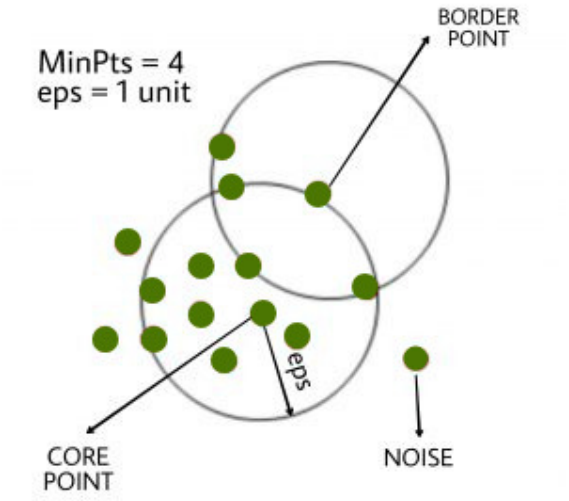
1. **eps**: It defines the neighborhood around a data point i.e. if the distance between two points is lower or equal to 'eps' then they are considered neighbors. If the eps value is chosen too small then a large part of the data will be considered as an outlier. If it is chosen very large then the clusters will merge and the majority of the data points will be in the same clusters. One way to find the eps value is based on the **k-distance graph**.
2. **MinPts**: Minimum number of neighbors (data points) within eps radius. The larger the dataset, the larger value of MinPts must be chosen. As a general rule, the minimum MinPts can be derived from the number of dimensions D in the dataset as, MinPts >= D+1. The minimum value of MinPts must be chosen at least 3.

These parameters can be understood if we explore two concepts called Density Reachability and Density Connectivity.

Reachability in terms of density establishes a point to be reachable from another if it lies within a particular distance (eps) from it.

Connectivity, on the other hand, involves a transitivity based chaining-approach to determine whether points are located in a particular cluster. For example, p and q points could be connected if p->r->s->t->q, where a->b means b is in the neighborhood of a.

In this algorithm, we have 3 types of data points.
Core Point: A point is a core point if it has more than MinPts points within eps.
Border Point: A point which has fewer than MinPts within eps but it is in the neighborhood of a core point.
Noise or outlier: A point which is not a core point or border point.



Steps Used In DBSCAN Algorithm

1. Find all the neighbor points within eps and identify the core points or visited with more than MinPts neighbors.
2. For each core point if it is not already assigned to a cluster, create a new cluster.
3. Find recursively all its density-connected points and assign them to the same cluster as the core point.
   A point a and b are said to be density connected if there exists a point c which has a sufficient number of points in its neighbors and both points a and b are within the eps distance. This is a chaining process. So, if b is a neighbor of c, c is a neighbor of d, and d is a neighbor of e, which in turn is neighbor of a implying that b is a neighbor of a.
4. Iterate through the remaining unvisited points in the dataset. Those points that do not belong to any cluster are noise.

DBSCAN Clustering

Q. Apply DBSCAN algo to the given data points and create clusters with minPts = 4 & epsilon (ε) = 1.9

Data Points:

P1: (3, 7)
P2: (4, 6)
P3: (5, 5)
P4: (6, 4)
P5: (7, 3)
P6: (6, 2)
P7: (7, 2)
P8: (8, 4)
P9: (3, 3)
P10: (2, 6)
P11: (3, 5)
P12: (2, 4)

Soln Use Euclidean distance & cal. distance b/w each points.

$$\text{Distance}(A(x_1, y_1), B(x_2, y_2)) = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

	P1	P2	P3	P4	P5	P6	P7	P8	P9	P10	P11	P12
P1	0											
P2	1.41	0										
P3	2.83	1.41	0									
P4	4.24	2.83	1.41	0								
P5	5.66	4.24	2.83	1.41	0							
P6	5.83	4.47	3.16	2.00	1.41	0						
P7	6.40	5.00	3.61	2.24	1.00	1.00	0					
P8	5.83	4.47	3.16	2.00	1.41	2.83	2.24	0				
P9	4.00	3.16	2.83	3.16	4.00	3.16	4.12	5.10	0			
P10	1.41	2.00	3.16	4.47	5.83	5.66	6.40	6.32	3.16	0		
P11	2.00	1.41	2.00	3.16	4.47	4.24	5.00	5.10	2.00	1.41	0	
P12	3.16	2.83	3.16	4.00	5.10	4.47	5.39	6.00	1.41	2.00	1.41	0

From P1 try to identify min distance
first horizontally & then vertically

P1: P2, P10 as 1.41 < 1.9

P2: P1, P3, P11

P3: P2, P4

P4: P3, P5

P5: P4, P6, P7, P8

P6: P5, P7

P7: P5, P6

P8: P5

P9: P12

P10: P1, P11

P11: P2, P10, P12

P12: P9, P11

Points	Status	
P1	Noise	Border
P2	Core	—
P3	N	Border
P4	N	Border
P5	C	—
P6	N	Border
P7	N	Border
P8	N	Border
P9	N	—
P10	N	Border
P11	C	—
P12	N	Border

We need to identify these noise points are border data points

P 9 remains noise here.
We will get 3 clusters centered at P 2, P 5, P 11

DBSCAN - Density based spatial clustering of applications with noise. It is based on notion of "clusters" & "noise"

eps :- It defines the neighborhood around a data point i.e. if the distance b/w 2 points is lower or equal to eps then they are considered neighbors.

STING – Statistical Information Grid in Data Mining

STING is a Grid-Based Clustering Technique. In STING, the dataset is recursively divided in a hierarchical manner. After the dataset, each cell is divided into a different number of cells. And after the cell, the statistical measures of the cell are collected, which helps answer the query as quickly as possible.

Grid-Based Method in Data Mining:

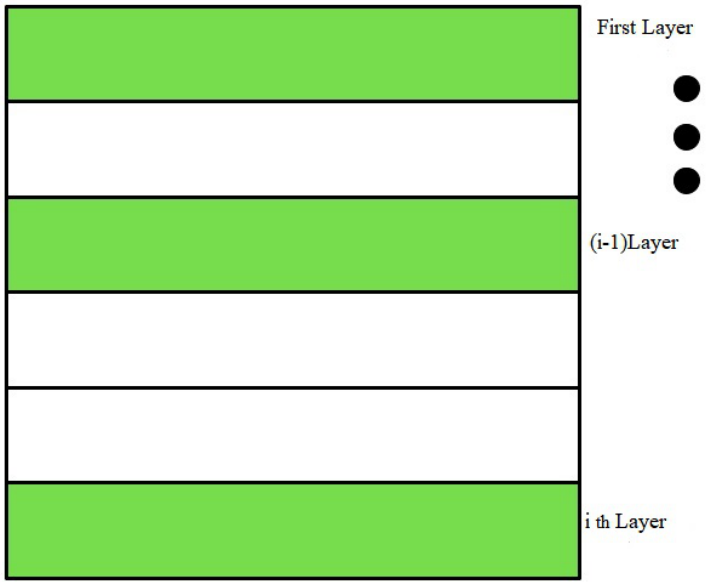
In Grid-Based Methods, the space of instance is divided into a grid structure. Clustering techniques are then applied using the Cells of the grid, instead of individual data points, as the base units. The biggest advantage of this method is to improve the processing time.

Statistical Information Grid(STING):

A STING is a grid-based clustering technique. It uses a multidimensional grid data structure that quantifies space into a finite number of cells. Instead of focusing on data points, it focuses on the value space surrounding the data points.

In STING, the spatial area is divided into rectangular cells and several levels of cells at different resolution levels. High-level cells are divided into several low-level cells.

In STING Statistical Information about attributes in each cell, such as mean, maximum, and minimum values, are precomputed and stored as statistical parameters. These statistical parameters are useful for query processing and other data analysis tasks.



The statistical parameter of higher-level cells can easily be computed from the parameters of the lower-level cells.

How STING Work:

Step 1: Determine a layer, to begin with.

Step 2: For each cell of this layer, it calculates the confidence interval or estimated range of probability that this is cell is relevant to the query.

Step 3: From the interval calculate above, it labels the cell as relevant or not relevant.

Step 4: If this layer is the bottom layer, go to point 6, otherwise, go to point 5.

Step 5: It goes down the hierarchy structure by one level. Go to point 2 for those cells that form the relevant cell of the high-level layer.

Step 6: If the specification of the query is met, go to point 8, otherwise go to point 7.

Step 7: Retrieve those data that fall into the relevant cells and do further processing. Return the result that meets the requirement of the query. Go to point 9.

Step 8: Find the regions of relevant cells. Return those regions that meet the requirement of the query. Go to point 9.

Step 9: Stop or terminate.

Advantages:

- Grid-based computing is query-independent because the statistics stored in each cell represent a summary of the data in the grid cells and are query-independent.
- The grid structure facilitates parallel processing and incremental updates.

Disadvantage:

- The main disadvantage of Sting (Statistics Grid). As we know, all cluster boundaries are either horizontal or vertical, so no diagonal boundaries are detected.

CLIQUE Algorithm

CLIQUE is a density-based and grid-based subspace clustering algorithm. So let's first take a look at what is a grid and density-based clustering technique.

- **Grid-Based Clustering Technique:** In Grid-Based Methods, the space of instance is divided into a grid structure. Clustering techniques are then applied using the Cells of the grid, instead of individual data points, as the base units.
- **Density-Based Clustering Technique:** In Density-Based Methods, A cluster is a maximal set of connected dense units in a subspace.

  CLIQUE Algorithm uses density and grid-based technique i.e subspace clustering algorithm and finds out the cluster by taking density threshold and a number of grids as input parameters. It is specially designed to handle datasets with a large number of dimensions.CLIQUE Algorithm is very scalable with respect to the value of the records, and a number of dimensions in the dataset because it is grid-based and uses the Apriori Property effectively. APRIORI APPROACH ?.

  Apriori Approach Stated that If an X dimensional unit is dense then all its projections in X-1 dimensional space are also dense.

  This means that dense regions in a given subspace must produce dense regions when projected to a low-dimensional subspace. CLIQUE restricts its search for high-dimensional dense cells to the intersection of dense cells in the subspace because CLIQUE uses apriori properties.

Working of CLIQUE Algorithm:

The CLIQUE algorithm first divides the data space into grids. It is done by dividing each dimension into equal intervals called units. After that, it identifies dense units. A unit is dense if the data points in this are exceeding the threshold value.

Once the algorithm finds dense cells along one dimension, the algorithm tries to find dense cells along two dimensions, and it works until all dense cells along the entire dimension are found.

After finding all dense cells in all dimensions, the algorithm proceeds to find the largest set ("cluster") of connected dense cells. Finally, the CLIQUE algorithm generates a minimal description of the cluster. Clusters are then generated from all dense subspaces using the apriori approach.

Advantage:

- CLIQUE is a subspace clustering algorithm that outperforms K-means, DBSCAN, and Farthest First in both execution time and accuracy.
- CLIQUE can find clusters of any shape and is able to find any number of clusters in any number of dimensions, where the number is not predetermined by a parameter.
- One of the simplest methods, and interpretability of results.

Disadvantage:

- The main disadvantage of CLIQUE Algorithm is that if the size of the cell is unsuitable for a set of very high values, then too much of the estimation will take place and the correct cluster will be unable to find.